NTNU
Norwegian University of
Science and Technology

Multi-label classification of a real-world image dataset

Nikita Uvarov

Applied Computer Science
Submission date: June 2017
Supervisor: Rune Hjelsvold, IDI
Co-supervisor: Faouzi Alaya Cheikh, IDI

Norwegian University of Science and Technology
Department of Computer Science

Acknowledgment

I would like to thank both of my supervisors, Prof. Rune Hjelsvold, and Assoc. Prof. Faouzi Alaya Cheikh for valuable guidance throughout this project. I would also like to acknowledge Norwegian News Agency for providing with the image collection used in this study and NVIDIA Corporation for their donation of hardware employed in the research. Finally, I want to thank my family and girlfriend for a big support and encouragement during this project.

Nikita Uvarov

Abstract

Deep neural networks have shown increasing performance in image classification recent years. However, most of the methods developed in research are created, trained and compared on standard, general image collections. While some of such datasets contain realistic pictures with multiple objects and interaction between them, the selection of pictures and categories is broad and designed for the training of automatic classification systems. Therefore, questions remain on how modern image classification approaches can be applied to real-world datasets. What challenges can arise in the implementation of systems based on such datasets and how can they be solved? What level of classification performance can be expected? This study looks into these questions and gives insights on building such classification systems for real-world image collections.

The Norwegian News Agency provided the author with a unique labeled dataset of one million images for the purpose of this research. The study consists of a trial and main experiment. The goal of the trial experiments was to learn necessary tools, test classification system implementation, training and testing pipelines as well as different architectural design choices. These set of experiments were performed on a subset of the original image collection with a limited number of categories. Insights from these experiments were further used to perform model training on a much broader set of manually selected categories. Two network architectures were employed and compared in the research: CaffeNet and GoogleNet.

Results from the study suggest a big potential of using pre-trained convolutional neural networks in solving the task of multi-label image classification on a real-world dataset. However, the study also highlights a number of challenges connected with developing such system in a realistic environment, including:

- A unique set of categories which requires either to train a new model or to reuse a pre-trained neural network with adjustments in the architecture to solve classification problems.
- Difficulties in category tree transformation connected with the decision of which categories include, exclude and merge.
- Categories which are unsuitable for automatic training purposes including contextual, abstract and ambiguous labels.
- Contextual images which can not be classified only by visual features and require additional information.

- Issues connected with the classification of objects and people located in the background of the pictures.
- Duplicates of images in the dataset which can cause issues for both training and testing processes.

The research gives further insights on the challenges mentioned, discusses their impact and possible solutions to them. The main limitation of the study is that only one dataset was employed in the research. Therefore, results are considered likely to be more generalizable to datasets similar to the one used in the study.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Topic covered by the project

The topic of this project is within the field of computer science, more specifically it relates to issues of automatic multi-label image classification. The study will focus on applying a transfer learning approach, which leverages existing pre-trained deep neural networks. In particular, the mentioned method will be employed to investigate possible challenges and the potential of applying deep convolutional neural networks in image classification for real-world datasets.

1.2 Keywords

Deep Learning; Neural Networks; Image Annotation; Image Classification; Multi-label Classification; Transfer Learning

1.3 Problem description

During past years significant steps forward has been made when it comes to image classification. In 2012 the system which utilized deep learning approach outperformed all other methods in ImageNet Large Scale Visual Recognition Challenge (ILSVRC) and achieved top 5 test error rate of 15.4% in single-label image classification task [3, 4, 5]. In four years the winner of ILSVRC 2016 improved this results even further with top-5 error rate of 2.9% using a new design of convolutional neural network (CNN) [6]. The progress in solving multi-label classification is also moving forward improving classification precision each year [1, 7]. All these methods and architectures are designed and tested on standard image collections like ImageNet [4] or PASCAL VOC [8], which were specifically created to compare proposed approaches and solutions in different papers [1, 2, 9, 10]. However, most of the datasets created and used by companies and universities are likely to differ from such standard datasets in ways including types of categories, balance in image distribution between categories, and an amount of systematic errors. Therefore, the questions remain on how these approaches can be applied to real-world datasets. What challenges can arise in the implementation of systems based on such datasets and how to solve them? What level of classification performance can be expected? This study will try to answer these questions and give insights on building such classification system for real-world image collections.

1.4 Justification, motivation and benefits

As mentioned earlier, image classification methods have evolved very fast recent years. However, most researchers create and compare methods using standard, tested and more general datasets of images, while companies that wish to use these methods are likely to have usually more unique and less structured real-world datasets. It is important to test how state of the art methods perform on such image collections. On the one side, this project will be interesting for researchers to know if current methods are working with the more realistic datasets. On the other hand, companies can benefit from more research on how to implement such systems in real-world environments, get insights on challenges that they could face with and recommendations on how to solve them in order to create a system that satisfies their needs.

1.5 Research questions

- What are the main challenges in building and training a multi-label image classification system on a real-world datasets?
- What level of performance can a multi-label classification system achieve using a fine-tuning approach for training on a real-world image collection?

In addition to this questions, the study also looked into how different neural network configurations can influence the speed of the training process as well as the end system performance. Some ideas on how the dataset can be improved to increase system precision will also be discussed.

1.6 Contributions

This study will give insights on potential challenges that can arise in the process of building a multi-label classification system for real-world datasets and will discuss possible solutions for them. The case-study implementation of such system, based on an image collection provided by Norwegian News Agency (NTB), will be presented together with achieved classification performance for different network configurations.

The research will also discuss how both discovered challenges and choices of network architecture can impact final results for the system. In addition, the study will also point out on possible directions for further research and experiments that should be performed to both validate and expand obtained results.

1.7 Clarification of terms and acronyms

- DNN – Deep Neural Network
- CNN – Convolutional Neural Network

- NTB – Norwegian News Agency (Norsk Telegrambyrå)
- RNN – Recurrent Neural Network
- ILSVRC – ImageNet Large Scale Visual Recognition Challenge
- HCP – Hypotheses-CNN-Pooling
- Standard image dataset – refers to the image collection specifically created for the purpose of classification methods comparison, for example PASCAL VOC [8], NUS-WIDE [11], ImageNet [4].
- Real-world dataset – image collection with annotations that was not specifically created for training automated classification systems

1.8 Ethical and legal considerations

The main legal aspect of this study lays in the access and usage of the NTB image collection, which is covered in the signed contract agreement included in Appendix A. This contract gives the author rights to use image dataset in the research as well as to use example images approved by the company for this master thesis and presentation.

2 Background

Image classification refers to a task that requires from method to determine the categories to which picture belongs. Such challenges that utilize standard datasets like ImageNet and PASCAL VOC limit this problem to a question if objects of particular classes presented on the image [4, 8]. However, real-world settings might require extending this question to more general one [12]. For instance, in this study, the search extended beyond merely objects to include other image properties including color, weather, action, and sceneries.

2.1 Deep convolutional neural networks

A standard neural network consists of computational units called neurons. Neurons are connected to each other in particular way to form a network. Neurons in such networks are organized in groups or "layers" for computational efficiency reasons since it allows to apply vector operations [13]. Each input connection to a neuron has dedicated weight. Non-liner *activation function* use these input weights to determine when to activate particular neuron and send a signal further [5]. Activation function can be different, however recent years ReLU function is mostly used in modern DNN due to its computational efficiency and good output results [14, 3]. The "depth" of the neural network is identified as an amount of layers except input one [5].

The process of *training* the neural network is optimization of each neuron input weights in such way that the particular input to a network will produce desired output. This optimization is achieved by first introducing loss function which represents how "far" given output is from the desired one. After that, on each iteration of training, by calculating gradients of a loss function on given weights, the system determines how it should adjust weights to decrease the computed loss [15]. Modern DNNs use backpropagation approach of gradient calculation due to its efficiency [16].

Convolutional Neural Networks (CNN) have the same core ideas as any other neural networks except that CNN architectures assume that the inputs are images [17]. Unlike conventional neural networks, instead of processing images pixel-by-pixel CNN is searching for patterns in image patches of different sizes. Each convolution can also be perceived as a feature extractor, such as the one that detects particular edge angle or color. Such convolution functions are also organized in layers where output features from one layer can be an input data for next convo-

lutional filters. Therefore, an end system consists of filters cascade, where, similar to human vision, low-level features, like edges, are recognized on the fist levels, and higher levels trigger on more complex features like different shapes and patterns [18]

The main difference between machine learning approach and deep convolutional neural networks is that the former one has an additional step of supervised feature extraction. In contrast, in deep learning approach feature extraction happens automatically by the internal network itself. Therefore DNN can be considered as a "black box" system. For instance, in the case of image classification, such system will have raw picture pixels as an input and classified category numbers as output.

To improve the generalizability of results and to be able to compare networks with different hyperparameters the common approach is to use dataset splitting where an image collection is partitioned into three parts [19]. The first part, *called training* set, is employed directly for training, while the second part, called *validation set*, is used to compare different architectures and control overfitting. Overfitting is a state of the system when it optimizes to a noise in a training data instead of searching the underlying patterns [13]. Such optimization can potentially reduce the generalizability of the system and should be avoided [13]. The usual way to control overfitting of the model is to check system performance on a validation set periodically during the training process [20]. If the loss computed on a training set gets smaller with more iterations, but validation loss increases or stays the same, then it is an evidence that the model is starting to overfit on the training set [20]. To be able to monitor the generalizability of the method, validation set should not contain any images from the training set. The third part of the dataset, which is called *test set*, is used for a final evaluation of the method. The separation between validation and test sets is important for the similar reason as with separation between training and validation sets. When different network implementations are compared using the validation set, and the best one is selected, hyperparameters can overfit on the validation set similar to the state when network is overfitting on the training set [19]. Meaning that chosen parameters will be optimized for the particular images in the validation set. Therefore, it is important to test the final system on a separate set of pictures, which CNN has not seen at any time during the training process. According to Andrew Ng, The common ratio between these tree sets is 60%/20%/20%, where the training set gets 60% of images while validation and test sets receive 20% each [21].

2.2 Multi-label image classification

Opposite to a single-label image classification, in multi-label case picture can belong to more than one category. Several labels can both describe all objects presented on the picture and characterize the objects or picture itself.

There are two group of multi-label image classification methods:

- That process parts of the images and use CNN to find a single label for each of the part [1, 7, 22],
- That train CNN on the whole picture, similarly to single label methods, but use different loss functions that take several labels into account [9].

An example of the first type of network is described by Wei et al. in their paper "HCP: A Flexible CNN Framework for Multi-Label Image Classification" [1]. The core idea of the proposed method is to reuse already developed CNNs that were trained to classify single-labeled pictures in a multi-label scenario. The authors proposed deep CNN infrastructure called "Hypotheses-CNN-Pooling" (HCP) to achieve this. Its architecture is shown on Figure 1.

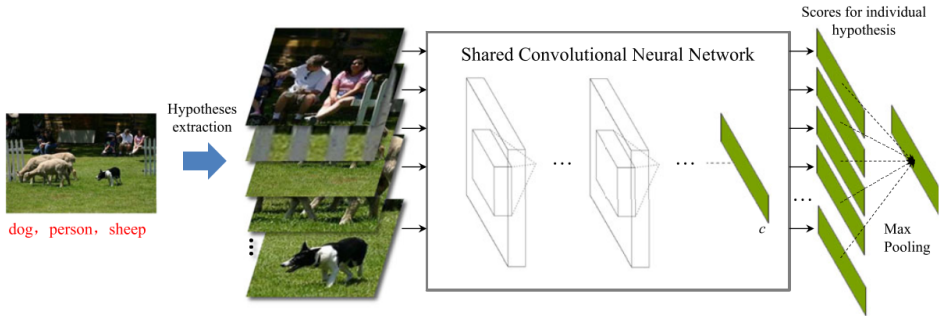


Figure 1: Hypotheses-CNN-Pooling infrastructure [1]

The whole classification pipeline proposed by the authors consists of the following steps:

1. Applying state of the art object detection technique (for example BING [23] or EdgeBoxes [24]) to get the set of hypothesis,
2. Using proposed hypothesis selection method reduce the number of initial hypotheses,
3. Use pre-trained CNN to assign single label for each hypothesis,
4. Use cross-hypothesis max-pooling to create the vector of labels found on the whole picture.

Authors claim of getting up to 90.9% of Mean Average Precision (which is defined in PASCAL Challenge [8]) on PASCAL VOC 2012 image dataset.

Yang et al. proposed similar infrastructure, but extended it with additional large-margin nearest neighbor (LMNN) CNN that incorporates knowledge of ground truth bounding boxes of the training set to improve results [22]. This extra layer tries to find similar objects in the test image to the ones extracted from the training set (ground-truth object set) independently from the other layer that employs more usual CNN. Results from the both layers are taken into account in the final prediction vector.

The method described by Girshik et al. in "Rich feature hierarchies for accurate object detection and semantic segmentation" paper also highly rely on the ground-truth bounding box during the training phase [25]. In particular, generated hypothesis are ranged based on the overlap with ground-truth box.

Oquab et al. proposed method that also use bounding box information to produce labels for training images [2]. Specifically, they propose to generate around 500 square patches from the picture and put a single label on each patch dependently on the overlap with the ground truth bounding box. Patches which intersect with two or more objects are filtered out, and the once which do not intersect with any objects are labeled as a background. Example training data generated from the image and ground-truth information is shown on Figure 2.

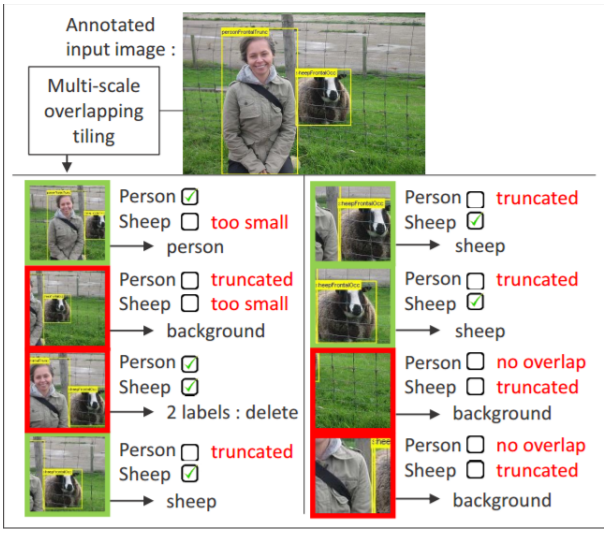


Figure 2: Training data generation example using patching method [2]

Currently, the first type of multi-label classification methods with image partitioning show improved performance compared to the ones that use the whole pictures [1, 7, 22]. However, such methods require having either ground truth bounding box encoded in metadata in addition to labels which identifies the loca-

tion of all objects on the image [26, 27], or pre-trained single-label classifier for the same set of categories [1]. In addition, this approach might be more suitable for searching for objects in the picture, but not perform as good to assign labels describing the whole image since it processes only part of a picture at a time. For instance, the dataset used in this study had label *alone* that separates images with single objects, or *landscapes* which characterize the whole picture. These kinds of categories might be challenging to classify using method that splits an image.

In contrast to previous papers, Gong et al. applied CNN to the whole picture without partitioning [9]. The main contribution of this study was not to produce new multi-label classification method, but to compare different loss functions in multi-label classification context. Results showed that weighted approximated-ranking loss (WARP) improved the classification performance of the categories with a small number of pictures which can be an essential factor for unbalanced datasets.

Wang et al. in their research were also trying to improve multi-label image classification performance for realistic image collections [12]. Authors point out on issues connected with methods that transforming multi-label task into a single-label one. Researchers identify the lack of modeling semantic dependencies between categories in such methods. The main reason for this is that such approaches treat image categories completely independently, while some of the labels might occur together more often than others. To solve this problem, the authors propose to utilize Recurrent Neural Network (RNN) in addition to CNN, which can model semantic relevance between images.

Most of the presented methods use so-called transfer learning technique in different ways to improve their results and to utilize the work from the other studies to their benefit. Brief description and motivation behind this approach presented in the next section.

2.3 Transfer learning

Usually, the usage of deep learning approach adds additional requirements: to have enough processing power and big enough image dataset. One of the most common methods to reduce these constraints called *transfer learning*. The main principal behind it is to reuse “knowledge” of a pre-trained neural network on one task and transfer this knowledge to another one [28, 2]. This idea appeared when people noticed that first layers of most CNNs learn to trigger on the same features like corners, edges, color conjunctions, therefore there is no need to discover them in each network [18].

There are two types of transfer learning [29]:

- With frozen layers, when layers copied from the pre-trained network do not

change their weights during the backpropagation phase, in other words, their learning rates are set to zero,
- Fine-tuning, when copied layers only start with pre-trained weights, but during the training phase adjust just their values like all other layers.

Oquab et al. describe in detail the usage of transfer learning for multi-label classification. In particular, they show how pre-trained network on one category set can be used to find labels for more general categories (for example source categories might include different dog breeds while target set may contain only general label "dog") or even entirely new ones [2]. Insights provided in this research will be applied in the project since target NTB dataset also has a different set of categories compared to the ones available in the standard image collections.

Recent studies also suggest that the usage of a pre-trained model can improve the generalizability of the final image classification system in contrast to the system trained on the target dataset from scratch [2, 29]. The transfer learning method employed in this project, which is described in the next chapter, allowed to reduce dataset size and performance requirements needed for successful system training.

2.4 Real-world datasets

Some datasets like PASCAL VOC or NUS WIDE are created from realistic pictures that contain several objects or people in different combination and interactions between them [8, 11]. Creators of such datasets, as well as other researchers, argue that real-world images are more likely to have multi-label nature by having both multiple entities in the picture as well as such category set that can describe image or object in several prospectives [8, 11, 12, 27]. Wang et al. poit out that in addition to objects, such images can contain labels that represent scenes, actions, and parts of some entities [12]. Authors also identify the challenge of small objects on such images that can be ignored by classification methods, especially when full pictures are processed in a neural network.

Except for the picture collection itself, dataset should also contain metadata for image and a defined set of categories. While datasets described earlier consist of realistic pictures, the set of categories and therefore images were designed and moderated for the purpose of further training and comparison of image classification systems. Therefore, there is still a question on how different state of the art methods can be applied to datasets that were created for other reasons and how their metadata utilization can be maximized. In addition to realistic pictures, such datasets can have additional challenges connected with a unique set of labels, a less balanced size of categories, a big number of both systematic and random errors.

2.4.1 Current state of the NTB dataset

As it was mentioned earlier, NTB provided their labeled image collection for the purpose of these project. Currently, NTB dataset contains around one million of manually annotated pictures and more than ten millions of images without labels. Therefore, the fist usage of the automatic image classification system for the company is to propagate annotations on other images from the collection. NTB has its unique set of categories organized in a tree structure. The company uses these tags to provide a solution for pictures search to their customers. Therefore the second application of the system would be to incorporate an image classification in this search solution. The dataset is specific to Norway and to particular topics including sports, finances, politics, and media. More in-depth analysis of the NTB dataset will be presented in Section 4.1.

3 Methodology

Methods used in the research will be described in this chapter. The trial and main sets of experiments were conducted in the study to answer the research questions. The purpose of the trial experiments was to get an understanding of classification system potential, to test its implementation, get a better understanding of time constraints when it comes to training and testing processes, to compare different solutions as well as to learn tools and framework used. Lessons learned during trial experiments on the limited set of image categories were further used to perform a full study with a more broad set of categories. A bigger scale of experiments was also possible due to donated GPU by the NVIDIA Corporation for this project (see section 3.3.5. The source code of all experiments and analysis conducted in the study is available online [30].

3.1 Metadata preparation

This section describes steps that were made to process NTB images metadata in order to store it in more suitable for this project format.

NTB uses proprietary text format called “tfo” to store images metadata. These files describe different image information including date, labels, description, and photographer’s name. For the purpose of this research, only image filename, SUBJECT_HEADING and STORAGE_TYPE fields were used. SUBJECT_HEADING contains a list of labels in the Norwegian language; therefore each label was translated into English using dictionary provided by NTB. 15 labels were missing translation and therefore were removed, a total number of 85 images were labeled with these tags.

Since all neural networks employed in the research were designed to train on images with three color channels, grayscale and CMYK images were filtered out from the collection. STORAGE_TYPE field, which, among other information, contain indication if the image is grayscale, was used for this purpose – pictures which had “svarthvitt”, “blackandwhite”, “monochrome” and “svartvit” values in the STORAGE_TYPE field were removed. However, it was empirically established that not all grayscale images were removed using this method. Therefore, the algorithm also read images from the disk and checked the number of color channels. The first method, therefore, was used only for optimization purposes: such images were filtered out without reading its content. From the total number of 966372

processed images, 54045 (5.59%) were grayscale, 2 had four channels (CMYK) and one picture was in the metadata, but was missing in the image collection itself.

As a result of the metadata processing, a Python dictionary was built with image filenames as keys and a corresponding list of labels as values. Since this process was time-consuming (around five to six hours on PC with HDD), the resulting dictionary was saved to disk using pickle [31] format and further read and reused in the experiments.

In addition to image metadata, NTB provided the author with categories tree definition. The tree was defined in a text-based format where TAB character was indicating the parent-child relationship. The tree was transcoded to newick [32] format by applying recursive function and further used to build in-memory tree representation via Python ETE toolkit [33].

3.2 Trial experiment

The goal of the trial experiment was to get insights on a potential of the system when applied for given image collection, as well as to check implementation, training and testing processes, and to discover possible issues which can be addressed in the further experiments.

The additional goal for one of the experiments was to test the potential speedup of the training process when LMDB is employed as an image data source instead of a Python Data Layer used in other experiments.

Further sections describe each part of the experiments in more details.

3.2.1 Selection of categories

The goal of the categories selection for the trial experiments was to find a small subset of the least challenging categories for automated training. Therefore, only categories with more than 1000 images were included. To further remove most of the abstract terms, categories that did not belong to the “entity” or “sport” classes of WordNet [34] were excluded from the selection. The “sports” class was selected specifically due to a good representation of sports images in the dataset. The initial choice of the categories was performed based on the flat list of NTB categories, without consideration of the original tree structure. In addition, each category in the resulting list of candidates was manually investigated and included or excluded based on the selection criteria described further.

The category was excluded based on its name if:

- It was a combined term. Examples: *moos-and-lichen*, *museums-and-art-galleries*.
- It was ambiguous. Examples: *direction* (the sign or human pointing the direction).
- The term was contextual rather than visual. Which means that it was not

possible to classify an image to this category based on only visual information and additional context was required. Examples: *used-cars*, *counterfeit-money*.

Random selection of images from each category was examined, and the category was included only when it met all criteria:

- All images contained entity or sport of interest, and it was clearly visible.
- All images were consistent in the interpretation of the term.

After selection of the categories, some of them were merged or renamed. The final list of 39 categories is shown in Table 1 together with the names of the NTB labels that were merged to form a new category.

Since the library interface for creation of LMDB entries allows assigning only one label to an image, the multi-label case of classification requires much more work. Therefore, for the separate experiment where LMDB [35] file was used as a pictures content data source (see section 3.2.3), skiing subcategories were chosen since these were mutually exclusive in NTB's dataset and single-label classification case was perfromed.

Category	NTB categories
skating	icehockey, skating, figure-skating
bridge	bridges
woman	women
umbrella	umbrellas
aeroplane	aeroplanes
crowd	demonstrations, supporters, crowds
signs	signs, traffic-signs
bus	buses
football	footbal, football-pitches, fotballs
park	parks, forests
harbour	harbours
hand	hands
running	running-[athletics], middle-distance-running, long-distance-running
flag	flags
firemen	firemen
shoe	shoes
skiing	skiing, alpine-skiing, nordic-skiing, biathlon, freeskiing, freestyle-skiing, slalom, snowboarding, giant-slalom, super-g, telemark-skiiing, downhill-skiing, nordic-combined, cross-country-skiing, ski-jumping, ski-orienteering, long-distance-ski-races, ski-flying, relay-races, skis-and-ski-poles, ski-trips
team-handball	team-handball
medal	medals
boat	boats, yachting, ferries, sailboats, passenger-ships, cruise-ships, fishing-boats
man	men
child	children, girls, boys
flower	flowers, bouquet-of-flowers
bicycling	bicycle-racing, bicycling, road-bicycle-racing, bicycles, cross-country-bicycling
dog	dogs
car	cars, ambulances, electric-cars, hybrid-cars, limousines, sports-cars, classic-cars, police-cars, automobile-racing, traffic
army	soldiers, the-armed-forces
sky	sky, clouds
snow	snow
tree	trees
person	full-length-portrait, portrait, persons
train	trains
landscape	landscape
norwegian-national-costumes	norwegian-national-costumes
beach-volleyball	beach-volleyball
helicopter	helicopters, rescue-helicopters
beach	beaches
triumph	sign-of-triumph
swimming	swimming-[sports], swimming

Table 1: Trial experiment categories selection

3.2.2 Dataset preparation

The base training dataset was created using selected categories. Distribution of images for each category is shown on Figure 3.

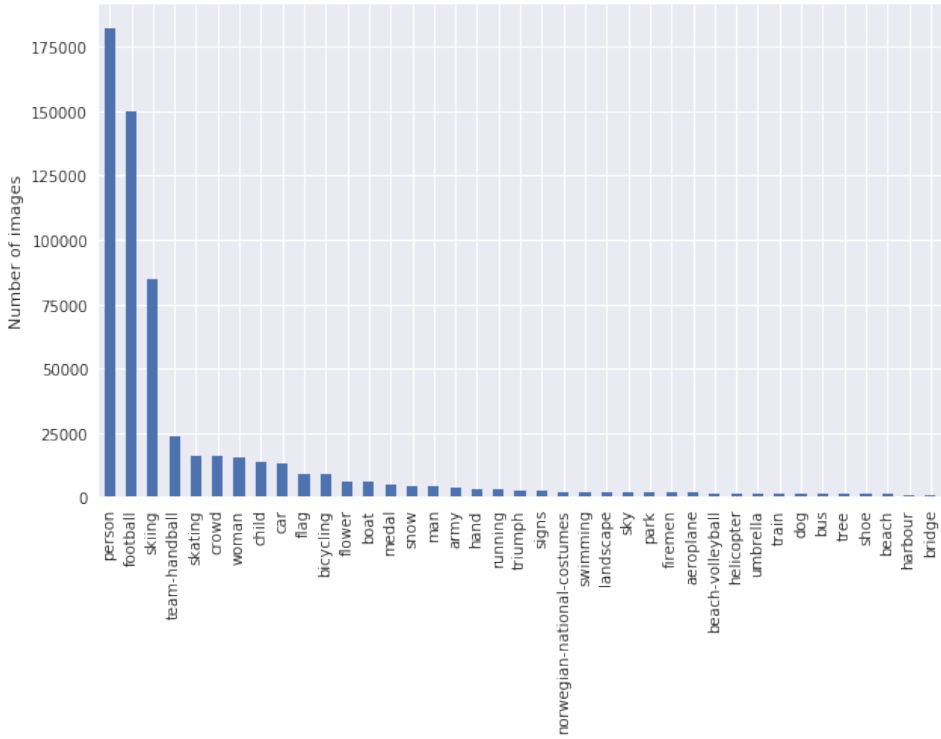


Figure 3: Category size distribution

The chart indicates a big imbalance in the number of images per category. Undersampling approach was used to solve this issue, where random images from the majority categories were removed from the dataset. There is additional challenge connected with a multi-label classification that rises when some of the categories are tightly coupled, meaning that one image can be part of several categories. In this case, by removing an image from one category, it will have to be removed from all other categories as well to keep consistency in the dataset. If the image will not be removed from the whole dataset and will belong only to the part of its actual categories, it can lead to false-positive errors (which are not errors originally) during the training [36]. The problem arise when the image has to be removed from the category with a small number of images initially. There are two choices in this case:

- To remove an image from all categories. In this case, it is possible to achieve state when no category has more pictures than the desired threshold, but this can lead to an even smaller number of images in underrepresented categories.
- To keep a picture in all categories. In this case, categories with a limited number of pictures will be preserved. However, it can be a situation when some categories size exceeded the desired threshold.

The decision on which choice to make depends on the particular case with the specific imbalance level, the level of intersection between categories, as well as a total number of images in all categories. Since some of the categories from the constructed dataset in this research had only 1000 images, the second option of keeping images in all categories was chosen.

The resulting balanced dataset with the threshold level of 10000 is shown on Figure 4. Due to the chosen threshold, *person*, *football*, *skiing*, *team-handball*, *woman*, *crowd*, *child*, *skating* and *car* categories were undersampled.

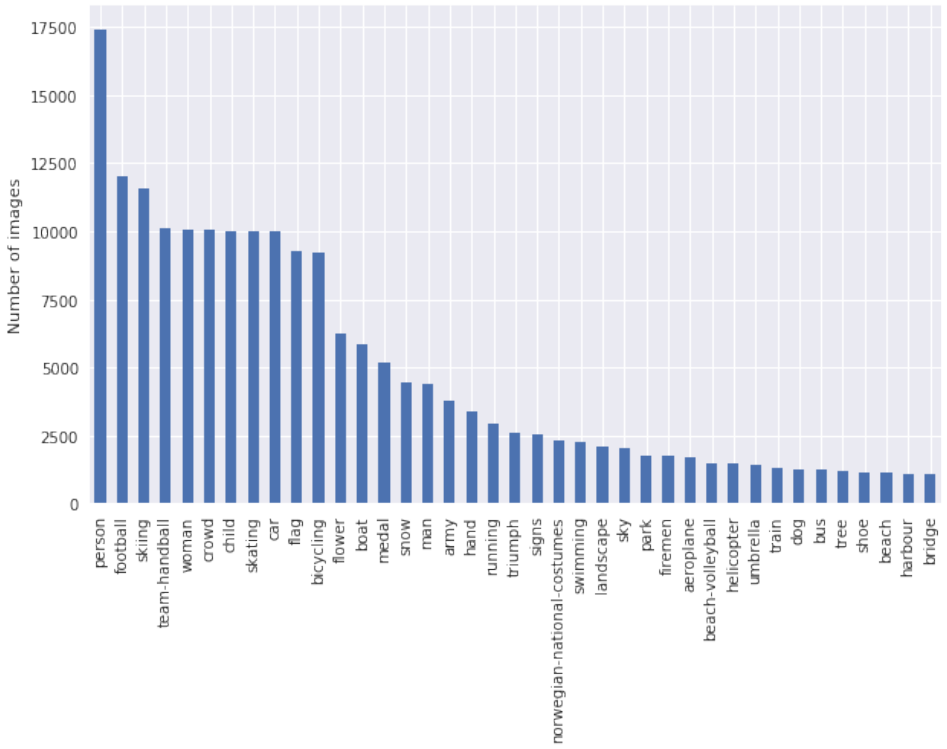


Figure 4: Category size distribution dataset balancing

In order to test how dataset improvement can influence resulting system performance, additional experiment was conducted where context content was filtered-out from some of the categories. In particular, pictures labeled with *portrait*, *press-conferences* and *coaches* were removed from all categories except *woman*, *umbrella*, *hand*, *person*, *norwegian-national-costumes*, *child*, *medal*, *triumph*, *man*. The motivation was to remove such pictures that can be classified to a particular category only using additional contextual information, for example to remove press-conference pictures in the category of football.

As described in Background chapter, after the dataset is generated, it has to be split into three parts: training, validation, and test. The training set is used directly in the training process. The validation set is used to control training process and to find the point of its termination. The test set is employed in the testing process to check the resulting system performance. The intersection of all of this sets should be an empty set, meaning that each image should be part of only one of them. Having separate testing set should potentially increase the generalizability of results since in this case system will be tested on new images, the ones that were not used during the training process.

Following the common recommendations [21], the original dataset was randomly split for each category with the ratio of 0.6, 0.2, 0.2 accordingly. For example, around 60% of the images from the category *person* were assigned to the training set, 20% – to the validation set and 20% – to the test set. The challenge on this stage is similar to the one described in the balancing step – in the case of a multi-labeling system, it is possible that some categories are tightly connected which can lead to unequally distributed images. For example, some images from the 60% selected for the training set from the *person* category can also have other labels, which can lead to the state when more than 60% of images from these categories are now part of the training set. In the trial experiments, a “priority” algorithm of dataset split was implemented and used, where the training set had higher priority than validation and test sets, meaning that each category from the training set had a higher chance to satisfy 60% percent requirement than ones from the other sets.

The base balanced dataset, as well as training, validation and test sets, were generated randomly for each experiment. Therefore, the difference in image samples can contribute to the variation of some results. It was changed in the main experiment to improve the validity of the comparisons of different system implementations.

3.2.3 Training process

Caffe [37] learning framework was used in the research. Specifically, pycaffe [38] Python interface was used to create specification, train, and test deep neural networks.

Due to the relatively old hardware available to the author during the trial experiments (see Section 3.2.5), CaffeNet [39] was chosen as a base neural network since it is a slightly changed version of AlexNet which was trained on the similar hardware [3].

The fine-tuning approach of the pre-trained network described in Section 2.3 was used in the study. Pretrained weights for the CaffeNet were downloaded from the Caffe Model Zoo [40].

Since the original CaffeNet (as well as AlexNet) was designed for a single-label classification on ImageNet dataset, the network structure had to be adapted for the multi-label purpose. Specifically, Data layer, the last InnerProduct layer, and SoftMax loss layers were replaced in the model. Since Caffe Accuracy layer was not adapted to work for the multi-label case during the time of the research, it was eliminated from the model. The weights for the rest of the layers were copied from the pre-trained model.

Data layer

Caffe framework can read image data during the training process from the different sources like database, memory, or directly from files on the storage [41]. Due to its flexibility, Python layer was used to supply network with image data from disk. However, the additional experiment was also conducted with LMDB [35] as a source of data to compare the speed of the training process. Since CaffeNet DNN was used in the experiment, all images were resized to 227x227 and RGB channels were swapped to BGR. In the case of Python data layer, these changes were made during the training process for each image, while when LMDB was used as a data source, these transformations were performed as a separate step before the training, and all image data was written to a newly created LMDB file.

InnerProduct layer

Since pre-trained weights for CaffeNet were optimized to classify 1000 categories from ImageNet dataset, the last fully connected layer had to be retrained from scratch. Therefore InnerProduct layer with the name “fc8” which had 1000 outputs was replaced with the new one with the 39 outputs (number of selected categories).

Loss layer

SoftMax loss layer used both in AlexNet and CaffeNet is only suitable for single-label classification. Therefore new loss function appropriate to a multi-label case

had to be applied. At the time of writing this thesis, the Caffe framework does not support mentioned earlier weighted approximate ranking (WARP) loss function. For this reason SigmoidCrossEntropyLoss layer was used in all experiments, which predicts the probability for each category and calculates loss in the following way [42]:

$$E = -\frac{1}{N}\sum_{n=1}^{N}[p_n \log(\hat{p}_n) + (1-p_n)\log(1-\hat{p}_n)]$$

Two solver algorithms were tested in trial experiments: Stochastic gradient descent [43] and Adam [44]. The following solver hyper-parameters were used for training:

- Base learning rate – 0.0001
- Learning rate policy – fixed
- Test iterations – 1000
- momentum – 0.9
- momentum2 – 0.999
- delta – 10^{-8}

The last three parameters were used only together with Adam solver algorithm. *momentum*, *momentum2* and *delta* parameters were set to the recommended values from the paper [44] (also default values in the Caffe framework).

Even though fine-tuning approach was used in the study, all layers had the same learning rate during the training process in contrast to reducing learning rates for first layers or even freezing them completely. This setup of training was based on the study were authors discovered that training all layers can give even better results compared to training only part of the layers [29].

In order to utilize all available GPU memory as well as to improve the stability of the training, the batch size of 200 was used for training and 20 for validation. Networks for all trial experiments were trained for 1000 iterations.

3.2.4 Testing process

The ”deploy“ version of the CaffeNet was used for testing where data layer was replaced with the direct Input layer, and SigmoidCrossEntropyLoss replaced with the Sigmoid layer. Each image from the prepared test set was preprocessed in the same way as for training, sent as an input to a trained network and predictions for each category were obtained and stored in the numpy matrix. This matrix was then used to calculate all metrics as described further.

To be able to compare different multi-label network implementations, *Average Precision (AP)* metric was in this study, which was defined in the PASCAL Visual Object Classes Challenge in the following way [8]:

> For a given task and class, the precision/recall curve is computed from a method's ranked output.Recall is defined as the proportion of all positive examples ranked above a given rank. Precision is the proportion of all examples above that rank which are from the positive class. The AP summarises the shape of the precision/recall curve, and is defined as the mean precision at a set of eleven equally spaced recall levels $[0, 0.1, ..., 1]$:
>
> $$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, ..., 1\}} p_{interp}(r)$$
>
> The precision at each recall level r is interpolated by taking the maximum precision measured for a method for which the corresponding recall exceeds r:
>
> $$p_{interp}(r) = \max_{\tilde{r}:\tilde{r} \geq r} p(\tilde{r})$$

Therefore, both *precision-recall curve* and *average precision* were calculated for each category using numpy vectorized computations and further used for models comparison.

3.2.5 Hardware

The following hardware setup was used to perform all computations for the trial experiments:

- **CPU** – Intel(R) Core(TM)2 Duo CPU E8400 @ 3.00GHz
- **GPU** – NVIDIA GeForce GTX 580
- **RAM** – 8 GB
- **Storage** – WD1002FAEX 1TB HDD (accessed trough 1Gbps Ethernet)

3.3 Main experiment

Lessons learned during trial experiments were used in the main set of experiments where the neural network was trained on a much broader and challenging selection of categories. These experiments were also performed on the new hardware, which allowed to test more complex neural networks. The following sections will describe the differences made for this set of experiments.

3.3.1 Selection of categories

Selection of categories for the main experiment was based on the whole NTB tree, including its structure (parent-child relationship). As a result, a new tree was created, adapted for training of the multi-label classification system. Issues and insights discovered during this process are presented in Section 4.3.1. This section describes selection criteria for categories as well as all steps taken to create the new tree.

The goal of this part of the research was to create a tree of categories that would be suitable for training the multi-label image classification system, maxi-

mizing utilization of the available metadata. Therefore decisions on how to change the existing tree, which categories to include, and which of them to merge into one were made based on:

- The available tree structure: the path from the root of the tree to the particular category, category children, parents, and siblings.
- The category name (translated from the Norwegian language as described in the section 3.1).
- The number of images in the category itself and the total number of images within all its descendants.

Unlike during the trial experiment, an investigation of a random sample of images from each category, to a get better understanding of the category meaning withing the NTB system, was not performed. There were two reasons for this:

1. The total number of categories in the NTB tree is 3437, which made it practically impossible to look into example pictures from each category, considering the time limitations of this project.
2. Since one of the applications of the resulting classification system is to provide a tag-based search for the end users it was considered that users would base their decision during the usage of such system on the name of the category itself, similar to the workflow used by the author of the research.

Categories were removed from the tree based on the following criteria:

- If the category name was completely contextual. Examples: *second-hand-cloth*, *used-cars*, *counterfeit-money*.
- If the category name was abstract. Examples: *love*, *daily-life*, *future*.
- If the name was a combined term. Examples: *childhood-and-youth-pictures*, *moos-and-lichen*, *fires-in-trains*.
- If the name was representing a combined action. Examples: *thriatlon* (different types of sports), *nordic-combined* (different types of skiing). The category *biathlon* was an exception and was included since even though it is also a combined sport, it is potentially visually recognizable by the unique combination of skiing and gun on the same picture.
- If the name could have a double meaning. Example: *direction* (human pointing somewhere or road signs).

In the NTB dataset, both leaves of the category tree, as well as parent categories, can contain images. To achieve a better level of granularity in the resulting system, in most cases only the most specific categories were included since parent labels can automatically be inferred from its children label. However, in some cases, parent category contained much more pictures than all its descendants com-

bined (for example 1259 images were labeled as general *dogs*, but only 25 images were labeled with the specific breeds). Therefore to maximize utilization of the available metadata, in some cases, parent labels were also included in the final selection of categories. The issue, in this case, was that in the original NTB metadata there were pictures labeled only with a specific category (for example with *huskies* breed), but not with its parent label (*dogs* in this case). The next algorithm was applied to fix this problem: if the category satisfies all criteria mentioned earlier and therefore should be included in the selection, and it has children categories, then all images from each of its children with the subclass relationship were included in this category.

For example, having the *buses* category tree, which is shown on Figure 5, all images labeled as *airport-buses*, *night-buses*, *sightseeingbuses*, *school-buses*, *veteran-busses* or *e-bus* (but not *bus-stops*) were included into the parent *buses* category. While *buses* category itself was included into final selection of categories, several of its children were excluded due to the contextual meaning: *airport-buses*, *night-buses*, *veteran-busses* and *e-bus*. Those labels were considered being practically impossible to classify only by their appearance.

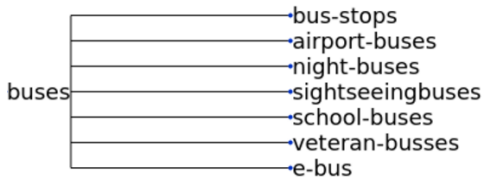


Figure 5: *buses* category sub-tree of the NTB category tree

3.3.2 Dataset preparation

Using the categories tree described in the previous section, a base dataset was created. Figure 6 shows the number of categories with an amount of images above the specified threshold. It is visible that the majority of categories has a low number of images. For example, more than half of the categories contain less than 50 images.

Due to a big number of categories and the multi-label case of classification, which makes it challenging to find particular threshold value that will suit the purpose, it was decided not to perform balancing of the dataset in the main experiment. One of the reasons why finding appropriate threshold can be a challenging task, in this case, is that some categories can be tightly connected and therefore removal images from one of them can also affect the number of images in other categories. Additionally, it was considered challenging to choose a single, justified value of a threshold for the purpose of the experiment.

A new method of splitting the dataset into training, validation and test sets

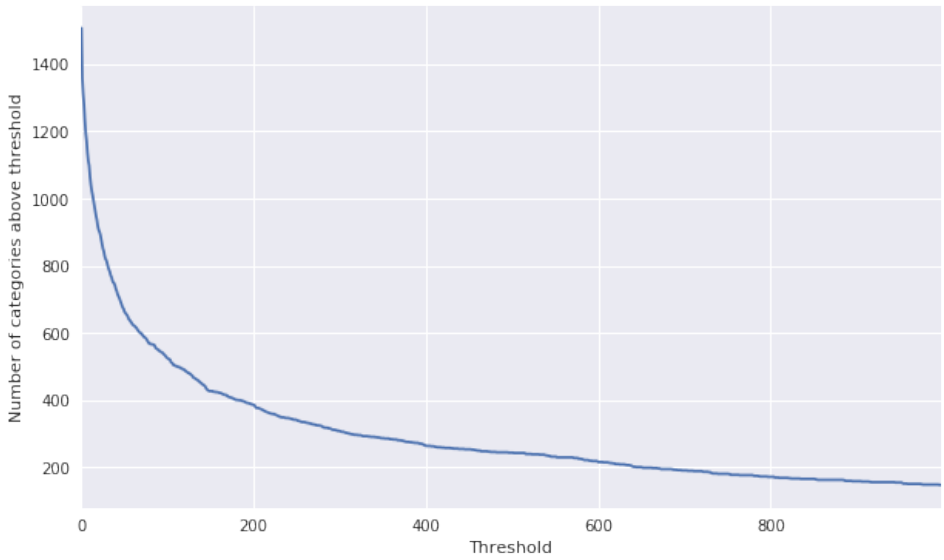


Figure 6: Number of categories with the size above particular threshold

was used. Instead of performing split on a per-category basis like in the trial experiments, the whole set of images was created and randomly divided into three parts with the specified ratio (60%/20%/20%). This method gives better average split on the subset for a multi-label case, as it will be shown in Section 4.3.2. As discussed in Section 3.2.2 of a trial experiment, there is a challenge of splitting multi-label dataset, particularly connected with the high level of coherency between categories. Since used splitting process is random, and to satisfy the requirement of having a minimum number of one image for each category in each of the three sets, it was empirically discovered that the minimum number of images in each category should be 50 for the given dataset. Therefore all classes below this threshold were removed living 663 categories in the final dataset.

To be able to compare results of different implementations, one image dataset was created and then used in training, in contrast to the trial experiments where the initial dataset, as well as training, validation and test splits, were random for each training process. This repeatability was achieved by setting random seed at the beginning of the splitting process and by reusing created LMDB files in further experiments.

3.3.3 Training process

Mostly the same training process described in Section 3.2.3 was used in the main experiments. The differences in the process are outlined in this section.

Due to a new hardware available during the main experiment phase, in addition to CaffeNet, classification system was trained on the newer GoogleNet DNN [45]. The same fine-tuning approach was applied. The CaffeNet was adapted to a multi-label case in the same way as during trial experiment.

Since GoogleNet introduced new InceptionLayer, which combines an output of the three fully connected layers on the last levels of the network, in addition to DataLayer all these three layers had to be replaced with new ones in order to train network on the new set of categories. Each of this three layers has their loss layer which was also substituted by a SigmoidCrossEntropyLoss layer. For the same reason as in CaffeNet, all three Accuracy layers were removed from the network.

Adam solver algorithm was used since it showed much better result compared to Stochastic Gradient Descent during the trial experiment.

The usage of LMDB as an image data source showed good results during the trial experiments. Therefore it was decided to adapt this solution to the multi-label case. While in the single-label case the image category is represented as single number in the database, in the multi-label case categories data have to be a vector of numbers or binary vector where value "1" on the ith position indicates that image belongs to the category i. Therefore, two LMDB files were created to achieve that: one for image data only, and one for labels which are represented as a ground-truth two-dimensional matrix.

Since the new GPU had more memory than the old one (12GB vs. 3GB), the batch size for CaffeNet was increased to 900 during the training phase, and to 200 during the validation phase, corresponding values of 128 and 16 were used for training of GoogleNet. The same hyper-parameters as during the trial experiment with Adam solver were used for training, except that test iterations were chosen to cover the whole validation set of images: 700 and 8600 for CaffeNet and GoogleNet correspondingly.

In contrast to the trial experiments, during the main experiments training process was terminated after a loss on the validation set was stabilized to get the most from the used dataset and network, and to avoid overfitting.

3.3.4 Testing process

The same testing process was used as during the trial experiment described in the section 3.2.4.

Similar to the adaptation of the deployment version of CaffeNet, DataLayer of the production version of GoogleNet was replaced with the Input layer with the corresponding dimensions. Only one from three fully connected layers are presented in the deployment version of GoogleNet specification, therefore only this layer was changed to the newly trained layer and its corresponding loss layer was replaced

with the Sigmoid layer.

3.3.5 Hardware

As mentioned earlier, the main set of experiments were performed on the new GPU donated by NVIDIA Corporation. For this reason, another PC compatible with this GPU was employed. The following hardware setup was used for the main experiments:

- **CPU** – Intel(R) Xeon(R) CPU E5-1620 v4 @ 3.50GHz
- **GPU** – NVIDIA Titan X Pascal
- **RAM** – 32 GB
- **Storage** – SK hynix SC308 256 SSD

3.4 Libraries and tools

This section gives a description of used libraries and tools in experiments as well as motivation for using them.

- **Caffe** [37] is a one of the most widely adopted in the industry deep learning framework. In addition to the training framework itself, it provides with a collection of pre-trained models from different research papers [40] and Python language bindings [38]. It also allows to run training on the CUDA [46] enabled graphics cards which significantly improves both training and testing speed [3].
- **NumPy** [47] is a Python library for scientific computing. This library was used in the project for computation optimizations trough vectorized calculations.
- **pandas** [48] is a Python library which provides efficient data manipulation trough different data structures as well as its visualization. It was used for tables representations and drawing charts.
- **Jupyter Notebook** [49] is a web application for live coding, visualization, numerical simulation etc. This application was used for both remote code editing and tables/charts visualizations.
- **ETE Toolkit** [33] is a Python framework for tree visualization and analysis. It was used for categories tree parsing (using newick [32] format), processing and visualization.
- **vagga** [50] is a tool that helps to describe and build development environment as well as to run processes inside of it. It also provides different levels of process isolation through Linux namespaces [51] kernel feature. It was used to specify the whole research project environment. Isolation properties that it gives, as well as full declarative system specification, should improve project reproducibility.

4 Results

4.1 Dataset analysis

This section presents the results of the initial analysis of the NTB dataset. This analysis was the basis for the further experiments design and implementation.

NTB categories are organized in a tree structure, the 31 top-level categories of which are shown in Table 2. The table also contains an amount of descendants each category has, as well as the total number of pictures that belong to its subtree.

Top level category	# of descendants	# of pictures
types-of-pictures-and-photographic-tec.	83	376328
sports	232	326424
human-life	217	230672
meetings-conferences-and-events	40	120355
finance-and-business-and-industry	283	106251
media	21	102824
politics	75	80164
royal-families	24	79014
art-and-culture	152	67474
the-legal-system-crime	163	62227
fashions-jewellery-and-clothes	77	52810
houses-buildings-and-construction	197	47501
transport	234	41010
miscellaneous	56	31416
leisure	127	26875
accidents-and-natural-disasters	97	24956
seasons-and-weather	37	24165
holidays-and-days-of-public-celebration	39	23871
nature	120	18058
health-and-health-services	132	14292
science-and-technology	111	14155
food-beverages-snacks-and-cooking	227	13061
war-and-military-affairs	59	11804
geography	20	11484
religion-and-philosophy-of-life	72	9531
industry-and-industrial-facilities	93	8839
schools-and-education	31	7757
community-life-and-social-conditions	21	6610
animals	330	6523
pollution-and-environmental-protection	33	5985
orqanizations-and-associations	2	2798

Table 2: Top level categories with total number of descendants and images

The whole category tree is too big to visualize it in any way. However, the main highlights and insights are provided further in this section. The tree consists of 3437 nodes, 3006 (87.5%) of them have at least one image, 2804 (81.6%) are leaf nodes, maximum depth of the tree is 6, mean depth is 3.45 with the standard deviation of 0.87.

Each image can belong to zero or more categories, meaning that categories were not designed to be mutually exclusive on any level of the categories tree. Top 20 categories with the most pictures in them are shown on the Figure 7. Provided NTB dataset contains 912324 of images, 805946 (88%) of them have at least one label, therefore belong at least to one category.

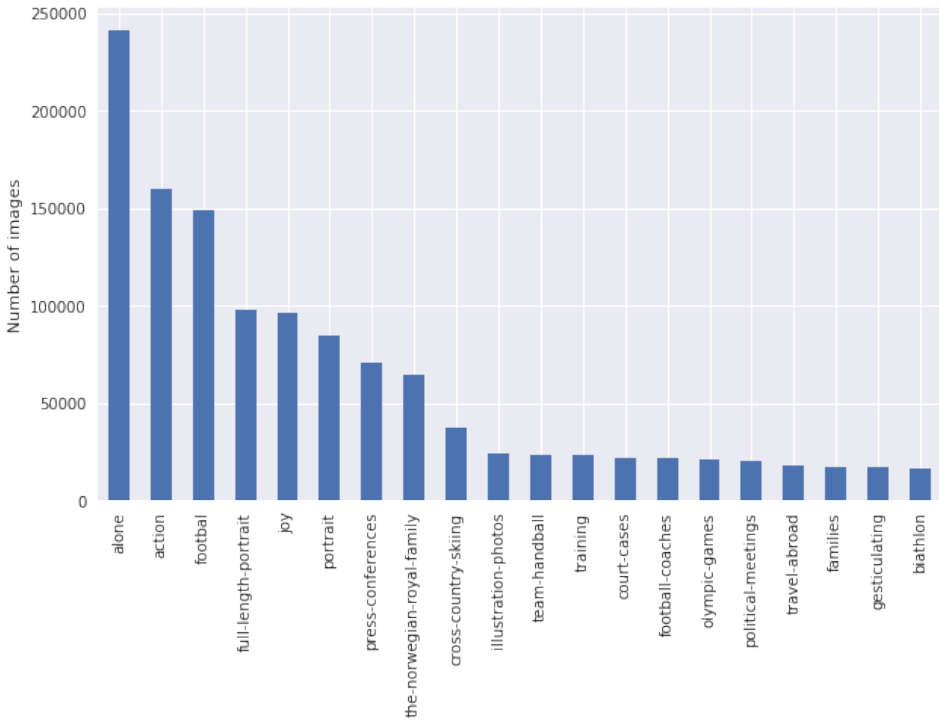


Figure 7: Distribution of images in top 20 categories

From both Table 2 as well as Figure 7 it is possible to see that the provided dataset is focused on sports, politics and finance topics, which is expected since NTB is a Norwegian news content providing company. Further analysis also shows that many categories are specific to Norway, for example *norwegian-royal-family*, *cross-country-skiing*, *the-king's-throne-speech*, *the-parliament-building*, *stave-churches*, and *norwegian-national-costumes*. The category types are also very diverse. For ex-

ample there are categories which represent simple objects (*tv-sets*, *cars*, *doors*), abstractions (*communism*, *neo-nazism*), group of people (*policemen*, *politicians*, *christians*), holidays (*christmas*, *national-days*), relationships (*grandchildren*, *daughters*), sports (*skiing*, *football*), and actions (*handshake*, *document-signing*).

Figure 8 illustrate the distribution of the number of labels per one image in the dataset. Most of the images have from two to four labels associated with them. This fact together with the non-mutual exclusive set of categories, as well as the real-world nature of the images (which potentially are more crowded), makes a multi-label classification system more suitable for this dataset than a single-label. There is a label *alone* in the NTB set of categories that is used to identify images which contain a single object. This label should not be confused with a single-label case of classification since even if there is only one object in the picture, in the multi-label case, it can belong to several categories which describe this object. For example, even though Figures 11b and 11a both contain label *alone*, they both also contain number of other labels that describe these pictures in different dimensions (kind of sport, gender, emotions etc).

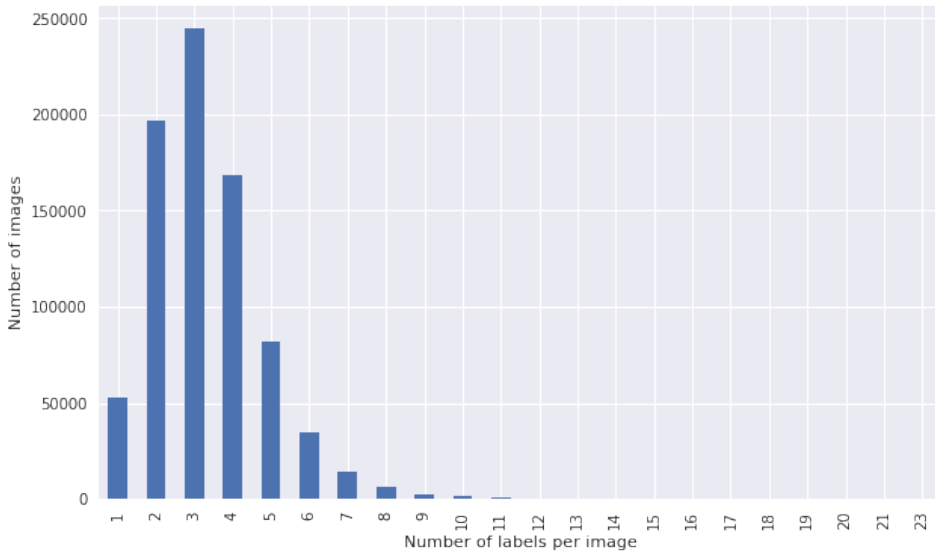


Figure 8: Number of labels per one image distribution

Analysis of the categories tree, as well as example images from different categories, revealed some issues connected with the dataset described further.

Mistakes and not visible entities

The dataset contains various classification mistakes like missing or extra labels on the picture. In many cases labeled object on the picture is not located in the center or even clearly visible. For example, picture labeled as *blueberries* is shown on Figure 9. No blueberries are visible in this image, the main entity visualized on the picture is a person. However, the image does not even contain the label *person*.

Figure 9: *blueberries*, *berries*, *autumn*, *illustration-photos*. Photo by Terje Bendiksby / Scanpix

Background objects

In some cases, labeled object is visible, but not located in the main scene of the picture or is part of the background. For example, on Figure 10 two pictures classified as *flowers* are shown, however, while in both pictures it is possible to see flowers, but they are either surrounded by other objects on the scene or are on the side of it.

Not consistent labels

Some labels are not consistent across the whole dataset. For example, *sign-of-triumph* label is applied on images that have a person with two hands raised up. However, often this label is missing on such images and only label *joy* is applied. This issue is illustrated on Figure 11, where from two visually similar pictures only one has *sign-of-triumph* label.

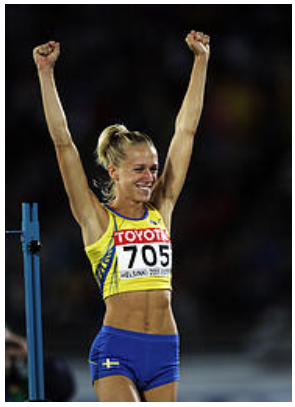
Different purpose of labels

Labels in the NTB dataset can have a various purpose. Most of the labels define some entity or action that is presented in the picture. However, some labels are designed to be a modification of other labels. For example, label *action* was de-

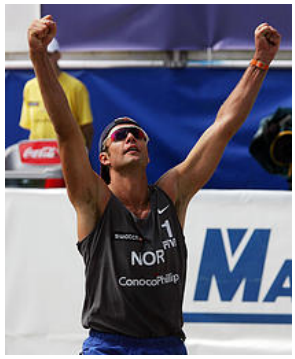
(a) *flowers*, *candles*, *apartment-houses-in-the-city*, *fires*, *death*, *house-fires*, *crime-scenes*. Photo by Stian Lysberg Solum / Scanpix

(b) *red-carpet*, *flowers*, *the-norwegian-royal-family*, *policemen*, *full-length-portrait*, *art-exhibitions*, *hats*, *the-dutch-royal-family*, *opening-ceremonies*. Photo by Fredrik Varfjell / Scanpix

Figure 10: Example pictures from *flowers* category



(a) *alone*, *athletics*, *high-jump*, *women*, *sign-of-triumph*, *joy*. Photo by Cornelius Poppe / Scanpix



(b) *alone*, *beach-volleyball*, *action*, *joy*. Photo by Alf Ove Hansen / Scanpix

Figure 11: Example of two similar images but with different labels. Only one image belongs to *sign-of-triumph* category

signed to be used together with any sports category like *football* to get football players in action. Mentioned earlier label *alone* serves to identify images withing particular category with only one object illustrated on them. This inconsistency can potentially influence resulting classification system performance.

Contextual categories

There are many contextual categories, where visual information is not enough to determine if image should belong to the category and additional context information is necessary. Example of such categories: *second-hand-cloth*, *used-cars*, *counterfeit-money*, *finance-debates*. More issues discovered during category tree evaluation are presented in Section 4.3.1.

Contextual images

There are many images which have an assigned category that can not be derived only from the picture itself, and the more contextual information is needed. For example, picture that is part of the category *politicians* as well as *fishfarming* and *fisheries-industry* is shown on Figure 12. While the first category can potentially be automatically derived using face-recognition approach, the second and the third categories are fully contextual. To assign these labels to the picture the system needs more additional information such as when and where the picture was taken, as well as the topic discussed in this place and time. Another example is press-conference or portrait pictures that are part of some sports category. From 151533 images of football, 151533 (10.4%) belong to the *coaches*, *press-conferences* or *portrait* categories as well.

Similar images

Due to a real-world nature of the used dataset, there are groups of pictures that were taken from one event. As a result, some images are visually very similar to each other. This issue can arise during the split of the dataset on training, validation and test parts. If two similar images end up in two different sets, it can potentially influence the validity of results and the reported precision of the system.

Figure 12: *politicians*, *fishfarming*, *fisheries-industry*, *travel-in-norway*. Photo by Gorm Kallestad / Scanpix

The reason for the mentioned issues can potentially be that the dataset, as well

as its categories tree, were not designed with the future automated classification system training in mind. They, however, can influence negatively such system performance. Further sections of this thesis will try to address and discuss these discovered issues from the implementation perspective of the multi-label image classification system.

The size and uniqueness of the NTB dataset make it perfect subject of the research on real-world datasets that were not originally designed for automatic classification training, and can be used to find possible solutions for presented issues that can also exist in other datasets of this kind. Broad, diverse and unique set of NTB categories also eliminates the possibility to direct usage of the systems trained on a more standard set of categories like ImageNet. Therefore, training system on a new selection of categories should be performed.

4.2 Trial experiment

As mentioned earlier, a set of trial experiments were performed for multiple reasons including getting a better understanding of the neural networks potential when trained on the available dataset, checking the training and testing implementations, and getting insights on which hyperparameters work better for this purpose.

The categories selection employed in the trial part of the research is described in Section 3.2.1. The first section presents results of the best network trained from the set of trial experiments. The next section shows performance improvement achieved by filtering out the part of contextual images from the dataset. A comparison of two solver algorithms is introduced in the third section, while the last section presents a comparison of the training speed using two different solutions for data layer of the neural network.

As described in Section 3.2.2, all trial experiments, except the comparison of performance with and without contextual images removed, were performed on different, randomly generated subsets of initial dataset. Therefore a direct comparison of the results from various experiments might not be precise enough. There can potentially be some variations in the results due to differences in the datasets. However, the results can give indications of whether one method of training is significantly better than another one.

4.2.1 Classification performance

Results presented in this section were obtained from the network trained using the Adam solver algorithm for 1000 iterations. The image dataset used to train the network was additionally filtered from the contextual images, this is explained further in the next section.

Training and validation loss curves are shown on Figure 13. The fact that both

of the loss curves go down is an evidence that overfitting is not taking place at this level of iterations.

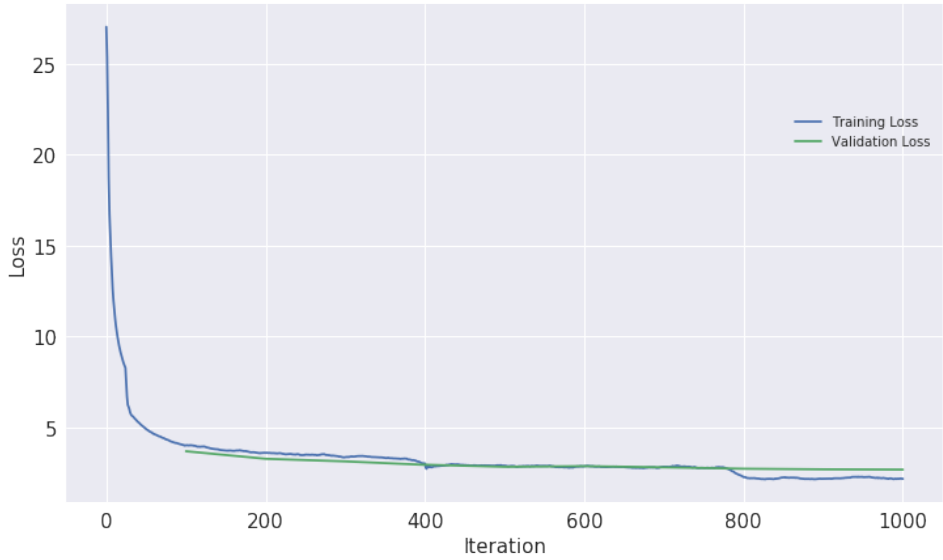


Figure 13: Training and validation loss curve

The resulting average precision for each category is shown on Figure 14. 10 (25.6%) out of 39 categories have an average precision of more than 80%. All these categories, except *cars* and *boats* are belong to sports. 29 (74.4%) categories have an AP of more than 60%.

The relationship between the average precision for each category and the sample size used for training is shown on Figure 15. The chart shows a non-linear dependency between the two values. The categories with the biggest sample sizes have the highest average precision values, however high values of average precision is also represented in categories with smaller sample sizes. Some of the categories with big sample sizes have a lower value of average precision. The potential reason for this can be that there are challenges connected with the category itself when it comes to automatic classification. The same reason could apply to the fact that some categories with similar values of the sample sizes have very different average precision (for example *bus* and *shoe*).

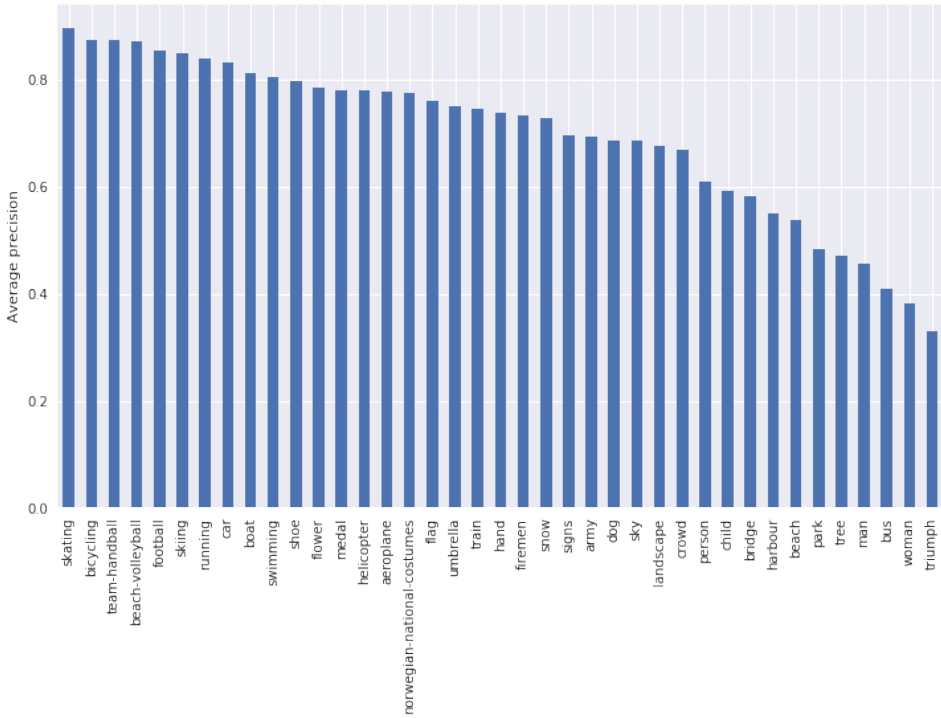


Figure 14: Average precision results

Figure 15: Average precision vs category sample size used for training

Figure 16 shows a comparison of average precision values for different iterations. As expected, average precision increases with further training. An additional observation is that further training appears to have a bigger effect on categories with lower values of average precision, compared with categories with higher values which got a lower increase in the precision. Therefore the difference between higher and lower categories decreases in line with the number of passed iterations.

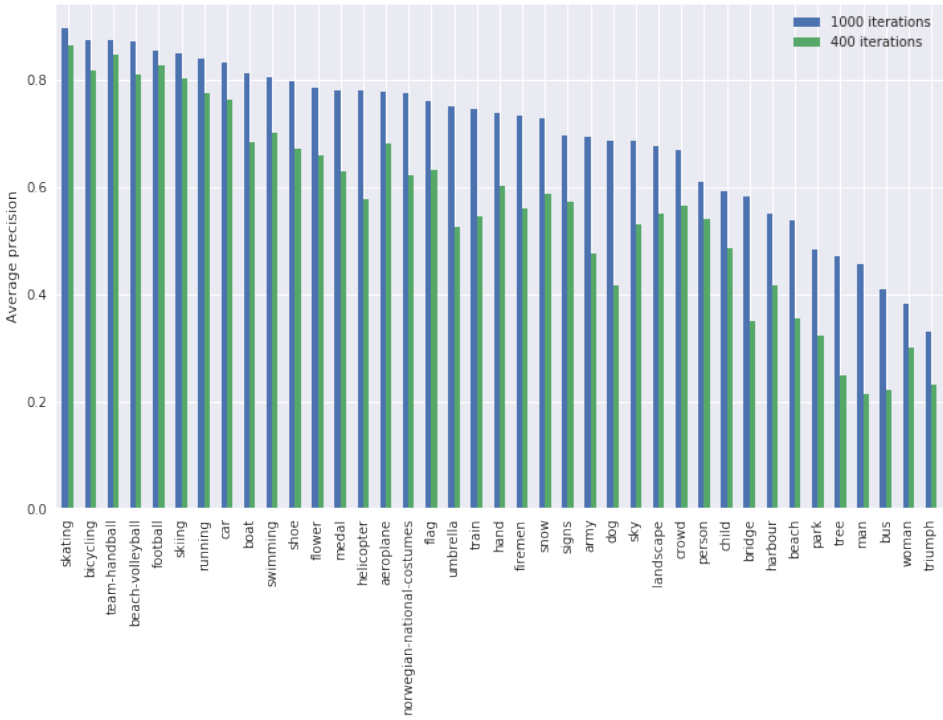
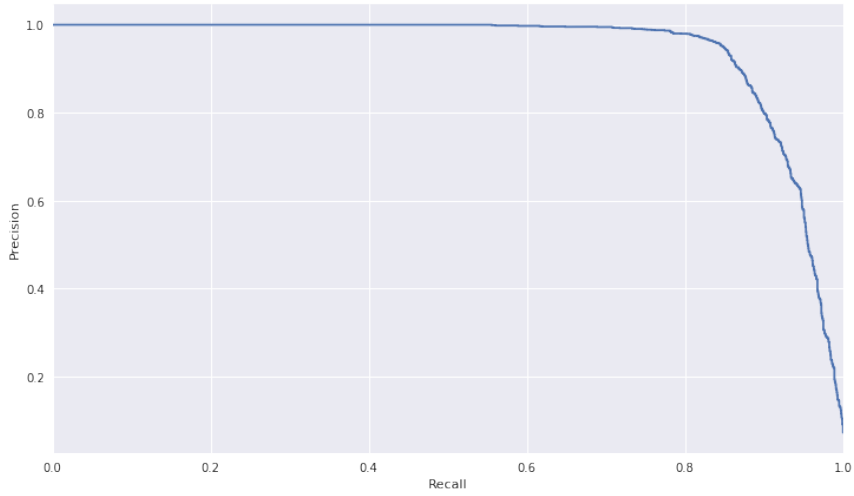
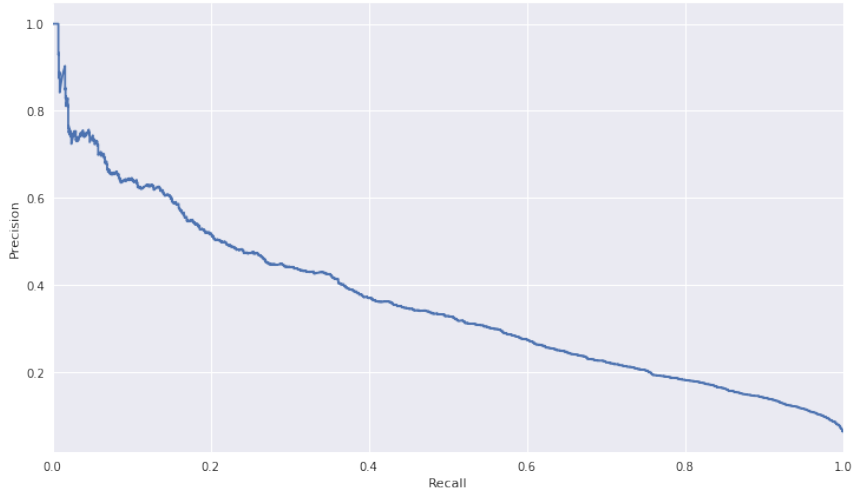


Figure 16: Comparison of the average precision for network on 400 and 1000 iterations

As described in Section 3.2.4, the average precision is calculated based on the precision-recall curve for each category. An example of such curve (for the best category of *skating*) is shown on Figure 17a. It shows that around 80% of the *skating* images can be found with almost 100% of precision. Figure 17b shows the precision-recall curve for one of the lowest categories (*woman*). In comparison to the previous chart, it is evident that the precision drops much faster with the higher values of recall.



(a) Precision-recall curve for *skating* category



(b) Precision-recall curve for *woman* category

Figure 17: Precision-recall curves for one of the best and worst categories

4.2.2 Context improvements

In order to show the potential improvement that can be achieved by filtering contextual images from the dataset, separate experiment was performed. As it was mentioned in Section 3.2.1, images labeled as *portrait*, *press-conferences* and *coaches* were removed from all categories except *woman*, *umbrella*, *hand*, *person*, *norwegian-national-costumes*, *child*, *medal*, *triumph*, *man*. The goal was to remove portraits of persons (such as portraits of sportsmen or their coaches) and press-

conference pictures since it was considered that in order to classify such images in certain categories (like *football* or *skating*) either additional information or a face-recognition system is needed.

The first network was trained for 1000 iterations on a randomly generated dataset (as described in Section 3.2.1), then contextual images were removed from the specified categories and a new network was trained for 1000 iterations on the filtered dataset. Results from both experiments are shown on Figure 18.

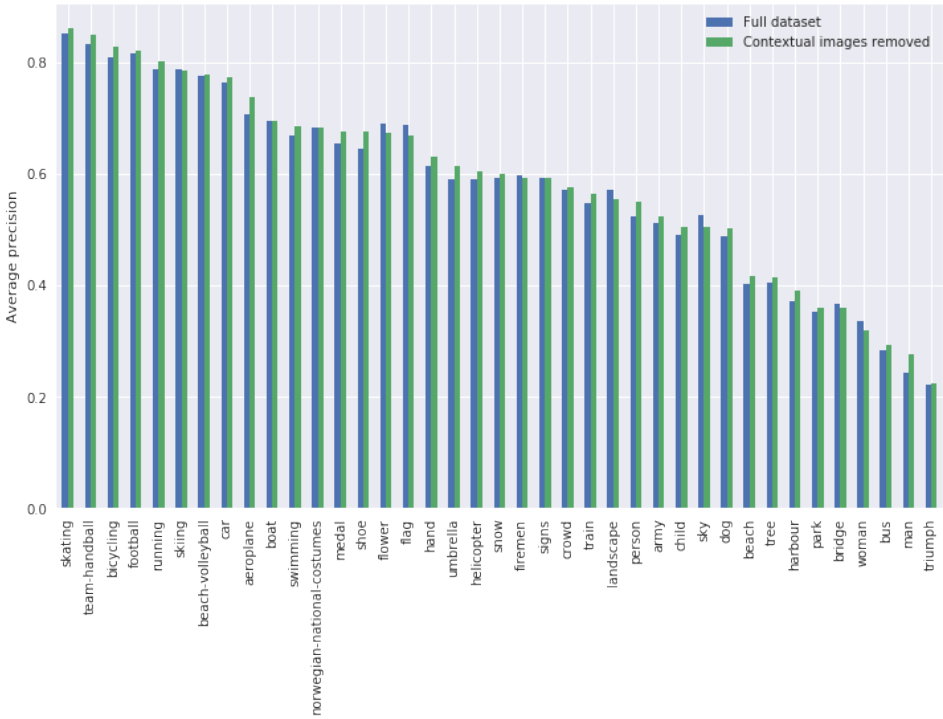


Figure 18: Comparison of average precision for network trained on the whole datasets and with contextual images removed

28 (71.8%) out of 39 categories got improvement in their average precision values including seven categories from those which were skipped in the filtering process and had the same dataset. Six of the categories reduced their average precision. One explanation of these results can be that in the used neural network implementation output probabilities from the final fully-connected layer are not entirely independent from each other since they can use on the same features from the earlier layers. Features learned on images from one category can also be used to produce probabilities for other ones. Therefore changes in a sample for one

category can influence end classification of the other.

Results give an indication that end system classification performance can be potentially improved by removing fully contextual images from the dataset. Section 5.1 contains further discussion on obtained results.

4.2.3 Adam vs SGD

An additional experiment was conducted to test which solver algorithm works better for the purpose of the research. As described in Section 3.2.3, two widely used algorithms were compared: Adam and Stochastic Gradient Descent (SGD). Two separate networks were trained for 1000 iterations each with its algorithm applied. Resulting average precision for both networks is shown on Figure 19.

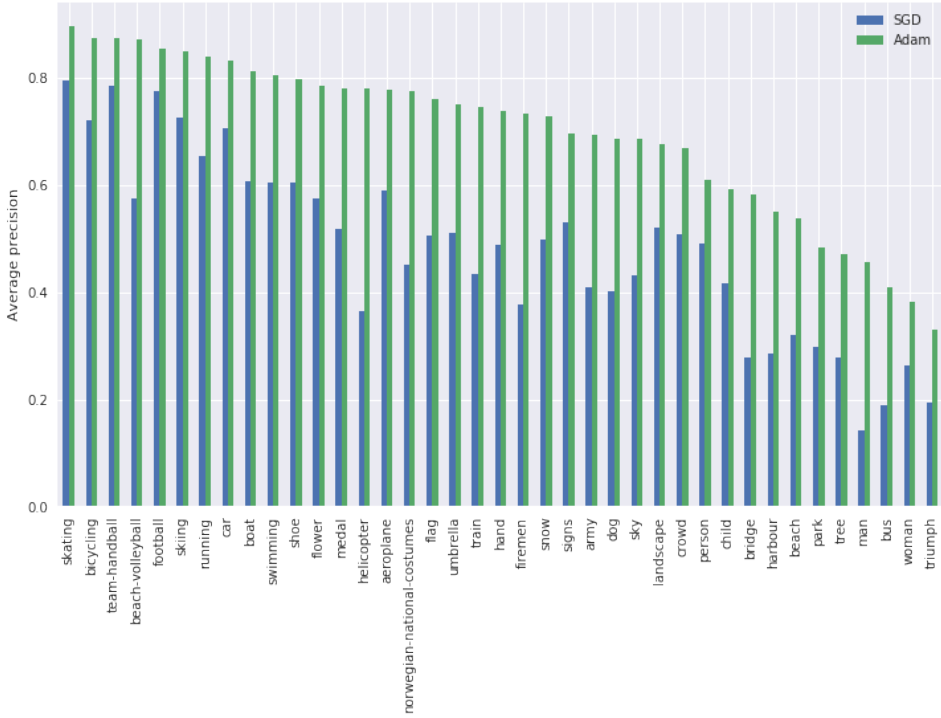


Figure 19: Comparison of average precision for network trained using SGD and Adam solver algorithms

Due to a significantly better average precision resulting from the training for the same amount of time, Adam solver was employed in all further experiments.

4.2.4 LMDB vs Python DataLayer

All trial experiments were performed on a relatively small selection of categories and a total number of images. This limitation allowed the author to perform experiments in a short amount of time. During main experiments, however, a much broader set of categories was employed. The total number of images used in the training and testing process was therefore also increased. Such increase could impact the overall training time of the system. Therefore, it was decided to investigate on possible optimizations of the developed system. Since all computational layers in the system pipeline were already leveraging GPU, the bottleneck of the whole system was seemingly the first Python data layer. The Python data layer was used in the trial set of experiments due to its flexibility. For the main experiments, however, flexibility was considered less valuable, and speed was a priority because of the increased dataset size. The Caffe framework supports other, more optimized ways to deliver image data inside the network. As described in Section 3.2.3, Lightning Memory-Mapped Database (LMDB) was chosen as an alternative source of image data.

An additional experiment was therefore performed to get insights on the potential speedup of the system training process while using this database. As it was mentioned earlier, due to the limited interface provided by the Caffe LMDB module, a single-label classification system was trained. Specifically, the skiing category was selected due to the mutually exclusive property of its descendants withing the NTB dataset. Two networks were trained during this experiment: one using Python layer as an image data source, and one that was loading images from the prepared LMDB file.

Based on the reported numbers provided by the Caffe framework, an average value of 0.1045 iterations per seconds was achieved by the network that used a Python data layer, while the other network was operating on 0.5381 iterations per second. Therefore, by using LMDB file as an image data source, the system achieved more than five times speedup during the training process.

The resulting average precision of the skiing type classification is shown on Figure 20.

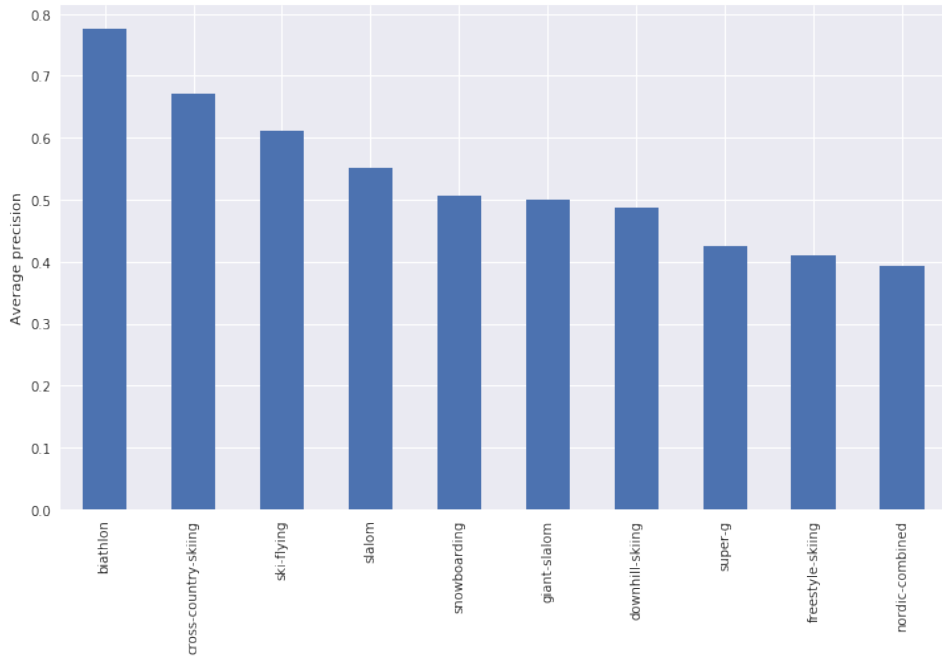


Figure 20: Average precision result of the skiing type classification

4.3 Main experiment

All lessons learned during the trial set of experiments were applied to perform experiments on a more broad set of categories. The goal of this part of the research was to use as much of dataset provided by NTB as possible. The first section gives insights on how initial NTB categories tree was changed and improved for multi-label classification system training. Comparison of the two dataset splitting methods is shown in the second section. The results of the two trained classification systems are presented in the last section.

4.3.1 Category tree evaluation and improvement

As described earlier, NTB provided the author with categories organized in a tree structure. Images in the dataset can be labeled with both parent and child categories. However, both neural network implementations adapted in this study, as well as other ones available in the ModelZoo [40] collection, were designed to be trained on a flat set of categories, without any utilization of the information about the relationship between them. Therefore, the first observation was that until existing neural networks can use this information during the training process, any categories hierarchical structure had to be transformed into a flat representation. Several automatic approaches for such transformation were tested, all of which

failed due to different reasons described further.

The first attempt of automatic approach was to include all categories without any modifications. The problem with this solution is that in many cases when there was subclass relationship between parent and child categories (for example *trees* and *oak-trees*) only part of the images from the child category was also labeled with the parent (only part of *oak-trees* were labeled as *trees*). This inconsistency in the way images are labeled had to be addressed since in another case in could influence the overall performance of the system. Therefore the second approach was to automatically include into the parent category all images of its descendants. It was discovered, however, that many relationships between categories were not of the subclass type. For example, *trees* $\rightarrow$ *leaves* (part-of relationship type), *football* $\rightarrow$ *football-pitches*, *war* $\rightarrow$ *resistance* and *banks* $\rightarrow$ *atms*. The main issue is that it is hard for an automatic system to know which type of relationship the terms have. The third automatic solution was to use only leaves of the tree and remove all other categories. While it would fix previous issues, this approach would require removing a big portion of metadata information that otherwise could be utilized by the system. One example of this would be to remove category of *football* and to keep only its descendants: *football-pitches*, *footballs*, *beach-soccer* and *penalty-kicks*. However, the *football* category contain 150016 images, while all its descendants combined have only 807 images. It was also discovered that it was not possible to make a decision on if parent category should be included or not based only on an amount of pictures in the category itself and its children due to inconsistency in the relationship types mentioned earlier. In addition to mentioned problems, the general issue with the automatic approach to category tree transformation is that many categories were not suitable for training of a classification system. The following challenges with categories were discovered through the tree evaluation:

- Some categories were abstract terms like *genocide*, *love*, *future* or *daily-life*.
- In some cases, category name could have a double meaning. For example, *direction* can mean the direction sign, or person pointing direction.
- Category names can be entirely contextual, which mean that to classify the image into this category, it is necessary to have additional information about it. Examples of such categories: *grandfathers*, *second-hand-cloth*, *used-cars*, *counterfeit-money*, *war-criminals*. An additional discovery was that some categories were classified as contextual only with the knowledge about specific nature of the dataset. For example, such category as *school-buses* can be distinguishable or not depending on the origin country of the specific dataset.
- Category name was a combined term. Examples: *childhood-and-youth-pictures*, *moos-and-lichen*, *fires-in-trains*. The last example category also shows that in some cases one category represented the connection of other ones, *trains* and

fire in this case. Multi-label classification system can potentially discover such connections automatically by assigning both labels to the image.

- Some categories defined a combined action. Examples: *thriatlon* (different types of sports), *nordic-combined* (various types of skiing). It can be hard to classify such categories since there are many pictures with only one type of sport visualized at the same time. The category *biathlon* is an exception since in this case carried gun can be an indicator of pictures from this kind of sport. *biathlon* category is also a good example of why in this case a manual selection of categories had to be performed rather than automatic one.

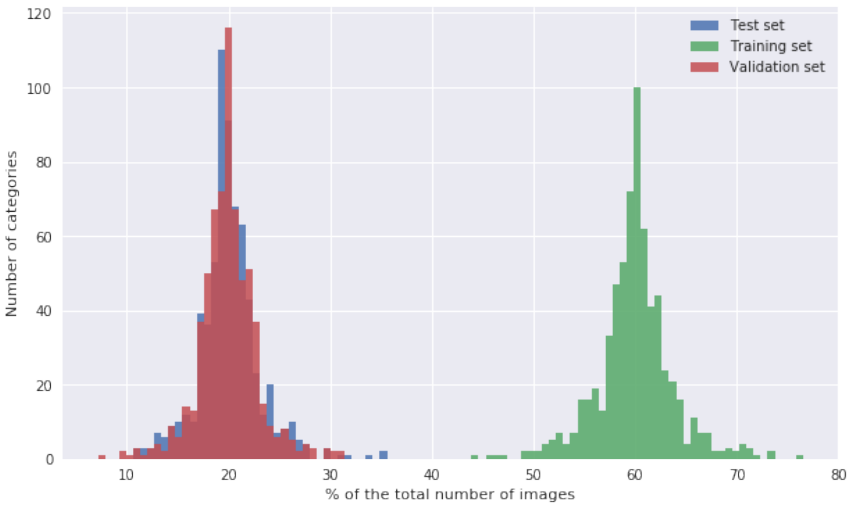
Due to all reasons described above the decision on which category to include had to be made for each case separately. Therefore all 3437 NTB categories were examined manually to create a new tree of categories, adapted for multi-label classification training. For reasons described earlier, it was decided to use leaves of the categories tree where is was possible, with a few exceptions, to get maximum granularity of the end classification. The overall category tree transformation method designed and used in this research is described in Section 3.3.1.

The final category set contained 1507 categories, from which 1440 (95.55%) were copied directly from the NTB dataset without any changes. The rest 67 categories were created by merging images from other categories. 23 (34.33%) of them were extended with pictures of all their descendants (example of such categories: *dogs*, *mushrooms*, *shoes*), while the rest of them were combined from a selected set of subcategories. An example of the new category created with such selection is *trees* that combined images of all tree types, but did not include pictures of such children categories like *leaves* or *branches* since it was a “part-of” relationship between them. Another example is category *person* that was created by merging images labeled with original *persons* NTB category as well as such categories as *women*, *men*, and even such context ones as *grandmothers* and *sons* which were not included in the final selection as a separate categories.

As described in Section 3.3.2 due to difficulties in dataset splitting connected with cross-categories connections in multi-label case, only categories with a minimal number of 50 images were included in the final set. Therefore from 1507 selected categories, only 663 were used for actual training.

4.3.2 Dataset split method

As described in Methodology chapter, different dataset splitting algorithms were used during the trial and main experiments. Comparison of the dataset distribution across training, validation and test sets of both methods is shown on Figure 21.



(a) Main experiment dataset splitting method

(b) Trial experiment dataset splitting method

Figure 21: Distribution of the image dataset across training, validation and test sets

Horizontal axes of both histograms show a percentage of included images from the original image dataset of the particular category, while the vertical axes represent an amount of labels with this proportion value. The histogram of the new splitting method shown on Figure 21a follows the curve of a normal distribution for all sets, with the mean values of the desired ratio (60% for the training set, 20% for both validation and test sets). Therefore most of the categories in all three sets

contain around the target amount of images with a smaller number of members below and above target values. Due to another, prioritized, method used in the trial experiments, distribution on Figure 21b has another shape. Validation and test sets had higher priority than the training set. Therefore both of them have higher mean value than the target 20%, which means that most of the categories from the validation and test sets contain more than 20% of images from the original dataset. The training set has a mean value lower than 60% since there were not enough images left after splitting them between the two other subsets.

The new splitting method was employed in all main experiments due to its better performance regarding the distance to the target split ratio.

4.3.3 Classification performance

Two pre-trained and adapted models were used in the main set of experiments: older and less computational demanding CaffeNet [39], and newer and more advanced GoogleNet [45]. Both systems were trained to classify 663 selected categories. This section presents achieved classification performance and comparison of these systems.

As described in Section 3.3.3 unlike during the trial experiments, a decision on termination of training process during the main set of experiments was based only on the validation loss curve. Specifically, training process continued until validation loss has reached a stable state.

Training and validation loss curves for CaffeNet based system is shown on Figure 22. Snapshot of the system at 2500 iteration was used for all subsequent computations since while training loss continued to go down being optimized by the solver algorithm, the validation loss remained almost the same in further iterations. In other words, after 2500 iterations the model started to overfit on the training set.

The same approach was used to determine termination point for GoogleNet model. Therefore, system snapshot at 35000 iterations was chosen as the most optimal one for the same reasons. Training and validation losses for GoogleNet model are shown on Figure 23.

There is a noticeable difference in the behavior of two training loss curves. It is visually evident that CaffeNet training loss is more stable than the GoogleNet one. This behavior is connected with different batch size used for training of these to networks. As mentioned in Methodology chapter, due to the bigger number of layers in GoogleNet than CaffeNet this model also requires more GPU memory to perform training, therefore the number of images that can be processed in one iteration is also smaller (128 images in GoogleNet compared to 900 in CaffeNet case for a given GPU). Since in the used setup the solver optimizes network weights

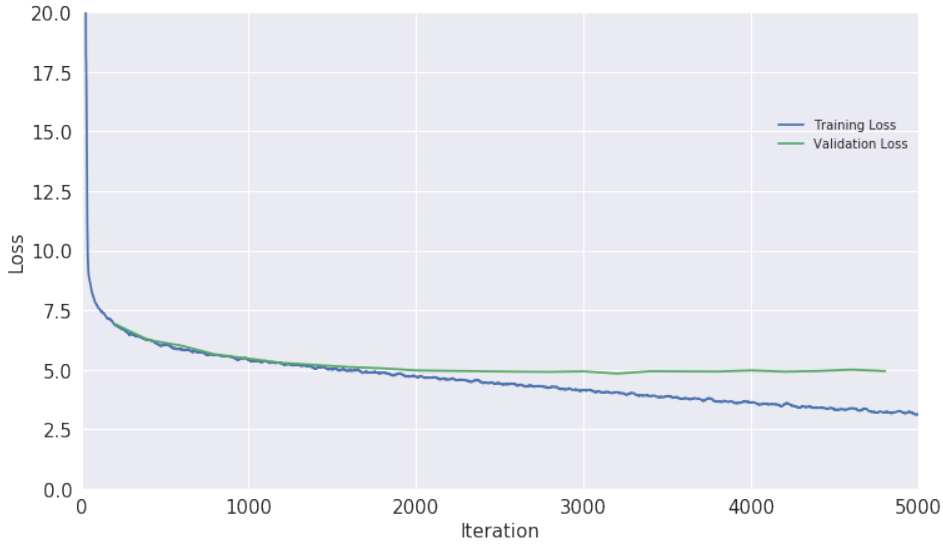


Figure 22: Training and validation loss curve for CaffeNet based model

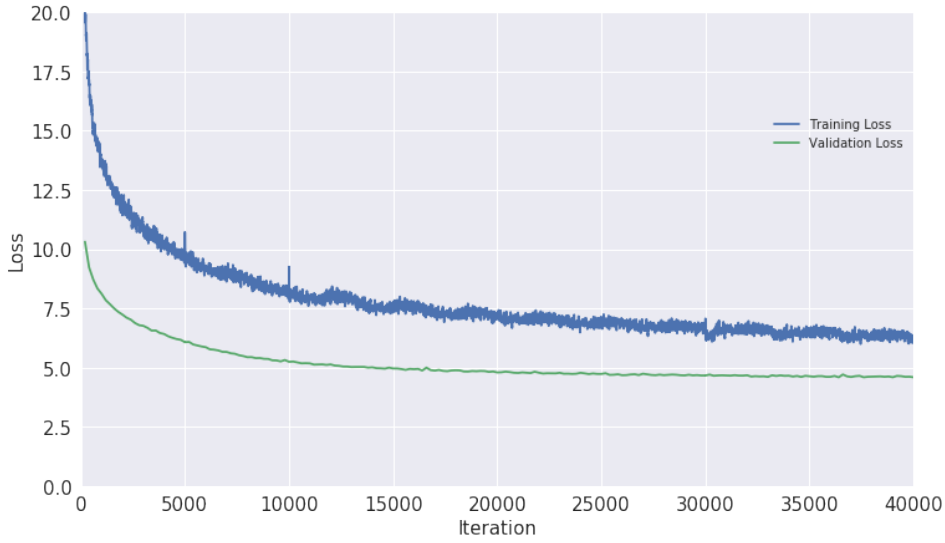


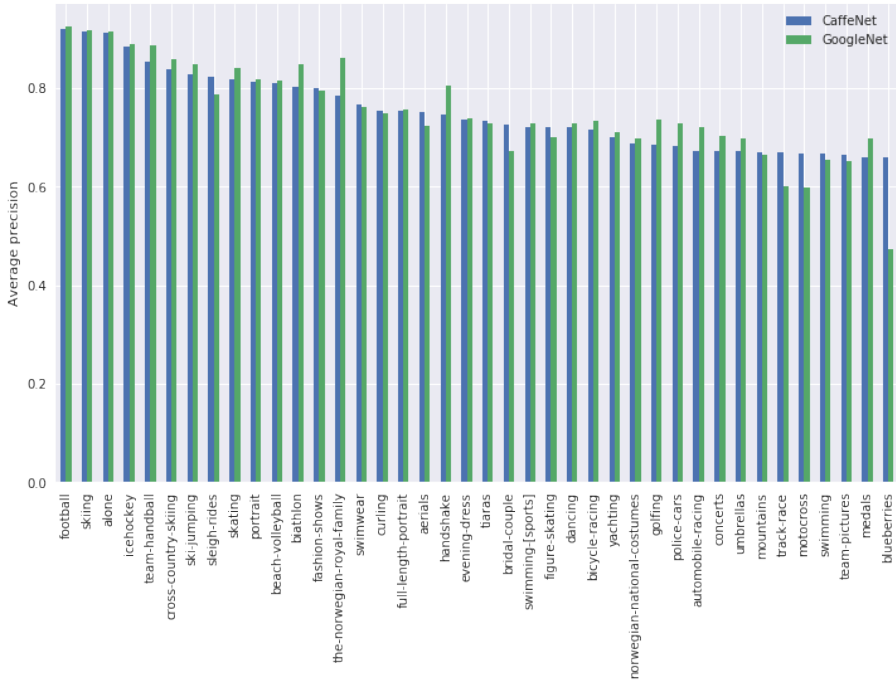
Figure 23: Training and validation loss curve for GoogleNet based model

based on a single iteration, smaller number of images lead to less generalizable changes in weights for the whole dataset. As a result, the difference in training loss between two iterations can be bigger.

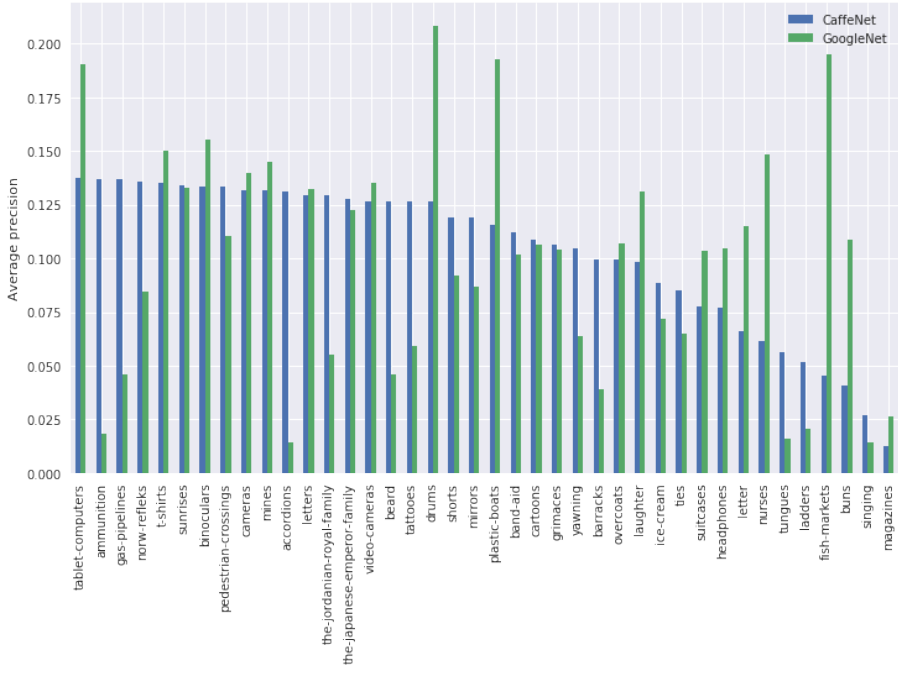
It is hard to show classification performance for each category due to a big num-

ber of them. Therefore, several charts were built to give insights on the general level of average precision of the trained classification systems. Average precision values for the top 40 categories of CaffeNet system are compared with values for the same categories from the GoogleNet model on Figure 24a. GoogleNet has better average precision in 26 (65%) of these categories. The same top 5 categories (*football*, *skiing*, *alone*, *icehockey*, and *team-handball*), in an identical order appear in both implementations. Similar to the trial set of experiments, sports categories were highly represented in the top categories for both CaffeNet and GoogleNet systems. 36 out of 40 categories have more than 400 images in the training sample. Specifically, the mean value for the size of the training sample for these categories is 22848.85 with the standard deviation of 47529.22.

The same comparison, but for the bottom 40 CaffeNet categories is shown on Figure 24b. There is a much bigger deviation in values between two implementations than on the previous chart. Out of the 40 categories, 38 contain less than 400 images in the training sample. The mean value for the size of the training sample for these labels is 409.52 with the standard deviation of 222.55.



(a) Top 40 categories



(b) Bottom 40 categories

Figure 24: Comparison of average precision results for best and worst categories for main experiment (sorted by CaffeNet values)

To compare performance for all categories the distribution of the categories across the series of average precision values is shown on Figure 25. The distribution for both systems covers almost the entire range of precision values. The GoogleNet based system have more categories in both the lower and higher ends of the precision scale. The whole distribution of this system is also more shifted towards the lower values, while CaffeNet based one is concentrated in the middle part of the range. The mean average precision value for the CaffeNet based system was 0.375 with the standard deviation of 0.173, for the system based on the GoogleNet the mean average precision value was 0.34 with the standard deviation of 0.184.

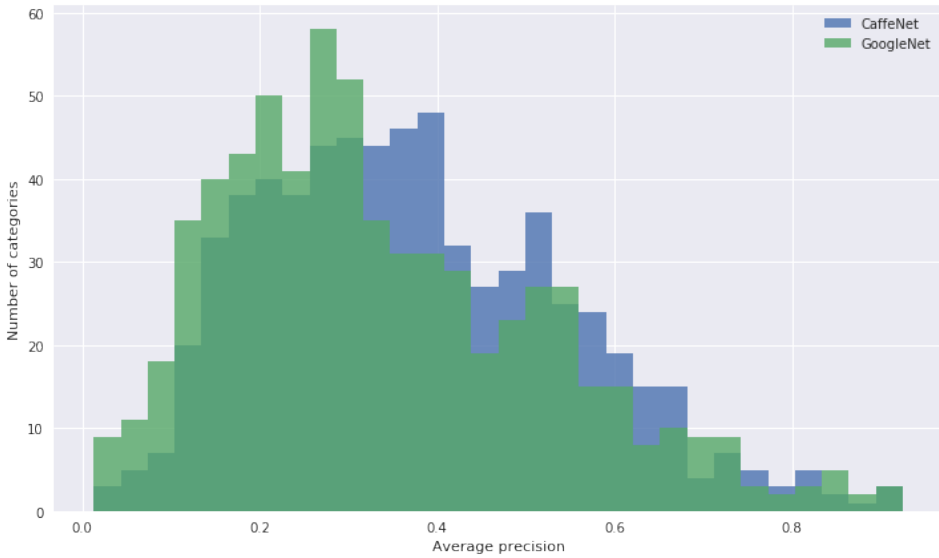
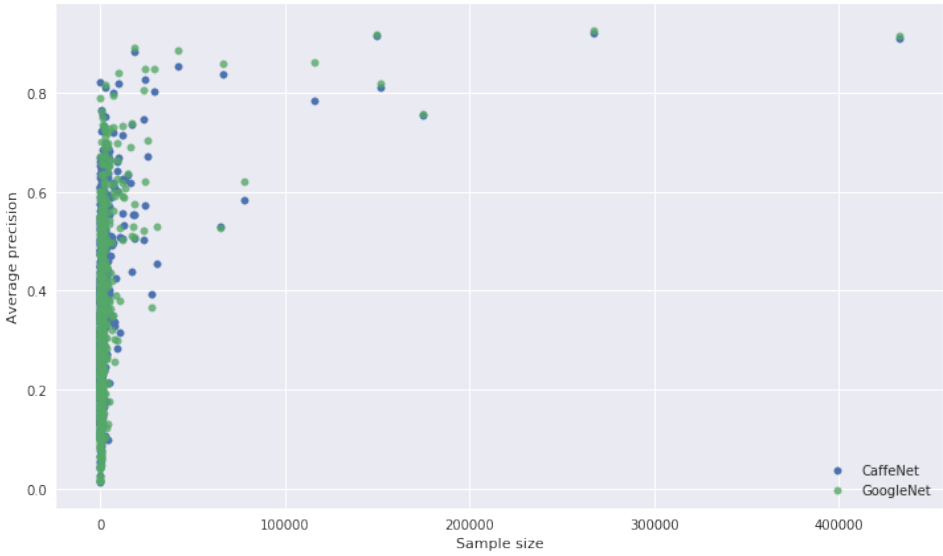


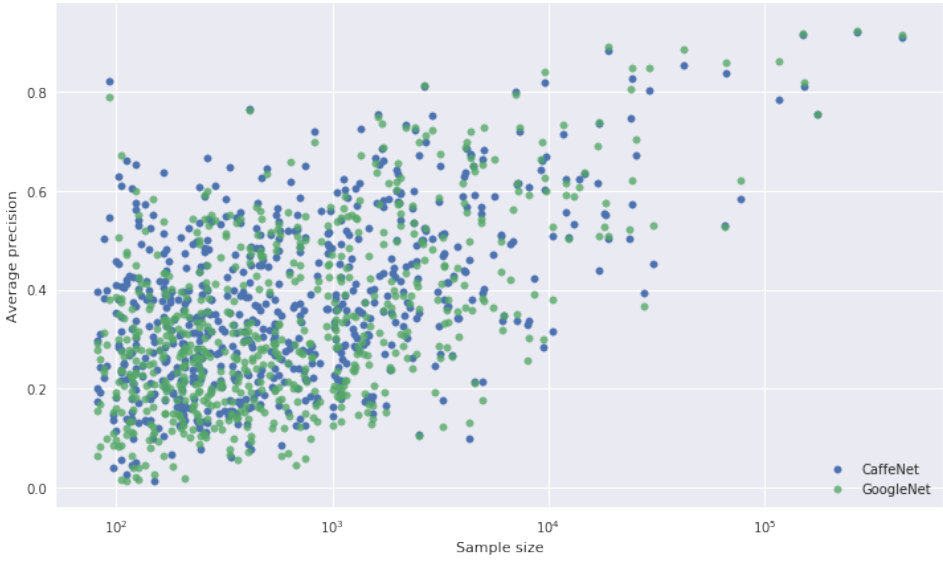
Figure 25: Comparison of CaffeNet and GoogleNet average precision distribution for the main experiment

The charts showed on Figure 26 give insights on how average precision is dependent on the sample size for both systems. While the first chart on Figure 26a has linear scale of horizontal axes, the second one on Figure 26b has logarithmic scale on the size axes. The curve has a similar shape as the one observed in the trial experiments. Until certain limit, the small increase in the sample size can give a big increase in the resulting average precision. After reaching the limit, the behavior is opposite – a big increase in the sample size result in a small increase of the end accuracy. The second chart also reveals a big difference in the average precision for the categories with similar sample sizes, which was also the case in the trial set of experiments. Confirming observations from the previous charts, GoogleNet

based system is concentrated in the lower range of the average precision for categories with a small number of sample images, while CaffeNet based system has more categories within the middle range of average precision values.



(a) Linear scale on horizontal axes



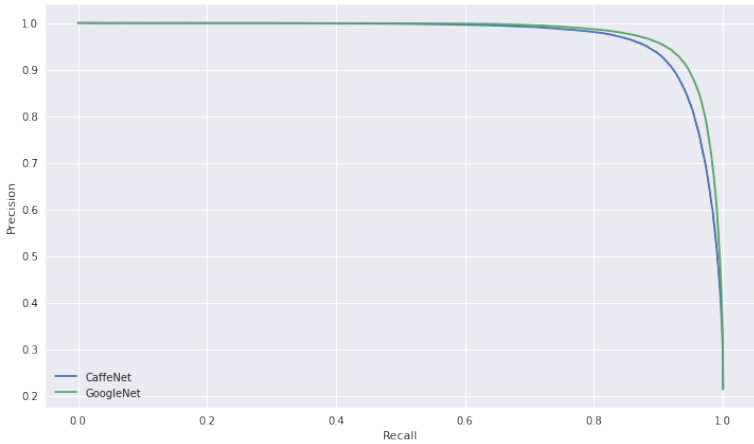
(b) Logarithmic scale on horizontal axes

Figure 26: Comparison of CaffeNet and GoogleNet average precision vs category sample size for the main experiment

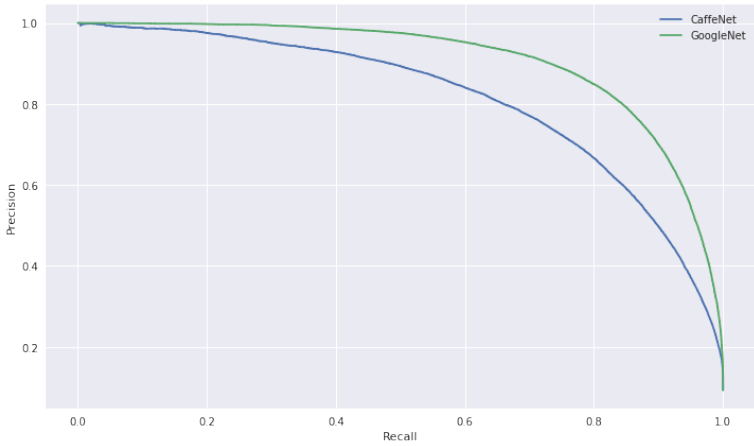
Examples of precision-recall curves for the three categories, *football*, *the-norwegian-royal-family*, and *distance-running*, are given on Figure 27. The *football category* was the category with the best average precision value for both implementations. However, GoogleNet based system has slightly higher average precision for this category, which is also represented on Figure 27a where its precision drops at a higher value of the recall level. The same pattern repeats for *the-norwegian-royal-family* category on Figure 27b. The opposite case is for *distance-running* category where CaffeNet implementation have higher precision values almost within entire recall range.

It should be mentioned that the actual system classification performance can be different than indicated by results from this section. The reason for this is a big amount of errors and inconsistency in the initial dataset described earlier. For instance, images that were correctly classified as *full-length-portrait* by GoogleNet based system but did not have this label in the NTB dataset are shown on Figure 28. Another example is the pictures shown on Figure 29 which were classified as *sign-of-triumph* by the system, but did not have the corresponding label in the NTB dataset. Since the NTB dataset was considered as a ground truth, such cases in all experiments were treated as errors of the classification system and lead to reduced average precision results for specific categories.

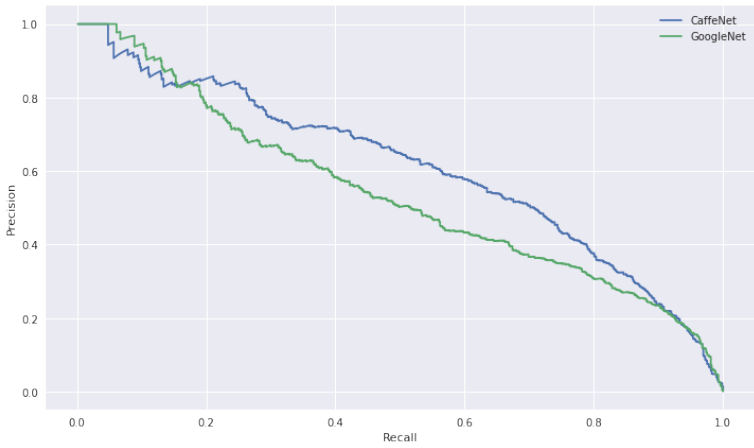
In addition, false negatives errors for some categories were investigated to get a better understanding of the reasons of the achieved results. This investigation confirmed the hypothesis about the negative influence of the contextual images on the average precision results. For example, from 279 images that were labeled as *automobile-racing* in the ground truth set but were not correctly classified to this category by the system 50 (17.9%) images were also part of the *portrait* or *full-length-portrait* category, 29 (10.4%) images were part of the *spectators* category, and 22 images were part of the *press-conferences* category; 96 (34.4%) of images in total were part of at least one of these categories.



(a) Precision-recall curve for *football* category



(b) Precision-recall curve for *the-norwegian-royal-family* category



(c) Precision-recall curve for *distance-running* category

Figure 27: Comparison of CaffeNet and GoogleNet precision-recall curves for two categories in main experiment

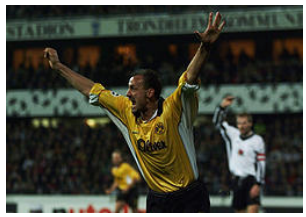
(a) Photo by Alf Ove Hansen / Scanpix

(b) Photo by Terje Bendiksby / NTB scanpix

(c) Photo by Stian Lysberg Solum / NTB scanpix

(d) Photo by Svein Ove Ekornesvåg / Scanpix

Figure 28: Examples of images classified as *full-length-portrait*, which were not labeled as such in the NTB database



(a) Photo by Cornelius Poppe / Scanpix

(b) Photo by Heiko Junge / Scanpix

(c) Photo by Stian Lysberg Solum / Scanpix

(d) Photo by Svein Ove Ekornesvåg / Scanpix

Figure 29: Examples of images classified as *sign-of-triumph*, which were not labeled as such in the NTB database

5 Discussion

The main challenges of creating a classification system based on a real-world image collection learned from this research will be presented and discussed in this chapter together with analysis on how those challenges and other factors could potentially influence final classification performance of the developed system. The main limitation of this study is that all experiments were performed on the single real-world dataset. Therefore the generalizability of all results and insights is in question. While results are considered likely to be more generalizable to datasets similar to the one used in the study, the discussion also reflects on how different challenges and observation can apply to other real-world datasets. The next sections will include descriptions of other research limitations as well as how future studies can address them and expand new knowledge even further.

5.1 Real-world dataset challenges

5.1.1 Unique set of categories

One of the main reasons to train neural networks on a real-world image collection instead of using standard datasets like ImageNet can be the unique nature of the final system purpose. A category set from the standard image database is often designed to be as general as possible since it should be suitable for wide range of applications. However, it might not be possible to solve some domain-specific tasks using such systems. The same applied to the NTB dataset employed in this research. As described in Section 4.1, this dataset was oriented towards Norway, sports, and politics and has its own, unique set of categories. The same challenge might arise for many other real-world datasets created for their particular purposes.

A unique dataset not only gives the opportunity to solve new problems, but it also brings additional challenges connected with it. The distinct nature of the dataset implies that features particular to it have to be learned by the system and can not be found in already pre-trained models. One result from this is that multi-label classification architectures that split an image into parts and turn a multi-labeling problem into a single-labeling case might perform less efficiently. The reason for this can be that these approaches depend on having a pre-trained single-label classifier for the same set of categories, which is considered unlikely to exist due to the unique nature of a dataset. More experiments and further research can validate this hypothesis. However, fine-tuning can still be applied in the case

of a real-world image collection since a number of studies showed that different pictures tend to have similar low-level features in convolutional neural networks [28, 2]. Another study revealed that fine-tuning can be applied even in cases when the original dataset was trained on an entirely different set of categories [29]. This observation suggests that classification systems built on top of real-world datasets can still leverage available pre-trained models. Results obtained in this research also confirm this. However, since the study was limited to only one set of categories, more experiments should be conducted to check to which extent real-world datasets can utilize pre-trained models.

5.1.2 Category tree transformation

Categories of the dataset used in this research were organized in a tree structure. It is considered likely that another real-world dataset of this kind might also have hierarchical categories definition. It was discovered that there are challenges when it comes to the transformation of this hierarchy to a flat structure used by modern neural networks. The primary objective of such change is to maximize available metadata utilization while keep or even improve the consistency of the labeling. Depending on the original category tree, a level of coherence in both relationship between categories (such as *part-of* or *one-of*) and the labeling rules, this process can be automatized to a particular level.

One of the key decisions that have to be made during the tree transformation is which parts of the categories tree should be included, excluded, or merged. There can be three approaches: to keep all nodes of the tree, to keep only leaves, or to use a combination of this two. On the one side, knowing the child category, it is always possible to automatically derive parent label, and therefore it should not be included as a separate category. However, in some cases, parent classes can also contain images (which was the case in the dataset used in this research), and by keeping only leaves of the tree, one can potentially remove a significant part of the metadata. To maintain consistency withing the categories, in cases when particular parent label was included in the final selection, its images should be extended with all pictures of its descendants. For instance, category *trees* should include pictures of all types of trees from the hierarchy. The issue here is to automatically find, which of the categories have subclass kind of relationship, and which not. Even if the designed hierarchy contains only subclass and superclass connections between labels, in some cases the meaning of the category should also be counted in the decision. For example, in the used dataset it was two parent classes with some descendants: *dogs* (with different breeds under it) and *fruits* (with types of fruits as children). *dogs* subtree can potentially be merged into the one category since all breeds have similar visual featured that system could use to classify images into

this category. Types of fruits, on the other hand, do not have common features and therefore combining all such images can potentially decrease the precision of the end system. Desired granularity of the system can also be a factor of if the final selection should include categories from the whole subtree or only the top category with merged images as described earlier. Other research also points out on impact different category separation techniques can have on end classification performance [27].

In the current implementation, the model does not incorporate semantic meaning and dependencies of the categories. However, as described earlier, some methods utilize such of information during the training process [12]. Further studies can investigate more on how such methods apply to real-world datasets and how existing connections in the category tree can be used to improve final results.

All described observations were based on the one category tree transformation. Therefore more studies can reveal additional challenges connected with the consistency withing categories and automatization of this process. Due to many issues with a tree structure of the used dataset, it was necessary to perform the manual analysis of the hierarchy for this study.

5.1.3 Unsuitable categories

The labels tree transformation also revealed challenges connected with categories not suitable for image classification system training. Around half of the NTB categories were classified as too abstract, contextual, ambiguous or combined terms. The main challenge here is that currently it is likely to be practically impossible to automatize the process of filtering categories by this criteria. One possible solution for this could be to use such system like WordNet to identify abstract terms. However, the discovery made in this research suggest that some labels might be contextual or not depending on the particular nature and context of the dataset itself. For instance, some non-contextual categories from the Norwegian image collection could potentially be considered as such in the dataset from another country. Also in some cases, a correct judgment requires the expert knowledge about particular categories. For instance, in order to classify *nordic-combined* category as combined action, one should know that this sport is a combination of different types of skiing. There are two general suggestions on categories selection that can be derived from the discoveries and insights of the study:

1. The category name should be straightforward, unambiguous, and clear.
2. It should be possible to assign an image to this category only by its visual content.

There are two main reasons for this:

1. This will increase the chance that all manual labelers will agree on the meaning of the term which potentially might lead to better consistency in the category itself.
2. If the end system will be used to provide a search solution for users, this can potentially help users to understand the meaning of the category and apply the filter accordingly.

More challenges in categories selection can potentially be found in further research that involves other real-world datasets.

5.1.4 Contextual images

In addition to contextual categories which require extra information about an image for classification, results also indicated challenges connected with contextual images. Such images were assigned to a particular category not by their visual features only but due to additional information available for manual labler, such as an event where the picture was taken, its date and place. The problem is that modern convolutional neural networks used to build classification system can currently utilize only visual information that is available on the picture. Therefore, such images potentially can negatively impact the average precision value of the final system as well as its actual classification performance.

While it is considered to be hard to remove this kind of images from the dataset fully automatically, the study suggests a method that can be used to improve existing dataset using semi-automatic approach. The method utilizes knowledge of the connections between categories as well as available dataset metadata to filter out contextual content from the particular categories. Due to time limitation of the project, this approach was tested only during the trial experiment where images labeled as *portraits*, *coaches* and *press-conference* were removed from other categories such as *football* or *skiing* since, in the case of sports classification, images from these classes were considered to be contextual. For instance, in this case, it was considered that press-conference picture could not be assigned to the *football* category only by a visual information but also having information about the topic of this event. While this method can partially solve the issue, it still requires an in-depth knowledge of the dataset to find such connections between different categories.

A search of such correlations can also be simplified by investigating false-negative error cases of the trained on the original dataset system. Such investigation was made on some results from the main experiment where it was discovered that significant part (34.4%) of the pictures with false-negative error type from *automobile-racing* category were also labeled as *portrait*, *full-length-portrait*, *spectators*, or *press-conferences*. By applying this approach to other labels, it is possible to obtain new

knowledge of the connections between different categories that can be further employed to improve original dataset and train model again. This method can also be applied on the next level where the trained classification system itself can be used to filter out images from the dataset based on the obtained knowledge. For instance, having a relatively good classifier of *portraits* it is possible to remove even those portrait images from sports categories that were missed to be labeled as such in the original dataset by mistake.

Proposed method can help to eliminate part of the contextual images automatically. However manual filtering of the pictures can potentially give even better results. More research should be done to further check the potential of the described approach.

5.1.5 Background entities

Standard single-label datasets like ImageNet contain images where the object is clearly visible and located in the center of picture [4]. However such multi-label datasets as PASCAL VOC are trying to include "real-world" photographs where several objects placed on the image scene in different combinations [8]. As described in Section 4.1, many images in the NTB dataset contain several objects in the picture, sometimes labeled objects are small or located in the background of the photograph. The similar case can be expected for other real-world datasets, which consist of images that were not selected specifically for classification of single objects [12]. A possible solution for this problem could be to use methods that partition the image and can find an object independently of its location [1, 7, 22]. However, as discussed earlier, it might not work with specific categories for which there is no pre-trained single-label classifier. Except for the identification of this challenge, no investigation of its influence on the final system performance was performed. Therefore, further studies should be conducted to examine its impact and to discover possible solutions for it.

5.1.6 Duplicates

The study also indicates that dataset can contain visually similar images taken from one event close to each other in time, which might also apply to the other real-world datasets. The main challenge here is to reduce a possibility of two similar pictures be assigned to different sets during the dataset split process. By applying various methods of image similarity calculation, it can be possible to reduce the number of such images. Such additional image metadata like date and place can also be incorporated in this process to reduce an amount of calculations by applying the algorithms to particular sets of pictures. Similar to the previous challenge, due to time limitations of the project, this study only indicates the possibility of such issue to accrue and do not try to address it. How this issue can influence average

precision results is discussed in further sections.

5.2 Classification performance

The primary metric of the system classification performance used in the study was average precision. While this metric is widely used in the research to compare various models [8], the real system accuracy can be different from the calculated value. As described in Section 3.2.4, the average precision level is computed based on the precision-recall curve for each label. This curve represents the possible state of the system for the category based on a particular configuration. Therefore, the final system setup can be adjusted according to the specific requirements for each label to achieve desired precision-recall rate from the possible range. The average precision metric can give indications on which of the implementations more likely will give better end system accuracy than others.

Results from the study indicate a big potential of deep neural networks used to solve classification problems when trained on real-world datasets. While some might expect that categories that represent physical objects would show significantly better average precision results, experiments conducted in the study revealed that mostly sports and action categories were on the top list. Further sections discuss how different challenges and properties of the system and dataset can potentially influence the final results.

5.2.1 Duplicates

As discussed earlier, it was discovered that due to the real-world nature of the dataset there is a possibility to encounter similar images in different sets during the dataset splitting process. These duplicated images can potentially influence both real and reported system performance. For instance, the loss value reported during the validation iterations can be influenced if validation and training sets contain copies of images. In the same way, average precision reported by the calculations can increase if test set will include images visually very similar to several pictures from the training set. This issue could potentially negatively impact the validity of the results regarding the generalizability of the classification system. However, as an extra validation, it is possible to perform an additional testing process on the set of images outside of the original dataset, which could decrease chances to encounter duplicate image in the test set. Due to time limitations of the project this issue was not addressed in the study, therefore more research should be conducted to investigate the number of such images and how it can effect experiments.

5.2.2 Contextual images

Results show that removing contextual images from the categories can increase the mean average precision of the system. However, while most of the categories im-

proved their precision values, some of them received worse results. Therefore, such filtering should be applied with caution. Findings also suggest that such filtering can also influence even those labels, where training sample was not changed. Such behavior can be connected with the fact that in deep neural networks last classification layer can use same subsets of features from the previous layers to determine probabilities for different categories. Therefore, by changing sample for one class, a model can learn slightly different features which then can impact other labels. Contextual images can potentially be also the reason for systems which are training on real-world datasets be more sensitive to overfitting since neural network can start to optimize weights to this contextual images from the training set. Therefore, it can also negatively impact the generalizability of the end system. However, results also showed that in some cases trained system was able to generalize on the visually recognizable images from the category correctly, and significant part of the "false-negative" errors consisted of contextual images. Since such pictures in the study were removed both from the training and test set, the difference in average precision can also be influenced not only by changed weights of the network but also due to more consistent selection in the test set itself. Therefore, more study should be done to investigate on the level of impact of such filtering on the system performance, as well as to get insights on its nature.

5.2.3 Training sample size

Multiple results from the experiments suggest that a small increase in a sample size can give a significant increase in the average precision to a particular level after which the growth slows down. This dependency might be the main reason why sports and action categories showed the best results in the experiments since those categories were the most represented in the dataset. However, the study also indicates that there is a broad range of average precision values for similar sample sizes. This situation can be connected with the quality of pictures of the particular category regarding consistency, diversity, and representability, as well as the different level of challenge related to the automatic classification of these categories. One possible way to improve the precision for a label with a small sample size is to extend image collection with pictures from the standard datasets where such category exists. However, this can also result in the loss of the class specific properties of this particular dataset. More studies with real-world datasets of different sizes should be performed to validate the findings of this research.

5.2.4 Ground-truth errors

Additional experiments on the trained system showed that due to inconsistency and errors in the original dataset the real system classification performance could be better than the one reported by the average precision metric. Investigation showed

that in some cases system correctly classified pictures that were not labeled accordingly in the ground-truth set. However, to calculate the actual system performance and its difference between reported one, a manual check of or error cases required. Therefore this study can only give indications of such behavior of the system and more research should be done to validate them.

5.2.5 Specialized classifier

Another utilization of the categories hierarchy can be to build specific classifiers for particular subtrees. In this case, the end system will consist of two parts. The first part will be a high-level classifier which is trained on merged and more general categories. This first classifier will then call a specific classifier for a particular subtree of categories. A small example of such system was built during the trial set of experiments where the main classifier could recognize general *skiing* pictures and then could call specific skiing classifier that can distinct between different types of skiing. The classification precision of such system for the particular category will be defined as multiplication of the accuracy of the first classifier for the parent category and the accuracy of the specific classifier for the category itself. Such system can potentially have better end performance since each neural network will have smaller set of categories and therefore more resources can be utilized to find distinctive features of different categories. The main disadvantage of this approach would be increased complexity in terms of building, training, and usage of the system. The study did not compare these two strategies but rather showed a proof of concept of such system. Therefore further research could reveal more challenges and opportunities of this method.

5.2.6 Network setup

This section will discuss how different network configurations, architectures and implementations used in the research influenced overall system performance.

Network architecture

The study compared two convolutional neural network implementations: CaffeNet and GoogleNet. Since the latter model is newer than the former one and showed better results when trained on the ImageNet image collection [45], one can expect to get improved average precision values on the system based on this model. However, results indicate that the CaffeNet based system had higher mean average precision value. Results also suggest that this system worked better for categories with small sample sizes, while GoogleNet based model was better at a classification of the categories with large training sample sizes. The implementation of the system also confirmed that GoogleNet model is more resource demanding model than CaffeNet and requires more time to converge. Therefore results give an indication

that investments in modern and advanced CNN architecture do not necessarily result in higher system performance. However, the study was limited only to two CNN designs. Therefore more research with different network architectures should be done to validate these observations.

Solver algorithm

Two solver algorithms were compared during the trial experiments. The Adam solver algorithm showed significantly better results than SGD during the training process. This difference can also be interpreted as that Adam solver finds a local minimum of the loss function faster than SGD. Running neural network training with both algorithms longer potentially could give similar average precision values for all categories. However, provided results suggest that in the current environment Adam solver converges in a shorter time. Other studies also show similar results [44]. While the decision on which solver method to chose should be made in each particular case, it can be argued based on the results that Adam solver may be a good first choice due to its performance characteristics.

Image data source

While image data source should not influence classification performance directly, it can impact the network training process regarding the speed and flexibility. Two approaches for the image data source were tested in the study: Python DataLayer and LMDB file. The latter method showed significantly better results in terms of the training speed by achieving the five-times difference compared to the former one. Part of the reason for such result can be explained by the fact that LMDB stores images as a decompressed and prepossessed array of bytes, while in the Python layer implementation each image was decompressed and cropped in runtime.

It worth mentioning that all image processing in the LMDB case happens before the start of the training, therefore it requires an additional step in the dataset preparation stage. However, since all main experiments were performed on the same dataset of images, this prepossessing was performed only once, and the same database file was further employed. An additional downside of this approach is extra storage space needed for the database file itself. While LMDB employed as an image data source performs faster than the Python data layer, the latter one gives more flexibility by enabling to make changes in a picture collection in runtime. Which of the approaches to choose depends on the requirements of the particular task.

Caffe framework also supports other types of data layers including LEVELDB, HDF5 data, MemmoryData [41]. This research was limited to compare only two different image data sources which were suitable to a corresponding part of the study. Therefore, more studies should be conducted to analyze other supported

sources from various perspectives.

Dataset splitting method

Image collections which consist of real-world pictures are considered more likely to be multi-label rather than single-label for reasons such as:

- Pictures can contain several objects or people on the scene
- The category set can have multiple dimensions including color, size, type of object or action, and place. Therefore several labels might be required to describe a single picture in this multidimensional space of categories.

As described earlier, the multi-label case can lead to a challenge of splitting such dataset into tree parts: training, validation and test sets. Two dataset splitting methods were tested in the research: the priority method used during trial experiments, and the random method applied in main experiments. However, due to time limitations of the project they were compared only by their direct properties, the influence of the dataset splitting method on the end system classification performance was not investigated. While the random method has better properties when it comes to a small difference between a desired and average actual ratio, some studies suggest that other methods could give better mean average precision value of the classification [36]. Therefore further research can include more splitting approaches to investigate their impact on generalizability and performance of the system.

6 Conclusion

The study investigated on how deep convolutional neural networks can be applied to solve the multi-label image classification problem for real-world datasets. Results give insights on what challenges can arise in building such classification system, what level of image classification performance can be achieved, and which factors can influence the end results.

Challenges and issues revealed during analysis of the NTB dataset and implementation of the classification system include:

- A unique set of categories which requires either to train a new model or to reuse a pre-trained neural network with adjustments in the architecture to solve classification problems.
- Difficulties in category tree transformation connected with the decision of which categories include, exclude and merge.
- Categories which are unsuitable for automatic training purposes including contextual, abstract and ambiguous labels.
- Contextual images which can not be classified only by visual features and require additional information.
- Issues connected with the classification of objects and people located in the background of the pictures.
- Duplicates of images in the dataset which can cause issues for both training and testing processes.

Each of the discovered challenges can potentially influence the final classification performance. Solutions for some of the issues can be automatized to a particular level. Results from the study also suggest that there is a potential in the automatic method of filtering contextual images described in the research.

Results from the experiments show a big potential of using pre-trained convolutional neural networks in solving the problem of multi-label image classification on a real-world dataset. The real accuracy of the system for a particular dataset will depend on its size and the extent of issues present including those discovered in the study. However, the results chapter can give an indication of which classification system performance level can be achieved when training on a dataset similar to the one used in the research.

Results suggest that employing more modern network architecture do not necessarily improve end classification performance. However, results also indicate that

different neural networks can perform better in different conditions. The two neural network architectures tested in the research showed different classification performance for various sizes of the training sample. A decision on which network configuration to use depend on the requirements of the particular task.

The main limitation of the study is that all experiments were performed on a single dataset. Therefore the generalizability of results, findings, and insights is in question. Results are considered likely to be more generalizable to datasets similar to the one used in the study. Experiments were designed to maximize reproducibility and internal validity. However, further studies should be done to validate obtained results.

Bibliography

[1] Wei, Y., Xia, W., Lin, M., Huang, J., Ni, B., Dong, J., Zhao, Y., & Yan, S. 9 2016. HCP: A Flexible CNN Framework for Multi-Label Image Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(9), 1901–1907. URL: http://ieeexplore.ieee.org/document/7305792/, doi:10.1109/TPAMI.2015.2491929.

[2] Oquab, M., Bottou, L., Laptev, I., & Sivic, J. 6 2014. Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 1717–1724. IEEE. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6909618, doi:10.1109/CVPR.2014.222.

[3] Krizhevsky, A., Sutskever, I., & Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in neural information processing systems*, 1097–1105. Neural Information Processing Systems.

[4] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3), 211–252. URL: http://link.springer.com/10.1007/s11263-015-0816-yhttp://arxiv.org/abs/1409.0575, doi:10.1007/s11263-015-0816-y.

[5] Schmidhuber, J. 2015. Deep learning in neural networks: An overview. *Neural Networks*, 61, 85–117. URL: http://www.sciencedirect.com/science/article/pii/S0893608014002135, doi:10.1016/j.neunet.2014.09.003.

[6] Stanford University. 2016. ImageNet Large Scale Visual Recognition Challenge 2016. URL: http://image-net.org/challenges/LSVRC/2016/results.

[7] Ren, S., He, K., Girshick, R., & Sun, J. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in neural information processing systems*, 91–99. Neural Information Processing Systems.

[8] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. 6 2010. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2), 303–338. URL: http://link.springer.com/10.1007/s11263-009-0275-4, doi:10.1007/s11263-009-0275-4.

[9] Gong, Y., Jia, Y., Leung, T., Toshev, A., & Ioffe, S. 12 2013. Deep Convolutional Ranking for Multilabel Image Annotation. *arXiv preprint arXiv:1312.4894*. URL: http://arxiv.org/abs/1312.4894.

[10] Chatfield, K., Simonyan, K., Vedaldi, A., & Zisserman, A. 5 2014. Return of the Devil in the Details: Delving Deep into Convolutional Nets. *arXiv preprint arXiv:1405.3531*. URL: http://arxiv.org/abs/1405.3531.

[11] Chua, T.-S., Tang, J., Hong, R., Li, H., Luo, Z., & Zheng, Y. 2009. NUS-WIDE: A Real-World Web Image Database from National University of Singapore. In *Proceedings of the ACM international conference on image and video retrieval*, 48. ACM.

[12] Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., & Xu, W. 2016. CNN-RNN: A Unified Framework for Multi-Label Image Classification. URL: http://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Wang_CNN-RNN_A_Unified_CVPR_2016_paper.html.

[13] Karpathy, A. CS231n Convolutional Neural Networks for Visual Recognition. Neural Networks Part 3. URL: http://cs231n.github.io/neural-networks-1/.

[14] Nair, V. & Hinton, G. E. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, 807–814. ICML. URL: https://pdfs.semanticscholar.org/a538/b05ebb01a40323997629e171c91aa28b8e2f.pdf.

[15] Karpathy, A. CS231n Convolutional Neural Networks for Visual Recognition. Optimization: Stochastic Gradient Descent. URL: http://cs231n.github.io/optimization-1/.

[16] Karpathy, A. CS231n Convolutional Neural Networks for Visual Recognition. Backpropagation, Intuitions. URL: http://cs231n.github.io/optimization-2/.

[17] Karpathy, A. CS231n Convolutional Neural Networks for Visual Recognition. Convolutional Neural Networks. URL: http://cs231n.github.io/convolutional-networks/.

[18] Zeiler, M. D. & Fergus, R. 2014. Visualizing and Understanding Convolutional Networks. In *European conference on computer vision*, 818–833. Springer International Publishing. URL: http://link.springer.com/10.1007/978-3-319-10590-1_53, doi:10.1007/978-3-319-10590-1{_}53.

[19] Karpathy, A. CS231n Convolutional Neural Networks for Visual Recognition. Classification. URL: http://cs231n.github.io/classification/.

[20] Karpathy, A. CS231n Convolutional Neural Networks for Visual Recognition. Neural Networks Part 1. URL: http://cs231n.github.io/neural-networks-3/.

[21] Ng, A. 2016. Nuts and bolts of building AI applications using Deep Learning. In *NIPS*. NIPS. URL: https://media.nips.cc/Conferences/2016/Slides/6203-Slides.pdf.

[22] Yang, H., Zhou, J. T., Zhang, Y., Gao, B.-B., Wu, J., & Cai, J. 4 2016. Exploit Bounding Box Annotations for Multi-label Object Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 280–288. IEEE. URL: http://arxiv.org/abs/1504.05843.

[23] Cheng, M.-M., Zhang, Z., Lin, W.-Y., & Torr, P. 2014. BING: Binarized Normed Gradients for Objectness Estimation at 300fps.

[24] Zitnick, C. L. & Dollár, P. 2014. Edge Boxes: Locating Object Proposals from Edges. In *European Conference on Computer Vision*, 391–405. Springer International Publishing. URL: http://link.springer.com/10.1007/978-3-319-10602-1_26, doi:10.1007/978-3-319-10602-1{_}26.

[25] Girshick, R., Donahue, J., Darrell, T., & Malik, J. 6 2014. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 580–587. IEEE. URL: http://ieeexplore.ieee.org/document/6909475/, doi:10.1109/CVPR.2014.81.

[26] Chen, Q., Song, Z., Dong, J., Huang, Z., Hua, Y., & Yan, S. 1 2015. Contextualizing Object Detection and Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(1), 13–27. URL: http://ieeexplore.ieee.org/document/6866901/, doi:10.1109/TPAMI.2014.2343217.

[27] Dong, J., Xia, W., Chen, Q., Feng, J., Huang, Z., & Yan, S. 2013. Subcategory-Aware Object Classification. URL: http://www.cv-foundation.org/openaccess/content_cvpr_2013/html/Dong_Subcategory-Aware_Object_Classification_2013_CVPR_paper.html.

[28] Pan, S. J. & Yang, Q. 10 2010. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345–1359. URL: http://ieeexplore.ieee.org/document/5288526/, doi:10.1109/TKDE.2009.191.

[29] Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. 2014. How transferable are features in deep neural networks?

[30] Uvarov, N. 2017. Master thesis supplementary source code. URL: https://github.com/uvNikita/master-thesis-src.

[31] 11.1. pickle — Python object serialization — Python 2.7.13 documentation. URL: https://docs.python.org/2/library/pickle.html.

[32] Felsenstein, J. The Newick tree format. URL: http://evolution.genetics.washington.edu/phylip/newicktree.html.

[33] Huerta-Cepas, J., Serra, F., & Bork, P. 6 2016. ETE 3: Reconstruction, Analysis, and Visualization of Phylogenomic Data. *Molecular Biology and Evolution*, 33(6), 1635–1638. URL: https://academic.oup.com/mbe/article-lookup/doi/10.1093/molbev/msw046http://mbe.oxfordjournals.org/lookup/doi/10.1093/molbev/msw046, doi:10.1093/molbev/msw046.

[34] Fellbaum, C., Fellbaum, & Christiane. 11 2012. WordNet. In *The Encyclopedia of Applied Linguistics*. John Wiley & Sons, Inc., Hoboken, NJ, USA. URL: http://doi.wiley.com/10.1002/9781405198431.wbeal1285, doi:10.1002/9781405198431.wbeal1285.

[35] Hedenfalk, M. LMDB: Lightning Memory-Mapped Database Manager (LMDB). URL: http://lmdb.tech/doc/.

[36] Sechidis, K., Tsoumakas, G., & Vlahavas, I. 2011. On the Stratification of Multi-label Data. In *Machine Learning and Knowledge Discovery in Databases*, 145–158. Springer, Berlin, Heidelberg. URL: http://link.springer.com/10.1007/978-3-642-23808-6_10, doi:10.1007/978-3-642-23808-6{_}10.

[37] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., & Darrell, T. 2014. Caffe. In *Proceedings of the ACM International Conference on Multimedia - MM '14*, 675–678, New York, New York, USA. ACM Press. URL: http://dl.acm.org/citation.cfm?doid=2647868.2654889, doi:10.1145/2647868.2654889.

[38] Caffe | Interfaces. URL: http://caffe.berkeleyvision.org/tutorial/interfaces.html.

[39] Berkeley Vision and Learning Center. 2015. BVLC CaffeNet Model. URL: https://github.com/BVLC/caffe/tree/master/models/bvlc_reference_caffenet.

[40] Berkeley Vision and Learning Center. Caffe | Model Zoo. URL: http://caffe.berkeleyvision.org/model_zoo.html.

[41] Caffe | Layer Catalogue. URL: http://caffe.berkeleyvision.org/tutorial/layers.html.

[42] Caffe | Sigmoid Cross-Entropy Loss Layer. URL: http://caffe.berkeleyvision.org/tutorial/layers/sigmoidcrossentropyloss.html.

[43] Bottou, L. 2012. Stochastic Gradient Descent Tricks. URL: https://www.microsoft.com/en-us/research/publication/stochastic-gradient-tricks/?from=http%3A%2F%2Fresearch.microsoft.com%2Fpubs%2F192769%2Ftricks-2012.pdf.

[44] Kingma, D. P. & Ba, J. L. 12 2015. Adam: a Method for Stochastic Optimization. *International Conference on Learning Representations 2015*. URL: http://arxiv.org/abs/1412.6980.

[45] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. 2015. Going Deeper with Convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–9. IEEE.

[46] NVIDIA. Parallel Programming and Computing Platform | CUDA | NVIDIA. URL: http://www.nvidia.com/object/cuda_home_new.html.

[47] NumPy — NumPy. URL: http://www.numpy.org/.

[48] Python Data Analysis Library — pandas: Python Data Analysis Library. URL: http://pandas.pydata.org/.

[49] Project Jupyter | Home. URL: http://jupyter.org/.

[50] Welcome to Vagga's documentation! — Vagga 0.7.2 documentation. URL: https://vagga.readthedocs.io/en/latest/.

[51] 2016. namespaces(7) - Linux manual page. URL: http://man7.org/linux/man-pages/man7/namespaces.7.html.

A Signed contract with NTB

NTBscanpix

LICENSE AGREEMENT

Norsk Telegrambyrå AS (NTB), Postbox 6817 St Olavs plass, 0130 OSLO and **Norges teknisk-naturvitenskapelige universitet i Gjøvik** (NTNU), Avdeling for informatikk og medieteknikk, Postbox 191, 2802 Gjøvik.

Whereas, the student Nikita Uvarov shall - under supervision by professor Rune Hjelsvold and professor Faouzi Alaya Cheikh – write a dissertation (masteroppgave) at NTNU Gjøvik in relation to Image Classification with use of Deep Learning.
Therefore, the parties agree the following:

1. GRANT OF RIGHTS
NTB grants NTNU a non-exclusive, non-transferrable right for Uvarov to access to NTB's downloadable, max preview-resolution pictures in NTBs image database, including a right for Uvarov to publish and present his (master) dissertation with no time limit. NTB commit to approve a minimum amount of 100 images for Uvarov to use in the master thesis and presentation.

Uvarov may during the term of this agreement, and the process of writing his dissertation, save pictures and metadata from NTB's database to his personal computer. NTNU is responsible for applying the necessary safety measures to avoid any misuse of the material. All material should be saved in a password protected folder. The personal computer should likewise not be accessed by third parties by remote or direct access to the computer.

NTNU shall mark all NTB material included in the dissertation, including an acknowledgment of the photographer's name. NTNU is responsible for clearing all rights related to the depicted in the material. NTNU is not allowed to process / manipulate material without the permission of NTB, except for smaller format changes and ./ or retouching in a way that does not alter the material content or character.

All rights other than those expressly granted above, are retained by NTB. NTNU does not acquire any copyright to NTB's material. NTB is granted a right to use all results, codes, material etc. which is related to the dissertation without any restrictions.

2. TERM AND TERMINATION
The duration of this agreement is from signature to August 1, 2017, upon which NTNU by Nikita Uvarov shall delete all material from NTB and make sure that the images are deleted from his personal computer.

If NTNU fails to comply with any of the safety measures of this agreement, this agreement shall automatically be terminated. NTNU shall compensate NTB for any claim, loss, cost or the like related to breach of the agreement.

3. CHOICE OF LAW AND LEGAL VENUE

This Agreement shall be applied in accordance with Norwegian law. Oslo City court is the agreed legal venue.

Oslo, 21.11.2016 — Oslo, 21.11.2016

For NTB — **For NTNU**

Norsk Telegrambyrå AS (NTB), Postbox 6817 St Olavs plass, 0130 OSLO 1