Challenge 2: Microscopy Image Challenge | sub7

MICHAEL RAFFELSBERGER

ACM Reference Format:
Michael Raffelsberger. 2022. Challenge 2: Microscopy Image Challenge | sub7. 1, 1 (June 2022), 1 page.

1 EXPLORATION

This challenge's goal is to classify microscopy images with 3 channels in a multi-class setting with 9 classes. We have 9632 labelled training samples and have to find predictions for 6869 unlabelled test samples. In the training set, the most frequent class has 2100 samples while the rarest class has 608 samples.

2 TRAIN-VALIDATION SPLIT

I performed a simple train-validation split with 80% training data ($n = 7705$) and 20% validation data ($n = 1927$). I did not create a small holdout set for a final unbiased performance estimate making the approach somewhat vulnerable to validation set overfitting.

3 FEATURE EXTRACTION

I used some pretrained models from `torchvision` solely for feature extraction without end-to-end training. This idea has the advantage that we have to pass the images through the big pretrained models just once making training very efficient. After extracting and saving the features, we can just use them to train any model capable of multiclass classification. I actually tried to use tree-based models (RF, GBM) first, but they turned out to not work very well in this setting. I made use of the following models to extract features from the images:

- **vgg19:** 512 features after applying global average pooling to the output of the `features` submodule.
- **resnet18:** 512 features using the output of the last layer before the fully-connected submodule.
- **efficientnet_b0:** 1280 features after applying global average pooling to the output of the `features` submodule.
- **vit_b_16:** 768 features using the output of the last layer before the `heads` submodule.

All images were normalized to $[0, 1]$, resized to $(h, w) = (224, 224)$ from $(h, w) = (64, 64)$ and standardized with the means and standard-deviation provided online for all pretrained `torchvision` models. Note that the features were similarly extracted the training, validation and (unlabelled) test images in the exact same manner. However, I created samples from the training set 6 times, each time with some random augmentation on the fly, including horizonal flips, vertical flips and random rotations.

4 TRAINING

To find a good model, I fit different models on the training dataset of shape $[6 \cdot 7705, 512 + 512 + 1280 + 768] = [46230, 3072]$. I tried some `scikit-learn` models first but switched to `PyTorch` after the MLP and the logistic regression performed best. In the end, I used an ensemble of three different MLPs with 1 to 3 hidden layers performing very similarly on the validation set. As loss function, I used the cross-entropy loss with class weights to alleviate the class imbalancedness. All features were standardized with the `StandardScaler`.

5 EVALUATION

To evaluate different models on the training and validation set, I primarily looked at the balanced accuracy score, which is also the final evaluation criterion for this challenge. I repeatedly experienced that the score on the submission server was around 3-4 percentage points below the score on my validation set.

6 FINAL FIT

Before predicting the unlabelled test set, I extracted the features for the combined training and validation set **6 times** again and refit three chosen model architectures from scratch. The final submission is based on the averaged probabilities predicted by the three models and yields a score of 0.853 on the public leaderboard.

7 DISCUSSION

My approach to extract features for all images a single time and then store and reuse the features in another model is facilitated by the small dataset size. It obviously has the disadvantage that we are less flexible with augmentations on the fly compared to training a network end-to-end. On the positive side, the approach allows for an extremely fast training procedure. Passing the images through the pretrained models a single time is not too time-consuming. Using the `RidgeClassifier`, which is very efficient for multiclass classification, after feature extraction would give a useful model within seconds for instance.

8 NOTE

Before sticking to the solution above, I also tried to design and train convolutional neural networks from scratch. In particular, I tried out an architecture similar to the GapNet-PL model from [1]. However, the highest leaderboard score I achieved was slightly below 0.8 which made me consider the powerful pretrained models instead. In the beginning, I was actually sceptical about them since they were not trained on microscopy images and I did not have the computational resources to fine-tune the pretrained weights. Fortunately, it turned out that they still managed to provide useful features even for novel image contents.

REFERENCES

[1] Elisabeth Rumetshofer, Markus Hofmarcher, Clemens Röhrl, Sepp Hochreiter, and Günter Klambauer. 2018. Human-level protein localization with convolutional neural networks. In *International conference on learning representations*.

Author's address: Michael Raffelsberger, k11772903@students.jku.at.