AIT Deep Learning Project work 2021 Spring
Image Classification of Sport Images

Nóra Almási ALMASI.NORA@SIMONYI.BME.HU
Budapest, Hungary
Budapest University of Technology and Economics

Barnabás Börcsök BORCSOK.BARNABAS@SIMONYI.BME.HU
Budapest, Hungary
Budapest University of Technology and Economics

1. Introduction

For the 2021 Spring semester of the AIT Deep Learning course we chose the project work of Image Classification of Sport Images. The problem is as follows: train a deep neural network, that tells for a given a picture what sport is being played by the people present. We gathered a dataset of 14,000 images, and then trained a Convolutional Neural Network to identify 22 different sports based on these images.

2. Previous solutions

As we searched Kaggle for datasets, we saw that our proposed problem was solved by others before. Some of them used pretrained nets like ImageNet, others just mixed the techniques that were also taught in this course. We decided to create a convolutional neural network solution from scratch rather than fine-tuning a pre-trained network.

3. Dataset

We downloaded multiple datasets from different sources. This Kaggle dataset served as the backbone of our collected images: `https://www.kaggle.com/ponrajsubramaniian/sportclassificationdataset`. We uploaded the collected data to the project's GitHub repository: `https://github.com/bobarna/dl-sport-detection/tree/master/data`.

We collected images of **22 different sports**, amounting more than **14 000 different images**. We separated the images into folders, based on the class of the given pictures. The list of supported sports is shown in Figure 4.

When we separated the train, validation and testing data, we paid attention not to mix them, and to keep each of these subsets representative of the whole dataset,

4. Proposed method

We propose a Convolutional Neural Network (CNN) for this classification problem. An overview of our architecture is illustrated in Figure 1. After 2 iterations of convolutions, the tensors get flattened. The flattened tensors are connected to a dense layer of 128 neurons, and then a classifier layer outputs a distribution over the available classes via a softmax function. We utilize max pooling after the convolutional layers, as well as dropout layers to reduce overfitting. Table 1 gives a detailed summary of the model's layers and parameters.

The data preparation pipeline is shown in Fig. 2. As our dataset contains images of various sizes, we had to resize the images to a suitable common size. After trying out various sizes, our proposed architecture works with images of 64×64 pixels, and 3 color channels. Resizing all the images happened

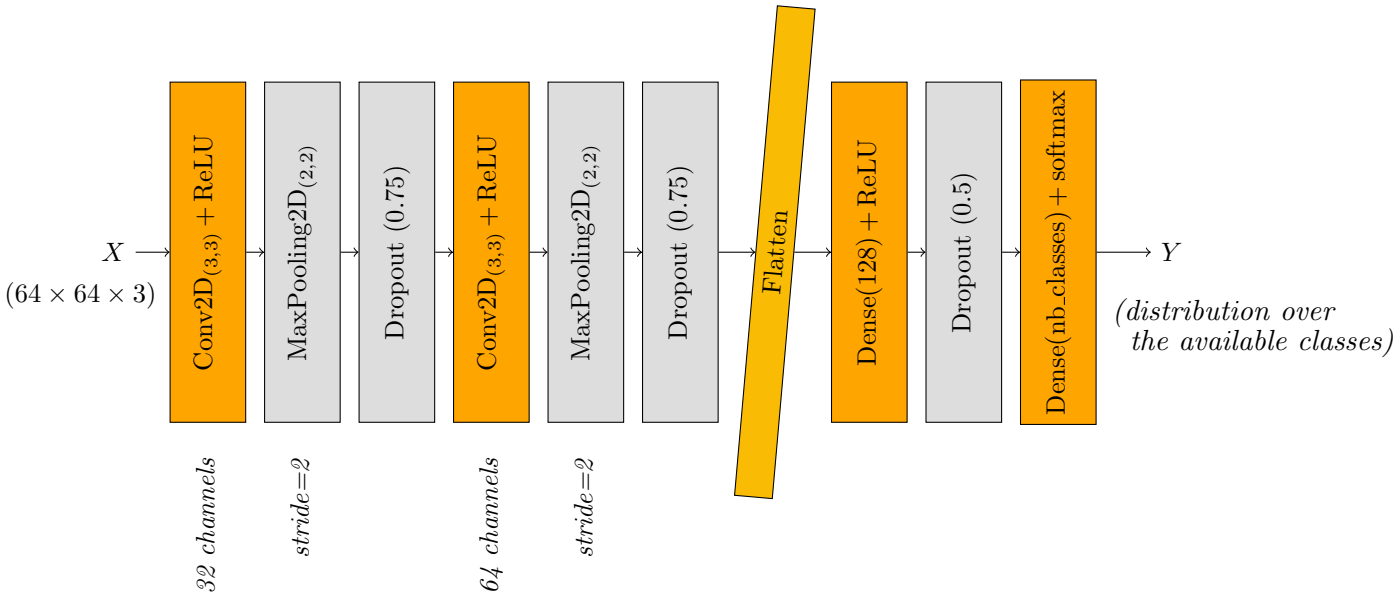

Figure 1: An illustration of our proposed method's architecture.

Layer (type)	Output Shape	Number of parameters
Conv2D	(batch_size, IMG_SIZE, IMG_SIZE, 32)	896
MaxPooling2D	(batch_size, 31, 31, 32)	0
Dropout	(batch_size, 31, 31, 32)	0
Conv2D	(batch_size, 29, 29, 64)	18,496
MaxPooling2D	(batch_size, 14, 14, 64)	0
Dropout	(batch_size, 14, 14, 64)	0
Flatten	(batch_size, 12544)	0
Dense	(batch_size, 128)	1,605,760
Dropout	(batch_size, 128)	0
Dense	(batch_size, nr_classes)	2,838
Total params: 1,627.990		
Trainable params: 1,627,990		
Non-trainable params: 0		

Table 1: Model summary

with the OpenCV library, with a bicubic interpolation over 4x4 pixel neighborhoods. These values make up the X tensors that are the inputs to the network. Then, they all get a corresponding one hot encoded vector Y that has all zeros, except for the index corresponding to the given picture's sport.

For training the model, we chose the categorical crossentropy loss function, as this is suited in multi-class classification tasks. In our classification problem the input can belong only to one category, and the network must decide which one. For optimizer, we chose the ADAM algorithm.

Our network takes in a (batch of) tensor(s) of size $(62, 62, 3)$ and outputs a vector, with a length that equals to the number of classes we would like to identify.

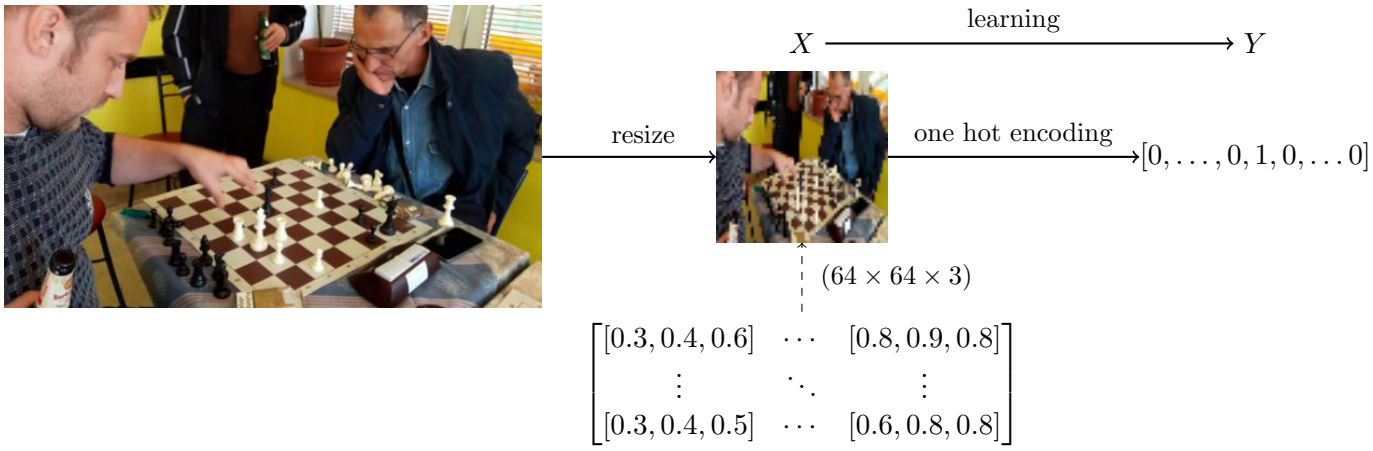

Figure 2: Our pipeline for creating the learning data.

5. Evaluation method

For training the model, we settled on a batch size of 64. The first training validation accuracy achieved was 50.87% after 76 epochs. Figure 3 shows the evolution of the loss and accuracy during the training. As seen on the above graphs, overfitting is noticeable. Early stopping (with a patience of 10 epochs) had to stop the training after 86 epochs.

After several additional training iterations (saving and loading the model again), we managed to get a validation accuracy of 67.24%, which resulted in a 68.38% test accuracy. We investigated the results to point out weaknesses and chances for improvement, see the next section for details.

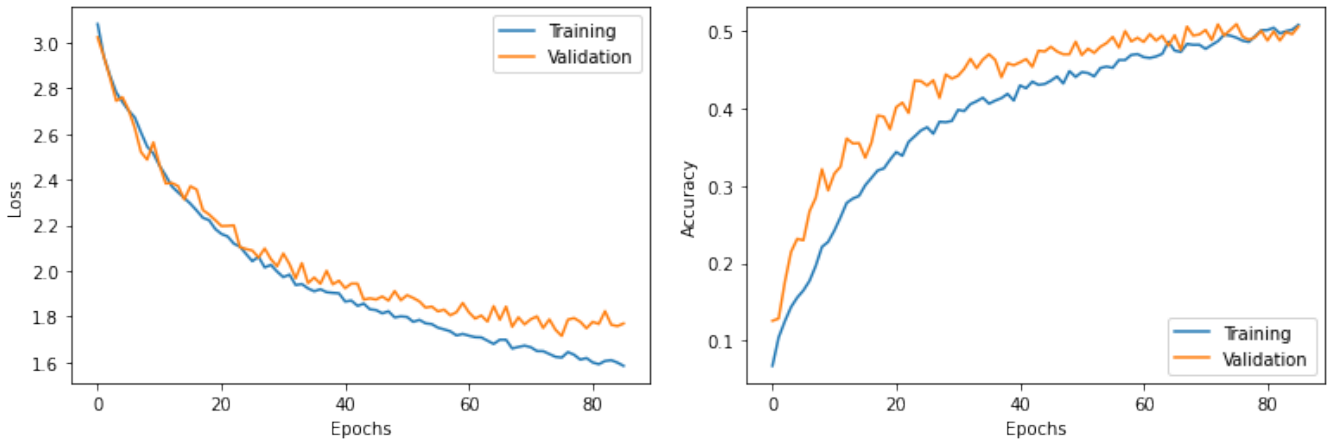

Figure 3: Loss (left) and Accuracy (right) while training.

6. Results

To understand the behaviour of our model, we evaluated the confusion matrix shown in Fig.4. We can see in the main diagonal that the choices were basically appropriate, there were some weak spots though. The varying in the diagonal colours is also due to the inequality of sport class image subset sizes. To point out weaknesses it is sensible to check the frequent missclassification types. On one hand, we can see a

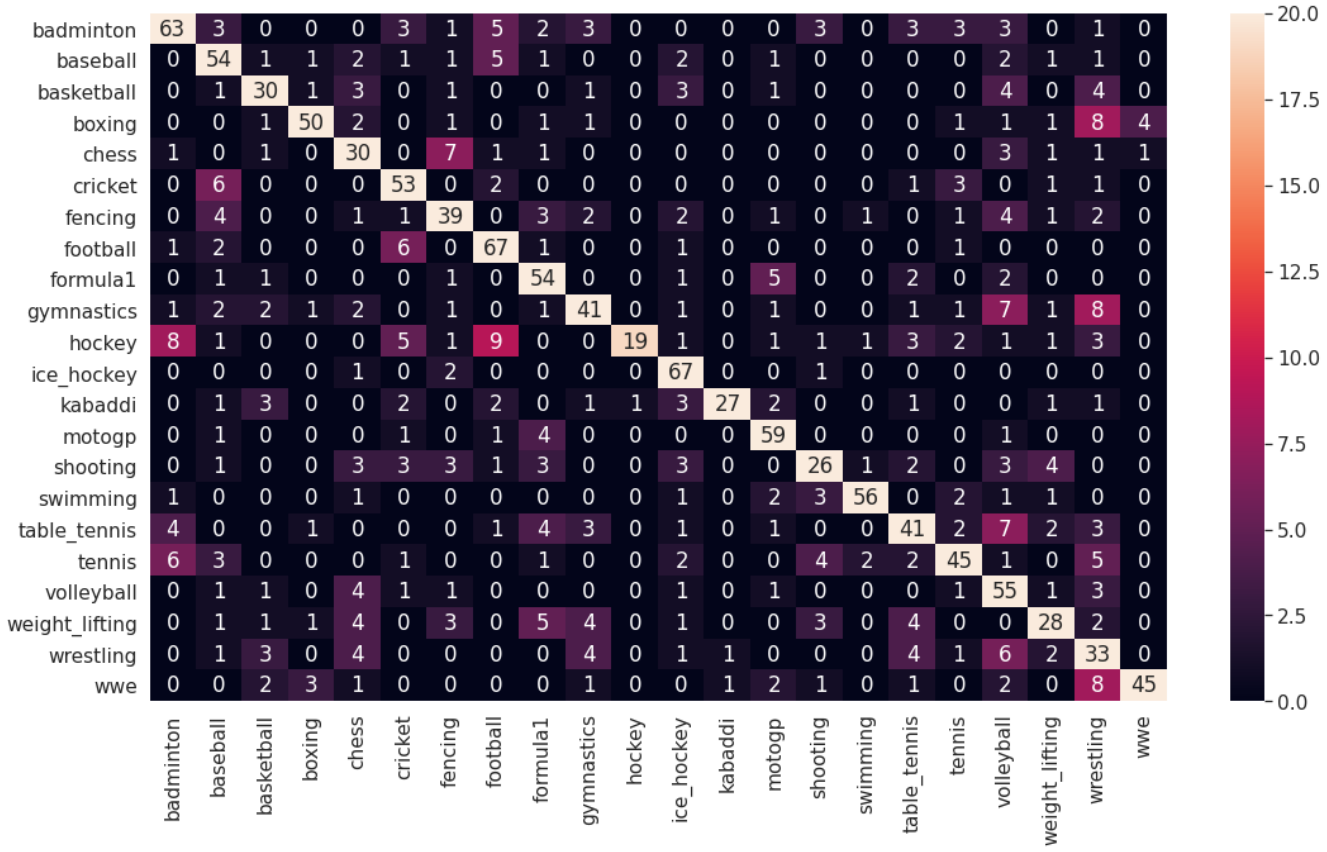

Figure 4: Confusion matrix of the solution with 68.38% test accuracy.

high number of mistakes in mixing up football with hockey, baseball with cricket or motogp with formula1. These are quite similar sports even for the human eye – a sport with cars, or one with teams running on grass, quite hard to distinguish. Therefore we cannot really hope to eliminate these faults. Maybe with a larger example set it could be improved. The majority of faults lie in this type, however, in some cases quite different sports are mixed up. Confusion of fencing with chess or volleyball with table tennis occurs (possibly because of the common features like 2 people facing each other or the net). One reason for that could be that 22 categories may be too much for this amount of samples. The other explanation is that the dataset is a little unclear. Volleyball pictures, for example, were taken through the whole match, not only when the ball was moving. We could have chosen images which really represent the active part of sports. However, it would have caused a model that can classify a narrower range of pictures, as it is trained on easy images. Instead, we could do the following to better our results. When the prediction is made, we are given a probability for each image class. Rather than choosing the maximum valuse, we could detect, whether the model is unsure about the decision. When there are 2 possible classes with a similar probability, they could be further investigated with another model. As choosing from 2 classes is way easier than choosing from 22, it could lead to an improvement.

7. Discussion

This report proposed a solution for a sports image classification problems. We described our dataset and the trained network in detail. We achieved a 68.38% test accuracy. Then we explained the results, pointed out weaknesses, and proposed chances for further improvement.