CS230

YOLO with embedded DenseNet Approach to detect Human Interaction (Computer Vision)

Rong Chen
Stanford University
jessyc@stanford.edu

Qin Gu
Stanford University
guqin@stanford.edu

Abstract

The existing computer vision AI model could classify numbers of human actions and human interactions. However, due to a large number of different types of human interaction, it's difficult to use one model to classify all the human interaction at this point. Instead of using one model to identify all the human interaction, we use two trained AI models to predict if there is human interaction in a video. Yolov3 has been trained to predict human objects and it was used to find the target frame and on top of Yolov3 we use another trained CNN model – DenseNet to further predict if there is human interaction in the video. This method helps to improve the overall accuracy and provides flexibility of changing or upgrading embedded CNN models to fulfill the different vision detection applications.

1 Introduction

Vision AI is important to many different applications, such as surveillance, robotics and so on. The existing AI models are able to classify certain human action and human interaction. It is important to detect human interaction in order to further recognize and analyze these interactions through AI models, such as predict the interactions and relationships between humans.

In this project, we implemented a method to detect interactions between two human subjects. We optimized the accuracy and performance by first detecting if at least two human bounding boxes are overlapping using Yolo, then detecting whether there is human interaction with a DenseNet model.

The input data is a video and output will be binary output indicating whether there is human interaction (interaction:1, non-interaction: 0). We focused on direct body contact type of human interaction in this project.

2 Related Work

Human interaction detection is a complicated task. Numerous research studies have been done in this area.

S. Jeba Berlin et al.[2] uses Harris corner points and the histogram to form the feature vector of the spatiotemporal volume to recognizing human interactions.

Vahdat et al.[3] uses a set of key poses to present each individual and formulated spatial and temporal relationships among key poses in their model.

Choi et al.[4] presents a spatial-temporal crowd context for the recognition of collective human activities.

Yu Kong et al.[5] introduces high-level descriptions called interactive phrases to express binary semantic motion relationships between interacting people.

A recent study [1] uses the Yolo method to analyze each frame from video to detect human interaction. In this method, the Yolo model analyses each frame and predicts the action classification.

To our best knowledge, previous researches are able to detect certain types of human interaction, but were not able to detect all the human interaction. The complex models may result in low performance.
In this project, we used YOLO with embedded DenseNet to detect human interaction in a video. This method helps to improve the overall accuracy and provides flexibility of changing or upgrading embedded CNN model to fulfill the different vision detection applications.

3 Dataset

In order to train the DenseNet to detect if there is human interaction or not, we needed a video dataset that included both human interaction and non-human interaction. We found that the UCF101 dataset included both above two types of videos. The dataset included 13,320 labelled video sequences in total (7.5 GB).
The raw UCF101 included more non-human interaction videos than human interaction videos. In order to balance the positive and negative samples in the dataset and re-label the videos correctly, we pre-processed the dataset as in the following pipeline [Figure.1].

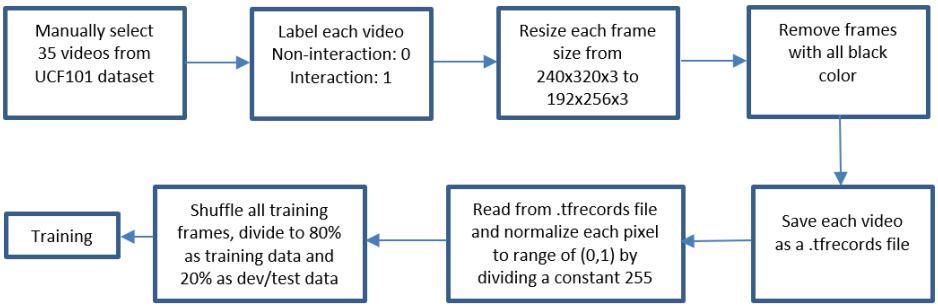


Figure. 1

In order to reduce the training size, we resize the original frame resolution 240(height) x 320(width) x 3 to 192(height) x 256(width) x 3. The training data was normalized to range of (0,1) by dividing constant value 255.
We used total 35 videos with 17 non-human interaction videos and 16 human interaction videos, details as following [Table. 1].

Class type	Videos	Frames	Training frames	Dev/Test frames
Non-human interaction	17	3097	4884	1222
Human interaction	18	3009		
Total	35	6106		

Table. 1

The entire dataset was divided to 80% as training data and 20% as dev/test data.
Figure.2 shows a human interaction sample and a non-human interaction sample.

Human interaction Non-human interation

Figure. 2

4 Method

For this project we used a two-step method to detect direct human interaction. We first used a pre-trained generic Yolo3 model to detect human subjects in each frame of a video. If at least two human bounding boxes overlapped, the candidate frame was fitted into our trained DenseNet model to detect for true human interaction.

4.1 Yolo Pre-prediction

Directly detecting human interaction in random video using our trained Densenet could require a relatively slow training process on a large dataset. To increase our pipeline efficiency, we first tuned and improved an existing generic Yolo3 model [4] to detect if at least two human

subjects are presented in each video frame. Then we used axis-aligned Bounding Box Collision detection algorithm [Figure.3] to check if the human bounding boxes overlapped.

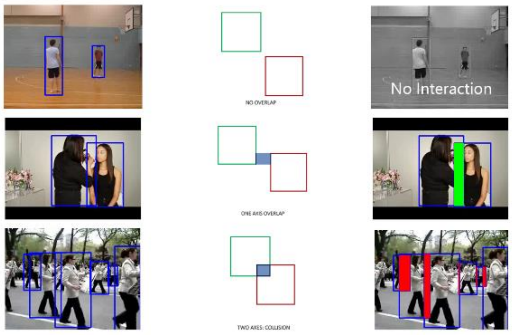


Figure.3

In our experiments, we extracted bounding boxes from human subjects with 80% confidence score from the preliminary Yolo pre-prediction. If no human, only a single person, or multiple persons without overlaps were detected, we quickly dropped the candidate frame [Figure.3 upper row] to classify it as no direct human interaction. If multiple persons with overlapping bounding boxes were detected, the candidate frame fed into the next DenseNet model specifically trained for detecting direct human interaction. We only used the second model if more than two overlapping persons were detected. In the above Figure.3 example, the middle row (with green intersection) had true human interaction while the lower row (with red intersection) had no direct human interaction even when overlapped bounding boxes are detected.

4.2 DenseNet Training and Prediction

In order to detect the human interaction, we needed a CNN model that was capable of extracting features, as well as relatively fewer parameters due to the constraint of training resource. DenseNet was designed for visual object recognition and establishes a connection relationship between different layers, the later layer was able to reuse the features from the earlier layers which helps to prevent the feature vanishes before reaching its destination. Densenet improves the accuracy caused by the vanishing gradient in high-level neural networks and requires fewer parameters than an equivalent traditional CNN. Because of these reasons, we selected DenseNet as the training model.
Refer to the DenseNet model structure in [Figure.4].

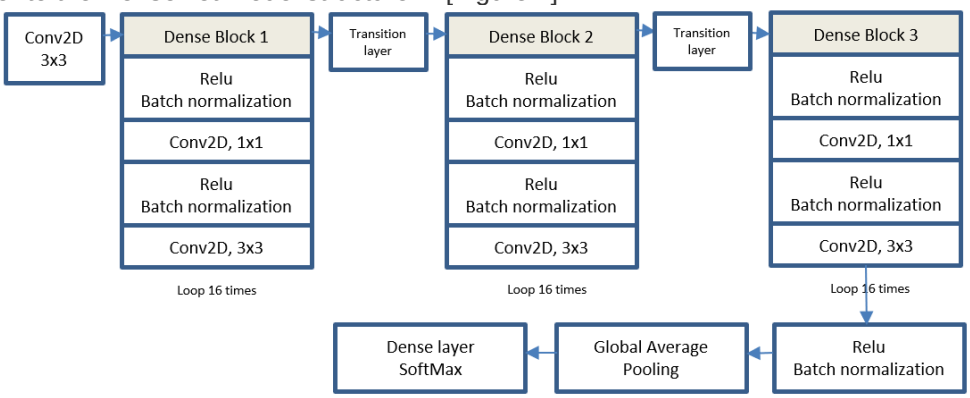


Figure.4

DenseNet has two main hyperparameters, depth and growth rate. Depth decides the network layers and growth rate regulates how much information is added to the network each layer. We used depth 100 and growth rate 12 in this model. The DenseNet model includes the L2 regulation.
After training, the model came out with acceptable accuracy with the initial hyperparameters. Refer to 5 Results and Analysis.

The output is a binary indicating whether human interaction exists (interaction:1, non-interaction: 0). We used categorical cross-entropy as a loss function.

$$\text{Loss}_{\text{train}} = -\sum_{i=1}^{N} y_i \cdot \log \hat{y}_i$$

The initial optimizer used for DenseNet is Gradient descent with hyperparameters learning rate=.1, momentum=0.9. We also tested Adam optimizer with hyperparameters learning rate=.001, beta_1=0.9, beta_2=0.999.
Both Gradient descent and Adam optimizer achieved similar accuracy and loss.

5 Results and Analysis

5.1 Training Result and Analysis

Due to each frame size and training resource constraint, we trained the model with mini batch size 8. We first trained the model with epoch 100, however the accuracy and loss did not converge. The accuracy can achieve around 70%. The convergence of training accuracy, test accuracy, training loss and test loss are as shown in [Figure 5].
After error analysis, we found there was a mislabeled frame in the video dataset. There was an all-black color frame in end of each video that caused this issue. After the mislabeled issue was resolved by removing the all-black color frame, the accuracy and loss started converge [Figure.6].
We increased the training epoch to 2000 and the training accuracy achieved 95.8%, as shown in [Table.2].

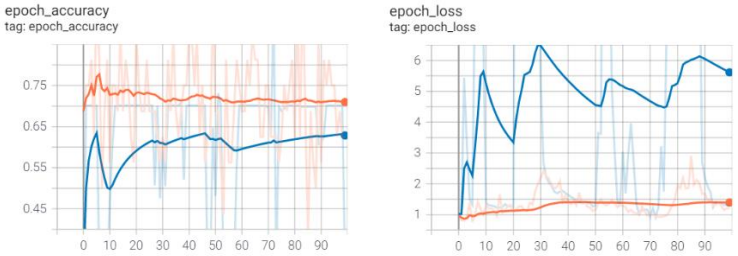


Figure.5

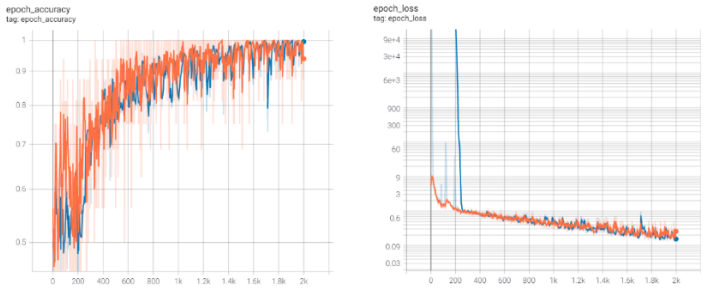


Figure.6

	Training accuracy	Dev/Test accuracy	Epoch
Original dataset	72.3%	63.8%	100
Dataset after remove mislabeled black frame	95.8%	100%	2000

Table.2

In order to have better visibility on how DenseNet detected the human interaction, we displayed the feature map using TensorBoard.
We printed feature maps of 3 Conv layers from the trained model. The 3 Conv layers are the first Conv2D layer, middle Conv2D layer and the last Conv2D layer and selected one feature map of each layer. For the detail feature map refer to Appendix [Feature Map.1] and [Feature Map.2].
Figure. 7 shows the feature maps of a non-human interaction sample. The feature map of the last Conv2D layer blurs out since there is no feature presented for human interaction.

Figure. 8 shows the feature maps of a human interaction sample. The feature map of the last Conv2D layer extracts the features present for human interaction.
Based on the printed feature maps, we have better understanding and visibility how the model extracts the features for human interaction and use it for prediction.

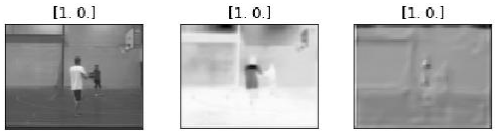


Figure.7

Figure.8

5.2 Prediction Result and Performance

We tested our two-step prediction framework on an Nvidia GTX 3070 GPU with 8GB video memory. The program can achieve real-time prediction while playback test videos. Compared with interactions manually labelled by human, we evaluated our framework prediction accuracy and recall from Yolo step1 alone, Densenet step2 alone, and the final prediction results.

Total video: 10 Total Frames: 1342 Interaction Frames: 675	Yolo + bbox overlapping	DenseNet	Final Prediction
Accuracy	78.5%	99.1%	93.2%
Recall	69.3%	97.8%	88.3%

Table.3 Similar videos from the UCF101 dataset

Total video: 10 Total Frames: 1342 Interaction Frames: 675	Yolo + bbox overlapping	DenseNet	Final Prediction
Accuracy	72.1%	75.1%	89.2%
Recall	80.8%	85.3%	86.4%

Table.4 Random videos from the UCF101 dataset

When testing on the small dataset with the same distribution as training data [Table. 3], Densenet achieves the best performance since it's specifically trained from the same distribution. Yolo alone performs relatively poorly because many videos contain overlapping subjects without direct true human interaction as mentioned in previous sections. The combined method can significantly increase the accuracy of Yolo step with the help of our trained Densenet. However, when running our framework on a larger generic dataset with more variants of motions [Table. 4], The combined method outperforms both Yolo and Densenet respectively. Therefore, we think it's a reasonable result that the combined framework works slightly less accurately on the test set very similar to the train set, but providing more potential for giving better performance on the larger generic test set, hence better usability on real-world practical data.

6 Conclusion and Future work

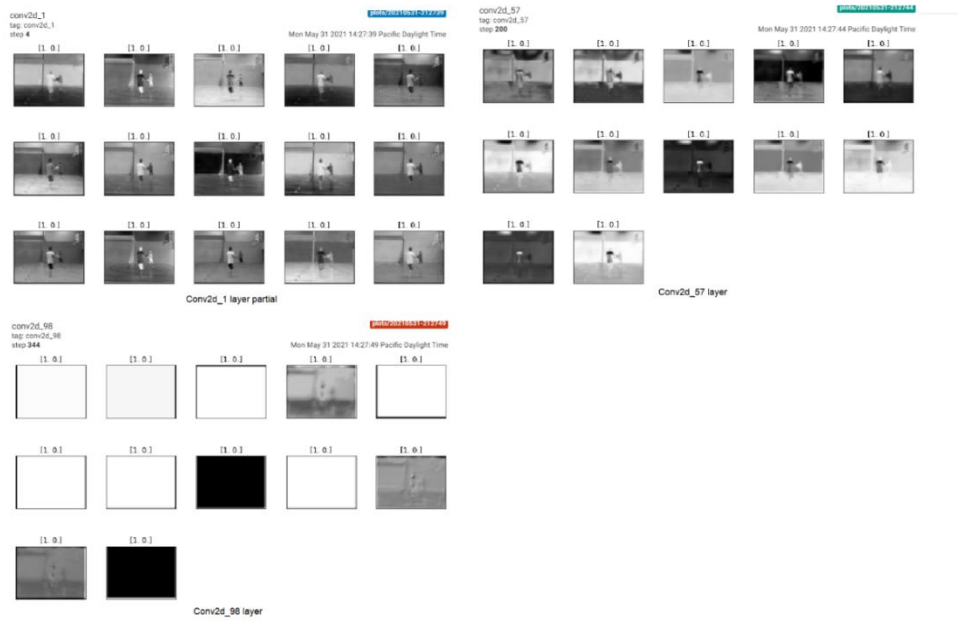
We implemented a program to detect human interaction using the Yolo and DenseNet plugin. We achieved 95.8% training accuracy with DenseNet by using techniques such as data normalization and error analysis to resolve the mislabeled samples.
The Yolo with embedded DenseNet Method helps to improve the overall accuracy and provides flexibility of changing or upgrading embedded CNN models to fulfill the different vision detection applications.
For future work, we may consider increasing the training data to improve the model scalability.

7 Contributions

We worked as a team and had a weekly meeting to discuss the project and our ideas. In summary, Rong pre-processed the training data and trained the DenseNet model, Qin worked on the Yolo model and final program assembly and result.

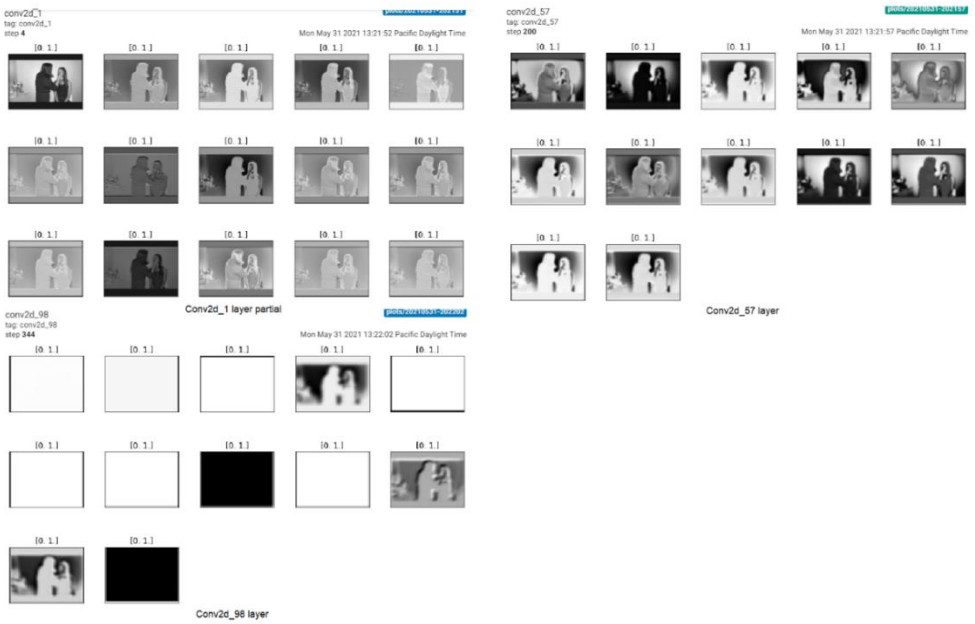
8 Appendix

[Feature map. 1] displays feature maps of 3 layers of a non-human interaction sample. The 3 Conv layers are the first Conv2D layer(partial), middle Conv2D layer and the last Conv2D layer.



Feature Map.1

[Feature map. 2] displays feature map of 3 layers of a human interaction sample. The 3 Conv layers are the first Conv2D layer(partial), middle Conv2D layer and the last Conv2D layer.



Feature Map.2

References

[1] S. Jeba Berlin; Mala JohnHuman Interaction Recognition through Deep Learning Network
[2] Vahdat, A., Gao, B., Ranjbar, M., Mori, G.: A discriminative key pose sequence model for recognizing human interactions. In: ICCV Workshops, pp. 1729–1736 (2011)
[3] Choi, W., Shahid, K., Savarese, S.: Learning context for collective activity recognition. In: CVPR (2011)
[4] Yu Kong, Yunde Jia, and Yun Fu Learning Human Interaction by Interactive Phrases
[5] Shubham Shindea, Ashwin Kotharia, Vikram Guptab "International Conference on Robotics and Smart Manufacturing (RoSMa2018) YOLO based Human Action Recognition and Localization"
[6] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger "Densely Connected Convolutional Networks"
[7] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi You Only Look Once: Unified, Real-Time Object Detection

Reference from Github:
[8] https://github.com/BIGBALLON/cifar-10-cnn
[9] https://github.com/zzh8829/yolov3-tf2
[10] https://github.com/zzh8829/yolov3-tf2/blob/master/docs/training_voc.md
[11] https://github.com/ferreirafabio/video2tfrecord