Aquila: Learning to Predict Subconscious Image Preference

Neon Labs Inc.

May 7, 2017

1 Training

In contrast to most (nearly all) applications of deep learning, our deep convolutional neural network 'Aquila' does not learn a mapping from inputs to labels. Instead, this mapping is learned over pairs of inputs and the one-to-one labeling of inputs is learned implicitly.

The training process is explained here. Each training step requires three items: I, W, and C. I is a matrix of training images, of size $B \times W \times H \times C$, where B is the size of the minibatch and W, H, and C are the width, height and number of channels in each image, respectively. $W \in \mathbb{R}^{B \times B \times D}$ is the 'win matrix' (see below) (D is the number of demographic bins we use) and $C \in \mathbb{R}^{B \times B \times D}$ is the confidence we have in the 'win ratio' (the ratio of the number of times image i was chosen over image j divided by the total number of times i and j have been compared) for each element in the win matrix.

1.1 Win Matrix

To begin, we are given B images and a list of tuples of the form $(p_1, p_2, ..., p_m)$, where $p_k = (i, j, d, c)$ indicates that image i was chosen over image j a total of c times by individuals in demographic d, $d \in [0, D)$.

We construct the 'win matrix' W by incrementing $w_{i,j,d}$ by c for each tuple in the list.

1.2 Confidence Matrix

The confidence matrix is not currently used; all win ratios are observed are treated as equally certain. However, in reality this is not the case as a ratio obtained with 3 comparisons is clearly not as certain as a ratio obtained by many times that many comparisons. One obvious means to relate the confidence in each win ratio would be:

$$\frac{1}{1 + CI}$$

Where CI is the confidence interval of the estimate, treating the outcomes as Bernoulli events and computing the Binomial proportion confidence interval. However, this is potentially overly conservative as in many cases we not only have comparisons of images i to image j but also comparisons of i to many other

images, as well as comparisons to j to many other images, which could inform our confidence in the outcomes of the i j comparisons.

1.3 Computing Loss

Given an image matrix I, Aquila assigns each item a score (as a float value), resulting in a length-B vector $\vec{s}$. Loss is computed using this vector and the win matrix W. Here, the demographic subscripts are dropped. This procedure is not changed by the demographic dimension; it is simply iteratively applied over each demographic, and the loss is summed at the end.

First, we construct S^{Δ}:

$$S^{\Delta}_{ij} = \vec{s}_i - \vec{s}_j$$

Next the win ratios are computed:

$$W^d = \max(W + W^{\intercal}, 1) \tag{1}$$

$$W' = W / W^d \tag{2}$$

Note that the maximum operator in (1) and the division operator in (2) are element-wise operations.

The two components of loss matrix L are computed:

$$K = \max(0, S^{\Delta}) \tag{3}$$

$$L_1 = K + \log(e^{-K} + e^{S^{\Delta} - K}) \tag{4}$$

$$L_2 = C \otimes S^{\Delta} \tag{5}$$

$$L = L_1 - L_2 \tag{6}$$

Note that the maximum operator in (3) is element-wise. The $\otimes$ operator in (4) is the Hadamard product. K is computed to prevent floating point overflows if any element of $|S^{\Delta}|$ is very large. L relates the cost of a correct prediction based on the separation between the Aquila-produced scores for images i and j. This cost is zero C is 1 for all elements and if the score of every chosen image is inf and the score of every rejected image is less than inf. Of course, in some cases a loss of zero is not possible: for instance, if i beats j and j beats k: there are two distinct chosen images and two distinct rejected images but only three images total.

The final loss is computed with respect to the observed wins and then normalized based on the number of number of potential comparisons:

$$\ell(I; \theta) = \frac{2}{|W'|} \sum (L \otimes W') \tag{7}$$