Human Activity Recognition Milestone Report 2

Menna El-Shaer

August 2020

In this second milestone report, we discuss modeling details using deep machine learning techniques. To review, our problem was given a set of 2D images of people interacting in an activity, can we estimate the locations of each person's joints which indicate that person's pose and using those predictions, can we determine the activity the person(s) in the image are participating in. The first problem is a regression problem where the inputs are the image pixels themselves and the outputs are the locations of the joints, while the second problem is a classification one that determines the activity in the image using those estimated keypoints from the regression stage. We will describe each problem separately in the following sections.

1. Regression Network (2D PoseNet)

1.1. Data Pre-processing

After the data wrangling step, we have the training and testing annotations defined in a Python dictionary data structure that can easily be read from the saved .json files. We will format this dictionary structure into a Pandas dataframe that can be conveniently used in building our deep learning models. Before using the images for training, it was necessary to normalize and scale the pixel values as inputs to our model are expected to be in [0, 1] range. We also resized all images to 256 x 256 shape; as a result, the annotated coordinates for the keypoints also needed to be rescaled to the new image size.

1.2. Network Architecture

We will utilize a pre-built network model for visual representation learning that was trained on the ImageNet-1k dataset. The choice of using a pre-trained model on the ImageNet dataset was due to the comprehensiveness of the image database. It is generally used for object detection tasks since it contains more than one million images that were manually annotated with object bounded boxes representing the detected object's identity and location. Objects' types in the database span numerous categories and can be considered general enough for most object classification tasks.

We used the ResNet50-v2 network architecture as the feature extractor layer; it outputs a 2048-dimensional vector for each input sample (image). The model we used for transfer training was the BiT-S R50x1 [1].

We added a fully connected 32-output layer to the feature extractor layer to perform the keypoints regression task.

1.3. Training

Training with the full dataset was performed under limited time and computational resources, as a result, only one epoch of training was run. In addition, no fine-tuning of the feature extraction weights or any hyperparameter tuning of the network was done. We also did not perform any data augmentation techniques on the training data such as image rotations or translations or brightness changes. We aim to explore such options in future work.

The training process was done in batches of 32 samples (images) using the Adam optimizer and the mean-squared error loss function. The resulting accuracy after one epoch was 16.45%.

We used the trained model to predict keypoints to input into the classifier network which will be used for activity identification.

2. Classification Network (ActivityNet)

2.1. Data Pre-processing

As inputs to the classification network, we will use the x and y coordinates of the keypoints resulting in 32 input features. Activities in the dataset are grouped into 20 categories with a label for each activity/category. There is a total of 410 activities which will be mapped to 410 output labels. We will however use the activity categories instead of the activities themselves to fit our network. Using the individual activities for classification will be explored in future work. The categories of the activities are originally encoded as strings in the dictionary, which we will need to re-encode as binary labels for our multi-label classifier network. Each of the 20 category classes will be one-hot encoded into a unique vector for representation in the model.

2.2. Network Architecture

We will implement a straightforward multi-layer perceptron network with an input layer of 32 nodes fully connected to three 100-node hidden layers followed by a single 20-node output layer. We will use ReLU activations for the hidden layers and a softmax activation function for the output layer. To fit the network, we use a categorical cross-entropy log loss function that is to be minimized by an Adam stochastic gradient descent optimizer.

2.3. Training Results

As with the regression network, we will train the model for one epoch (one pass through the training data) to minimize our time and computation resources. A single epoch training with the whole training dataset led to a training accuracy of 22%.

3. Discussion

The next logical step would involve some experimentation typical in deep learning modeling. First, the pre-trained weights we used for feature extraction and selection were used as is and not trained on our specific dataset. A next step would be to unfreeze the feature extraction layers and retrain using images from our dataset. We could also fine-tune and experiment with different hyperparameter values for the entire network when computing the best weights. We hypothesize this will improve our network's total prediction performance since the extracted features post training will be from the same dataset used to train the rest of the network. Second, a completely different architecture could be used for the feature selection process; we used the ResNet50 architecture in this project since it has been shown to work well on general images, but a different architecture could be better suited to images of human body poses specifically. Third, due to our constrained time and computational resources, we only trained the model for one epoch; a minimum of ten epochs or complete passes through the training data is usually done in deep learning modeling. Since training is done in an optimization manner, finding the best network parameters requires multiple passes through the dataset until convergence is reached. Consequently, multiple epoch training is necessary to ensure convergence. Finally, to prevent overfitting and ensure model generalization, evaluating the model on the test dataset and achieving a good prediction accuracy on the test data is crucial. We intend to conclude our project with model validation and performance evaluation.

References

[1] A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly and N. Houlsby: Big Transfer (BiT): General Visual Representation Learning.