Video activity classification

Kousik Rajesh

February 2020

Introduction

Task: To perform video action classification on the Breakfast actions dataset.
Link to colab notebook: Notebook link.
Link to dataset: Drive Link
Link to Github repo: Github repo

I have split the task into two subproblems task classification and activity classification although the problem statement only asks to do activity classification I have attempted both

Task classification

There are 10 breakfast items (i.e., cereals, coffee, fried egg, milk, salat, sandwich, tea, scrambled egg, pancake, juice) Each video segment consists of a person making one of these 10 breakfast items.

Classifying such a task requires one to consider the actual order of different actions. The capability of an LSTM to learn sequential patterns would be quite useful in this scenario.

My model consists of two stacked Bidirectional LSTM layers, each cell of which has a 400 length output vector. The concatenation of both the forward and backward pass output gives an 800 length output vector. Finally, max pooling is applied over the output from each LSTM cell to get a vector of size 800. A dense softmax layer follows, which outputs the softmax probability for each of the 10 classes.

Table 1: Model description

Layer (type)	Output Shape	Parameters
Bidirectional LSTM	(None, None, 800)	2563200
Bidirectional LSTM	(None, None, 800)	3843200
MaxPooling1D	(None, None, 800)	0
Dense	(None, None, 10)	8010

Table 2: Test results on given test data

Test split	Loss	Accuracy
Split 1	0.6129	0.7579
Split 2	0.6449	0.7583
Split 3	0.5142	0.7898
Split 4	0.5722	0.7847

Activity classification

We have seen the methodology to perform task classification now coming to the action classification task: There are 48 different activities as given in the mapping.txt file. To perform activity classification first I tried using a simple neural network, the reasoning being that each feature vector could be assumed to be independent of the previous and no previous information is necessary to classify any frame. Using such a network with 2 hidden layers I was able to obtain an accuracy of 87.41 %.

Then I tried an LSTM architecture as mentioned in the problem statement to see if there is any significant improvement in accuracy but did not find any.

Miscellaneous

At first glance the task classification accuracy isn't too high(approx 80%) however this is due to insufficient training time. Due to time and memory constraints on Google colab and also the painfully slow training process of bidirectional LSTM's I was able to train each training data bundle for one epoch only. Considering this I feel the accuracy is pretty high.
Also I used Keras for quick prototyping as due to ongoing exams in my institute I didn't have much time to concentrate here.