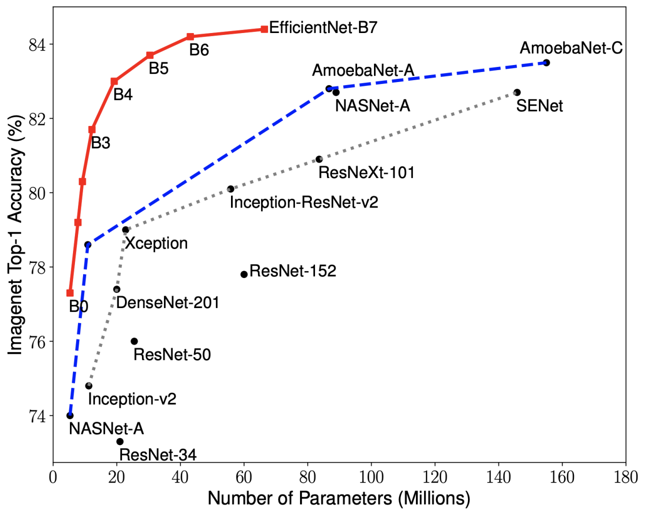
Throughout the challenge, we used a bottom-up approach starting from simpler models progressively increasing the complexity.
Since we noticed that data were equally distributed among the three classes, we selected for the validation set the same percentage of images from each class (stratified sampling). What we wanted to achieve is that the distribution of classes in the validation set reflects the distribution of classes in the test set.

Firstly, we faced the problem by building a CNN model from scratch, very similar to the one we saw in class, as a baseline model to establish a minimum model performance to which all of our other models can be compared.
In terms of image preprocessing, we applied a normalization of the input trying with different resizing from 256x256 to 512x512.
Inspired by the architecture of the VGG model, chosen because of its structure easy to understand and implement, we trained several models with different depth and hyperparameters. In order to find the best hyperparameters, we automatized the process of grid search. We used ReLU as activation function, He-uniform as kernel initializer, and RectifiedAdam as optimizer since further research showed that are the most suitable for the task. Indeed, RectifiedAdam let us to obtain a more generalizable deep neural network , completing training in fewer epochs.
Chosen the best model with respect to the performance, applying only Early stopping, the result suggested that the model is in need of more significant regularization to address its rapid overfitting. That's why we resort to additional regularization techniques such as dropout, data augmentation, weight decay, and batch normalization. We obtained the desired effects achieving 0.8333 accuracy on the test set.

At this point, we applied both transfer learning and fine-tuning, known as advanced techniques usually used for tasks where the dataset has too little data to train a very deep full-scale model from scratch.
We investigated a series of pre-trained models such as InceptionV3, InceptionResNetV2, and EfficientNet-B5, B6, and finally B7, for their astonishing results in the ImageNet challenge. For all of them, we used their respective image size and preprocessing function, replacing their top classification network.
As expected, due to the difference between our dataset and the dataset of ImageNet, the use of only transfer learning was not enough performing, pointing out the inevitable necessity of using fine-tuning in addition to transfer learning. EfficientNetB7 showed to perform better than the others, due to its brilliant accuracy-number of parameters trade-off. Here, follows some important considerations which come from our research on how to properly use fine-tuning:

- When we perform fine-tuning is critical to do it only after the model with frozen layers has been trained to convergence on our dataset. If we mix randomly-initialized trainable layers with trainable layers that hold pre-trained features, the randomly initialized layers will cause very large gradient updates during training, which will destroy our pre-trained features.

- It's also critical to use a very low learning rate, because we are training a much larger model than the first round of training, on a dataset that is not very big. As a result, we are at risk of overfitting very quickly if we apply large weight updates. Here, we only want to readapt the pre-trained weights in an incremental way. That's why we used a learning rate of 5e-4 with a ReduceOnPlateau option.
- Furthermore, the BatchNormalization layers contained in the pre-trained model need to be kept frozen. In this way, the BatchNormalization layer will run in inference mode, and will not update its mean and variance statistics.
- Lastly, in EfficientNetB7 each block needs to be all turned on or off. This is because the architecture includes a shortcut from the first layer to the last layer of each block. Not respecting blocks also significantly harms the final performance.



Comparison among EfficientNet and other popular CNN models in terms of ImageNet accuracy vs. model size (Link to the paper)