Università degli studi di
Milano-Bicocca

Advanced Machine Learning

Final Project

Image auto-orientation

Authors:

Matteo Campironi - 801850 - m.campironi@campus.unimib.it
Serena Di Maggio - 821063 - s.dimaggio2@campus.unimib.it

January 21, 2021

Abstract

In the last years, deep learning has developed more and more and in particular convolutional neural networks are widely used in computer vision for various task, such as image and video recognition or classification. This work addresses the problem of image orientation detection by trying to determine the correct angle between 0°, 90°, 180° and 270° of a set of collected images. Indeed, this is a fundamental aspect for any field that deals with images and can not be ignored. To solve this problem initially one model has been trained from scratch, but better performances have been obtained thanks to transfer learning on three different classical networks, VGG16, MobileNetV2 and DenseNet201.

1 Introduction

Image orientation detection is a task that can easily be performed by human beings. On the other hand, automatically recognizing the orientation of a picture is a challenge in the field of computer vision and digital image processing. Nowadays a huge number of pictures are produced due to the proliferation of digital cameras and smartphones. However, while shooting a picture, the camera is not always held in the correct position which results in a wrong displayed image. Modern devices have built-in accelerometer and gyroscope to help to correct the rotation of a photo, but is a computationally intensive task to be done in real time. For this reason usually all the information is stored in Exif data of the image. Unfortunately this technique is not consistently applied across different applications. It may happen that the Exif data is manipulated or deleted across different software and therefore the image may be displayed in its original state. Even though any angle is possible it is common to assume that the images have been taken in one of the four orientations: 0°, 90°, 180°, 270°. Since it requires an high level of scene understanding, humans use object recognition and contextual scene information in order to correctly orient images. An extensive study shows that the accuracy of a human observer is close to 98% when using all available semantic cues from high-resolution images [1]. Almost all imaging applications require that images are correctly oriented before processing and heavily rely on this assumption. Since the manual correction of picture orientation is a tedious and time-consuming activity, this work is focused on providing a solution for this problem, based on convolutional neural networks.

2 Datasets

In theory, to train a network to predict the correct orientation is required a dataset of images annotated with how much their rotation angle vary from the upright orientation. To collect such a dataset the API provided by flickr were used. Flickr is a popular photo-sharing and hosting service, as well as an online community, with advanced and powerful features. Since the goal was to build a model that can recognize the orientation of a typical customer photo, the dataset had to be as generic as possible. In order to do that the pictures were sampled from the most popular tags of all time and of the week [2]. Due to the API and computational limitations it was possible to download only 2000 images per tag. After that, the correct orientation of each image was manually checked and all pictures with the watermark of the author were removed, to avoid that the network could exploit this kind of information. There also were ambiguous images where it was hard or impossible to tell the correct orientation, like a symmetrical plate of pasta seen from above, so they were removed from the dataset. During the cleaning process it was noticed that the tag *sports* contained mostly images of cars. To prevent such an important category from being deleted it was decided to sample 2000 images from some of the 25 world's most popular sports [3].

The final dataset consists of 37910 unique images from 27 categories (as shown in Figure 1) , then it was split into training (80%) and test (20%), while the validation set was extracted from the training set (20%). Finally all the possible rotations were added artificially to each of the set to obtain a balanced dataset, for a total of 151.640 pictures of various resolution.

airplane	drink	night
animals	fish	park
beach	flower	people
bike	food	pets
boat	home	portrait
book	insect	road sign
car	landscape	street
city	music	sunset
dance	nature	**sports**

sports

athletics	boxing	handball	swimming
badminton	cricket	hockey	table tennis
baseball	football	MMA	tennis
basketball	golf	rugby	volleyball

Figure 1: List of the chosen tags to build the dataset.

2.1 Padding and data augmentation

Since the dataset is composed of images with various resolution it was decided to pad the pictures as necessary with black pixel, in order for the input to be a square. Cropping and resizing were other two possible solution, but the former could have removed an important part of the picture, while the latter could have changed the aspect ratio too much. Some examples can be seen in the figure below.

To reduce overfitting data augmentation was performed. Using *ImageDataGenerator* random transformations to input samples were applied during network training. Keras offers a little choice on what kind of manipulation can be done on an image, mostly based on rotating and cropping the pictures which were not feasible for this work. Hence it was decided to use `brightness_range` $= (0.1, 0.9)$, which changes the brightness of the image of a value randomly drawn from the given range and `channel_shift_range` $=$ 150.0, which shifts the channels using a value extracted randomly.

Figure 2: Some images from the dataset.

2.2 Test datasets

To test the performances of the models on pictures different from the ones from flickr, some known dataset were chosen and all the possible rotations were added artificially.

- **Pascal VOC 2012** [4]: this dataset contains the data from train-validation set of the 2012 PASCAL Visual Object Classes Challenge.
- **SUN2012** [5]: famous dataset from MIT which counts 16.873 images taken in various scenes.
- **INRIA Holidays** [6]: the Holidays dataset is a set of images which mainly contains some personal holidays photos. with a large variety of scene types (natural, man-made, water and fire effects, etc).

3 The Methodological Approach

3.1 Custom Model

The first model proposed for image orientation detection has been built from scratch. The idea was to develop a simple neural network, with a restricted number of layer due to the limited resources available. Indeed, even training a not so deep model can require a lot of time and memory and has high computational costs.

The structure of this custom model is illustrated in the following table:

Table 1: Structure of the custom model.

Layer	Number of Filters/Units	Kernel or Pool Size	Activation Function
Conv2D	64	3x3	ReLU
Conv2D	64	3x3	ReLU
MaxPooling2D		2x2	
BatchNormalization			
Conv2D	128	3x3	ReLU
Conv2D	128	3x3	ReLU
MaxPooling2D		2x2	
Dropout (0.5)			
Dense	4		Softmax

The choice of this configuration is derived from the various attempts made, given the innumerable possible combinations to create a convolutional neural network. This is the one with the best results achieved. However, the performance of the model can not be considered optimal. For this reason, a new approach to the problem has been adopted, transfer learning, which takes into account pre-trained models and exploits their stored information to obtain better results.

3.2 Transfer Learning

State-of-the-art models available by Keras are different and numerous. Only three of them have been selected, with the intention of using the deeper ones to obtain the best results, while the more compact networks could be used on low performance devices.

For transfer learning, the following CNNs were chosen:

- **VGG16**, the famous model consisting of stacked convolutional layers with 3x3 kernel sized filters, that won the ILSVRC-2014 challenge [7];
- **MobileNetV2**, a neural network that runs efficiently on mobile devices and it is characterized by 17 bottleneck blocks, each one containing 1x1 expansion layer, 3x3 depthwise separable convolutional layer and 1x1 projection layer; this model combines the efficiency of the classical version of MobileNet with the advantages of the residual connections, typical of ResNet [8];
- **DenseNet201**, so called because it contains 201 layers grouped in dense blocks with transition layers between them; its peculiarity is that every layer has access to all its preceding feature maps, for a total of L(L-1)/2 connections (L = number of layers), and instead of adding the output of the previous layer to the following, such as in ResNet, it concatenates them [9].

In the first two cases, the final dense layers were excluded when loading the models and were replaced by a fully connected layer, with 128 neurons and activation function `Relu`, followed by the classification layer with `softmax` activation function and output dimension 4, according to the different class orientations considered. Both these layers were regularized with l1-norm and lambda parameter 1e-4. Moreover, it was also added a `dropout` layer before and after the fully connected one, with rate respectively 0.7 and 0.5, in order to regularize the networks. Also in the case of DenseNet201 the loading of the model did not include the final dense layer, but then it was replaced by a `GlobalAveragePooling2D` layer with size 2x2, followed by a dropout layer with rate 0.7 and the classification layer, the same as above.

These models were all pre-trained on the ImageNet dataset, traditionally used in the annual Large Scale Visual Recognition Challenge.

3.3 Settings

While building the models, some parameters has been configured:

- `loss function`: categorical cross entropy;
- `metric`: accuracy;
- `optimizer`: adam.

Considering that the dataset is balanced, the use of accuracy as metric appears as a valid choice. The learning rate of the optimizer has been modulated by the `PolynomialDecay` schedule, decreasing from an initial learning rate of 1e-04 to an end learning rate of 2e-07 in 10000 decay steps [10]. Another fundamental parameter is the preprocessing function of ImageDataGenerator, that has been applied to the images of input of all the models used for transfer learning. It was set using the `preprocess_input` method provided by Keras for each of the correspondent network. For the custom model, instead, the rescale parameter has been used; it has been set to 1./255 and applied after all other transformations, rescaling inputs into the $[0, 1]$ range. Finally, all the network has been trained for 20 epochs and two different callbacks available by Keras were employed: `EarlyStopping`, in order to stop training the model when the validation loss did not improve for two successive epochs (`patience` = 2); `ModelCheckpoint`, used to save the model that has achieved the best performance on the validation loss during the epochs. After some testing it was noticed that loading pre-trained weights on ImageNet, and then training the network end-to-end resulted in the best validation performance.

4 Results and Evaluation

In this section some experimental results of the models proposed are reported. For each of the four networks the plots of the accuracy and loss are shown and all the results are summarized in Table 2, 3, 4 and 5. The dotted vertical line refers to when the early stopping occurred.

4.1 Custom model

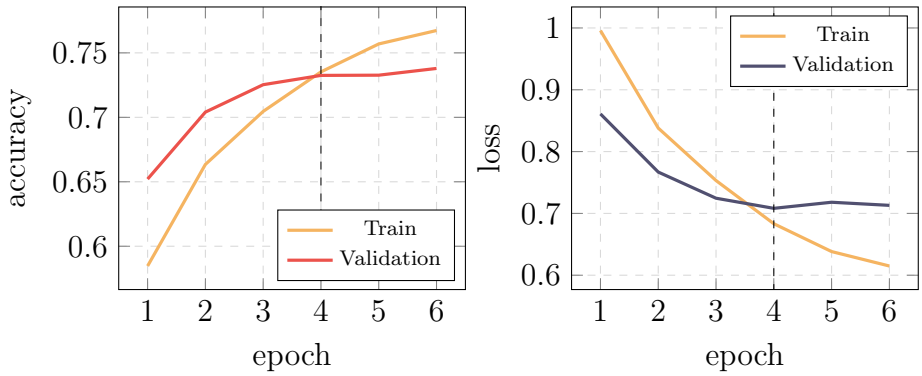

Figure 3: Plots of accuracy and loss per epoch of our custom model.

accuracy (%)	validation accuracy (%)	loss	validation loss
73.53	73.25	0.6833	0.7082

Table 2: Results for our custom model.

4.2 VGG16

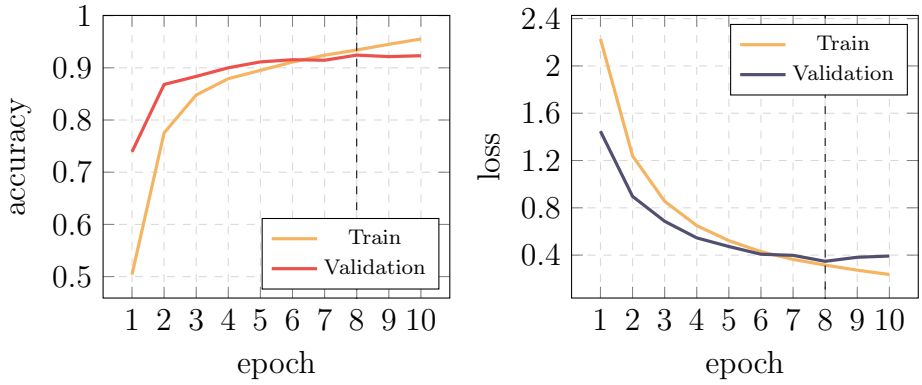

Figure 4: Plots of accuracy and loss per epoch of VGG16.

accuracy (%)	validation accuracy (%)	loss	validation loss
93.42	92.43	0.3156	0.3477

Table 3: Results for VGG16.

4.3 MobileNetV2

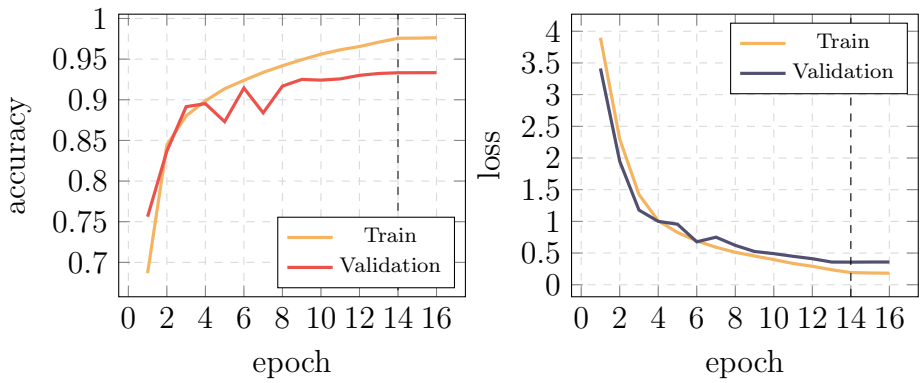

Figure 5: Plots of accuracy and loss per epoch of MobileNetV2.

accuracy (%)	validation accuracy (%)	loss	validation loss
97.56	93.32	0.1923	0.3568

Table 4: Results for MobileNetV2.

4.4 DenseNet201

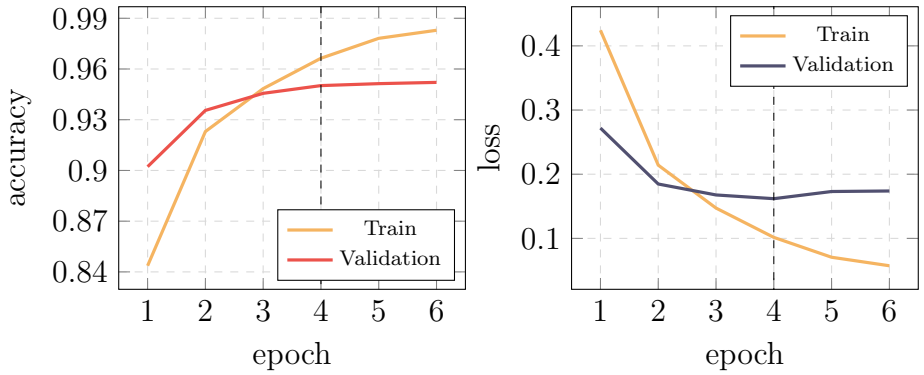

Figure 6: Plots of accuracy and loss per epoch of DenseNet201.

accuracy (%)	validation accuracy (%)	loss	validation loss
96.64	95.02	0.1016	0.1619

Table 5: Results for DenseNet201.

4.5 Test results

The following table summarizes the results obtained by the four models on the test sets proposed in 2.2.

Table 6: Accuracy (%) on different test sets of the four models.

	Flickr (ours)	VOC 2012	SUN2012	INRIA Holidays
Custom model	74.18	69.10	69.96	71.40
VGG16	94.57	92.71	91.86	88.21
MobileNetV2	94.53	92.48	92.03	87.58
DenseNet201	**96.55**	**95.02**	**94.82**	**91.25**

5 Discussion

Although the restricted number of layers, the custom model is able to reach an accuracy of 73% and shows no signs of overfitting thanks to the early stopping implementation. On the other hand transfer learning produced much better results, likely because of the pre-trained weights on ImageNet, whose great variety of images has allowed to cover those present in the train set, and also because of the deeper structure of the models. Comparing the values of the accuracy between the training set and the validation set, it is evident that MobileNetV2 suffers of overfitting, even though its performances are great. Slightly better results are obtained by VGG16, whose training was stopped at the 8th epoch and do not present particular problem. After training and validating all the models, DenseNet201 can be considered the network with best performance while the custom model gets the worst results. DenseNet201 always outperforms the other models, showing a more suitable structure for the detection of the image orientation. By looking at the loss function plot one can notice that the validation loss quickly reaches a plateau and starts increasing at epoch 4. Since the shape of the curve may suggest an high learning rate, different types of learning rate schedules offered by Keras were tested, without a noticeable change. Another important consideration could be made: regardless of the model, the highest values of accuracy are always obtained for the Flickr test set. This could be explained by the fact that the types of the images here are very similar to the ones used to train and validate the model. The drop in performance on the INRIA dataset may be due to some images that have an orientation angle which does not fall in any

of the 4 categories. However, the very good performances obtained on Pascal VOC 2012 and SUN2012 show that the models are able to generalize well on a different kind of images.

In the future, some improvements can be made. For example, implementing a custom generator could give the possibility of using a bigger variety of data augmentation techniques, while building a custom learning rate function could be useful to help reach even better performances. With more computational power the dataset size could be enlarged and some hyperparameter optimization could be done.

6 Conclusions

Automatic image orientation detection is a relevant and not straightforward problem. In this work some well known deep learning structures have been modified in order to detect the correct orientations of the pictures. Experimental results on several datasets shows that the proposed approach works very well on this task. In particular, the transfer learning technique is the key that leads to the best performances, as demonstrated by the quantitative results in Section 4. Disadvantages of deep neural networks is that to reach better generalization, large datasets are required and making the network deeper would result in additional necessary computational power which is not always desirable. Future works will focus on model optimization and generalization by getting more data on categories that were not explored.

References

[1] G. Ciocca, C. Cusano, and R. Schettini, "Image orientation detection using lbp-based features and logistic regression," *Multimedia Tools and Applications*, vol. 74, no. 9, pp. 3013–3034, May 2015. [Online]. Available: https://doi.org/10.1007/s11042-013-1766-4

[2] Tags - flickr. [Online]. Available: https://www.flickr.com/photos/tags

[3] 25 world's most popular sports (ranked by 13 factors). [Online]. Available: https://www.totalsportek.com/popular-sports/

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[5] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," pp. 3485–3492, 2010.

[6] Inria holidays dataset. [Online]. Available: http://lear.inrialpes.fr/people/jegou/data.phpholidays

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[8] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2019.

[9] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2018.

[10] P. Fischer, A. Dosovitskiy, and T. Brox, "Image orientation estimation with convolutional networks," vol. 9358, 10 2015, pp. 368–378.