# YOLO-based Adaptive Window Two-stream Convolutional Neural Network for Video Classification

Charles Han
hcs@stanford.edu

Chao Wang
cwang17@stanford.edu

Evelyn Mei
Evelyn66@stanford.edu

## Abstract

*Human action recognition in videos is an important task in computer vision research. The challenge is to capture information from still frames and motion between frames then combine these two streams in a way that optimizes class prediction. Our contribution is two folds. First, we developed an adaptive window approach for detecting human actors in videos and forming a temporal stream based on human crops rather than central crops. Second, we demonstrated that YOLO is a fast and effective method for human detection in UCF-101 dataset. Our model performance exceeded baseline model by a notable margin, and provides valuable guidance for future work.*

## 1. Introduction

Convolutional Neural Networks (CNN) have been adopted widely for image classification problems. As CNNs demonstrate significant success in learning powerful and interpretable image features [7], more and more researchers start to deploy CNN on video classification problems. The main challenge is to capture not only the appearance information present in single, static frames, but also complex temporal evolution.

Among various video classification tasks, human action recognition is the key problem due to its wide application in surveillance camera, robotics and human computer interface (HCI) [8] [9]. Robustly classifying real videos (UCF-101 [6]) into arbitrary free-form activities is a challenging task mainly because of background clutter, viewpoint changes, and drastically diverse dynamics in the observed motion [3].

Existing research [2] has shown that a two-stream approach, namely spatial stream plus temporal stream performed significantly better than training on raw stacked frames. This is inspired by two-stream hypothesis -- the human visual cortex contains two pathways: the ventral stream (which performs object recognition in the scene) and the dorsal stream (which recognizes motion). In order to get rid of the background noise and represent the action dynamics, we propose to detect and segment out human action from the background scene.

In this project, we investigate how single-shot object detection method [11] can be integrated into the existing two-stream video classification pipeline to leverage its performance. In particular, we adopt You Only Look Once (YOLO) [4] [10] object detection approach to localize human action. Specifically, our two-stream architecture includes one spatial stream which performs action recognition from still video frames, and one temporal stream where adaptive cropping windows are generated by the fine-tuned YOLO detector and frames are cropped and stacked to represent the action from motion. Both streams are then fed into two separate CNNs and further fused together to get the classification result.

## 2. Related work

Classic video classification approach usually involves three major steps [13]: First, extracting local visual features that describe a region of the video, either densely [15] or as a sparse set of interest points [16]. Second, the extracted features are encoded into high-level local spatial-temporal video descriptions [14]. Lastly, a classifier (such as an SVM or Softmax) is trained on the resulting representation to cast the final class prediction.

There has also been a number of attempts for developing a deep architecture for video recognition. Karpathy and his colleagues [1] were one of the first pioneers in using deep convolutional neural networks for video classification. They suggested a multiresolution, foveated two-stream architecture. Input frames are fed into two separate streams of processing: a low-resolution full image frame and a high-resolution center crop. Their key assumption was that main information of the video was captured in the center of the frame. In addition, this method directly feeds video frames into CNN without considering the motion between consecutive images. Therefore, model is expected to implicitly learn spatial-temporal motion-dependent features, which can be a difficult task.

Subsequently, a research that addressed the shortcomings mention before is Two-Stream Convolutional Networks for Action Recognition in Videos [2]. In addition to a spatial stream, which operates on original video frames, they developed a temporal stream using Optical Flow. Optical flow is defined as a set of displacement vector fields between pairs of consecutive frames. Essentially, optical flow represents the difference between frames and captures motion-dependent features. However, this approach is susceptive to changes in camera orientation and position. Nevertheless, this study gave us fresh thoughts for extracting motion from videos.

A state-of-the-art research on this topic is two-stream semantic region based CNNs (SR-CNNs) [3]. They leveraged existing human/object detectors and proposed an architecture which leverages semantic cues (e.g. scene, person and object) for action understanding. Specifically, for an image that has more than one person, they tried to capture the "actor" rather than "bystanders". They also described how they recovered missing detection results in individual frames and refined locations of bounding boxes. This is a very thought-provoking project that we look up to. Wang and his colleagues [3] demonstrated that, for current action recognition benchmarks, scene context acts as a very strong cue for action recognition. Incorporating human action using R-CNNs has given action recognition accuracy a significant boost.

However, approaches like R-CNN includes complex pipeline for generating potential bounding boxes in an image, running a classifier on these proposed boxes and post-processing to refine these boxes. This is slow and hard to optimize as each of these components must be trained separately. Thus, we propose to use YOLO to provide semantic information about the activities portrait in the videos over time. YOLO was first proposed by Joseph Redmon etc. in 2016 [4] where they construct object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. In the YOLO framework, a single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation, which leads to extremely fast detection speed. However, YOLO also suffers from a variety of shortcomings such as high localization errors and low recalls compared to state-of-the-art detection systems. In 2017 YOLO evolves into its second version YOLOv2 (YOLO 9000) by pooling a variety of techniques such as batch normalization, dimension clusters, direct location prediction and multi-scale training [10]. In this project, we integrate fine-tuned YOLOv2 detector into our two-stream CNN architecture, generating bounding boxes to localize the human action in the temporal stream.

## 3. Method

The workflow that we have adopted for this study is depicted in figure 1. We take each video and decode them into images at a 5 FPS (frames per second). For each class of videos, we take 20 of them as training data and 5 as test data. During training and test time, we randomly sample 64 videos from training and test pools respectively. For each video, we feed it into two streams. In the spatial stream, we randomly sample one image from this video, and pass it through a CNN which eventually generates a Spatial Loss. In the temporal stream, we randomly select 10 consecutive frames from this video, get the human crop of each frame, and stack the 10 crops together. The method that we used to generate human crops is You Only Look Once (YOLO), which will be explained in greater details in the following

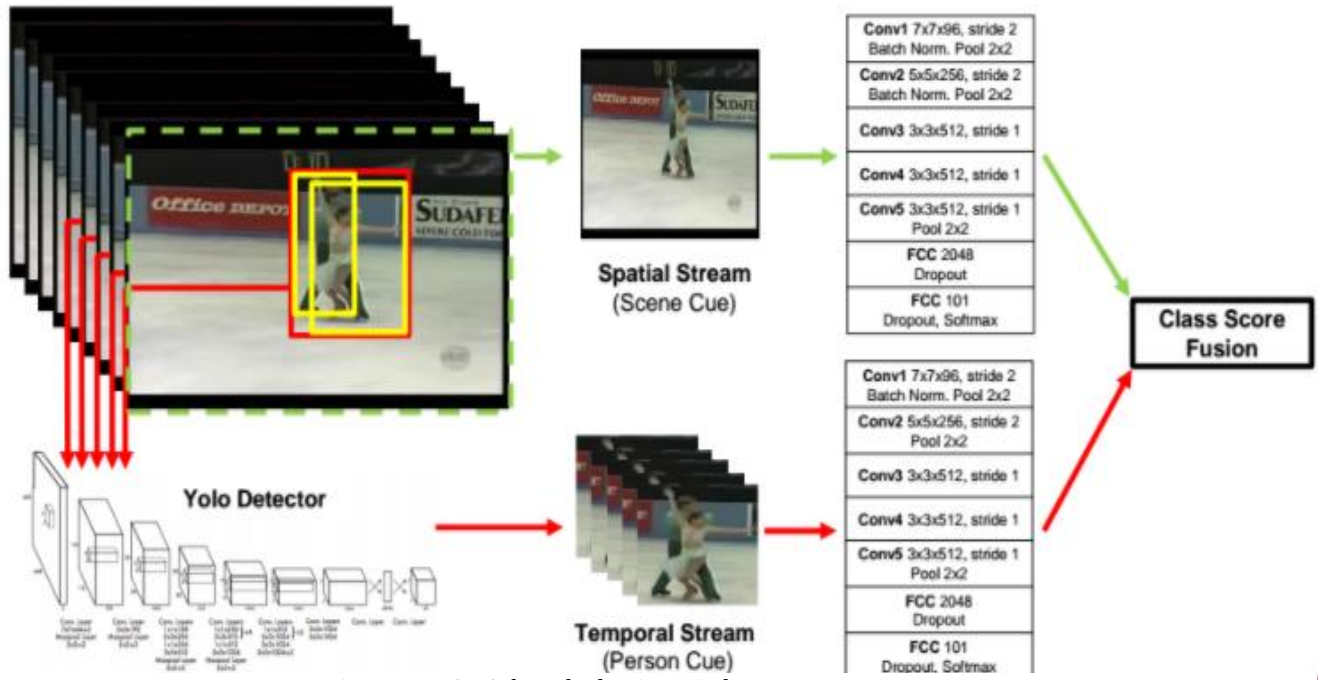

Figure 1: YOLO-based adaptive window two-stream structure

section. We feed the stacked human crops into a CNN, which outputs a Temporal Loss at the end. Finally, we fuse the Spatial Loss and Temporal Loss together to produce a Class Score Fusion, based on which class predictions are generated.

## 3.1 You Only Look Once (YOLO)

YOLO is a new approach to object detection. It frames object detection as a regression problem which outputs spatially separated bounding boxes and associated class probabilities. YOLO is faster than Region-proposal methods because it predicts bounding boxes and class probabilities in one evaluation by a single network.

The architecture of YOLO is relatively simple. It consists of 24 convolutional layers followed by 2 fully connected layers. The activation function of the network is Leaky ReLu. Multiple bounding boxes and corresponding class probabilities are simultaneously predicted by a single convolutional neural network [4].

Compared to traditional methods of object detection, YOLO has several advantages. First, YOLO is extremely fast. The algorithm doesn't need a complex pipeline since it frames detection as a regression problem [4]. Second, YOLO predicts bounding boxes and labels at the same time, which is a significant improvement from traditional 2-step methods. Third, YOLO learns generalizable representations of objects and reasons globally about the image. When it comes to unexpected inputs or entirely different domain, YOLO is less likely to fail.

YOLO divides the entire input image into an S × S grid. Each grid cell predicts B bounding boxes, together with their associated confidence scores. Each confidence scores represents the expectation that the box contains an object and indicates the accuracy of the box. There are 5 prediction parameters in each bounding boxes: x, y, w, h, and confidence.

The loss function of YOLO combines different aspects of the algorithm. It is shown as follows:

$$L = \lambda_{\text{coord}} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{\text{ij}}^{\text{obj}} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \ + \lambda_{\text{coord}} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{\text{ij}}^{\text{obj}} [(\sqrt{w_i} - \sqrt{\widehat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\widehat{h}_i})^2] + \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{\text{ij}}^{\text{obj}} (C_i - \hat{C}_i)^2 \ + \lambda_{\text{coord}} \sum_{i=0}^{s^2} \sum_{j=0}^{B} 1_{\text{ij}}^{\text{noobj}} (C_i - \hat{C}_i)^2 + \sum_{i=0}^{s^2} 1_{\text{i}}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c)\right)^2 \qquad (1)$$

Where $C_i$ denotes the confidence of class i, $1_{\text{i}}^{\text{obj}}$ denotes if object appears in cell i and $1_{\text{ij}}^{\text{obj}}$ denotes that the j th bounding box predictor in cell i is "responsible" for that prediction.

Note that, classification error counts towards loss function only if an object is present in that grid cell. Besides, bounding box coordinate error is only penalized in the loss function if that predictor has the highest coverage of the object, and thus is "responsible" for the detecting this object [4].

## 3.2 Two-stream Model

For this action recognition task, we adopted a two-stream architecture, which contains a spatial stream and a temporal stream (figure 1). The spatial stream takes one frame from each video. It carries information about scenes and objects described in the video. This is a useful clue, since some actions are highly related to particular objects. The temporal stream concentrates on actions of people. We first use YOLO detector to crop out humans from the image. In the case where there are multiple people in the image, we combine the bounding boxes to form a larger bounding box which covers all people. We take the cropped images and stack them to represent motion across frames. The temporal stream significantly improves classification accuracy compared to one-stream approach. The system uses deep CNN in each stream. We combine the softmax scores from each stream using different methods in the late fusion phase.

## 4. Dataset

We use UCF-101 Human Action dataset [6] as it offers a good balance between number of action classes and the variety of actions. Besides, it is adopted in almost all research papers in the video classification area. This dataset contains 101 action classes. Each action class consists of 25 groups and each group has 4 to 7 video clips. The whole dataset contains 13,320 video clips. The clips in one group share some common features, such as back-ground or actors. The mean clip length is 7.21 sec and the total duration is 1600 mins. The min clip length is 1.06 sec and the max clip length is 71.04 sec. All clips have fixed frame rate and resolution of 25 FPS and 320×240 respectively. We pre-process all the videos, discretized into frames of images at 5FPS. We further split the entire UCF-101 dataset into train and test set by a ratio of 4:1 -- in each action classes, the first 20 groups of videos belong to training set and the other 5 groups belong to test set. The following figure2 shows some of the action categories in UCF-101:

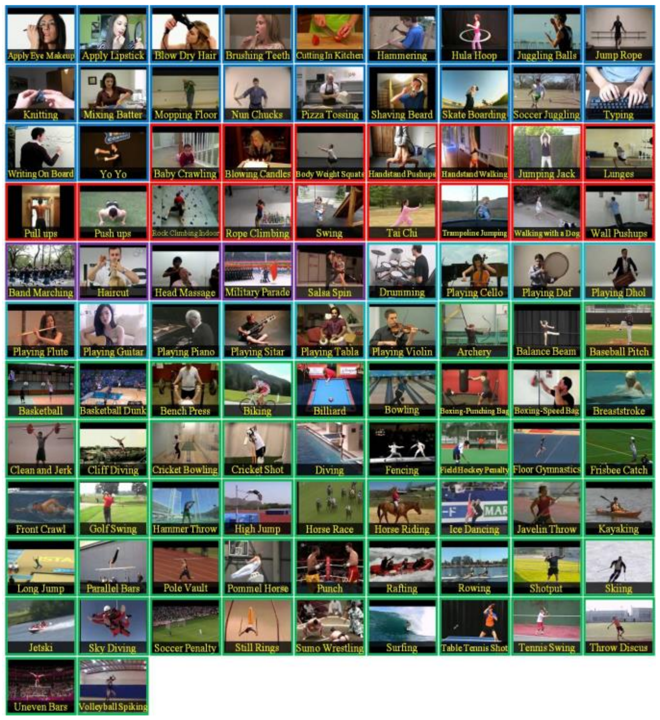

Figure 2: UCF-101 Action Recognition Samples

## 5. Experiments

The layer configuration of our spatial and temporal CNNs is schematically shown in Fig. 1. It corresponds to CNN-M-2048 architecture [17]. All hidden weight layers use the Rectified Linear Unit (ReLU) activation function; maxpooling is performed over 3× 3 spatial windows with stride 2. The spatial and temporal CNNs are almost identical except that the second normalization layer is removed from the latter to reduce memory consumption.

In the training procedure, the spatial and temporal CNNs are first trained separately and later fused together. 64 videos were randomly sampled from the training set as one batch. We ran 50 epochs (10000 iterations) in total. The network weights are learnt using the mini-batch RMSprop optimization.

RMSprop is an adaptive learning rate method proposed by Geoff Hinton in [19]. RMSprop divides the learning rate by an exponentially decaying average of squared gradients:

$$E[g^2]_t = 0.9E[g^2]_{t-1} + 0.1{g_t}^2 \tag{2}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t \tag{3}$$

Where $E[g^2]_t$ is the mean square of gradients at time t. It is taken as a moving average. $\eta$ is the learning rate. It is divided by the square root of $E[g^2]_t$ plus an error term.

We started with learning rate equal 1e-3, which was the default learning rate in RMSprop. However, in our experimentation, we saw huge fluctuation. Therefore, we then reduced our learning rate to 5e-5. It turned out to be very slow. Through a series of experimentation, we found that 1e-4 was the optimal learning rate which minimized the loss.

At each iteration, a mini-batch of 64 samples is constructed by sampling 64 training videos (uniformly across the classes), from each of which a single frame is randomly selected. In spatial net training, a 224 × 224 sub-image is randomly cropped from the selected frame. In the temporal net training, we use YOLOv2 detector to generate an adaptive window to crop the original frames into sub-images, which are later resized to 224 × 224 as well. The YOLOv2 detector is pre-trained on VOC 2007 [18] (we download weight from pjreddie.com/darknet) then we fine-tune the detector to identify "Person" (category 14) only. We combine multiple bounding boxes by drawing a single larger bounding box to cover them all.

In our two-stream approach, each stream produces a loss, namely Spatial Loss and Temporal Loss. We tested several ways to combine these two.

The first method is to take maximum over Spatial Loss and Temporal Loss.

$$\text{Fusion Loss} = \max(\text{Spatial Loss}, \text{Temporal Loss}) \tag{4}$$

We confirmed Wang's observation in [3] that the Max didn't provide good result. We believe this is because max fusion does not take advantage of the complementary contribution among the two streams.

The second method is to take a simple average over the losses produced by the two streams.

$$\text{Fusion Loss} = 0.5 \times \text{Spatial Loss} + 0.5 \times \text{Temporal Loss} \tag{5}$$

We saw 11% improvement in test accuracy from the Max.

From our one-stream experiments, we noticed that the testing accuracy of spatial stream was 0.39, while that of temporal stream was 0.51. Given temporal stream is more accurate than spatial stream, we proposed a weighted sum method. The weights are in proportion to accuracies.

$$\text{Fusion Loss} = 0.44 * \text{Spatial Loss} + 0.56 * \text{Temporal Loss} \tag{6}$$

We observed further loss reduction from the simple average method. Now that the weighted sum method worked, we hope to improve on the weights by training the model to tune this parameter.

$$\text{Fusion Loss} = \mathrm{w_s} * \text{Spatial Loss} + (1 - \mathrm{w_s}) * \text{Temporal Loss} \tag{7}$$

However, it was not as good as the fixed weight method. The loss curve became more unstable.

For efficiency, we evaluated the four fusion methods by running 150 training iterations for each method. The loss curves generated by these four fusion methods are plotted in figure 3.

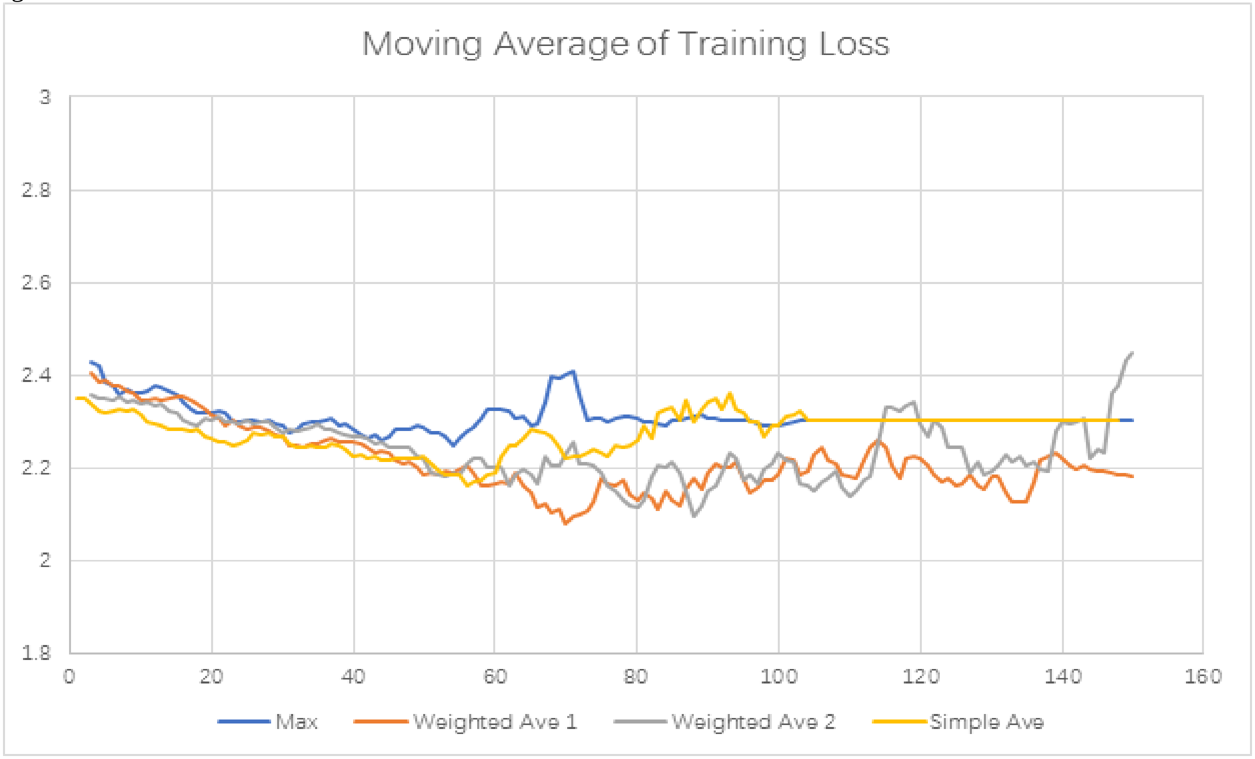

Figure 3: Training loss reduction with four fusion methods. Weighted average with predetermined weights is more stable and has continued momentum for loss reduction.

## 6. Results

Our primary metric is classification accuracy on a test set. We first tuned hyper-parameters for our network on 10 classes of videos. Then we freeze the hyper-parameters and trained on all 101 classes in UCF-101.

### 6.1. Training

We trained our network using Google Cloud Platform Compute Engine with GPU instances. Each iteration takes about 50ms on 8 CPU, 1 Pascal GPU virtual machine over the cloud. In our two-stream model, we first conducted clip-level random sampling. Then we randomly sampled one image from a clip to feed into the spatial stream, and we also sampled 10 consecutive frames to feed into the temporal stream. Videos that are shorter than 2 seconds have less than 10 frames. We repeat the last frame to make up for the full 10 frame sequence.

### 6.2. Quantitative Result

We compared training and testing accuracy among spatial, temporal and Two-stream network (table 1).

| Accuracy | Spatial Stream | Temporal Stream | Two-Stream with Fusion |
|---|---|---|---|
| Training | 80% | 95% | 98% |
| Testing | 39% | 51% | 68% |

Table 1 Training and testing accuracy among Spatial, Temporal and Two-stream network

Two-stream network with fusion achieved 98% accuracy on training set and 68% accuracy on the test set. The temporal stream alone achieved 95% training accuracy and 51% testing accuracy, whereas the spatial stream alone achieved 80% and 39% accuracy in training and test sets respectively.

During the training process of our Two-stream Network, the optimal learning rate 1e-4 produced steady loss reduction (figure 4).

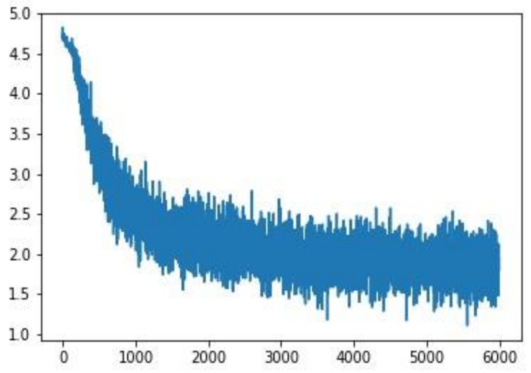

Figure 4: Learning curve for two-stream CNN

The confusion matrix produced by the best performing Two-stream Network is in figure 5.

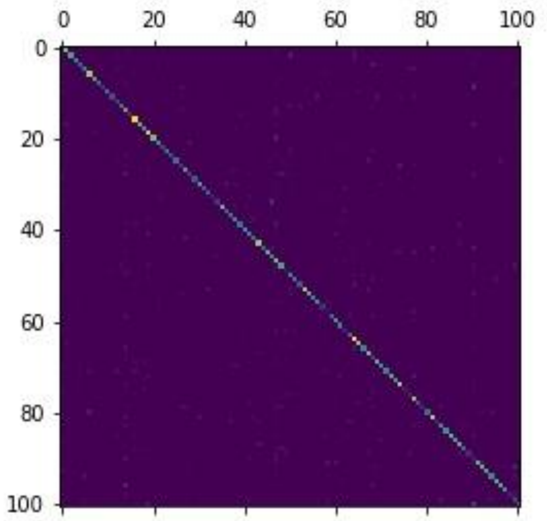

Figure 5: Confusion matrix for 101 classes

Most classification results are concentrated along the diagonal, while we still see noise distributed across the board.

6.3. Comparison with Previous Work

Comparing against our baseline model, Karpathy, et al. (2014) our model has improved test accuracy by 4% (table 2). It shows that a YOLO based adaptive window method outperforms fovea stream center crop. However, comparing with state-of-the-art, for example the paper published by Wang and his colleagues [3], there's significant room for improvement.

In Wang's paper, they utilized an additional stream – Object Stream. It captures the object that the person is interacting with. In actions like "Blow hair dry" and "Apply eye makeup", the person crops are usually human faces and the backgrounds are also similar. Therefore, object information could be very crucial for classification.

|  | Katharthy, et al. (2014) | Two-stream model in this paper | Wang, et al. (2017) |
|---|---|---|---|
| Testing accuracy | 65.4% | 68% | 92.6% |

Table 2 Testing accuracy compared with state of the art

7. Discussion

Compared with Karpathy's multi-resolution CNN architecture which comprises of a low-resolution context stream and a high-resolution fovea-stream center crop, our approach replaced the fovea-stream with a YOLO-based temporal stream which extracts human from the video and thus improved the classification performance.

However, the person cue is not perfect. We have seen a number of cases where there is no person in the crop or only an incomplete crop. Some cases are caused by how the original video was shot. For example, we have noticed that as camera zoomed out quickly, the person in the scene is too "thin" to have enough pixels for the YOLO detector to work. Other cases are caused by camera alternating front and back, such that the same person is shot from completely different angels. This further increased the difficulty in capturing common traits in videos of the same class.

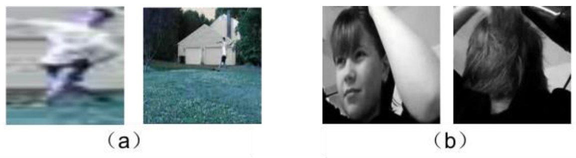

Figure 6: YOLO misdetection: (a) Camera zooming out too quickly to lose the Person cue. (b) Camera alternating front and back

8. Conclusion and Future Work

We proposed a two-stream video classification model with a spatial stream and human-crop temporal stream based on CNNs. We found that the fused two-stream model achieved significantly higher accuracy than each individual stream, and the temporal stream outperformed spatial stream.

As mentioned before, our future work will focus on improving human-detection accuracy. The current YOLO model that we used was trained on VOC07 dataset. Training YOLO on a human recognition dataset, and tuning hyper-parameter and threshold specifically for recognizing human will likely improve human detection accuracy. Moreover, we plan to try out region-based proposal methods such as faster RCNN, which has also seen great success in this field of research.

In addition, we plan to further improve network structure and classifiers. The first idea that we want to try out is to implement Long Short-term Memory (LSTM) network or other Recurrent Neural Network (RNN) structures on temporal stream so as to better capture temporal information. Secondly, we will further optimize fusion method, fine tune hyper-parameters, and learning rate. Last but not the least, we will explore explicit handling of camera zooming and alternation so as to build a more robust video action recognition system.

## References

[1] Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 1725-1732).

[2] Simonyan, K., & Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems* (pp. 568-576).

[3] Wang, Y., Song, J., Wang, L., Van Gool, L., & Hilliges, O. (2016). Two-Stream SR-CNNs for Action Recognition in Videos. BMVC.

[4] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. (2016). You Only Look Once: Unified, Real-Time Object Detection. CVPR

[5] Zuxuan Wu, Xi Wang, Yu-Gang Jiang, Hao Ye, Xiangyang Xue. (2015). Modeling Spatial-Temporal Clues in a Hybrid Deep Learning Framework for Video Classification. ACM MM.

[6] Khurram Soomro, Amir Roshan Zamir and Mubarak Shah, UCF101: A Dataset of 101 Human Action Classes From Videos in The Wild, CRCV-TR-12-01, November, 2012.

[7] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional neural networks. arXiv preprint arXiv:1311.2901, 2013.

[8] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Dense trajectories and motion boundary descriptors for action recognition. International Journal of Computer Vision, 103(1):60–79, 2013.

[9] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In CVPR, pages 4305–4314, 2015.

[10] Redmon, Joseph, and Ali Farhadi. "YOLO9000: Better, Faster, Stronger." arXiv preprint arXiv:1612.08242 (2016).

[11] Liu, Wei, et al. "SSD: Single shot multibox detector." European Conference on Computer Vision. Springer International Publishing, 2016.

[12] M. A. Goodale and A. D. Milner. Separate visual pathways for perception and action. Trends in Neurosciences, 15(1):20–25, 1992.

[13] J. Liu, J. Luo, and M. Shah. Recognizing realistic actions from videos “in the wild”. In CVPR, 2009.

[14] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, C. Schmid, et al. Evaluation of local spatio-temporal features for action recognition. In BMVC, 2009.

[15] H. Wang, A. Klaser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In CVPR. IEEE, 2011.

[16] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005.

[17] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In Proc. BMVC., 2014.

[18] Everingham, Mark, et al. "The PASCAL visual object classes challenge 2007 (VOC2007) results." (2007).

[19] Geoffrey Hinton, N Srivastava, and Kevin Swersky. 2012. Lecture 6a overview of mini–batch gradient descent. Coursera Lecture slides https://class. coursera. org/neuralnets-2012-001/lecture,[Online].