Which whale is it, anyway?

Face recognition for right whales using deep learning

Robert Bogucki*, Marek Cygan, Maciej Klimek, Jan Kanty Milczek, Marcin Mucha
deepsense.io

***Contact Information:**
Robert Bogucki
Chief Science Officer at deepsense.io
Warsaw, Poland

Phone: +48 508 083 091
Email: `robert@deepsense.io`

Abstract
With fewer than 500 North Atlantic right whales left in the world's oceans, knowing the health and status of each whale is integral to the efforts of researchers working to protect the species from extinction. To interest the data science community, NOAA Fisheries has organized a competition hosted on Kaggle.com. The challenge was to automate the right whales recognition process using a dataset of aerial photographs of individual whales - currently a painstaking and lengthy, manual process. In the poster, we outline the winning solution. It is based on deep learning and convolutional neural networks.

Introduction

The goal of the competition was to recognize individual right whales in photographs taken during aerial surveys. There were 447 different right whales in the data set. The photographs had been taken at different times of day, and with various equipment. They ranged from very clear and centered on the animal to taken from afar and badly focused.

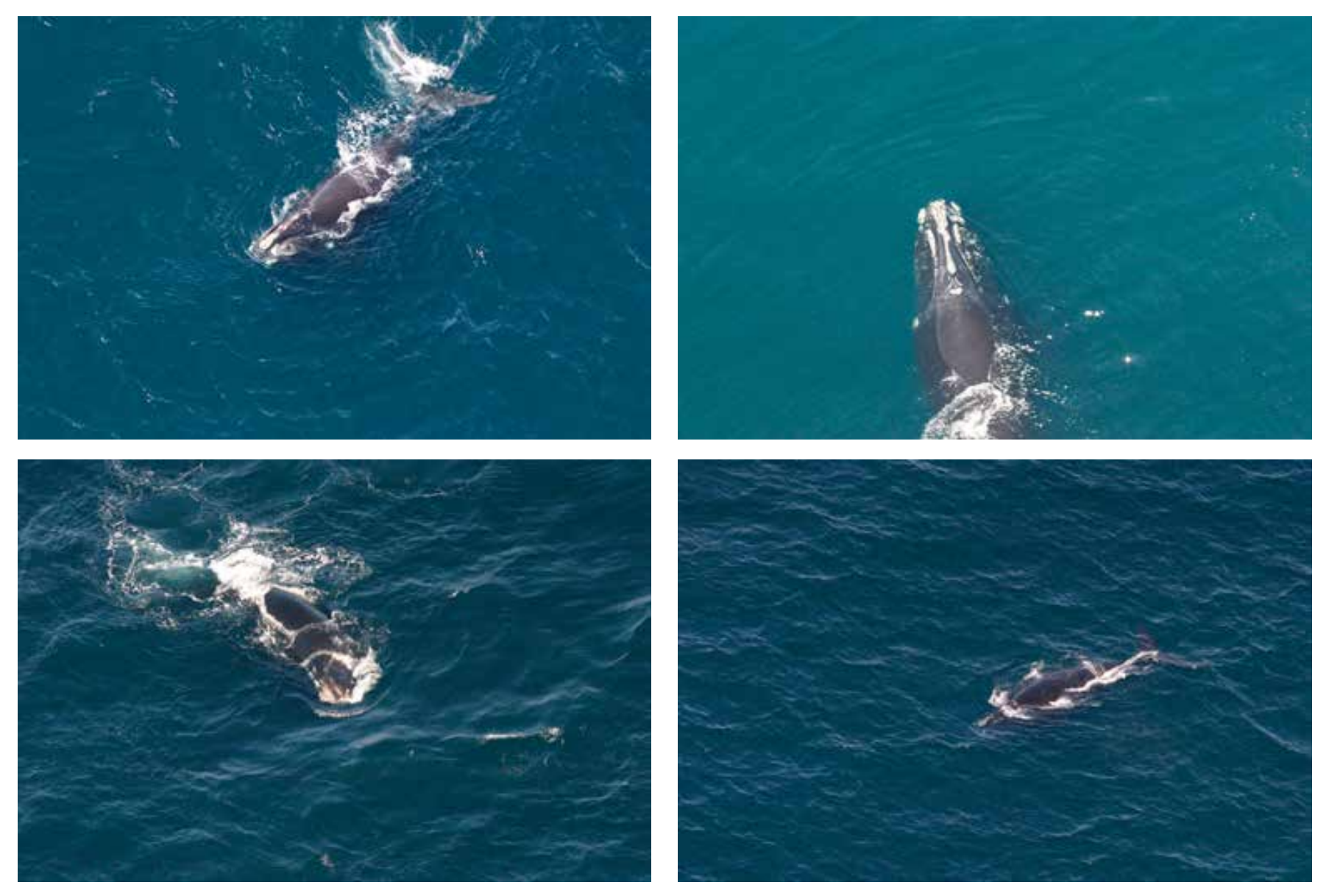

Figure 1: Sample images from the dataset

The training set consisted of 4,544 images and was not balanced. The number of pictures per whale varied a lot: the most common ones had around forty, a great deal had over a dozen, yet around twenty had just a single photo. The contestants were asked to provide a probability distribution over all 447 whales. The solutions were then judged by multiclass log loss, also known as cross-entropy loss.

Another challenge, is that images representing different classes (i.e. different whales) were very similar to each other. This is somewhat different than the situation where we try to discriminate between, say, dogs and cats. This posed some difficulties for the neural networks that we had been training — the unique characteristics that make up an individual whale, or that set this particular whale apart from others, occupy only a small portion of an image and are not very apparent. Helping out our classifiers to focus on the correct features, i.e. the whales heads and their callosity patterns, turned out to be crucial.

Outline of the solution

Convolutional neural networks (CNNs) have proven to do extraordinarily well in image recognition tasks, and our solution has used them to a great extent. It consisted of the following steps:

1. head localizer (using CNNs),
2. head aligner (using CNNs),
3. training several CNNs on passport-like photos of whales (obtained from previous steps),
4. averaging and tuning the predictions.

One additional trick that has served us well is providing the networks with some additional targets. If the additional targets depend on the part of the image that is of particular interest for us (i.e. head and callosity pattern in this case), this trick should force the network to focus on this area. Also, the networks have more stimuli to learn on, and thus have to develop more robust features, sensible for more than one task, which should limit overfitting.

To implement the solution we used Python, NumPy and Theano. To create manual annotations we employed Sloth (a universal labeling tool), as well as an ad-hoc Julia script. To train our models we used two types of Nvidia GPUs: Tesla K80 and GRID K520.

"Passport photos" of whales

Before training the final classifier, we took some preliminary steps to provide it with good quality, standarized shots of whales' heads. The general idea of this approach can be thought of as obtaining a passport photo from a random picture where the subject is in some arbitrary position. This boiled down to training a head localizer and a head aligner. The first one takes a photo and produces a bounding box around the head, still the head may be arbitrarily rotated and not necessarily in the middle of the photo. The second one takes a photo of the head and aligns and rescales it, so that the blowhead and bonnet-tip (roughly speaking, the top and bottom points of the head) are always in the same place. Both of these steps were done by training a neural network on manual annotations for the training data.

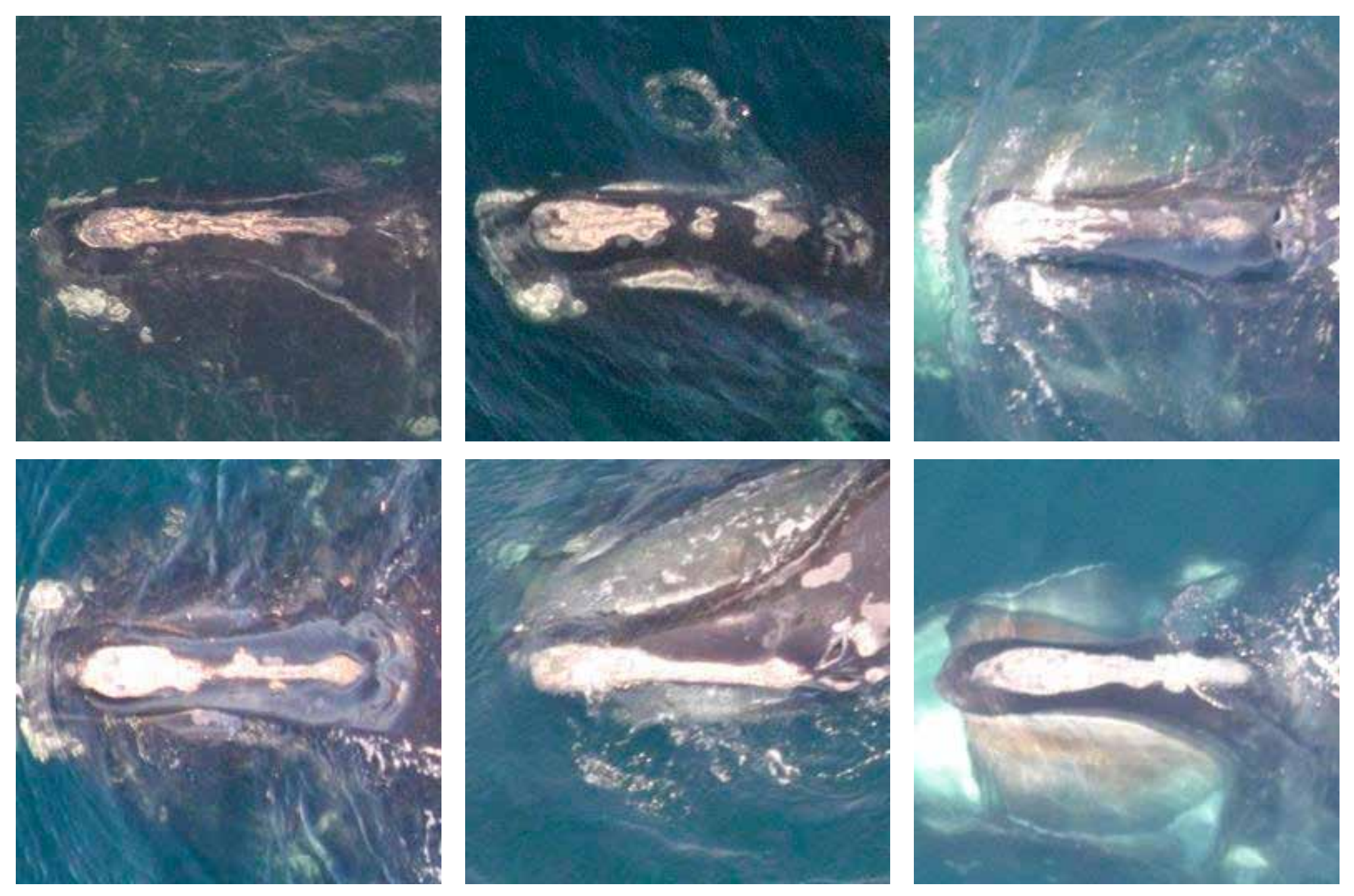

Figure 2: "Passport photos" of whales

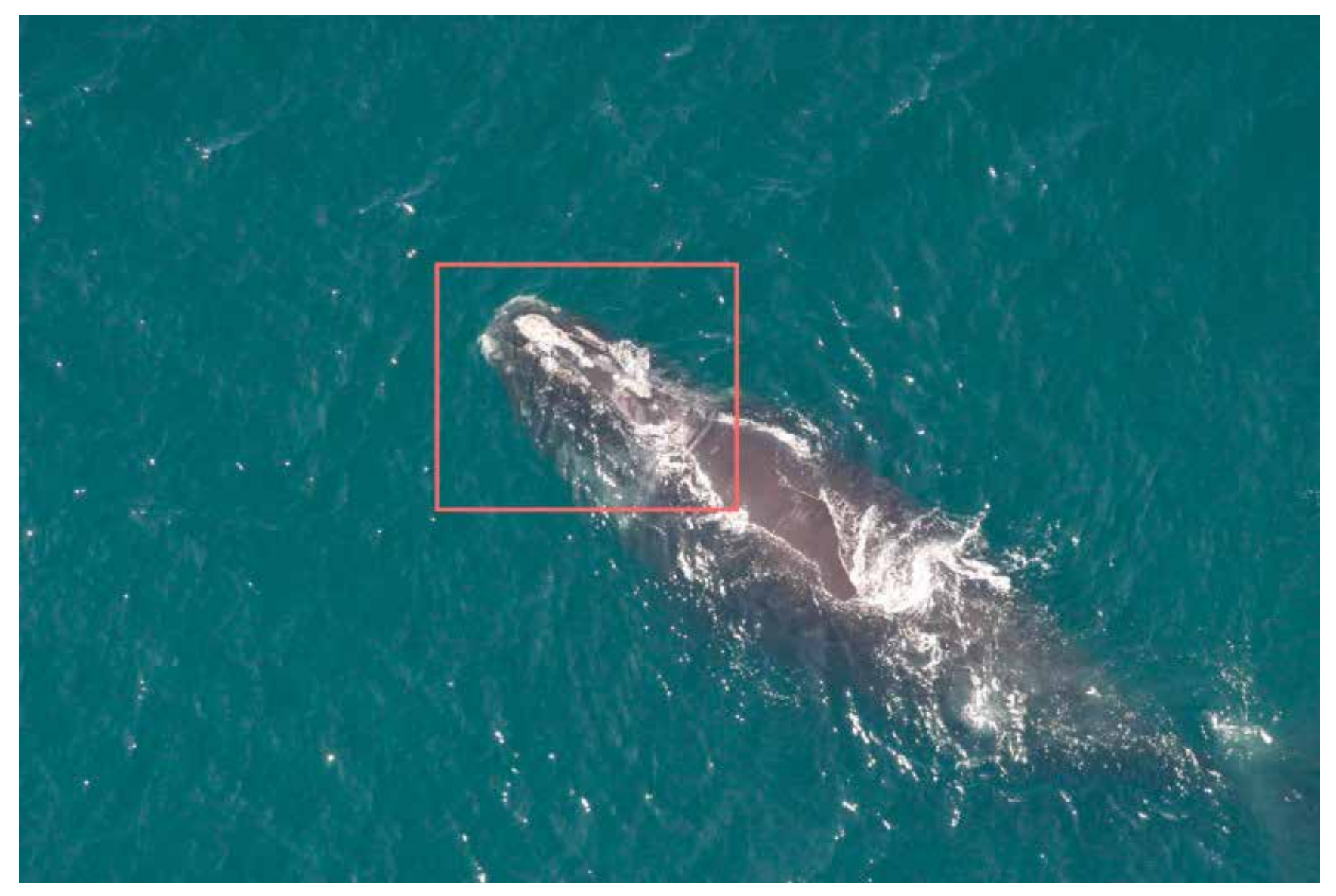

Figure 3: Output from the head localizer

Localizing the whale

We trained a CNN which takes an original image and outputs coordinates of the bounding box around the whale's head. Although this is clearly a regression task, instead of using L2 loss, we had more success with quantizing the output into bins and using Softmax together with cross-entropy loss.

Aligning the whale

The next step was to align the photos so that they all conform to the same standards. The idea was to train a CNN that estimates the coordinates of blowhead and bonnet-tip. Having them, one can easily come up with a transformation that maps the original image into one where these two points are always in the same position. Once again we proceeded to train CNN to predict quantized coordinates. Although we do not claim that it was impossible to pinpoint these points using the whole image (i.e. avoid the previous step), we now face a possibly easier task — we know the approximate location of the head.

Final classifier

Almost all of our models worked on 256×256 images and shared the same architecture. All convolutional layers had 3×3 filters and did not change the size of the image, all pooling layers were 3×3 with 2 stride (they halved the size). In addition all convolutional layers were followed by batch normalization and ReLU nonlinearity. We employed a mild data augmentation together with test-time augmentation (over 20 random augmentations). In addition, the networks had to output an additional target describing the continuity of the callosity pattern.

Regularization

For all models, we have used only L2 regularization. However, separate hyperparameters were used for convolutional and fully connected layers, with a higher value for the latter.

One should also recall that we were adding supplementary targets to the networks. This imposes additional constraints on weights, enforces more focus on the head of the whale (instead of, say, some random patterns on the water), i.e. it works against overfitting.

Training and validation

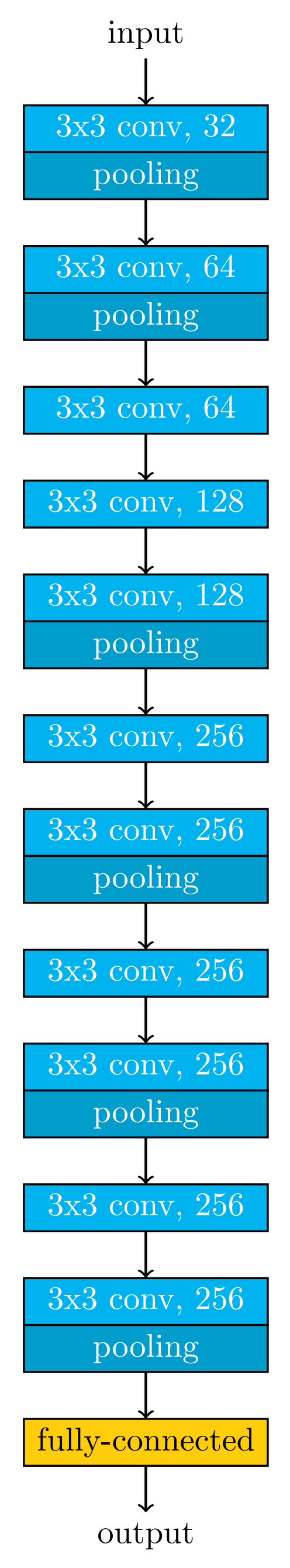

Figure 4: Main net's architecture

We trained almost all of our models with stochastic gradient descent (SGD) combined with 0.9 momentum. Usually we settled after around 500-1000 epochs. During the training process we used quite slow exponential decay on the learning rate (0.9955) and also manually adjusted the learning rate from time to time. After increasing the learning rate, the networks error went up a lot, yet it tended to settle on a lower level after a few epochs. Our best model also used another idea — it was first trained with Nesterov momentum for over a hundred epochs, and then we switched to using Adam (adaptive moment estimation). We couldn't achieve similar loss when using Adam from the very beginning. The initial learning rate may not have mattered much, but we used values around 0.0005.

We used a random 10% of the training data for validation. Moreover, after we had decided that a model is good enough to use, we proceeded to reuse the validation set for training. For most models we' added the validation set to the training one and ran it for 50-100 additional epochs. This was meant to account for a relatively small dataset and the issue of whales appearing only in our validation set.

Combining the predictions

We have ended up with a number of models scoring in a range of 0.97 to 1.3 according to our validation (actual test scores were better). We averaged them together and combined with a simple transformation of raising the predictions to a moderate power (in the final solution we used 1.45). This translated into roughly 0.1 improvement in the log loss. We have not scrutinized this enough, yet one may hypothesize that this trick worked because our fully connected layers were strongly regularized, and were reluctant to produce more extreme probabilities. The final log loss was around 0.59, which should translate into about 87% accuracy.

Acknowledgements

A special thanks to Christin Khan and NOAA for presenting the data science community with this exceptional challenge. We hope that this will inspire others. Of course, all this would be much harder without Kaggle.com, which does a great job with their platform. We would also like to thank Nvidia for giving us access to Nvidia K80 GPUs — they've done a great job.