Connectionist Temporal Classification for Group Activity Recognition in Videos

Bicheng Xu
School of Computing Science
Simon Fraser University
bichengx@sfu.ca

Abstract

Sequence of group activities recognition in videos is a currently less stressed research area in computer vision. In this work, we address this issue by employing the Connectionist Temporal Classification (CTC) model [1]. The task is given a sequence of video frames containing a group of people, predicting the sequence of activities that the group performs. We build a model consisting of VGG-19 Net [5], one layer of bi-directional Long-Short Term Memory (LSTM) recurrent neural network, and the CTC model to solve this task. To do the task, we construct a new volleyball dataset using the volleyball game videos available on YouTube[1]. The dataset contains 100 sequences of video frames, with one label for each sequence without time alignment. Our proposed model presents a reasonable result on the new volleyball dataset.

1 Introduction

Group activity recognition is a popular research area in the field of computer vision. Deng et al. [2] propose a joint framework integrating graphical models and deep neural networks to recognize group activity in images. Ibrahim and Muralidharan et al. [3] present a 2-stage deep temporal model for group activity recognition in videos. However, for group activity recognition in videos, current work mainly focuses on single activity recognition. In this work, we deal with recognizing the sequence of activities that the group performs in a video sequence.

Long-Short Term Memory (LSTM) recurrent neural network is a good sequence learner. It can learn the temporal information in a sequence. Yeung et al. [7] formulate a recurrent neural network-based agent, which interacts with a video over time. In this work, we use one layer of bi-directional LSTM recurrent neural network to extract the temporal relationship among the video sequence.

Connectionist Temporal Classification (CTC) model is first introduced by Graves et al. [1] to deal with speech recognition tasks. The CTC model is basically a dynamic programming algorithm. It interprets the output from a recurrent neural network into a labeling, in which adjacent labels are different. In the computer vision area, Huang et al. [4] use a similar idea to do weakly supervised action labeling in videos. Their work aims for personal action recognition, but we focus on group activity recognition.

[1] https://www.youtube.com/

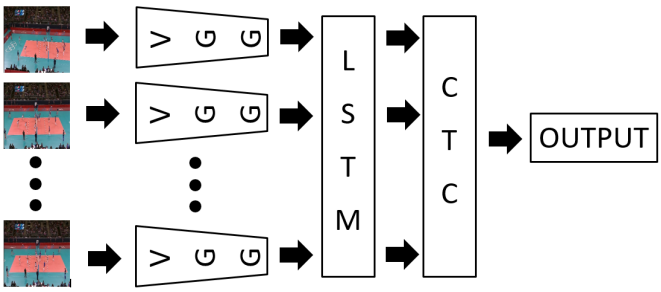

Figure 1: Our model structure. The model consists of VGG-19 Net [5], one layer of bi-directional LSTM recurrent neural network, and the CTC model.

2 Task and Dataset

2.1 Task Specification

Our goal is to given a sequence of video frames containing a group of people, recognize the sequence of activities that the group performs. The input is a sequence of video frames, and the output should be the sequence of activities that the group performs.

2.2 The Dataset

Since we find that there is no previous related dataset for this kind of task, we collect the dataset by ourselves. Using the volleyball game videos available on YouTube[2], we manually cut 100 sequences of video frames from 10 volleyball games, with 10 group activity labels. The number of frames of these sequences ranges from 63 to 529. Each sequence is labeled with a sequence of group activity labels without time alignment. The length of the labeling is in the range from 2 to 17.

For our collected volleyball dataset, the 10 group activity labels are as follows. They are left serve, left set, left spike, left pass, left winning point, right serve, right set, right spike, right pass, and right winning point. Each sequence describes the group activity from serving the ball to a winning point.

3 Approach

We develop a neural network model which consists of VGG-19 Net [5], one layer of bi-directional Long-Short Term Memory (LSTM) recurrent neural network, and the CTC model to solve the task.

In the current approach, We first use the VGG-19 Net [5] trained on the ImageNet ILSVRC-2014 dataset [6] to extract the features of the frames. We use the output of the fc7 layer of the VGG-19 Net [5] as the feature to represent the frame. Then we feed the feature to the one layer of bi-directional LSTM network to do supervised training. During the training, only the weights of the LSTM network will be updated.

Although a better way is to unified train the VGG-19 Net [5] and the LSTM recurrent neural network together, which should be a future work, our current approach produces reasonable results. The model structure is shown in Figure 1.

[2]https://www.youtube.com/

Table 1: The testing and last-time validation error rates for the three experiments

SAMPLING METHOD	INITIAL LEARNING RATES	TEST / VAL. ERROR RATES
Every Frame	0.0001	46.28% / 62.73%
Every Other Frame	0.0001	58.67% / 57.76%
Every Other Frame	0.001	**47.93% / 47.20%**

4 Experiments

4.1 Implementing Method

We use TensorFlow[3], a open source software library for machine intelligence, to implement our whole model. Our code for the VGG-19 Net [5] is modified from the code in the GitHub tensorflow-vgg repository[4].

4.2 Database Setting

We first permute all the sequences in the dataset in a random order. Then we divide the randomly permuted sequences into three parts, 60 sequences for training, 20 sequences for validation, and the remaining 20 sequences for testing.

4.3 Parameter and Input Setting

We conduct three experiments with different initial learning rates, or different input sampling methods. For all the experiments, when doing the training, the initial learning rate is either 0.001 or 0.0001. The momentum is set to 0.9, and the number of training epochs is 15,000. We do a validation test every 100 epochs. The batch sizes for all the training, validation, and testing are set to 20. For the bi-directional LSTM recurrent neural network, the number of hidden nodes is 128.

The number of frames in each sequence in our dataset ranges from 63 to 529. For the input of the neural network, we use two different sampling methods. The first method is using every frame in the sequence as the input to the neural network. The second method is to use every other frame in the sequence as the input. In this case, the number of frames in the input sequence ranges from 32 to 265. From the experiment results, it shows that there is no much difference between these two sampling methods. This means that in our future work we can use the second sampling method, which is more efficient for the time and space needed when conducting experiments.

4.4 Training Method and Error Measurement

We use the momentum algorithm, a stochastic gradient descent method, to train our neural network. To measure the error of the network output, we calculate the edit distance between the network result and the ground-truth labeling.

4.5 Results

We report the results for three different experiments. The first one uses every frame as the input with initial learning rate 0.0001. The second experiment uses every other frame as the input with initial learning rate being 0.0001. The third experiment also uses every other frame as the input but with initial learning rate 0.001. Our current approach produces reasonable results on our new volleyball dataset. The testing error rates and the last-time validation error rates are listed in Table 1.

[3] https://www.tensorflow.org/

[4] https://github.com/machrisaa/tensorflow-vgg

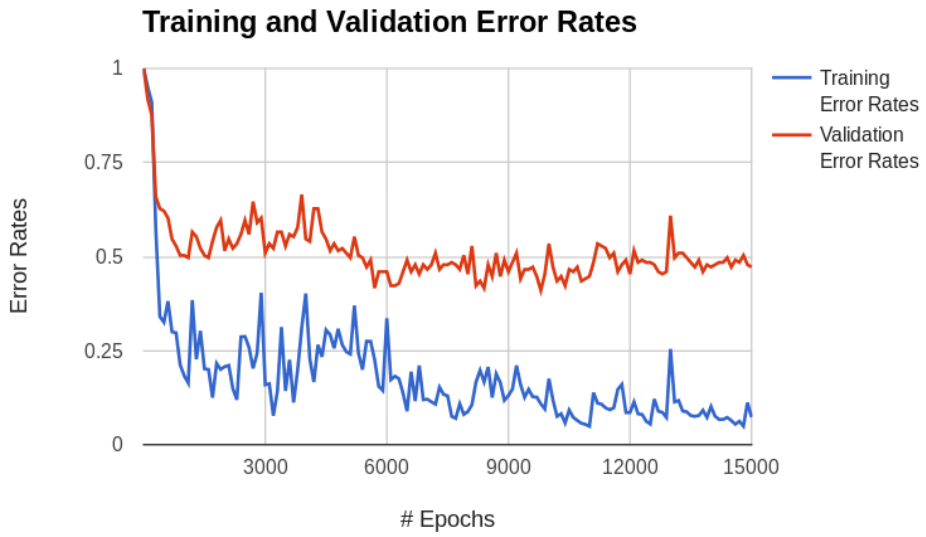

Figure 2: The changes of the training and validation error rates with the number of epochs increasing for the case using every other frame as the input and initial learning rate 0.001.

4.5.1 Experiments Using Every Frame

When the initial learning rate is 0.0001, after training 15,000 epochs, the training error rate is 0.0%, the last-time validation error rate is 62.73%, and the testing error rate is 46.28%.

We also try to set the initial learning rate to 0.001 when doing experiments, but sometimes this parameter setting will generate a **nan** loss. This means that this parameter setting is not stable, so we do not report the results for this setting.

4.5.2 Experiments Using Every Other Frame

When the initial learning rate is 0.0001, after training 15,000 epochs, the training error rate is 0.0%, the last-time validation error rate is 57.76%, and the testing error rate is 58.67%.

When the initial learning rate is 0.001, after training 15,000 epochs, the training error rate is 7.27%, the last-time validation error rate is 47.20%, and the testing error rate is 47.93%. The changes of the training and validation error rates with the number of epochs increasing for this case are shown in Figure 2. The trends of the changes for other experiments are similar. A sample testing result for this setting is shown in Figure 3.

4.6 Results Analysis

All of the three results shown above exist over-fitting but this is reasonable. The size of our dataset is limited, and we only use 60 sequences to train the LSTM recurrent neural network whose maximum timestep is either 529 or 265. Besides, in the current approach, we just use the original weights of the VGG-19 Net [5] trained on the ImageNet ILSVRC-2014 dataset [6]. Future work should involve training the VGG-19 Net [5] and the LSTM recurrent neural network together as an end-to-end unified framework.

From the three results in terms of testing error rate and the last-time validation error rate, the experiment using every other frame with initial learning rate 0.001 performs better than any other two experiments. This shows that in the future experiments, the setting with sampling every other input frame and initial learning rate being 0.001 can be a good choice.

Input: A sequence containing 125 frames.

Output:

Right serve, Left pass, Left set, Left spike, Right winning point

Ground-truth labeling:

Right serve, Left pass, Left spike, Left winning point

Figure 3: A sample test output for the case using every other frame as the input and initial learning rate 0.001.

5 Conclusion

Long-Short Term Memory (LSTM) recurrent neural network is a good sequence learner. Connectionist Temporal Classification (CTC) model [1] is a good interpreter for the output from a recurrent neural network. We propose a model consisting of VGG-19 Net [5], one layer bi-directional LSTM recurrent neural network, and the CTC model to recognize a sequence of activities performed by a group of people in a video sequence. Our current approach produces reasonable results on our newly collected volleyball dataset, and more future work is needed.

6 Future Work

Future work is supposed to involve the following four aspects, collecting more sequences of frames, unified training the whole framework, trying other sampling methods for the input frames, and building baseline models to compare with.

The total number of sequences in our current dataset is 100, which is quite limited to train a bi-directional LSTM recurrent neural network. More sequences are needed to get a good testing result.

Unified training the whole framework including VGG-19 Net [5], the LSTM recurrent neural network, and the CTC model is needed. Since the current weights for the VGG-19 Net [5] are trained for a classification task, not a recognition task. Unified training should produce much better experiment results.

Doing sampling on the input frames can reduce the time and space needed when training the neural network. The current results show that in some settings doing sampling on the input can achieve better results. Other sampling methods should be tried.

Some baseline models can be built without using the CTC model to interpret the result from a recurrent neural network. For example, the method proposed in [8] can be a good idea.

Contributions

Since our group only has one people, me, I do all the work for the project.

Acknowledgments

We would like to thank the instructor, Professor Greg Mori, and all the TAs of the course for helping with this work.

References

[1] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *International Conference on Machine Learning (ICML)*, Pittsburgh, USA, 2006.

[2] Z. Deng, A. Vahdat, H. Hu, and G. Mori. Structure Inference Machines: Recurrent Neural Networks for Analyzing Relations in Group Activity Recognition. In *Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, 2016.

[3] M. Ibrahim, S. Muralidharan, Z. Deng, A. Vahdat, and G. Mori. A Hierarchical Deep Temporal Model for Group Activity Recognition. In *Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, 2016.

[4] D.-A. Huang, L. Fei-Fei, and J. C. Niebles. Connectionist Temporal Modeling for Weakly Supervised Action Labeling. In *European Conference on Computer Vision (ECCV)*, Amsterdam, Netherlands, 2016.

[5] K. Simonyan, and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR)*, San Diego, USA, 2015.

[6] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[7] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei. End-to-end Learning of Action Detection from Frame Glimpses in Videos. In *Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, 2016.

[8] A. Karpathy, and L. Fei-Fei. Deep Visual-Semantic Alignments for Generating Image Descriptions. In *Computer Vision and Pattern Recognition (CVPR)*, Boston, USA, 2015.