Land cover classification: Squeezing the lemon with AutoML

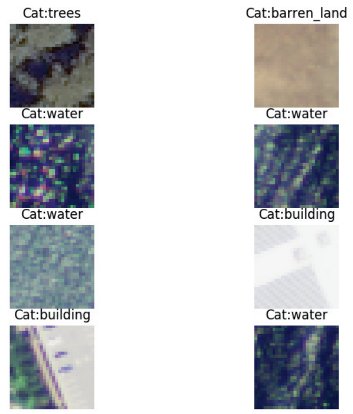
9.4.2019
Timo Grossenbacher
ZHAW CAS MAIN
Module "Deep Learning"
timo@timogrossenbacher.ch
Code: github.com/grssnbchr/ml_dl_assignment_2019

The task & data set

The goal of this assignment is to correctly predict one out of six different land cover classes given an input aerial image. The dataset at hand is the SAT-6 Airborne Dataset as available on Kaggle [0]:

> 405'000 labelled 28x28px images with 4 channels (RGB + NIR)
> Test set size: 81'000

Only 10% of the original data were retained, and only 80% of that were used as training data (25'920 images). 20% were left as hold-out validation set for preliminary tests (6480 images).



The "shallow" baseline

A "shallow" (as opposed to "deep") model, namely a Random Forest with ~1800 trees, was trained on the same training data as part of the ML course module. It reached an accuracy of 99.0% on the whole test set.

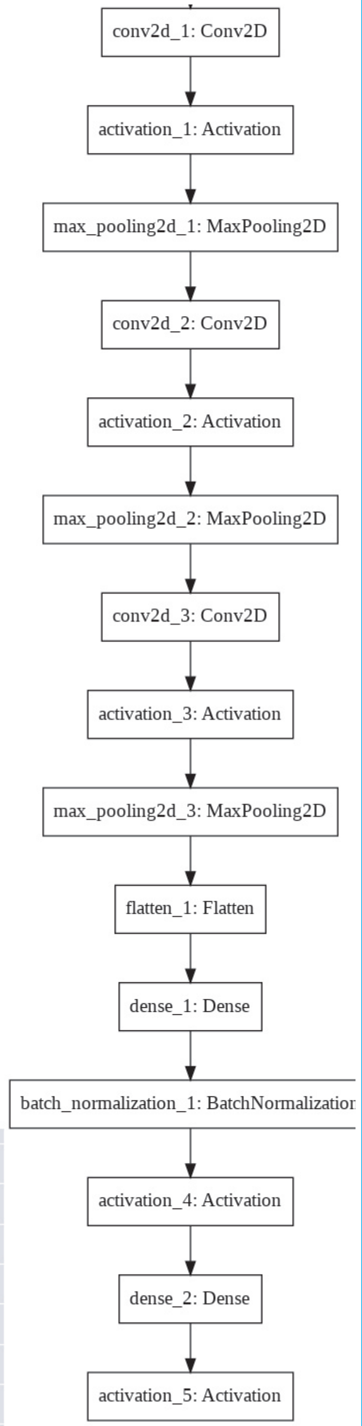
Before training, extensive preprocessing was applied to the data: A fifth channel (NDVI) was calculated, the pixel values were standardized, and the dimensionality was heavily reduced through the calcuation of mean and st. dev. of each channel.

Simple CNN approach

In the first iteration, a simple convolutional neural network with 3 convolution and 3 maxpooling layers each and a total of ~700'000 trainable parameters was trained on the training data (see architecture on the right side).

The data were fed to the model as is (no preprocessing as in baseline). The model was trained for 30 epochs, which took about 40 minutes with GPU acceleration. After this, the model reached an accuracy of 97.3% on the test set.

As can be seen from the below training history, the model suffered from overfitting early on. After around 15 epochs, the validation accuracy didn't improve anymore.



AutoML with AutoKeras

AutoML, specifically Neural Architecture Search (NAS), is an optimization technique where the best hypothesis space is searched automatically. This takes considerable resources but is easy to use, i.e. with Google's AutoML or with the open source AutoKeras package [1]:

```
autoclf = ImageClassifier( )
autoclf.fit(X_train, y_train, time_limit=3 * 60 * 60)
autoclf.final_fit(X_train, y_train, X_val, y_val, retrain=True)
```

After 2 hours of search (with max. 15 epochs per model), AutoKeras had tried out 3 different architectures. The best architecture had an accuracy of 99.2% on the whole test set – which is only slightly better than the baseline but quite a bit better than the simple CNN. However, the resulting architecture is much more complex and has ~10 times more parameters. A small part of it can be seen on the right side.

Conclusions & Learnings

1. Deep learning models for image classification are not always better than "shallow" models like Random Forests (and they take longer to train).

2. AutoML only makes sense if a) small improvements have a large impact b) enough resources (distributed GPUs) are available.

3. However, AutoKeras found a reasonably good CNN without any effort on the user-side.

References: [0] https://www.kaggle.com/crawford/deepsat-sat6, [1] Auto-Keras: Efficient Neural Architecture Search with Network Morphism (2018). Haifeng Jin, Qingquan Song, and Xia Hu.