INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia de Eletrónica e Telecomunicações e de Computadores

Automatic Identification of *Not Suitable For Work* images

DANIEL PESCA RODRIGUES BICHO

Master Degree

Projecto Final para obtenção do Grau de Mestre
em Engenharia Informática e de Computadores

Orientadores : Prof. Doutor Artur Jorge Ferreira
Prof. Doutor Nuno Miguel Soares Datia

Júri:

Presidente: Prof. Doutor Tiago Miguel Braga da Silva Dias

Vogais: Prof. Doutor Gonçalo Caetano Marques
Prof. Doutor Artur Jorge Ferreira

October, 2019

INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia de Eletrónica e Telecomunicações e de Computadores

Automatic Identification of *Not Suitable For Work* images

DANIEL PESCA RODRIGUES BICHO

Master Degree

Projecto Final para obtenção do Grau de Mestre
em Engenharia Informática e de Computadores

Orientadores : Prof. Doutor Artur Jorge Ferreira
Prof. Doutor Nuno Miguel Soares Datia

Júri:

Presidente: Prof. Doutor Tiago Miguel Braga da Silva Dias

Vogais: Prof. Doutor Gonçalo Caetano Marques
Prof. Doutor Artur Jorge Ferreira

October, 2019

Abstract

Web Archiving is a research field that addresses the problem of how to collect portions of the World Wide Web (WWW) to ensure that a given piece of information is preserved in an archive for future researchers, historians and the general public. It is important to make available and discoverable this unique and historically valuable information. Arquivo.pt is a Web Archiving initiative that provides a full-text search method to access all its data. Also, it is developing an Image Search system to enhance its data access. There are huge amounts of visual content at Arquivo.pt stored as image files. Some of these contents, such as nudity and pornography, can be offensive for the users.

This work studies and implements a methodology to automatically identify this kind of image content, known as Not Safe For Work (NSFW), using deep neural network methodologies. An evaluation dataset is built using Arquivo.pt data and two pre-trained neural network models, ResNet and SqueezeNet, are evaluated and improved for this task using this dataset. Afterwards, the proposed solution is integrated into the Arquivo.pt Image Search System, enabling the filtering of these problematic images.

The evaluation of the models reported 93% accuracy for the ResNet neural network model and 72% for the SqueezeNet neural network model. The accuracy of the models were improved afterwards, the ResNet neural network increased to 94% and the SqueezeNet to 89%.

The work is currently in production at https://arquivo.pt/images.jsp.

Keywords: Image Classification, Deep Neural Networks, Web Archiving, Not Safe For Work.

Resumo

Web Archiving é um ramo de investigação que endereça o problema de como recolher porções da World Wide Web de forma a garantir que esta informação é preservada num arquivo para futuros investigadores, historiadores e público em geral. É importante tornar disponível e acessível esta informação valiosa, única e histórica. O Arquivo.pt é um Arquivo da Web que providencia um método de pesquisa por texto para aceder a todos estes dados. Um sistema de pesquisa por imagem está também a ser desenvolvido para melhorar o acesso aos dados. Existem muitos conteúdos visuais no Arquivo.pt. Parte destes conteúdos, como conteudos com nudez e pornografia, podem ser ofensivos para os utilizadores.

Neste trabalho, estuda-se e implementam-se metodologias para automaticamente identificar este tipo de conteudos de imagem, conhecidos como NSFW, através de redes neuronais. Um conjunto de avaliação é construido usando os dados do Arquivo.pt e dois modelos de redes neuronais previamente treinados, ResNet e SqueezeNet, sao avaliados e melhorados para esta tarefa. De seguida, é integrado no sistema de pesquisa por imagem do Arquivo.pt, permitindo a filtragem destas imagens com contéudo problemático.

A avaliação inicial dos modelos reporta 93% de taxa de acerto para o modelo de rede neural ResNet e 72% para o modelo SqueezeNet. A taxa de acerto destes modelos foi melhorada posteriormente, o modelo ResNet aumento a taxa de acerto para 94% e o SqueezeNet para 89%.

Este trabalho encontra-se actualmente em produção em https://arquivo.pt/images.jsp.

Palavras-chave: Classificação de Imagens, Redes neuronais profundas, Arquivamento da Web, Não seguro para trabalhar.

Contents

List of Figures

List of Tables

Acronyms

AI Artificial Intelligence. 32

ANN Artificial Neural Network. 17

ARC ARChive file format. 12–14

AUC Area Under the receiver operating characteristic Curve. 27

BLAS Basic Linear Algebra Subprograms. 30

BoVW Bag of Visual Words. 7, 8

BVLC Berkeley Vision and Learning Center. 30

CDX Capture inDeXes. 12, 14

CNN Convolutional Neural Networks. 8, 18, 43

CV Computer Vision. 32

DAG Directed Acyclic Graph. 18

DL Deep Learning. 8, 9, 38

DNN Deep Neural Network. 9, 17

FC Fully-Connected. 9

FN False Negatives. 29–31

FP False Positives. 29, 31

GAN Generative Adversarial Networks. 32

GIF Graphics Interchange Format. 39

GPU Graphics Processing Unit. 38

HTML HyperText Markup Language. 1, 26

HTTP HyperText Transfer Protocol. 14

NSFW Not Safe For Work. v, vii, 4, 7, 9, 29, 39, 40

NSFW Not Suitable for Work. 3

ROC Receiver Operating characteristic Curve. 27

SFW Safe For Work. 4, 29

SGD Stochastic Gradient Descent. 23, 24, 34, 35

SIFT Scale-Invariant Feature Transform. 8

TN True Negatives. 29

TP True Positives. 29

URL Uniform Resource Locator. 12, 39

WA Web Archiving. 1

WAIR Web Archiving Information Retrieval. 2

WARC Web Archive file format. 4, 12–14, 37, 38

WWW World Wide Web. v, 1

1

Introduction

Web Archiving (WA) [30] is a research field that addresses the problem of collecting portions of the World Wide Web (WWW) to ensure that information is preserved in an archive for future researchers, historians and the general public.

The most common way used by Web Archives to collect this information for preservation means is through Web crawlers such as Heritrix [33], specialized at archiving the Web. It automates the process of harvesting Web pages and preserving their contents. These contents include any resource type, such as HyperText Markup Language (HTML), style sheets, JavaScript, images, videos and metadata related to these resources such as access times, resource mime-types and content length.

Choosing what to be preserved is a hard challenge, since there isn't enough storage space to keep everything and the amount of data available on the Web is permanently growing.

There are several WA initiatives worldwide that try to preserve the WWW. Some Web Archives have more narrow scopes, preserving only some very specific kinds of pages such as institutional Web pages (European Commission Historical Archives[1]), while others try to preserve the entire national top-level domain (UK Web Archive[2]), or the entire Web (Internet Archive[3]).

[1] http://ec.europa.eu/historical_archives/index_en.htm
[2] https://www.webarchive.org.uk/ukwa/
[3] https://archive.org/

Arquivo.pt[4] is one of those initiatives. Its main goal is to preserve Web pages from the Portuguese national Web domain (*.pt*) as well as Web pages that contain Portuguese related important information. It also acts as a research infrastructure making its contents searchable and available in open access to researchers and to the general public.

It is important to make available and discoverable this unique and historically valuable information. Without the tools for users to find the desired information, the usefulness of Web Archives is hampered.

To accomplish the ability to search, Arquivo.pt provides a full-text search system to all its data. There are significant studies and efforts to improve the Web Archiving Information Retrieval (WAIR) capabilities. For instance, Miguel Costa tries to improve the information search on Web Archives, by exploring the temporal information that is intrinsic to them [7].

Following this task to provide better searching capabilities to this important information, Arquivo.pt is developing an Image Search service. This service enables image retrieval capabilities to Arquivo.pt preserved contents, presenting an interface in which users can perform queries and the service will try to retrieve images related to the user query.

Figure 1.1 presents the prototype of the Image Search service. A text box is available for users to fill with query terms like *Merkel* and the service displays images

[4] http://arquivo.pt

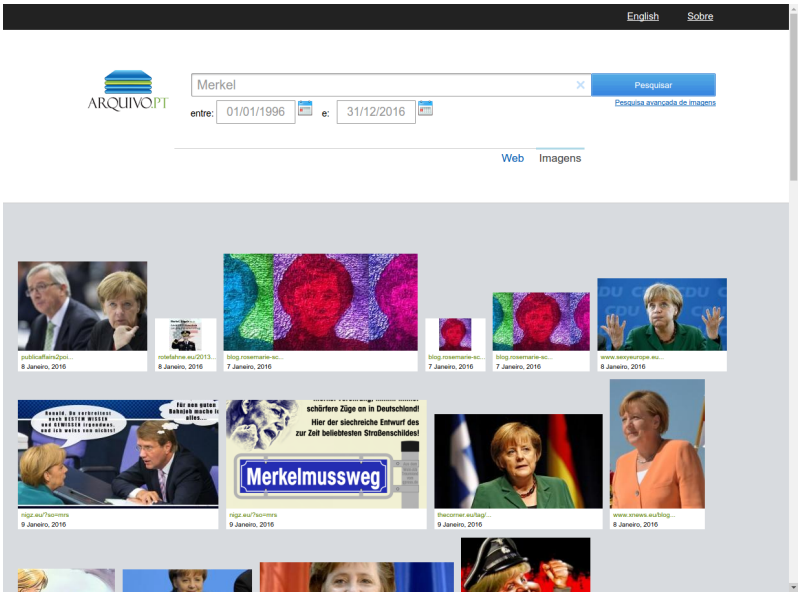


Figure 1.1: Example of Arquivo.pt Image Search functionality.

related to the user query. On this specific use case, the service retrieves images about *Angela Merkel*. Each image is clickable, and when a click is performed, the service presents the Archived Web Page from which the images were found.

1.1 The Problem

There are huge amounts of visual content on the Web. One part of this visual content is Not Suitable for Work (NSFW) for most users, because it contains offensive or explicit images (naked persons, violence and pornography, for instance). The exposure to these types of content is particularly critical for children and young persons.

The Image Search service retrieves images based on the filename, alternative text and the surrounding text of an image presented on a Web page. Therefore, due to the nature of the Web, there are no guarantees that the images retrieved to answer a search query will not contain some offensive content, even with an apparently not problematic query. For instance, a website that got hacked for Web spam can exhibit this behaviour.

An example of this problem using the Image Search service is shown in Figure 1.2, in which an apparently not offensive query term *angela* was fulfilled on the system. The retrieved results have content that can be considered offensive to the

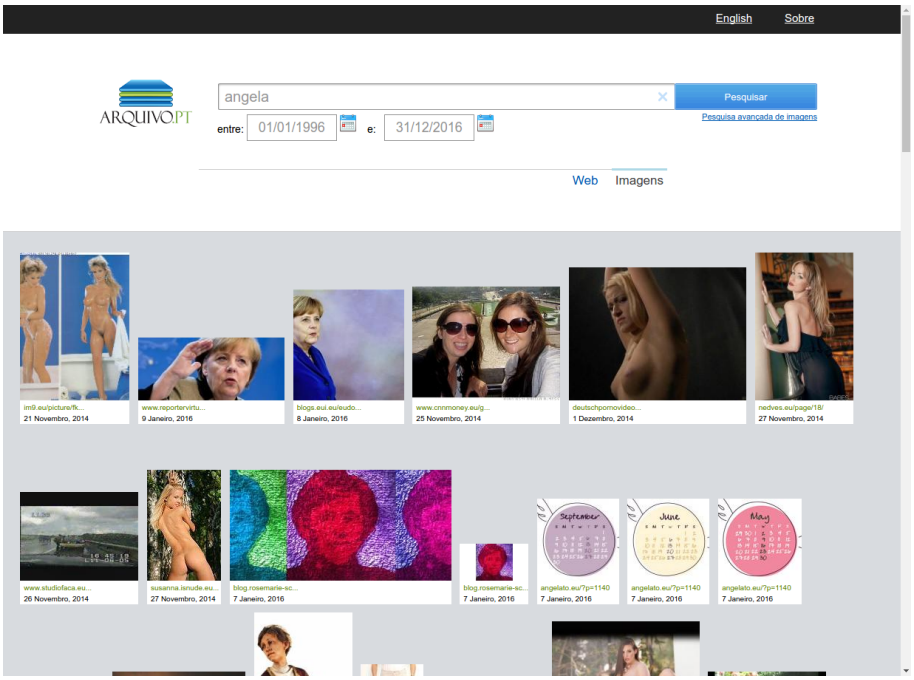


Figure 1.2: Example of Arquivo.pt Image Search problematic content.

users.

There are several types of NSFW content. The type that Arquivo.pt is trying to identify and filter out is pornography.

Detection of NSFW content on the archived Web pages from Internet is challenging due to the scale (billions of images) and the diversity (small to very large images, graphic, color images, among others) of image content.

1.2 Proposed Solution Outline

The proposed solution automatically identifies the image content, classifying it as NSFW or Safe For Work (SFW). This classification can then be used by the Image Search service to filter out the content, for instance providing an option of 'Safe Search' or 'Not Safe Search' to the users of Arquivo.pt. This is similar to how Google and other Search Engines work.

Providing image content analysis capabilities to Arquivo.pt would also enable the extraction of other information from the images, providing better tools and retrieve results based on this extra information.

The application must be able to perform classification of the image contents in a reasonably time frame. The number of contents available at Arquivo.pt, for each collection, can be very large. As of the time of this writing, one broad crawl to the *.pt* top-level domain can collect a total amount of 18 Terabytes of compressed data.

The proposed system is modular to be used for other use cases by switching the underlying model, for instance, get other type of information from the images. A Web Service interface for real time classification is also implemented. The workflow of the proposed solution is depicted in Figure 1.3, and it represents the proposed solution.

Figure 1.3: Workflow of the classification system.

Arquivo.pt Web Archive file format (WARC) files have diferent types of content. From these WARC files, the images to classify need to be extracted, then they are analysed in order to get a feature vector. This vector will be used by a classifier to discriminate if a image is NSFW or SFW, and finally the results will be stored.

1.3 Thesis Outline

This thesis is composed by the chapters descrived as follows. Chapter 1 presents the motivation behind this work, the solution outline as well as this thesis structure.

Chapter 2 describes related work and previous research on similar Image Recognition problems.

Chapter 3 presents an overview about the Arquivo.pt infrastructure and its main data characteristics. This chapter provides the context to understand how Arquivo.pt gets its data and what type of data needs to be processed. It also gives an infrastructure description to understand how the integrated solution can be performed.

Chapter 4 explains neural network models, that are used at the evalutions and improvement experiments.

Chapter 5 describes how the evaluation dataset was built. This dataset is the basis for the classification models evaluations and improvements. The results obtained with each model, such as classification accuracy are reported. Also, an evaluation about classification speed (images/second) of each model was measured using different math libraries as the backend and different types of hardware that were available at Arquivo.pt.

The integration of the classification system at Arquivo.pt is reported in Chapter 6. Chapter 7 provides the main conclusions on this work as well as some future work directions.

2

Related Work

The problem of automatically identify NSFW content is not new. Along the years, there has been a significant amount of research to provide, and to improve methods that can accurately identify these contents. Section 2.1 presents the first methods of image classification based on skin detection. More robust methods are presented in Section 2.2 using Bag of Visual Words (BoVW). Finaly, Section 2.3 presents the usage of neural networks to handle this type of tasks.

2.1 Skin Detection Methods

In 1999, an automatic system to detect if human nudes are present in an image is proposed [13]. It uses methods to mark skin-like pixels combined with color and textures properties. These marked regions are then analyzed by a specialized grouper which attemps to group a human figure using geometric constraints on the human structure. Based on that, the system decides if a human is present or not.

The first methods [51] that tried to solve this problem were based on skin-detection algorithms in order to identify regions of interest, then they analyzed the features of these skin regions to decide whether they were pornographic or not. An example of these methods is the POESIA filter [32], an open source implementation of a skin-color-based filter. The performance of those methods relies on the accuracy of the skin detection algorithm and the extracted features, which are usually

hand made. They present high false positive rates in images such as beach and sports.

2.2 Bag of Visual Words Methods

Other techniques that showed adequate image classification results was through BoVW [10]. These techniques extract, from an image, a set of visual features represented as words, similar to the *bag-of-words* for document classification, building a vocabulary vector with the number of occurrences of these visual words representing local image features. Those local images features usually are derived from detecting keypoints or local descriptors usually Scale-Invariant Feature Transform (SIFT) [27] variations. A classifier that uses these representations is then trained to classify the image content as pornographic or not.

2.3 Neural Networks Methods

Recently, Deep Learning (DL) has showed state of the art results in almost all tasks of image recognition, specifically, Convolutional Neural Networks (CNN) have been widely used on image recognition tasks [25]. New CNN architectures have continuously been published with improved accuracy of the standard ImageNet [42] classification challenge, as presented bellow (top-5 error):

- ALexNet (2012) [25] - 16.4%;
- ZF Net (2013) [50] - 11.7%;
- GoogLeNet (2014) [41] - 6.7%;
- Residual Networks (2015) [17] - 3.6%;

The use of DL combines both feature extraction and classification, so there is less involvement of a designer in terms of selecting the features or the classifier.

The downside of these techniques is the amount of training data and the infrastructure that they require. Nowadays, improved training datasets such as ImageNet are available. Pre-trained models have been published openly to be used by people without the need to train a Neural Network from scratch. An example

of this kind of initiatives is Yahoo! release of an open source model to identify NSFW images, specifically, pornographic images [29].

OpenNSFW is a Deep Neural Network (DNN) solution released by Yahoo! for classification of NSFW images. The model is publicly available to be used for classification by developers, but the image dataset that Yahoo! applied for training is not available, due to the nature of its contents.

First, they have trained a Residual Network architecture model [17] with ImageNet 1000 classes dataset. Then, they replaced the last neural network layer, a 1000 node Fully-Connected (FC) layer, with a 2 node FC layer and fine-tuned the model to the specific task of NSFW classification, using their NSFW images dataset. This model can be reused and fine-tuned for each specific use case and dataset. The model is published using Caffe [23], an open source DL framework developed by Berkeley AI Research[1].

Inspired by OpenNSFW, other models are being made publicly available to solve this problem but focusing in architectures that can keep a good accuracy using less hardware resources. An example is NSFWSqueezeNet [34] that uses a SqueezeNet [18] architecture that is designed to use a small number of parameters and memory, so it can be run on more common hardware.

[1] http://bair.berkeley.edu/

3

Arquivo.pt Infrastructure

This chapter contains an overall description about Arquivo.pt infrastructure, its data and how it is obtained, in order to provide the system environment context, and data provenance context.

In Section 3.1, the Arquivo.pt system components are presented. The following subsections explain each of these systems. Section 3.1.1 describes the system responsible for gathering Arquivo.pt contents. Section 3.1.2 presents the system that provides the access to these contents, trying to keep the *look and feel* of the old Web pages. Section 3.1.3 presents the Arquivo.pt full text search system. Section 3.1.4 addresses the Image Search System where the image classification solution is integrated. Finally, Section 3.1.5 describes the Arquivo.pt indexing system. Section 3.2 describes the type of data that constitutes Arquivo.pt.

3.1 System Components Description

Figure 3.1 illustrates the logical organization of Arquivo.pt infrastructure, composed by five key system components: Crawler System, Replay System, Text Search System, Image Search System and Indexing System, each one with their own functionality and responsability. The boxes represent a software component and the cylinders the data, needed by the components (inputs) or produced by them (outputs). The arrows represent the input/output relation between the components and this data. For instance, in the Crawler System, both Heritrix [33]

and Brozzler [26] software components produce as output the ARChive file format (ARC)/WARC data, which is used by the Replay System or by the Indexing System as input data to generate other data such as Capture inDeXes (CDX) [2] and Lucene[1] data sources.

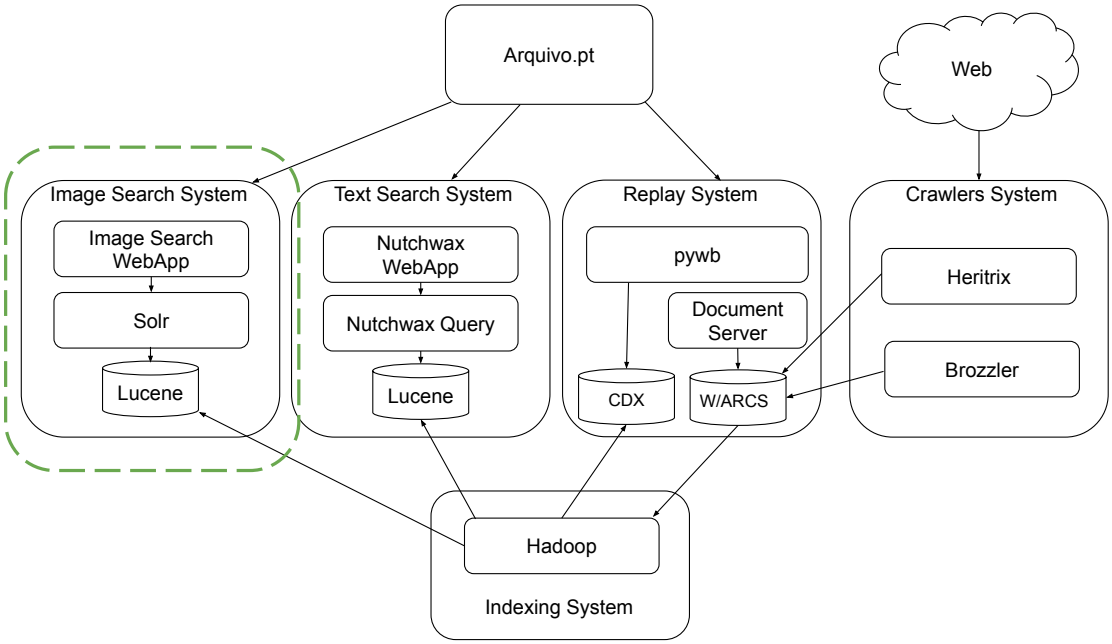


Figure 3.1: Arquivo.pt Infrastructure.

3.1.1 Crawler System

The crawler system job is to harvest the Web, collecting and storing its contents. It uses crawlers such as Heritrix [33] to perform this task. The crawlers are configured with a list of starting Uniform Resource Locator (URL) (seeds) and with a specific scope and budget. An example of configuration is for instance to crawl only URLs from *.pt* top-level domain and to download up to 10 000 URL per host, for performance reasons. It then starts to crawl those seeds, discovering more URL to crawl and preserve them. Other crawler used by the system is the Brozzler [26], that uses Chromium[2] instances to render the Web pages instead of only fetching the resource's representation. These crawlers can discover and to preserve more content with higher quality with the downside that they demand more hardware resources, and their processing is more demanding. All contents

[1] http://lucene.apache.org/
[2] https://www.chromium.org/

that are fetched by the crawlers are stored in ARC and WARC [12] files to be used by other systems.

Arquivo.pt uses four different Web crawlers scopes to gather information to be preserved:

- Broad Domain - top-level domain *.pt* crawls and suggested websites.
- Daily - daily crawls of Portuguese news websites.
- Special - thematic crawls, for instance Portuguese elections and top-level *.eu* domain crawl.
- High Quality - high quality crawls to selected websites.

3.1.2 Replay System

The Replay System is responsible to reproduce the preserved contents page back to the user trying to provide a similar experience and navigability as the original Web page. The software used for this kind of task is commonly named Wayback Machine. There are several implementations of Wayback Machines, for instance OpenWayback [19] written in Java. Arquivo.pt uses for this the PyWB [24] a Wayback Machine written in Python. These systems use special indexes named CDX [2] that map the URL and their position within the WARC files.

3.1.3 Text Search System

The Text Search System is responsible to provide a way to find information at Arquivo.pt. Traditionally, Web Archives only allow to search for a Web page through the URL. Arquivo.pt provides more search capabilities allowing users to search through query terms. The system will display preserved Web pages such that their contents are related with the submitted terms. This system uses a modified NutchWAX [11] implementation and the Lucene indexes generated by the Indexing System.

3.1.4 Image Search System

The Image Search System is the new component that is being developed and it provides the components to enable image search at Arquivo.pt. It is composed

by an Image Search WebApp that answers the image search requests that come from Arquivo.pt frontend and uses Solr[3] to answer the query. The indexes to the system are created at the Indexing System similary from the Text Search System but with a different Hadoop job. The classification system will be integrated at this job.

3.1.5 Indexing System

The Indexing System is responsible for all the index work so that the preserved contents can be searched and reproduced. It is composed by a Hadoop[4] Cluster that processes the Terabytes of ARC/WARC files (stored by the Crawler System) and produces Lucene indexes to be used by the full-text Search System and CDX indexes to be used by the Replay System.

3.2 Arquivo.pt Data Characterization

As of 8 October 2019, Arquivo.pt has a total amount of 6 966 589 866 preserved Web files, gathered in 3 967 593 ARC and WARC compressed files fulfilling 368 Terabytes of disk space storage.

The ARC/WARC file formats specify a method for combining multiple digital resources into an aggregate archival file together with metadata information. Each WARC file is a concatenation of one or more WARC records. Each WARC record consists of a header with metadata information, followed by a content block with the corresponding resource, such as images, text documents, or any type of resource found on the Web.

Each crawl has different configurations. For instance, the broad domain crawlers don't download files from the Web with more than 10 MB, but the special crawls and high-quality crawls don't have those restrictions. Moreover, they are configured to accept all mime-types found on the Web. For this reason, the type of data that can be found at Arquivo.pt is very widespread and heterogeneous.

Table 3.1 reports the top 10 mime-types[5] found during a 2017 *.pt* top-level domain crawl. The presented mime-types are reported by the Web servers using the HyperText Transfer Protocol (HTTP) meta information, when each resource

[3] http://lucene.apache.org/solr/
[4] https://hadoop.apache.org/
[5] https://www.iana.org/assignments/media-types/media-types.xhtml

is collected. In some rare cases, the reported mime-type is wrong regarding the real resource type provided by the Web server.

Table 3.1: Number of resources mime-types collected on Arquivo.pt last broad domain crawl.

% Amount	Number of URL	mime-types
79.60	237 966 251	text/html
10.03	29 997 689	image/jpeg
2.45	7 328 179	image/png
0.77	2 305 834	application/pdf
0.76	2 271 770	application/javascript
0.72	2 145 481	text/xml
0.62	1 869 902	application/rss+xml
0.62	1 861 165	application/json
0.60	1 820 236	image/gif
0.58	1 783 630	text/css
3.25	1 783 630	all others

Table 3.2 presents the top 7 mime-types regarding image contents that were found during the last broad domain crawl. The most common image type is the *jpeg* with 76% of the total images followed by *png* with 18%. Although only 4% of the images are *gif*, this type can present an extra challenge to classify because the images may have an animation.

Table 3.2: Mime-types distribution for image type contents.

% Amount	Number of URL	mime-types
75.59	30 145 538	image/jpeg
18.37	7 328 179	image/png
4.56	1 820 236	image/gif
0.98	389 839	image/svg+xml
0.34	135 720	image/x-icon
0.09	38 577	image/pjpeg
0.06	22 717	image/bmp

Since the origin of the images is the Web and come from multiple different websites, these images can have many sizes, different resolutions and any type that is allowed on the Web.

4

Neural Networks

An Artificial Neural Network (ANN) is a computing system that was inspired by a biological neural network, found in every animal and humans. Instead of the conventional programming approaches, where we tell the computer system how to solve a problem, these systems try to solve a problem without a human telling how to solve it. These systems learn to perform tasks and to solve problems by considering examples, generally without being programmed with any task-specific rules.

The big challenge about these systems is how to train them to solve a problem. The number of iterations and observations that the system needs can be computationally expensive, in order for the system to reach an acceptable solution.

These Neural Networks are composed by simple structures of a function that maps an input value to an output value. These structures can be aggregated and pipelined to form a neural network. Networks with a significant amount of neurons and layers are Deep Neural Network (DNN).

DNN are the latest development research on neural networks. Essentially, a DNN is an ANN with more intermediate layers. It's a new take on learning representations from data that puts an emphasis on learning successive layers of increasingly meaningful representations. It often involves tens or even hundreds of successive layers of representations, all learned automatically from exposure to training data.

These networks are characterized by the number of parameters, layers (depth)

and neurons they have. There are several types, but more specifically for image recognition problems there has been a huge success with the Convolutional Neural Networks (CNN) [15].

4.1 Input Data

For image classification, the provided input data to the network is given by the image pixels. Depending on the architecture and its goal, each neural network accepts a fixed size input data. A common input data size value for a neural network that classifies color images is $3 \times 256 \times 256$ array, in which each array unit corresponds to a pixel value from a specific RGB channel. Each array value can be seen as a *feature*, from a *feature vector* with 196 608 dimensions total. It is based on these *features* that the neural network will decide which class the images belong to, after passing them forward to the network.

4.2 Models

A neural network is a Directed Acyclic Graph (DAG) of layers. The simplest example of a neural network is a linear stack of layers that maps a single input to a single output.

Figure 4.1 presents the typical blocks that a Neural Network has. Input X is the data and it will be forward through all the layers of representations. At the end of the network, a prediction score will be inferred, and based on this score, a loss function is applied to measure how far the prediction is from the true labels. With this measure, the weights of each layer are updated in order to minimize the loss score.

There is a broad range of network topologies. For instance, two-branch networks, multihead networks, inceptions blocks, among others. In this work, we made experiments with two different topologies, Resnet [17] and SqueezeNet networks [18], described in section 4.2.1.

These topology networks define a *hypothesis space*. This space constrains the space of possibilities to solve a specific problem. It is within this constrained space that we try to find a useful representation of the input data, that will be mapped for the desired output.

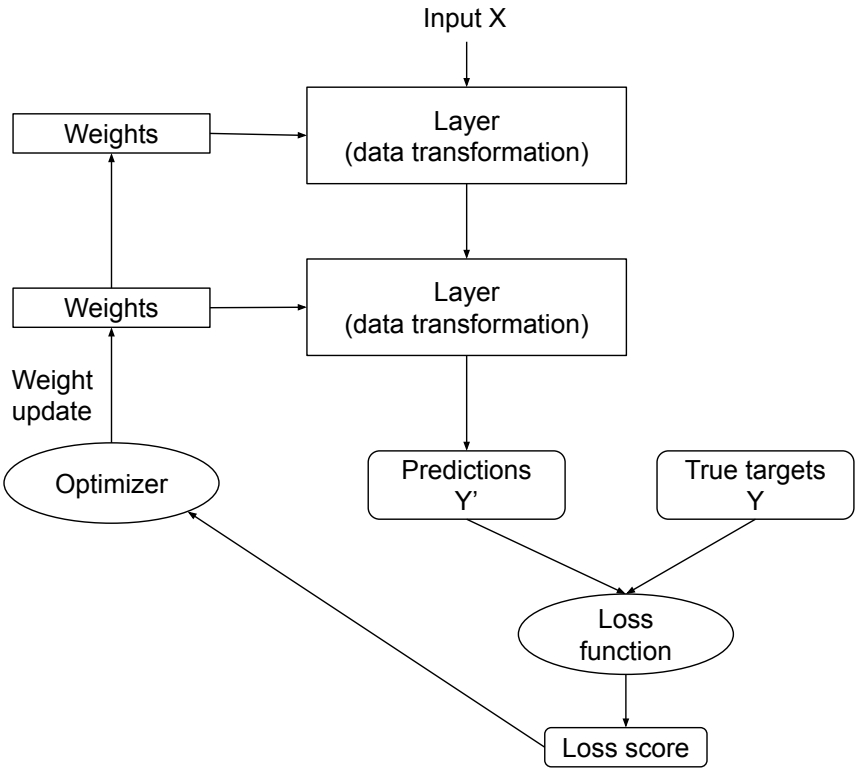


Figure 4.1: Learning diagram of a neural network.

4.2.1 Convolutional Neural Networks

4.2.1.1 ResNet Neural Networks

Deep networks are hard to train because of the vanishing and exploding gradient problem [14]. With the gradient value being backpropagated to the first layers, repeated multiplications may make the gradient infinitely small. As a consequence, the network learning tends to halt, specially in the first layers. The same happened in the reverse way, in which the weights get saturated and the network learning also halts. Several approaches were used to address these problems, such as, normalized initialization [14], rectifiers [16] and normalization layers [21].

With the increasing depth of these networks, some network nodes get saturated and the accuracy degrades. Such degradation is not caused by overfitting. The ResNet network model addresses this degradation problem by introducing shortcut connections between the layers, as depicted in Figure 4.2.

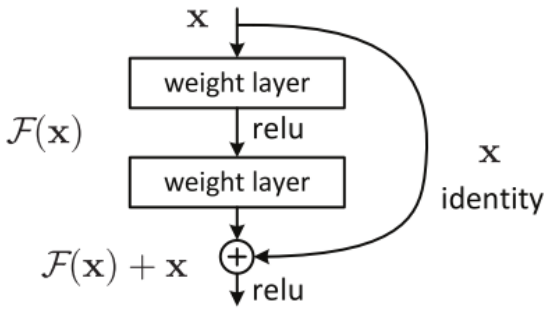


Figure 4.2: Identity shortcut connection.

The authors [17] had the premise that building deeper networks by adding more layers should not degrade the network training performance. Stacking identity mappings upon the current network would make the network perform exactly the same. So, this deeper model should not produce a training error higher than its shallower models. Their hypothesis is that letting the stacked layers fit a residual mapping is easier than letting them directly fit the desired underlying mapping. Their paper shows that these networks are easier to optimize, and that a higher accuracy is reached due to the increased depth of the networks.

4.2.1.2 SqueezeNet Neural Networks

SqueezeNet [18] network model was designed to be a model with a smaller architecture, while preserving the accuracy of bigger models. Having a smaller architecture means that the neural network has fewer parameters and layers and it has multiple advantages:

- More efficient distributed training.
- Less overhead exporting new models.
- Be able to run on Field-Programmable Gate Array (FPGA) and embedded circuits.

The authors of the network model accomplish these goals, by using the following strategies:

- Strategy 1 - Replace 3x3 filters with 1x1 filters.
- Strategy 2 - Decrease the number of input channels to 3x3 filters.

- Strategy 3 - Downsample late in the network so that convolution layers have large activation maps.

The goal of these strategies is to reduce the number of parameters (weights) of the neural network. With the above strategies in mind, the authors organized the convolution filters in a *Fire Module*.

The Fire Module illustrated in Figure 4.3 is composed by a *squeeze* convolution layer which has only 1x1 filters, squeezing the incoming data. It also has an *expand* convolution layer that has a mix of 1x1 and 3x3 convolution filters, expanding the depth of the data again.

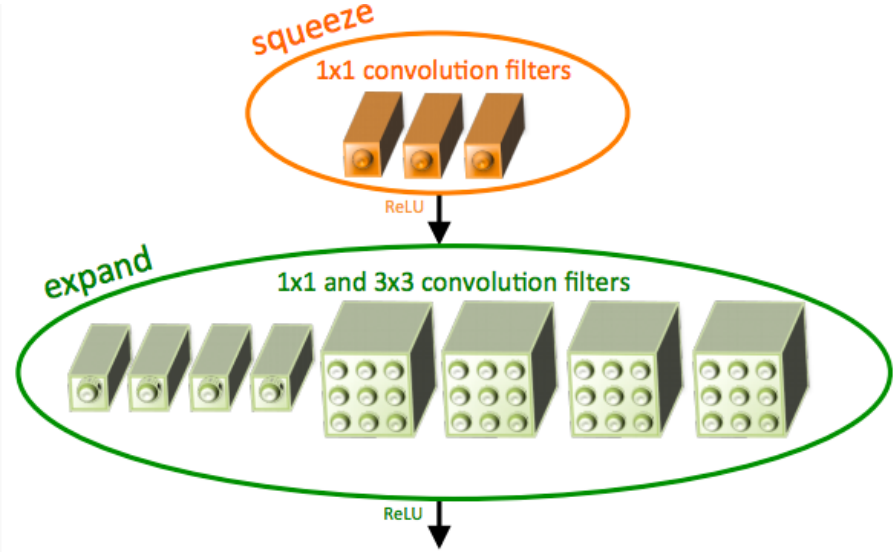


Figure 4.3: SqueezeNet Fire Module.

4.3 Loss Function

To be able to find an useful representation that models and solves the problem at hand, we need to somehow have feedback on how well the representation is performing. The loss function, also known as objective function gives us that feedback. A loss function tells us "how good" our model is at making prediction for a given set of parameters. The choice of the right loss function is very important, because the networks take any shortcut to minimize the loss. Thus, if this loss function is not well suited for the problem we are trying to solve, the network can end doing things we didn't pretend.

There are several loss functions that can be applied to different problems:

- *Cross-Entropy* - binary classification problems.
- *Categorical Cross-Entropy* - many-class classification problems.
- *Mean Square Error* - regression problems.
- *Connectionist Temporal Classification* - sequence-learning problems.

Since the problem addressed in this work is a binary classification task, we used a Cross-Entropy loss function [22].

4.3.1 Cross-Entropy

Cross-Entropy is a quantity from the field of Information Theory [8, 40] that measures the distance between probability distributions. In this case, between the ground-truth distribution and our models predictions distribution, giving us a metric to verify how far is our model from correctly classifying the input data.

Surprisal is the degree to which one is surprised to see some result. We will be more surprised with an outcome with lower probability in comparison with an outcome with high probability. If y_i is the probability of the outcome i, then its surprisal s (or self-information) can be represented as:

$$s = \log\left(\frac{1}{y_i}\right). \tag{4.1}$$

Since the surprisal of the individual outcomes is known, we can estimate the surprisal for the overall event. This is defined as the Entropy (e) [8, 43], and it is calculated with a weighted average of the surprisal of each outcome (n is the number of outcomes):

$$e = \sum_{i=1}^{n} y_i \log\left(\frac{1}{y_i}\right). \tag{4.2}$$

If the actual outcome of an event is p_i, but the estimated probability is q_i, then the event will occur with a p_i probability but the surprisal measure will be determined by q_i since it was the estimated probability. This is the Cross-Entropy and it is defined by:

$$c = \sum_{i=1}^{n} p_i \log\left(\frac{1}{q_i}\right). \tag{4.3}$$

Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value (between 0 and 1). Cross-entropy loss increases as the predicted probability diverges from the actual label. A perfect model would have a log loss of 0.

From another perspective, minimizing cross entropy is equivalent to maximizing the log likelihood of our data, which is a direct measure of the predictive power of our model.

4.4 Optimizer

The Optimizer is the mechanism that will update the network weights based on what the networks are predicting and its loss function. Progressively, the incremental updates the optimizer produces will lower the loss score, making the network prediction more accurate.

The Optimizer falls into two types of algorithms. The first order optimization algorithms in which the loss function gradient is used to minimize the network loss. They are known as Gradient Descent algorithms. The second order algorithms, in which the second order derivative is used to minimize the network loss. With second order algorithms its possible to know if the first derivative is increasing or decreasing and with that the function curvature. The second order derivative is much more costly to compute, therefore the first order optimizers are more used. For this work, a Stochastic Gradient Descent (SGD) algorithm is used [37].

4.4.1 Gradient Descent

Gradient descent is an optimization algorithm to minimize a loss function by interactively updating the models parameters in the opposite direction of the loss function gradient. Thus, moving in the direction of the steepest descent as defined by the negative of the gradient (4.4).

The method intuition is the following: it starts at a given point x_0 and computes the gradient at the point $\nabla f(x_0)$.

Then, a step of length η is taken on the direction of the negative gradient. A new point is found $x_1 = x_0 - \eta \nabla f(x_0)$ and the same procedure is performed until it reaches a minimum. This minimum can be local or global and its found by testing if the norm of the gradient is zero: $\|\nabla f(x)\| = 0$.

There are several practical concerns even with this basic algorithm to ensure both that the algorithm converges (reaching a minimum) and that it does so in a fast way, by an adequate choice of the following parameters:

- Learning Rate (η) - The size of these steps is called the learning rate. With high learning rate it can cover more ground each step. However, we risk overshooting the lowest point since the slope of the hill is constantly changing. With a very low learning rate, we can confidently move in the direction of the negative gradient since we are recalculating it so frequently. A low learning rate is more precise, but calculating the gradient is time-consuming, so it will take us a very long time to reach the solution.
- Stopping Criteria - Normally, it is not possible to reach a full convergence either because it will be too slow, or because of numerical issues such as precision effects, due to a finite representation. The commonly used criteria are: a maximum number of iterations, the gradient norm be smaller than a given threshold or the normalized difference in the function value be smaller than a given threshold.

The type of solver used was the SGD. It updates the network weights (W) using a linear combination of the negative gradient $\nabla L(W_t)$ and the previous weight update V_t,

$$V_{t+1} = \mu V_t - \eta \nabla L(W_t), \tag{4.4}$$

$$W_{t+1} = W_t + V_{t+1}. \tag{4.5}$$

The *learning rate* (η) represents the weight of the negative gradient, and the *momentum* (μ) the weight of the previous update.

5

Models Evaluation and Improvement

Two neural network models were chosen to be evaluated for the NSFW classification task: the OpenNSFW and the NSFWSqueezeNet.

Section 5.1 describes how a ground-truth dataset was build, to perform this evaluation. This dataset is needed so we can properly test the models accuracy at solving this kind of task, identify their main limitations, and consequently try to improve them. Section 5.2 presents the hardware used to perform these evaluations. The model evaluation is reported in Section 5.3. Section 5.4 addresses the identification of the wrongly classified image content. Section 5.5 presents techniques that were used to increase the ground-truth dataset and finally Section 5.6 describes the experiments performed to improve the models.

5.1 Building the Initial Evaluation Dataset

The proposed dataset is composed by 17 655 images manually labeled from Arquivo.pt with 8 273 labeled as NSFW and 9 382 as SFW (Table 5.1).

Table 5.1: Evaluation Dataset.

	SFW	NSFW	Total
Labeled Dataset	9 382	8 273	17 655
Non-Labeled Dataset	-	-	18 626

These images were acquired from Arquivo.pt using two methods. The first method to acquire the images was throught the existent Beta Images Indexes. These are Lucene indexes provided by the Solr platform[1]. Each index document is an image resource with its corresponding metadata, for instance the image width and height, URL of the origin site and text extracted from images tags properties. The Solr platform exposes a REST API that was used to execute queries to return images to be manually labeled.

The second method was throught Arquivo.pt Text Search API [3], querying the API to retrieve Web pages, and from those Web pages the images contained were extracted to be manually labelled.

On Arquivo.pt the total of images that belong to the NSFW class is much less than the images from the SFW class. The main dificulty at this task is to find enough images from the NSFW class in order to build a dataset with a significant number of images and with both classes balanced in terms of the number of images.

Arquivo.pt Text Search API was queried to retrieve Web pages that contain the terms *porn* and *blowjob* or *fuck*. Those terms are usually associated with this kind of contents [39]. It was used another keyword beside *porn* to narrow down the retrieved results. Web pages that contain these two keywords are more likely to have pornographic content, increasing the amount of relevant images retrieved.

With the list of the archived Web pages URL returned, each archived Web page was scrapped by their HTML img tags and the available archived images downloaded. A total of 18 000 images were retrieved this way.

With the Beta Images Indexes the query term *porn* was used and the images manually labeled. Then it was queried from the indexes images from hosts that were identified as being related with this kind of content. The hosts were associated to NSFW contents throught the images that were labeled in the previous query.

A significant number of noisy images were being returned, such as image banners and icons. To reduce the amount of this type of images, only images with resolution above 150x150 pixels were accepted firstly. Later, 1040 samples of SFW images and 451 samples of NSFW images with spatial resolutions below 150x150 were labeled in order to be able to test the models with this kind of images.

[1] http://lucene.apache.org/solr/

5.2 Evaluation Hardware Specifications

The available hardware to evaluate these models is a laptop class Lenovo Y50 with 8 GB RAM, a GeForce GTX 860M as GPU and a Intel(R) Core(TM) i7-4710HQ CPU @ 2.50GHz.

The models were also tested using server class hardware available at Arquivo.pt infrastructure. The server is a Dell PowerEdge R730xd model with 256 GB RAM and an Intel(R) Xeon(R) CPU E5-2620 v3 @ 2.40GHz.

5.3 Models Evaluation

One of the evaluation metrics to compare the models was the Area Under the receiver operating characteristic Curve (AUC) established by the Receiver Operating characteristic Curve (ROC). This is a common evaluation metric used on binary classifiers, and it evaluates the classifier varying a cut-off threshold [28]. Using the AUC gives us a broader sense about the model classification performance. It is useful to evaluate the models before a threshold value were not chosen as cut-off. Also, the common classification metrics such as accuracy were used.

Another metric used as evaluation is the amount of images a model can classify per second with a specific hardware. The use of this metric shows us the cost of the model in terms of computation resources. With limited resources, a model with better speed classifying images can be used as a tradeoff for choosing a model with less accuracy, for instance when the model needs to be used online.

The models were evaluated using the dataset described in Section 5.1. For each model, all the images were preprocessed and passed through the network and the AUC was plotted.

5.3.1 Image Preprocessing

Before each image is passed throught the network, a preprocessing phase must be performed. This preprocessing depends on several factors depending on the library used to read the images and the underlying model and how it was trained.

The models are using as input a $b \times 3 \times 256 \times 256$ vector, where $b >= 1$ is the batch size (the number of images that will be classified in a batch) and the other

indexes correspond to the number of image channels, image width and image height (196 608 dimensions total per image). Depending on the library used to read the images, the input vector must be arranged to match the neural network models input. On this evaluation test, the python Pillow library [45] for image processing was used. Caffe natively expects the color channels ordered as BGR instead of RGB, so a transposition is needed to switch the channels.

Also, the mean of the data where the models were pre-trained, is subtracted, and the values scaled from the [0,1] range to the [0,255] range. At last, a bilinear image resizing to the 256 $\times$ 256 spatial resolution is performed. This image preprocessing is exactly the same that it was used on the training phase of the pre-trained models networks, ensuring the best results.

5.3.2 Initial Evaluation Results

Figures 5.1 and 5.2 show the ROC curve for each model. OpenNSFW has the highest AUC with 0.98 and the NSFWSqueezeNet with 0.85.

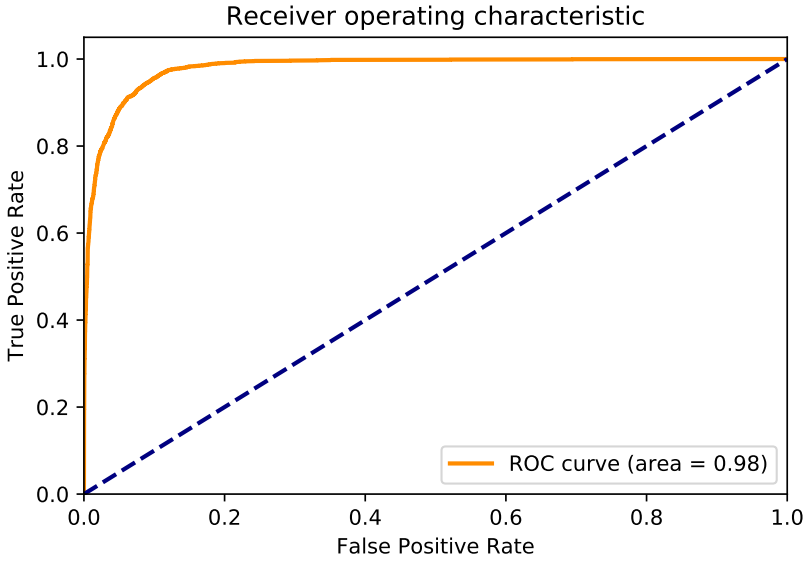


Figure 5.1: ROC Curve evaluation of OpenNSFW.

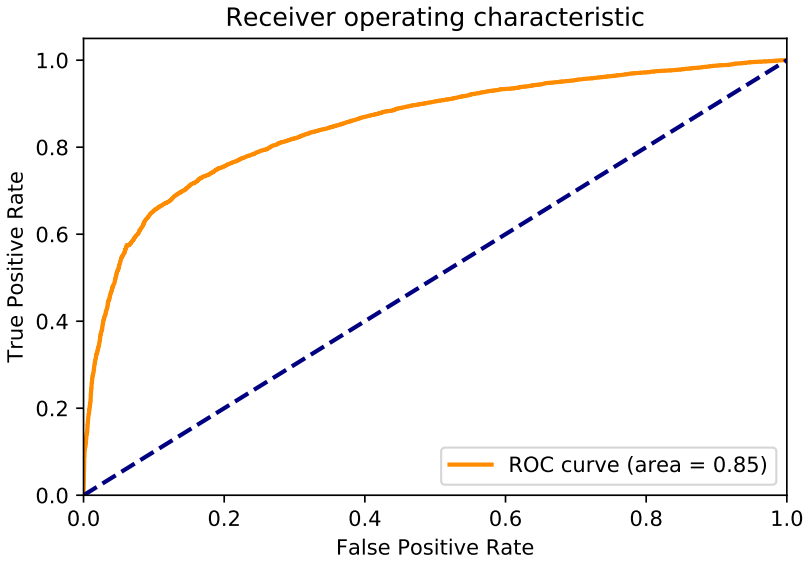


Figure 5.2: ROC Curve evaluation of NSFWSqueezeNet.

Other evaluation metrics were calculated such as *accuracy*, *precision*, *recall*, and *f-score*, defined as

$$accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}}, \tag{5.1}$$

$$precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{5.2}$$

$$recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{5.3}$$

$$f\text{-}score = 2 \times \frac{precision * recall}{precision + recall}. \tag{5.4}$$

True Positives (TP) is the number of NSFW images correctly classified as NSFW and True Negatives (TN) the number of SFW images correctly classified as SFW. False Positives (FP) is the number of SFW that were classified as NSFW and False Negatives (FN) is the number of NSFW images classified as SFW.

Tables 5.2 and 5.3 present the results for the OpenNSFW model. The confusion matrix shows that the classifying error is balanced between both labels (SFW and NSFW), with more error on the FP (652) comparing to the FN (567). This trade-off is good here, since classifying a NSFW image as SFW is worse than the reverse.

Table 5.2: Evaluation Metrics for the OpenNSFW model.

	Precision	Recall	F1-Score	Accuracy
NSFW	0.92	0.93	0.93	-
SFW	0.94	0.93	0.93	-
Average / Total	0.93	0.93	0.93	0.93

Table 5.3: Confusion Matrix of the OpenNSFW model.

	NSFW'	SFW'
NSFW	7706	567
SFW	652	8730

Tables 5.4 and 5.5 report the corresponding results for the NSFWSqueezeNet. On this model most of the errors being made are FN, which is a serious error. As expected, the OpenNSFW has the best scores with 0.93 accuracy in comparison to the 0.72 from NSFWSqueezeNet. Looking at their corresponding confusion matrix we can see that the OpenNSFW model error predictions are balanced among both NSFW and SFW images, but NSFWSqueezeNet is doing lots of errors classifying NSFW images as SFW.

The models were also tested with different hardware described in Section 5.2, as well as using different math subroutines backends and Caffe distributions.

There are several Basic Linear Algebra Subprograms (BLAS) libraries that can be used as a backend to Caffe [23]. These libraries optimize linear algebra routines such as vector additions, multiplication, matrix multiplication and fast Fourier transforms.

Choosing a library that improves these operations to a specific hardware can give a huge performance boost to the classification speed. The libraries used to evaluate the performance of classification were OpenBLAS [47] and MKL [20].

It was also evaluated using a Intel Caffe distribution, a fork from Berkeley Vision and Learning Center (BVLC)[2], Caffe dedicated to improve the framework

[2] https://github.com/BVLC

Table 5.4: Evaluation Metrics for the NSFWSqueezeNet model.

	Precision	Recall	F1-Score	Accuracy
NSFW	0.92	0.45	0.60	-
SFW	0.66	0.96	0.79	-
Average / Total	0.78	0.72	0.70	0.72

Table 5.5: Confusion Matrix of the NSFWSqueezeNet model.

	NSFW'	SFW'
NSFW	3697	4576
SFW	333	9049

Table 5.6: Classification speed performance (img/sec) with different hardware and backends setups.

	OpenNSFW	NSFWSqueezeNet
BVLC Caffe i7 + OpenBLAS	7	17 img/sec
BVLC Caffe GTX860M CUDA	40	77 img/sec
Intel Caffe CPU Xeon + MKL	9	28 img/sec

performance when running on CPU, particulary, Intel Xeon processors with architectures from or more recent than Haswell[3] and Intel Xeon Phi processores[4].

Table 5.6 reports the classification speed obtained with different setups. The NSFWSqueezeNet model has the best classification speed, being able to classify 77 images per second in comparison with the OpenNSFW model, with 40 images per second, both using the GPU GTX860M setup.

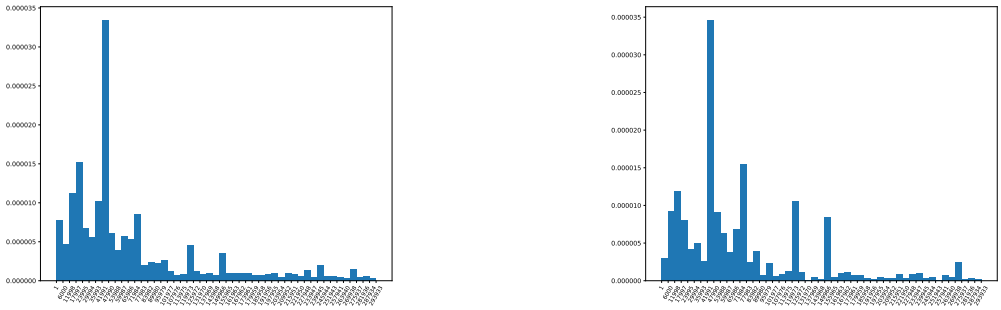
5.4 Analyzing False Positives and False Negatives

In order to understand what type of images the OpenNSFW model is classifying wrongly, so we could try to improve it, the FP and FN were analysed. There could be some common patterns on the type of images that it tends to wrongly classify. For instance, could the image's spatial resolution be a main source of classification error? To answer this, Figure 5.3 plots the normalized histogram representing the images resolution distribution and the images resolution normalized histogram of the wrongly classified images. The wrongly classified images resolution distribution matches the images dataset resolution distribution, so there is no evidence that the image resolution is the cause of these errors.

Some patterns were identified manually inspecting the wrongly classified images, as false positives the most typical error are images with women with a significant amount of nudity, but not explicit. At false negatives samples common

[3] https://ark.intel.com/content/www/us/en/ark/products/codename/42174/haswell.html

[4] https://www.intel.com/content/www/us/en/products/processors/xeon-phi/xeon-phi-processors.html



(a) All images resolution distribution.

(b) Misclassified images resolution distribution.

Figure 5.3: Images resolution distribution comparison between the dataset and the misclassified images.

occurrences are sexual acts with animals, and sexual explicit animations. This is expected, since the model was trained to filter pornography.

5.5 Data Augmentation

In the area of Computer Vision (CV) and Artificial Intelligence (AI) often, there is not enough data to train algorithms to properly handle tasks such as classification. Usually models trained on small datasets do not generalize well enough. To overcame this problem, techniques of data augmentation can be applied to increase the available dataset and consequently improve the algorithms accuracy on those tasks. It is known that applying simple data augmentation techniques such as cropping, rotating and flipping input images reduces the overfitting and class imbalance problems typically found in small datasets [46]. More complex techniques such as elastic distortions are also used for data augmentation [38]. More recently, Generative Adversarial Networks (GAN) have been proposed to generate images of different styles [9, 35].

Augmentator [6], a image data augmentation library for machine learning was used to generate new images by applying transformations such as mirror, shearing and flipping. Figure 5.4 presents an example of this type of transformations.

(a) Original image. (b) Augmented image.

Figure 5.4: Image augmentation technique.

5.6 Improving the Models

Transfer Learning is a technique where a neural network is first trained on a base dataset and task, different from the main task to be solved, and then previously learned features are transferred to a second neural network to be trained on a new dataset and task [49]. When the dataset available for the new task is much smaller than the base dataset, this technique can be very efficient to train a large neural network without overfitting [49]. The authors of the NSFWSqueezeNet and OpenNSFW trained these networks using this technique, using a pre-trained model on the ImageNet dataset and then applied to this specific task.

These models can continually be improved through fine-tuning, a process were some parameters are adjusted to better perform on the given task. There are several methods to fine tune a neural network model, for instance, training the network with our labeled data but performing only small adjusts to the last layer, or adjusting more layers, for instance, the last 3 layers. All the network layers can also be retrained, using the network weights from a pre-trained model. These weights usually are very good initial weights to start training a network instead of using other weights initialization techniques [31].

Experiments to evaluate if a model can improve its accuracy and loss had been made. Both models were retrained using a 4-fold methodology on the dataset built and the following heuristics were applied:

- Retraining each model for 1 epoch and augmenting the dataset.
- Retraining each model for 5 epochs and augmenting the dataset.

One epoch means one pass of the full training set. It contains several iterations where each iteration is a training image being passed forward through the network.

5.6.1 NSFWSqueezeNet Improving

To improve the NSFWSqueezeNet model, all network model layers were updated using the SGD solver with the following training parameters:

- Learning rate policy - poly.
- Power - 1.0.
- Momentum (α) - 0.9.
- Base learning rate (η) - 0.001.
- Weight decay - 0.0000001.

During the network model training, the learning rate parameter was updated following a polynomial decay:

$$\eta_{iter+1} \leftarrow \eta_{iter} * \left(1 - \frac{iter}{max\_iter}\right)^{power}. \tag{5.5}$$

Table 5.7: NSFWSqueezeNet fine-tuning accuracy and loss.

Model	4-Fold Accuracy	4-Fold Loss
NSFWSqueezeNet 1 Epoch Augmented 10K	0.88 ± 0.002	0.28 ± 0.004
NSFWSqueezeNet 5 Epoch Augmented 10K	0.89 ± 0.002	0.27 ± 0.006

There was a significant accuracy improvement, from the initial model accuracy of 72% to 89% accuracy. The training network configuration, solver configuration and training logs are available online[5].

[5] https://github.com/arquivo/SafeImage/tree/master/training/SqueezeNet

Table 5.8: OpenNSFW fine-tuning accuracy and loss.

Model	4-Fold Accuracy	4-Fold Loss
OpenNsfw 1 Epoch Augmented 10K	0.92 ± 0.003	0.20 ± 0.006
OpenNsfw 5 Epoch Augmented 10K	0.94 ± 0.004	0.16 ± 0.007

5.6.2 OpenNSFW Improving

This model is more computationally expensive to train. With the limited hardware available and time constraints, an attempt to improve it was made, freezing all the network layers and retraining just the last fully-connected network layer and the softmax layer.

The solver used was also a SGD with the same learning parameters as the model above.

- Learning rate policy - poly.
- Power - 1.0.
- Momentum (α) - 0.9.
- Base learning rate (η) - 0.001.
- Weight decay - 0.0000001.

There was a very small model's accuracy improvement (from 93% on Table 5.8, not significant because the accuracy number is being rounded to 2 decimal places.

The training network configuration, solver configuration and training logs are available online[6].

[6] https://github.com/arquivo/SafeImage/tree/master/training/Resnet

6

Solution Integration

The developed solution to classify image content as NSFW or SFW needs to be integrated on the Arquivo.pt Image Search indexing workflow. The Image Search indexing workflow extracts from the WARC files the images and the related metadata. The NSFW classification score will be an extra metadata field to be added to this information.

The Image Search indexing workflow is composed by one Hadoop job that processes each WARC file and splits each one into smaller units named WARC Records. Each WARC Record contains a Web resource representation which is processed by a Hadoop map. The Hadoop maps extracts the text and images available at those WARC Records and builds an index record with the image and its derived metadata.

There are two possible solutions to integrate the image classification step at this workflow.

The first one would be a complete new phase at the Image Search indexing workflow. At the first phase of the Image Search indexing logic, all the images and related information would be extracted from the WARC Records and kept in an intermediate data structure for classification. Then, a new Hadoop job would be launched to classify those images and store the classification results in a persistent dictionary where the key would be the image URL, and the value would be the classification score.

The final indexing phase would be to use the classification scores information with the metadata extracted from each record and produce the final Solr index.

This solution has the problem that the classification step needs to be done in Hadoop nodes with specific resources such as Graphics Processing Unit (GPU) for this type of workload. The Hadoop version 0.14 available at Arquivo.pt does not have any possibilities to declare what kind of resources a Hadoop job needs. This kind of feature is only available with Hadoop 3.0.0 through the YARN [44] resource manager. There are some alternatives, for instance CaffeOnSpark [48] developed by Yahoo! in order to handle this type of problem. But to implement it, the indexing logic would need to be rewritten to use Spark instead of Hadoop, and the lack of documentation and the complexity to use this custom solution could be a problem.

The second solution would be to use a message queue to classify images and provision worker nodes, with the required hardware resources, such as GPU to fetch from this queue the images to be classified. The advantages of this solution is that it completely decouples the dependency between the indexing system and the classification system. With this decouplement other DL backends, like TensorFlow [1] can be used, changing only the type of worker being used.

This solution can also be scaled horizontally by adding more workers, and eventually, more message queues with a load balancing solution. The adopted solution was to use a message queue. It was chosen because of its decoupling properties and the current available resources at Arquivo.pt. It was used a Redis message queue [36] instance as the message broker to handle all the classification requests [4]. There was not any particular reason for picking Redis, just for the previous usage experience with it, any other message queue such RabbitMQ[1] or ActiveMQ[2] could handle the task.

[1] https://github.com/rabbitmq
[2] https://github.com/apache/activemq

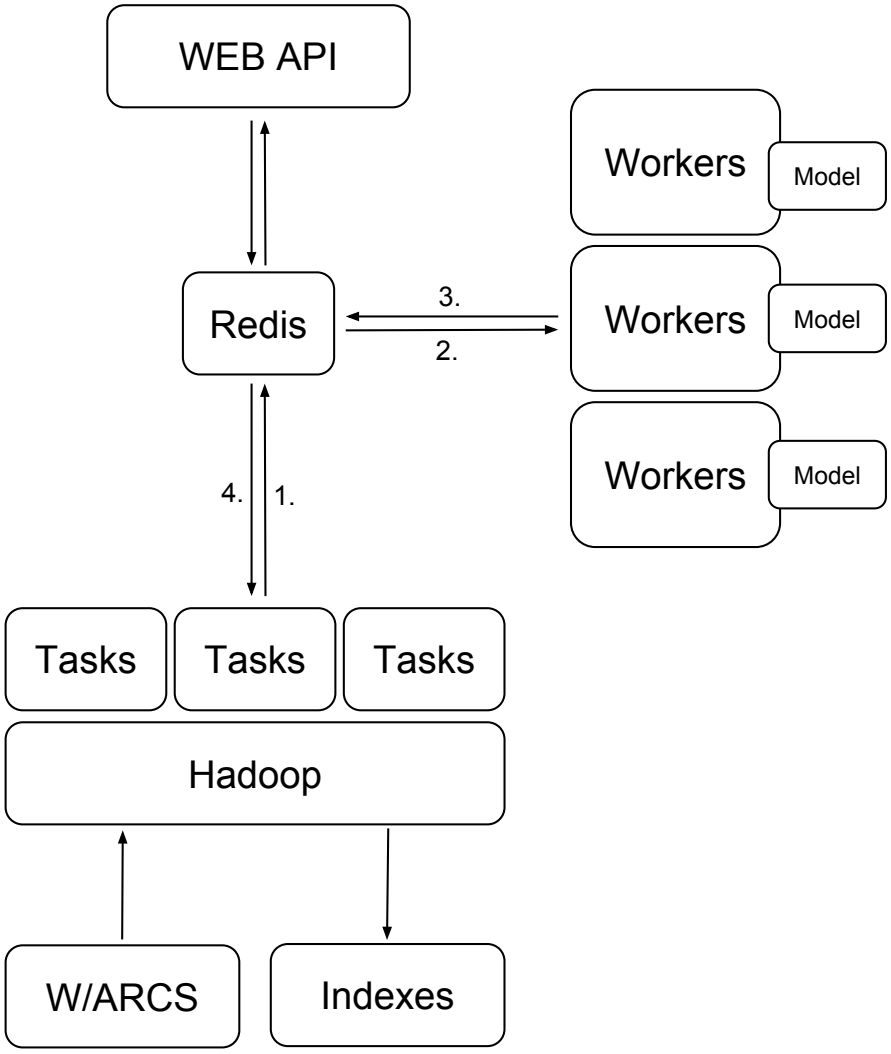


Figure 6.1: Integration of the classification system with the Image Search Indexing phase.

On the proposed solution, multiple workers are polling from Redis images queued to be classified [5].

Figure 6.1 presents a diagram of this solution. The Hadoop cluster processes the WARC data and generates the Solr indexes:

1. The Hadoop tasks process the WARC and the extracted images are queued to the Redis Message Queue.
2. The Workers then fetch the images from the queue and perform the classification.
3. The classification result is reported back to the message queue.

4. Finally, the Hadoop tasks fetch the classification result and proceeds with the indexing workflow, generating the Solr indexes.

The Web API also handles the classification requests, posting the images to be classified to the Redis Message Queue.

6.1 Classifying Images with Multiple Frames

Although not being the mime-type with higher number of URLs at Arquivo.pt data, Graphics Interchange Format (GIF) images pose an extra problem to the classification system because some of them can be composed by multiple frames. Classifying only the first frame is not enough with this type of images. Any of the subsequent frames within the GIF image can have some NSFW content. To solve this problem before classifying those images with a final score, each individual frame needs to be classified, and then the final classification score assigned to that image is the largest one observed among all frames, as depicted in Figure 6.2.

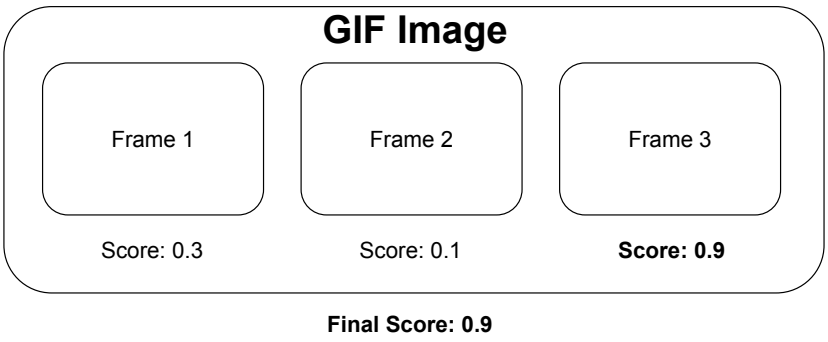


Figure 6.2: Classification of a GIF image.

6.2 Evaluation

The image classification task is performed offline, therefore the extra workload that the system needs to perform is very low. For each indexed documented representing an image, a field with name *safeimage* was added, with values ranging between 0 and 1. When performing a query, Solr filters out NSFW images based on the value of this field that needs to be superior to 0.5. For 6 468 single term

queries requested to Solr with the filtering of NSFW images, the average query time was 9 ms. Without the filtering, it was 1 ms.

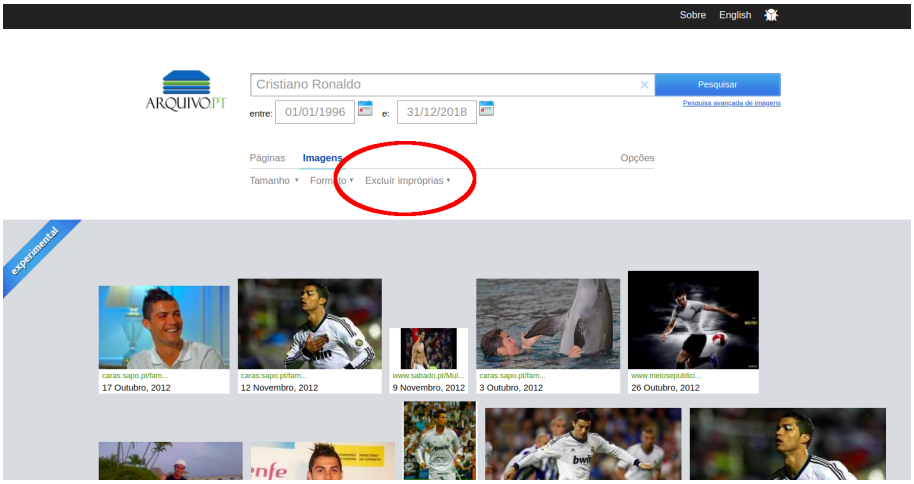


Figure 6.3: Image filtering interface integration.

Figure 6.3 presents the image filtering integration on the user interface. The users can opt for excluding this kind of image contents or not.

7

Conclusions

Arquivo.pt provides access to more than 6 770 423 012 preserved resources. In order to users find the information they want among all Arquivo.pt contents, information retrievel tools are provided to find pages with a specific text and an Image Search system is under development to expand its searching capabilities.

There are huge amounts of visual content on the Web, and therefore also in Arquivo.pt. The Image Search system provides an easy access to this visual content. But some of this visual content can be offensive for users (for instance naked persons, violence, pornography). And also due to webspam and how the system works, this type of content can be shown to users even when they do not intentionally look for it.

To mitigate this problem, a solution that automatically identifies images with this type of content was developed and integrated into Arquivo.pt Image Search system. The solution uses a CNN to identify this content type and provides the identification result as a signal to the Image Search system, which uses it to hide this content from the results.

Two neural networks pre-trained to solve this kind of task were evaluated against a dataset built with Arquivo.pt information. Then, a fine tuning process was performed to improve the model's accuracy, identifying this type of content.

The initial evaluation of the models reported 93% accuracy for the Resnet model and 72% for the SqueezeNet model. After fine tuning, the model's accuracy,

was improved by 1% on the Resnet model (94% accuracy) and by 17% on the SqueezeNet model (89% accuracy).

The best model was integrated in the Image Search system indexing workflow, using a Redis Message Queue as a broker to be able to distribute and scale the classification work among several workers. Also, it is currently in production at Arquivo.pt (https://arquivo.pt/image.jsp).

7.1 Future Work

As future work, the models accuracy can be improved building a larger dataset and using more recent Deep Neural Network models, while at the same time increasing the number of training epochs. Also, instead of only using 2 classes NSFW or SFW, categorizing with more classes could help users to better define their needs. An example is having the choice of Strict, Moderate or Off, like the search engine DuckDuckGo[1] does.

Also, they can be expanded to identify others types of NSFW content, for instance, violence, offensive symbols, among others.

[1] https://duckduckgo.com/

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Gregory S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian J. Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Józefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Gordon Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul A. Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda B. Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *CoRR*, abs/1603.04467, 2016. URL http://arxiv.org/abs/1603.04467. (p. 38)

[2] Internet Archive. Internet Archive: CDX File Format Reference @ONLINE, January 2018. URL https://archive.org/web/researcher/cdx_file_format.php. (pp. 12 e 13)

[3] Arquivo.pt. Arquivo.pt API v.0.2 (beta version), March 2018. URL https://github.com/arquivo/pwa-technologies/wiki/Arquivo.pt-API-v.0.2-(beta-version). (p. 26)

[4] Daniel Bicho. Image Search Project Prototype, March 2018. URL https://github.com/arquivo/ImageSearch/tree/nsfw. (p. 38)

[5] Daniel Bicho. SafeImage Project, March 2018. URL https://github.com/arquivo/SafeImage. (p. 39)

[6] Marcus D. Bloice, Christof Stocker, and Andreas Holzinger. Augmentor: An image augmentation library for machine learning. *CoRR*, abs/1708.04680, 2017. URL http://arxiv.org/abs/1708.04680. (p. 32)

[7] Miguel Costa. *Information Search in Web Archives*. PhD thesis, Faculty of Sciences of the University of Lisbon, December 2014. URL http://xldb.fc.ul.pt/xldb/publications/Costa:InformationSearchIn:2014_document.pdf. (p. 2)

[8] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Elements of Information Theory. Wiley, 2006. ISBN 9780471748816. URL https://books.google.pt/books?id=EuhBluW31hsC. (p. 22)

[9] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A. Bharath. Generative Adversarial Networks: An Overview, 2018. ISSN 10535888. (p. 32)

[10] T. Deselaers, L. Pimenidis, and H. Ney. Bag-of-visual-words models for adult image classification and filtering. In *2008 19th International Conference on Pattern Recognition*, pages 1–4, Dec 2008. doi: 10.1109/ICPR.2008.4761366. (p. 8)

[11] FCCN. Arquivo.pt NutchWAX, January 2018. URL https://github.com/arquivo/pwa-technologies. (p. 13)

[12] International Organization for Standardization. ISO 28500:2017 information and documentation – WARC file format, January 2018. URL https://www.iso.org/standard/44717.html. (p. 13)

[13] D.A. Forsyth and M.M. Fleck. Automatic detection of human nudes. *International Journal of Computer Vision*, 32(1):63–77, Aug 1999. ISSN 1573-1405. doi: 10.1023/A:1008145029462. URL https://doi.org/10.1023/A:1008145029462. (p. 7)

[14] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*, 2010. (p. 19)

[15] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, and Gang Wang. Recent advances in convolutional neural networks. *CoRR*, abs/1512.07108, 2015. URL http://arxiv.org/abs/1512.07108. (p. 18)

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.

CoRR, abs/1502.01852, 2015. URL http://arxiv.org/abs/1502.01852. (p. 19)

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.90. URL http://ieeexplore.ieee.org/document/7780459/. (pp. 8, 9, 18 e 20)

[18] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. SqueezeNet. *arXiv*, 2016. ISSN 0302-9743. doi: 10.1007/978-3-319-24553-9. (pp. 9, 18 e 20)

[19] IIPC. Openwayback - Java Wayback Machine @ONLINE, January 2018. URL https://github.com/iipc/openwayback. (p. 13)

[20] Intel. Intel® Math Kernel Library (Intel® MKL) | Intel® Software, March 2018. URL https://software.intel.com/mkl. (p. 30)

[21] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL http://arxiv.org/abs/1502.03167. (p. 19)

[22] Katarzyna Janocha and Wojciech Marian Czarnecki. On loss functions for deep neural networks in classification. *CoRR*, abs/1702.05659, 2017. URL http://arxiv.org/abs/1702.05659. (p. 22)

[23] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. (pp. 9 e 30)

[24] Ilya Kreymer. PyWB - Python Wayback Machine @ONLINE, January 2018. URL https://github.com/ikreymer/pywb. (p. 13)

[25] Alex Krizhevsky, IIya Sulskever, and Geoffret E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information and Processing Systems (NIPS)*, pages 1–9, 2012. (p. 8)

[26] Noah Levitt. Brozzler - Distributed browser-based web crawler @ONLINE, January 2018. URL https://github.com/internetarchive/brozzler. (p. 12)

[27] T. Lindeberg. Scale Invariant Feature Transform. *Scholarpedia*, 7(5):10491, 2012. doi: 10.4249/scholarpedia.10491. revision #153939. (p. 8)

[28] Charles X. Ling, Jin Huang, and Harry Zhang. AUC: A better measure than accuracy in comparing learning algorithms. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 2671, pages 329–341, 2003. ISBN 3540403000. doi: 10.1007/3-540-44886-1_25. (p. 27)

[29] Jay Mahadeokar. Open NSFW model code @ONLINE, January 2018. URL https://github.com/yahoo/open_nsfw. (p. 9)

[30] Julien Masańs. *Web archiving*. Springer, 2006. ISBN 3540233385. doi: 10.1007/978-3-540-46332-0. (p. 1)

[31] Sarfaraz Masood, M. N. Doja, and Pravin Chandra. Analysis of weight initialization techniques for Gradient Descent algorithm. In *12th IEEE International Conference Electronics, Energy, Environment, Communication, Computer, Control: (E3-C3), INDICON 2015*, 2016. ISBN 9781467373999. doi: 10.1109/INDICON.2015.7443734. (p. 33)

[32] Daoudi Mohamed. POESIA - Filtering Software @ONLINE, January 2018. URL http://web.archive.org/web/20051030090955/http://www.poesia-filter.org:80/. (p. 7)

[33] Gordon Mohr, Michele Kimpton, Micheal Stack, and Igor Ranitovic. Introduction to Heritrix, an archival quality web crawler. In *4th International Web Archiving Workshop (IWAW04)*, Bath, UK, 2004. URL http://www.iwaw.net/04/proceedings.php?f=Mohr. (pp. 1, 11 e 12)

[34] Jesse Nicholson. NSFW squeezenet, January 2018. URL https://github.com/TechnikEmpire/NsfwSqueezenet. (p. 9)

[35] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *CoRR*, abs/1712.04621, 2017. URL http://arxiv.org/abs/1712.04621. (p. 32)

[36] Redislabs. Redis, May 2018. URL https://redis.io/. (p. 38)

[37] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016. URL http://arxiv.org/abs/1609.04747. (p. 23)

[38] Patrice Simard, Dave Steinkraus, and John C Platt. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis. *Proceedings of the 7th International Conference on Document Analysis and Recognition*, pages 958–963, 2003. ISSN 15205363. doi: 10.1109/ICDAR.2003.1227801. (p. 32)

[39] slate.com. Words banned from Bing and Google's autocomplete algorithms., March 2018. URL http://www.slate.com/articles/technology/future_tense/2013/08/words_banned_from_bing_and_google_s_autocomplete_algorithms.html. (p. 26)

[40] James V. Stone. Information theory: A tutorial introduction. *CoRR*, abs/1802.05968, 2018. URL http://arxiv.org/abs/1802.05968. (p. 22)

[41] 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. URL http://arxiv.org/abs/1409.4842. (p. 8)

[42] Stanford University. Imagenet @ONLINE, January 2018. URL http://www.image-net.org/. (p. 8)

[43] Sriram Vajapeyam. Understanding Shannon's entropy metric for information. *CoRR*, abs/1405.2061, 2014. URL http://arxiv.org/abs/1405.2061. (p. 22)

[44] Vinod Kumar Vavilapalli, Arun C. Murthy, Chris Douglas, Sharad Agarwal, Mahadev Konar, Robert Evans, Thomas Graves, Jason Lowe, Hitesh Shah, Siddharth Seth, Bikas Saha, Carlo Curino, Owen O'Malley, Sanjay Radia, Benjamin Reed, and Eric Baldeschwieler. Apache Hadoop YARN: Yet another resource negotiator. In *Proceedings of the 4th Annual Symposium on Cloud Computing*, SOCC '13, pages 1–16, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2428-1. doi: 10.1145/2523616.2523633. URL http://doi.acm.org/10.1145/2523616.2523633. (p. 38)

[45] wiredfool, Alex Clark, Hugo, Andrew Murray, Alexander Karpinsky, Christoph Gohlke, Brian Crowell, David Schmidt, Alastair Houghton, Steve Johnson, Sandro Mani, Josh Ware, David Caro, Steve Kossouho, Eric W. Brown, Antony Lee, Mikhail Korobov, Michał Górny, Esteban Santana Santana, Nicolas Pieuchot, Oliver Tonnhofer, Michael Brown, Benoit Pierre, Joaquín Cuenca Abela, Lars Jørgen Solberg, Felipe Reyes, Alexey Buzanov,

Yifu Yu, eliempje, and Fredrik Tolf. Pillow: 3.1.0, January 2016. URL https://doi.org/10.5281/zenodo.44297. (p. 28)

[46] Sebastien C. Wong, Adam Gatt, Victor Stamatescu, and Mark D. McDonnell. Understanding Data Augmentation for Classification: When to Warp? In *2016 International Conference on Digital Image Computing: Techniques and Applications, DICTA 2016*, 2016. ISBN 9781509028962. doi: 10.1109/DICTA.2016.7797091. (p. 32)

[47] Zhang Xianyi. OpenBLAS : An optimized BLAS library, March 2018. URL http://www.openblas.net/. (p. 30)

[48] Yahoo! yahoo/CaffeOnSpark, May 2018. URL https://github.com/yahoo/CaffeOnSpark. (p. 38)

[49] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *CoRR*, abs/1411.1792, 2014. URL http://arxiv.org/abs/1411.1792. (p. 33)

[50] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 818–833, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10590-1. (p. 8)

[51] Huicheng Zheng, Mohamed Daoudi, and Bruno Jedynak. Blocking Adult Images Based on Statistical Skin Detection. *Electronic Letters on Computer Vision and Image Analysis*, 4(2):1–14, 2004. ISSN 1577-5097. doi: 10.5565/rev/elcvia.78. (p. 7)