TU Graz Data Team
iMaterialist Challenge (Furniture) at FGVC5e

Authors / Verfasser: Adrian Spataru, Max Deineko, Aleksandar Bobić

Overview

Project Scope

The goal of the iMaterialist kaggle challenge was to classify furniture and home goods in photos automatically. The main obstacles of the challenge were that the images of the same object could have been taken from different angles, different objects could have been highly similar to each other, there might have been different lighting in the pictures and much more. The challenge was organized by the organizers of the FGVC5 workshop which is a workshop for data scientists focused on fine-grained visual categorization.

Approach

- First, the images had to be downloaded from the URLs provided by the challenge organizers and transformed into an appropriate form. Once preprocessed they were saved as thumbnails. Third, the images were passed through various pre-trained models and features were extracted from these models. Finally, the features were merged and passed through a custom model which was trained on them and used for classification.
- The final model achieved an average error of 0.18416 on the public kaggle leaderboard and 0.18138 on the private one. The model was tested in the public and private leaderboard on 30% and 70% of all the test data respectively. The conclusion was that a decent result can be achieved by using simple approaches like transfer learning. However, for a better result, the models would have to go through the process of fine-tuning.

Details on the Approach

Approach

To achieve the best solution possible with limited resources an ensemble of different pre-trained models with a custom model added on top for classification was built. This approach was chosen as it proved to be useful in past competitions of a similar kind. The Keras API was used for work with the neural networks instead of raw TensorFlow as it enabled the developers to have a faster workflow and cleaner codebase. In addition, the Jupyter notebooks tool was used for development as it provided many advantages to traditional linear programming approaches such as code blocks which could be executed in a non-linear manner and a more intuitive way of documenting code.

Detailed description of the approach

To fully understand the problem and the approaches that can be taken, further reading of previous similar kaggle challenge solutions was required. The analysis indicated that the best way of getting optimal results was to use neural networks by combining multiple large pre-trained models and later try to fine-tune them.

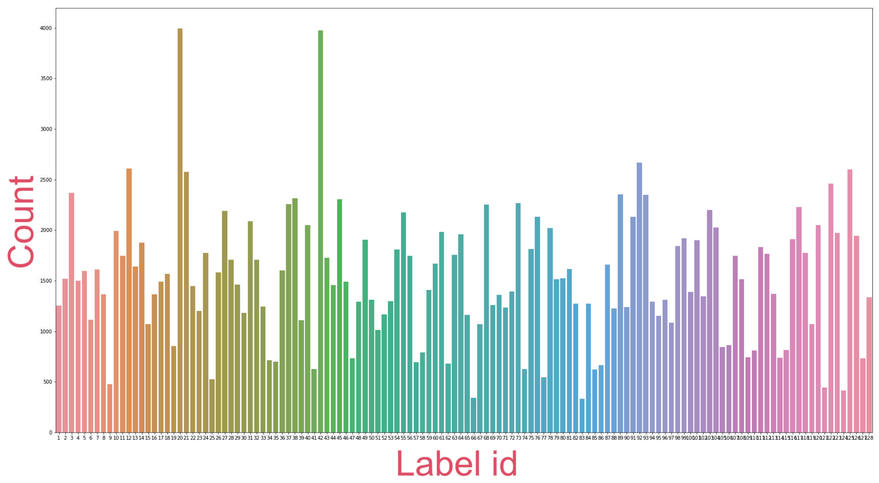
However, before creating the neural network ensemble, the data sets had to be downloaded. This resulted in a large number of images which had to be rescaled to thumbnails.

Once the images were processed, they had to be passed to different pre-trained neural network models using a process called bottlenecking. In this process, features are extracted from the models by omitting the classifying layers of the networks. These features are later used for training a custom neural network (or just a few custom layers) which is fine-tuned to the problem domain (in our case to classifying an image into one of the 128 classes).

To achieve better accuracy, we had to combine multiple models into a model ensemble. Their features were joined by weighting based on the accuracy of the single models on the validation set and passed through the custom model which was learning from them. The final solution was an ensemble of the following pre-trained models: Nasnet, ResNet50, VGG, InceptionV3 and Densenet.

The features were gathered using multiple machines and took approximately 10 hours per model. The size of the features was much smaller than the size of the primary images and could, therefore, be easily transferred to a machine where the final training was done. The custom model had an input layer, a forward connected layer and an otput layer. Due to hardware limitations, we were not able to perform fine-tuning on the models.

Label distribution



Data Set

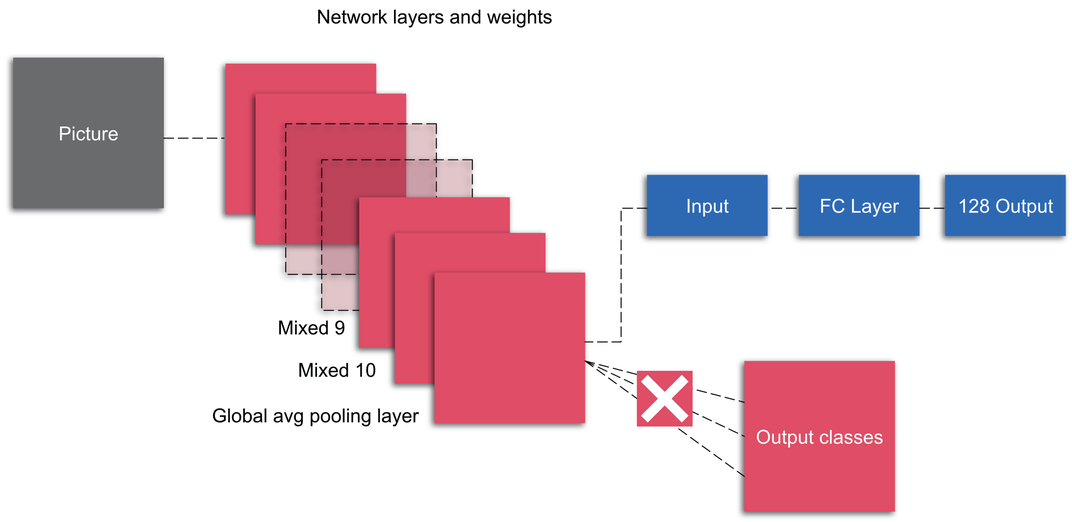
Extended Problem Statement

The data consisted of 3 separate sections and was provided the shape of JSON files. The first file was train.json contained the URLs of images from all furniture and home goods classes and a ground truth label for each image. The validation.json had the same form as the train.json. Finally, the test.json only provided the image URLs and no labels. The fact that the images were provided in the form of URLs meant that they had to be downloaded which was a long process by itself. The label names were omitted to avoid any hand-labeling attempts.

Statistics About the Data Sets

The data set consisted of 194,828 training images, 6400 validation images, and 12,800 testing images. There were 128 different classes (objects) in these images. The original data was around 30GB in size and therefore had to be downscaled to smaller thumbnails. In addition, all images were not of the same size. Thus they had to be rescaled to thumbnails of same size.

Transfer Learning - Bottlenecking



Results

The final Data team submission was an ensemble of Nasnet, ResNet50, VGG, InceptionV3 and Densenet. It had an average error of 0.18416 on the public and 0.18138 on the private leaderboard. Which was a good improvement compared to the initial submission which consisted only of Nasnet and had an average error of 0.28260 on the public board and 0.29978 on the private board. This result brought the TU Graz Data team to the 161. position among 436 contestants.

Discussion

Considering the fact that the solution was created using an ensemble of pre-trained models and did not include any fine tuning to the problem at hand, the solution is working relatively good. In the real world, this would be a useful classification tool as it outperforms any random guessing and could be used in aiding human classifiers. However, the average error could have been drastically reduced if the models were fine-tuned and additionally trained on the problems data set. Due to the hardware limitations, this was not possible as it would have required a lot of processing power from powerful GPUs. The same result could be achieved using ordinary notebooks but in a much longer timespan which would exceed the contest deadlines and would be impractical. Additionally, a good way of improving the results would be to apply different transformations (such as rotation, skewing, mirroring) on images and generate a lot of new images. This would vastly increase the data size and enable the models to be more accurate. In addition, random cropping could be used for training purposes. This would aid in avoiding overfitting and additionally help the models learn from only partial images of objects. Further, the class imbalance could have been taken into account which would have lead to less biased results. In conclusion, the results were satisfactory, and the model could be used for practical purposes.

TU Graz – Institute of Interactive Systems and Data Science
8010 Graz, Inffeldgasse 16c, Austria, Tel.: +43 (316) 873-5624
office.isds@tugraz.at, https://www.tugraz.at/institute/isds/home/