MABE Task 1 – classical classification: approach

Ben Lansdell

May 2021

1 Overview

1. Generate predictions of a 1DCNN neural network, using the set of distances between all key-points as input to the network, rather than the raw positions. Predictions are made through a 5-fold cross validation prediction procedure. Denote the predicted probabilities X_{CNN}.
2. Generate a set of handcrafted features based on the MARS model [1]. Denote the handcrafted features X_{HC}.
3. Through a separate 5-fold cross validation prediction procedure, generate predictions of a set of basic ML models, trained on $[X_{CNN}, X_{HC}]$. Call the prediction of these models a new set of features, X_{ML}.
4. Train an XGB model on the features $[X_{CNN}, X_{HC}, X_{ML}]$.
5. Perform some final tweaks of the output of the XGB model to finalize predictions.

2 More details

Each point above is elaborated on here.

2.1 CNN model

The model is based on the baseline code provided by the organizers. That is, it is a CNN with the following architecture. Let a convolutional block have the following structure:

ConvBlock(ch):
Conv1D(ch, s)
BatchNorm()
ReLU()
Dropout(d)

For dropout rate $d = 0.5$, convolution kernel size $s = 5$, and number of channels ch.

Then the network as a whole has the structure:

1D CNN network
Input
BatchNorm()
ConvBlock(128)
ConvBlock(64)
ConvBlock(64)
Flatten()
Dense(n_b)
Softmax()

For n_b the number of behaviors, in this task, $n_b = 4$. Instead of inputting raw position features to the network, we used the set of distances between all keypoints being tracked (both between points within the same mouse, and points between different mice). This gives 91 features per frame, let's denote as $f_t \in \mathbb{R}^{91}$ for frame t. Frame t's input to the network is the concatenation:

$$x_t = [f_{t-T}, f_{t-T+1}, \ldots, f_t, \ldots, f_{t+T-1}, f_{t+T}] \in \mathbb{R}^{91\times(2T+1)}$$

For a window size $T = 50$.

The model was trained for 15 epochs with the Adam optimizer, with a learning rate of 0.0001, and a batch size of 128.

To be used as as a base model in a stacking procedure we generate predictions of this network in a 5-fold cross validation procedure on the entirety of the training set. Folds are generated at the video level to avoid leakage of data from a training video into the validation set. This produces 5 trained 1DCNNs, one for each fold. To run inference on the test set (for submission), the test set is pushed through each of the 5 networks, and the output probabilities are average to produce the final predictions.

Call these predicted probabilities X_{CNN}.

2.2 Hand-crafted features

The 1DCNN prediction probability features are appended to a set of hand-crafted features that may also contain information about the behaviors. These are based on the set of features used in [1]. A short description of each of these is:

- Overall centroid of each mouse (all key points)
- Centroid of just the head of each mouse (nose, ears, neck)
- Centroid of the hips of each mouse (hips, tailbase)
- Centroid of the body of each mouse (neck, hips, tailbase)
- Distance of each mouse from left/right side of cage
- Distance of each mouse from front/back side of cage
- Distance to closest side of the cage for each mouse
- Orientation (absolute angle) of centroid head to centroid hips vector of each mouse
- Orientation (absolute angle) of tail to neck vector of each mouse

- Orientation (absolute angle) of neck to nose vector of each mouse
- Angle between left ear and neck and right ear and neck vectors, for each mouse
- Major and minor axis lengths, ratio between major and minor axis lengths, and area of ellipse fit to the set of key points for each mouse. These are computed through an SVD.
- Ratio of the areas of the ellipse of resident to intruder mice
- Relative velocity of mice tangent to the vector between their overall centroids
- Relative velocity of mice perpendicular to the vector between their overall centroids
- Distance between the overall centroids, divided by the length of the major axis of the ellipse of each mouse
- Relative angle between the centroid head-¿centroid body vector and the vector connecting the overall centroids of the mice, for each mouse
- Relative angle between the nose-¿neck vector and the vector connecting the overall centroids of the mice, for each mouse
- Derived from above: indicator variable if this angle is less than 45 degrees – the mouse is looking at the other
- Intersection of union of bounding boxes of the two mice
- Kinematic features: speed and acceleration of all of the centroids of each mouse. A step size of 3 frames is used for each calculation.
- As with the CNN model, all distances between all keypoints are also computed

For all but the last two items listed above, the following procedure is used to capture something of the dynamics of the feature, at a given frame t. The feature, g_t, is computed over a window of size τ, and the minimum, maximum, mean and standard deviation of the feature value within this window are computed. These statistics are computed for window sizes $\tau = 1, 5, 10$. These statistics form the final set of numbers for that feature.

Call the set of features X_{HC}.

2.3 First level stacking

Concatenate features from the previous two sections $X_1 = [X_{CNN}, X_{HC}]$ to form a feature set. Through the same cross-validation procedure described above, generate predicted probabilities for a set of ML models on features X_1.

The models are:

- Naive Bayes classifier, applied to z-scored features
- Random forest classifier, using entropy as the loss function
- Random forest classifier, using the gini purity measure as the loss function

- Extremely randomized trees classifier, using entropy as the loss function
- Extremely randomized trees classifier, using the gini purity measure as the loss function

Parameters are otherwise kept to their default in scikit-learn (version 0.24.1).

The models' output probabilities produce a new set of 20 features (5 models $\times$ 4 behaviors). Call these features X_{ML}.

2.4 Final model predictions

The final predictions are made on the feature set $X_2 = [X_{CNN}, X_{HC}, X_{ML}]$. XGBoost is used to generate the final predictions. A random hyperparameter search using 5-fold cross validation found the following optimal parameters:

- subsample: 0.6
- min_child_weight: 1
- max_depth: 3
- gamma: 1.5
- colsample_bytree: 1

The rest of the parameters were kept as their default in the xgboost python package (1.4.1). Once these hyper-parameters have been found, the model is retrained on the entire training set.

Call the output of this model, the prediction probabilities, Y_{XGB}.

2.5 Post-ML optimizations

After this set of predictions are produced they are tweaked slightly to get the final output.

First, since the macro averaged F1 performance on only the labeled behaviors was the important metric for this task, the final predictions are based on reweighing the output probabilities to optimize this F1 score. This was a simple random parameter search over the probability simplex in $\mathbb{R}^4$ to find optimal weights w^*. Call the reweighed prediction probabilities

$$\tilde{Y}_{XGB} = w^* Y_{XGB}.$$

Second, to try to better try to capture the dynamics of the behaviors themselves (e.g. the fact that each has a typical duration), a standard, discrete state HMM model was used to infer the final behavior predictions. The observations of the HMM are the predicted behavior of the reweighed XGB model, along with the vector of reweighed predicted probabilities of each behavior, discretized. The hidden state is the true behavior. The emission and transition distributions of the HMM can then both be estimated from the training data and the XGB model predictions. Inference then gives the final set of predictions $\tilde{Y}_{HMM}$.

Both of these steps were trained on the entire training set (no cross-validation or anything).

References

[1] Cristina Segalin, Jalani Williams, Tomomi Karigo, May Hui, Moriel Zelikowsky, Jennifer J. Sun, Pietro Perona, David J. Anderson, and Ann Kennedy. The mouse action recognition system (mars): a software pipeline for automated analysis of social behaviors in mice. *bioRxiv*, 2020.