Immaterialist Fashion - Report

Mick van Hulst Dennis Verheijden Roel van der burg Brian Westerweel
Joost Besseling

May 29, 2018

1 Introduction

This report describes the results for the first competition of the course Machine Learning in Practice. The source code for this project can be found by going to the public Github repository.

During the project each of the team members were assigned specific tasks. Do take note that our approach can be considered agile as we switched roles constantly and helped each other out where we could:

- **Mick van Hulst:** Exploratory data analysis, GCP, Batch generator, implementing ResNet
- **Dennis Verheijen:** Implementing networks, general code cleaning, weight determination
- **Roel van der Burg:** Exploratory data analysis, implementing ResNet
- **Brian Westerweel:** Implementing custom F1-measure, Object-detection, Report
- **Joost Besseling:** Object-detection, Ensemble algorithm

2 Problem statement

We have chosen the immaterialist fashion challenge on Kaggle. This challenge consisted of a data set of approximately 1.1 million images which had to be assigned one or multiple labels. In total there are 228 labels that could be assigned of which most don't make sense for the human mind.

3 Data set

Kaggle provided a training, validation and test set, combined they consist of approximately 1.1 million images. For each image there is a list containing one or more labels. As mentioned earlier for most of the labels it is unclear why images have a certain label (figure 1).

Figure 1: Example images for label 24.

When looking at the individual images label 24 could either be a shirt, necklace, skirt or bracelet. However, apparently there is a more general feature in this images that is represented by label 24 which we couldn't think of.

Another important characteristic of this data set is that it is quite imbalanced (figure 2). There are around 20 labels which occur more than a 100.000 times each while most of the other labels have roughly 30.000 occurrences.

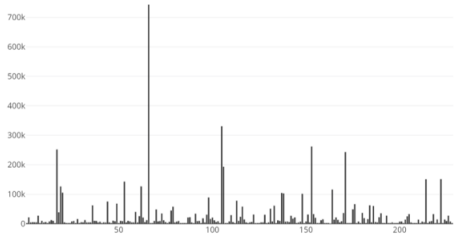

Figure 2: Data distribution, each value on the x-axis representing the label and on the y-axis are the occurences for the labels.

The imbalance in the data set posed the challenge of properly training our network by making sure that it doesn't label all images with the most occurring label. This can, for example, be done by setting weights to the different classes so incorrectly labeling a rare label is more heavily penalized than incorrectly labeling a very common label.

Data Preprocessing

The data for this challenge didn't need a lot of sanitizing. However, the images were of different sizes and because convolutional neural networks need a static input size we needed to make sure that all images that were fed to the network had the same size. This is accomplished by implementing a patch generator that cut same-sized patches from the source images.

It has to be noted that due to the gigantic disc size that was needed for the data we decided to download the images at 85% of the available quality.

4 Models

During the project multiple approaches have been tried to come up with the final solution. First we

Method	F1-measure (micro-averaged)
Feature Based Approach v1	0.60
Feature Based Approach v2	0.56
Convolutional Neural Network	0.49
Vanilla LSTM	0.52
Bidirectional LSTM	0.93

Table 1: Summary of the achieved results, the F1-measure scores are computed using Kaggle.

implemented an Inception network and trained it on the training set to establish a baseline. After that we trained Inception on a subset of the training data to learn the labels that have more than 100.000 occurrences and on a subset to train the labels that have less than 100.000 occurrences. Besides that, we tried to create a model based on object detected features.

For the loss of all models binary cross entropy is used, because the model has to output a probability for each label. Based on the probability thresholds are determined for each label on when the label actually is there and when not. So after applying the thresholds the output is a list with for the labels associated with an image an '1' and the labels not belonging for that image an '0'.

4.1 Object detection

For the object detection model we first predicted objects for each image using a pre-trained ResNet50 [1] trained on the open image dataset [2]. For each image we used the found objects as features for that image and used those features to train another neurel network.

However, because the object detection network very often detected just a few objects. Because the object detection network resulted in very small features per image we found very early on that this network would not yield any noteworthy results. Therefore, this model has not been further developed.

4.2 ResNet50

The ResNet we used is based on Keras' version [1], where the last layer was unfrozen and fine-tuned to our target data set.

TODO: MORE EXPLANATION + RESULTS

4.3 MobileNet

MobileNet again is based on a version of Keras [3]. In this case the last 5 layers are unfrozen and trained on the training images of this challenge.

TODO: MORE EXPLANATION + RESULTS

4.4 InceptionV3

The InceptionV3 used in this research is based on the pre-trained network from Keras [4]. Here the last two layers where unfrozen and fine-tuned to the target images.

TODO: MORE EXPLANATION + RESULTS

5 Ensembling

TODO: MORE EXPLANATION + RESULTS

6 Reflection

This section briefly discusses some things that stood out during this project.

References

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

[2] Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Shahab Kamali, Matteo Malloci, Jordi Pont-Tuset, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://storage.googleapis.com/openimages/web/index.html*, 2017.

[3] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[4] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.