Machine Learning report: Assignment2

Kun Lu 22060598
Mengya Ding 22060581
College of Information Science and Electronic Engineering, Zhejiang University

Abstract

This article briefly introduces the methods we take in the task of *Action Recognition in Videos*, which is an assignment in the Machine Intellengence course at the full of 2020, Zhejiang University, instructored by Dr. Haoji Hu. In this short paper, we explored some possible *attention mechanisms* in the task of action recignition. We first designed two kind of attention mechanism on the time scope and space scope, which we call the temporal attention and spatial attention respectivley, and then introduce some methods to realize it. Extensive experiments on HMDB-51 dataset have shown the effectiveness of the proposed approach.

1 Introduction

Action recognition aims to recognize the true action from a given video sequence, and is a natural extension to image classification. Except from the challenges come with image classification, action recognition is faced with more complicated scenarios, unexpected motions, and long time variance. Key motions are usually done within a few frames, and the main objects in a video usually takes only part of the video, meaning that a large certain temporal and spatial information is actually useless. This invites our detailed discussion on the attention-based action recognition task.

2 Methods

In this section, we will briefly introduce the temporal and spatial attention which maight bring some benefits in the action recognition task. We starts from temporal attention, and then two kinds of spatial attention.

2.1 Temporal attention

In the real life, one clip of video might last for seconds or even minutes, but the key information are only contained in part of them. We designed a simple method to leverage the spatial attention so as to provide a time-variant weight for varied video frames.

We use a simple way to calculate the weights of each frame. Supposing $v = \{f_1, f_2, ..., f_T\}$ is a sampled video clip, we send each frame f_i into the LSTM cell. At the last timestep, the cell state c_t is believed to have a global insight into the content of a video. Thus we can use several CNN layers to produce a fixed-length weight $w_1, w_2, ..., w_T$ for each frame which was used as the input. After producing the weight template, the features of a video is represented as 1

$$\mathbf{o}_f = \sum_{t=0}^{T-1} w_i \cdot o_t \tag{1}$$

where o_t indicates the output of LSTM cell at time step t. The attention-weighted features could be used for downstream tasks, i.e, classification.

2.2 Spatial attention

2.2.1 Grid attention

Persons usually occupy only part of an image, with much useless background. Directly feeding a whole image or video to classification task are thus prone to overfit to some irrational textures. To alleviate this issue, we can also generate a self-adaptive model to guide the network into real and key components in a video.

We propose two kinds of spatial attention. First we use a weakly supervised one, which we call the grid attention. This work is inspired by show and tell (also called the *soft attention*). We use Θ to denote a feature extraction module, with its feature $\Theta(v)_{ij} \in \mathbb{R}^{H \times W \times C}$. To produce the attention weights $w_i j$, where $i = 0, 1, ..., H$ and $j = 0, 1, ..., W$ denotes the spatial importance, we need to find a task-based embedding vector as the input, denoted by e_v. In practice, this embedding vector could be a simple CNN network. Grid attention mechanism is formulation as follows 2:

$$\mathbf{o}_f = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} w_{ij} \cdot \Theta(v)_{ij} \tag{2}$$

2.2.2 Object-based attention

Though grid attention provide a simple and effective guidance, we find it not that effective as we have expected. This is probably explained by the fact that a weakly-supervised task may not behave well when certain properties dominate the work. Taken these points into consideration, we further introduce the object-based attention method.

Algorithm 1 Ways to select the bounding box in object-based attention.

for T frames **do**
  • detect the bounding box for human with confidence > 0.7
  if one person detected **then**
    • save the bounding box for this frame
  else if multiple target principals detected **then**
    • use the union bounding box
  else
    • use the whole image as the bounding box
  end if
end for
• use the average bounding box for this video

As shown in Algorithm 1, the proposed method utilizes a pre-trained object detection module called Faster R-CNN to generate the bounding boxes. For the sake of stability, we used the mean position for each coordinate. Each bounding box has four values, representing the area from upper-left to the lower-right position. Once we have got the bounding box $bbox_{i,j}$ for one video, we can formulate a similar attention as 2 do:

$$\mathbf{o}_f = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} bbox_{ij} \cdot I(i,j) \cdot \Theta(v)_{ij} \tag{3}$$

where $I(i, j)$ is an identical function $\in \{1, 1.5\}$, indicating whether the corresponding area in the feature map is human or not. Those areas contains persons are given a higher weight.

Then, we use one max pooling, followed by another conv layer and average pooling to extract image features.

3 Implementation details

We have tried two kinds of backbones as our baseline. First we use a LSTM-based structure, where a pre-trained Resnet-101 is used as the feature extractor. We performed spatial on the last conv layer, whose size is $20 \times 20 \times 1024$. Temporal attention is performed on the LSTM cell. We have also tried to use a 3d CNN i.e., R(2+10)D as the feature extractor. In this case, the feature size of is $4 \times 20 \times 20 \times 256$, where 4 is the temporal dimension. Second, we use a pre-trained 3d conv method i.e., R(2+10)D to better visualize the effectiveness of object-based attention. In this case, we only fine tune the last fully connected layer for both models with and without object-based attention models, where the only difference is our proposed attention mechanism.

We use python-cv2 library to clip 16 frames evenly for each video instead of FFmpeg, and resized them to $160 \times 160 \times 3$. **To evaluate the performance, we have downloaded the raw HMDB-51 dataset and followed the same train and test split as given in the course, where *smile* class is not used. Note that we strictly obey the test and train split as the course required.**

Since the extracting the features is relatively time-consuming, we provide a *preprocess.py* file to save the features extracted by Resnet or 3D conv models in the hard disk. In this way, we significantly speed up the training process.

We provide a Pytorch implementation of our work. Pre-trained models are available on [1].

4 Results

During our experiment, we found that spatial attention did not bring an obvious improvement. Since this part was done with another machine, and considering the time limit, we do not list it here. The scores might boost a little bit, but is far from satisfactory.

The remaining models are performed with CNN feature extractor + RNN classification procedure, where the only difference is the backbones and the proposed attention mechanism. Here, $Resnet$ means Resnet-101; $_{avgpool}$ means the pooling layer before the last fc layer; $RCNN$ means Faster R-CNN, and $_{conv}$ means the last conv layer.

Note that we report the result generated by model (e) in the txt file, since we are only allow to use RNN-based model, as required by the README file. However, we do find a better result using 3d conv. Thus the accuracy rate in the submitted txt file is much lower (about 0.44).

Table 1: Ablation study on the proposed attention mechanism.

index	mdoels	top-1 score
(a)	$2dResnet_{avgpool} + LSTM$	0.401
(b)	$2dResnet_{conv} + LSTM$	0.425
(c)	$3dResnet_{conv} + LSTM$	0.416
(d)	$2dResnet_{conv} + LSTM + RCNN$	0.417
(e)	$3dResnet_{conv} + LSTM + RCNN$	0.449
(f)	$(2+1)Dfinetune_{fc}$	0.576
(g)	$(2+1)Dfinetune_{fc} + RCNN$	0.595

References

[1] `https://github.com/lukun199/Action-Recognition-in-Videos`