Human activity recognition from RGB videos

Undergraduate Thesis

Submitted in partial fulfillment of the requirements of
BITS F421T Thesis

By

Apoorva Gondimalla
ID No. 2015B2A70650G

Under the supervision of:

Dr. Ravi Kiran Sarvadevabhatla
&
Dr. Swati Aggarwal

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, GOA CAMPUS
December 2020

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, GOA CAMPUS

Abstract

Bachelor of Engineering (Hons.) Computer Science

Human activity recognition from RGB videos

by Apoorva Gondimalla

Human activity recognition from videos attributes to myriad of real life applications primarily dealing with human-centric problems. As Deep Learning set to become the heart of automation, recent work on vision-based human action recognition focuses on designing complex deep learning models for the task. Majority of these solutions are modeled for large training datasets. However, collecting and processing video data is usually very expensive and time consuming. Thus achieving equivalent performance with low data is very much essential. In addition to this, due to lack of depth information, RGB only videos perform poorly in comparison to RGB-D video based solutions. But acquiring depth information, inertia etc. is costly and requires special equipment, whereas RGB videos are available in ordinary cameras. This work deals with a specific action recognition task to automate surveillance at a manufacturing company. In this regard, the solution attempts to obtain significant performance for activity recognition from RGB only videos using low training data, thereby addressing both the issues through various techniques such as data augmentations, auto encoders etc.

Acknowledgements

I sincerely thank Dr. Ravi Kiran for giving me this opportunity to explore applicability of deep leaning in the field of computer vision under his guidance at Center for Visual Information Technology(CVIT) lab, International Institute of Information Technology, Hyderabad(IIITH). I also thank Dr. Avinash Sharma for guiding me during the thesis and Dr. Swati aggarwal for being the co-supervisor.

Contents

List of Tables

Abbreviations

MSE	**M**ean **S**quared **E**rror
CE	**C**ross **E**ntropy
CMC	**C**umulative **M**atching **C**haracteristic
NN	**N**earest **N**eighbours
AE	**A**uto **E**ncoder

Chapter 1

Human activity recognition from RGB videos

1.1 Problem Statement

Video Surveillance is used for various applications such as monitoring, investigation, hazard prevention etc. Manufacturing companies often use the data obtained from video surveillance for quality control activities such as improving the existing workflows by process-time analysis, process-accuracy evaluation etc.

Manual surveillance requires a large dedicated security group monitoring the workers and manually noting the data (In the specific case we dealt with there were processes with duration less than 30 seconds!). This task is not only exhausting but requires huge labour as each inspector can ideally monitor only one worker. In addition, this could potentially impose unease for the worker who is under constant observation of another human being.

Thus an automated surveillance system, which could categorize and recognize actions from a video largely reduces labour cost and strain while allowing comprehensive analysis through structured and detailed data. However, sequential action recognition is yet in its initial stages of improvement with many challenges. Therefore for a practical scenario, segmenting the video into smaller clips and performing action recognition on those segments is a more plausible solution.

This project attempts to automate classification of videos into sequence of actions by predicting action for video segments of certain interval. At first, the complete video is split into multiple overlapping segments and these segments are passed through our model for recognition.

The videos consisted of workflows related to manufacturing of various parts at a Japanese automobile company. I have contributed to research, design and development of a deep learning model for the action recognition.

1.2 Literature Review

In early days, traditional algorithms like the HMM [3] and SVM [9] have been proposed for developing predictive models for activity recognition. With the advancement of deep learning in recent years it has been possible to perform automatic high-level feature extraction to achieve promising performance in many areas. Deep learning architectures have been further explored extensively for video based human activity recognition and have shown encouraging results. The survey papers [4] and [2] establish the essence of deep leaning methods and the progress achieved in this field.

There are various challenges in action recognition ranging from human detection to action segmentation from a video. Skeleton information from RGB-D video has been widely studied to improve recognition accuracy [8], [5]. However, acquiring depth information, inertia etc. is costly and requires special equipment, on the other hand RGB video streams are available in ordinary cameras.

Detecting actions from RGB only videos witnessed the state-of-the-art architectures such as 3D-convolution networks(C3D)[6], Inflated 3D convolutions(I3D)[1] etc which helped overcome the limits encountered and explore more challenging tasks. Although these methods abundantly dealt with large publicly available datasets, it is often not possible to get such large amount of data as collection and annotation of videos is an expensive and time consuming process. Hence, this project attempts to achieve significant performance with a small dataset.

1.3 Data: Structure and Characteristics

The data provided by the manufacturing company has 7 different video types known as "operators" with different sequence of activities. These actions in each video are not mutually exclusive. Total number of actions in all the operators together is 75. The characteristics of the data were analysed using various plots like, Samples per class, Frames per class, Distribution of frames per instance per class etc through pie charts, bar graphs and violin plots. This investigation resulted in the following observations:

1.3.1 Characteristics of Data

- Small Data Size: The amount of annotated data available is very small.
- Skewed Data: The variation of number of samples per class is very large. Some classes have as low as one sample, while some have about 400 samples.
- Occlusion: The camera position is diagonally above the worker's head and conveyor belt. Thus sometimes the action is occluded by the worker's head.
- Colour: the worker's cap and the product are black in colour which could result in perceiving the product and his head together as one object.
- Similarity: There are actions with large similarity in hand movements or objects used. These would require high level feature recognition from the video for differentiation.
- Overlap: Some actions start before the previous action ends, while some actions take place simultaneously.
- Disturbance: Sometimes the view from the camera also covers part of neighbourhood operator's work space. This might lead to recognition of his action over the worker under inspection.
- Variability: Few actions could consists of two or more different types of work, such as "tightening the product" could include tightening the screw or the lid. Given the small amount of data present this would make the classification harder.
- Fixed Setup: Since all the action are taking place indoors in the same environment, the effect on classification due to variations in the surroundings is minimal.
- Predictable sequence: The actions taking place follow a known work flow, thus to a certain degree, probability could be used to predict next action in the sequence.
- Classes(Action types): The names of the actions are descriptive. That is, phrases explaining the action instead of single words. Thus similar actions could have similar phrases. For example classes like "pick up the jar", "pick up the tape" both are pick actions and this could be interpreted from their class names alone.

1.3.2 Initial Challenges

The data provided was manually annotated by the company workers. Manual annotation is always prone to errors, especially more in this scenario since the number of classes was also large and the actions were domain specific. As the data is small, few wrongly labelled annotations could have large impact on the model training.

Data cleaning was performed by rectifications such as adjusting intervals of actions precisely, correcting wrongly labelled annotations, removal of unknown/junk class labels etc. Further comparative analysis between actions based on similarities resulted in combining and removing few classes.

The videos consisted 3 different camera views, namely top, diagonally top and the other one slightly lower to the diagonal top view. Among these the top view videos could not be used because the head of the worker occludes 80-90% of the actions taking place making them redundant for recognition.

For videos, frame extraction is primary for performing any kind of feature analysis. The videos provided did not have uniform frame rate requiring re-annotation of action segments with frame numbers instead of given time stamps.

1.4 Performance Metrics

The standard performance metrics Precision, Recall and F1 Score were used for evaluation. To interpret the overall performance, median value of these metrics across all the classes was considered and the median absolute deviation was plotted as error bar.

Some of the classes in the data had only 2 samples. Since these cannot be used for any training, they were excluded. Though some of the other classes had at least 3 samples, having two training samples and one testing sample would not be sufficient to establish a consistent result.
Thus the data was evaluated in two different set of classes:

- Set1: All the classes having >=3 samples
- Set2: All the classes having >=20 samples

1.5 Baseline Experiments

Evaluating the given dataset with existing architectures provides a perception of the complexity and learning capability of the models allowing further analysis and improvement over a base model.

1.5.1 Existing architectures

State-of the-art neural network architectures pre trained on indoor action datasets such as charades, Something-Something and other datasets namely Sports1M, Imagenet etc were studied.

Based on the model performance and dataset similarity the following architectures were chosen for baseline analysis:

- **C3D:** 3D Convolutions[6]
  - Pretrained Dataset : Sports1M
  - Feature Dimension : 4096
- **I3D:** Inflated 3D Convolutions[1]
  - Pretrained Dataset : ImageNet + Kinetics400
  - Feature Dimension : 1024
- **TSN:** Temporal Segment Networks[7]
  - Pretrained Dataset : ImageNet
  - Feature Dimension : 1024
- **TRN:** Temporal Relational Reasoning in Videos[10]
  - Pretrained Dataset : Something Something-V2
  - Feature Dimension : 2048

I worked on evaluation of the dataset on C3D and TRN architectures. The features were extracted from the fully connected layer just before the softmax layer.

1.5.2 Establishing baseline for the performance

K-Nearest Neighbour was performed on the extracted features from the four architectures to set a baseline for the performance. The characteristics of the evaluation are:

- Train-test split = 70:30
- k values : ranging from one to ((minimum number of samples/class) * 0.7)
- Calculated median precision, recall and F1 score based on K-Nearest Neighbours
- Calculated median precision, recall and F1 score based on distance weighted K-Nearest Neighbours

1.5.3 Results

Among all the four architectures, I3D gave about 0.2 greater F1 score than the other architectures. A comparative analysis of the median results and Cumulative Match curve consistently indicated I3D as a significantly better architecture. Thus the features of I3D were chosen for further processing.

1.5.4 Cross Validation

A 3-fold cross validation was performed using the K-NN experiment. Each fold consists of 67% training and 33% testing split ratio. The difference between results for the folds was large which indicated bias in the data set. Hence the 3-fold setup is continued for evaluation throughout all the experiments further performed.

1.6 Data Augmentation: Dealing with low training data

After preprocessing, various data augmentation techniques were used to increase the training data size and therefore mitigate the data scarcity problem. This not only provides larger data for training but also helps to increase the robustness of the model developed by introducing variability.
Three types of spatial augmentation were generated:

- Crop: This targets change in camera position. Random crop of 5% of each frame in a sample from one of the four corners.
- Flip: Targets change in direction. Flip each frame of a sample about the vertical axis(i.e, Horizontal flip)
- Rotate: Targets change in camera angle. Randomly rotate each frame in a sample by +5 or -5 degrees.

Five temporal augmentations were generated:

- Temporal skip: Targets delayed actions or temporal differences in same action performed by different workers. Random removal of 20% of the frames.
  5 different sets of 20% frames were removed to generate 5 different temporal augmentations. This ensures that the primary part of the action is present.

After the augmentations were added to the training data and evaluation on K-NN showed a slight improvement in the performance. This ensured that the augmentations had no negative effect and could be used for training. Evaluating CMC curves with inclusion of augmented data clearly showed feature extraction from I3D architecture gave the best results.

1.7 Auto Encoders

Auto encoders allow efficient encoding of information into low dimension feature vectors. This helps in removal of unnecessary information and therefore in training the model more efficiently with low data. The concept of auto encoders states that if a higher dimension vector can be compressed to a lower dimension vector without any loss of information, the lower dimension vector essentially retains the core aspects of the particular image or action.

The feature vectors extracted from I3D are of 1024 dimension. Thus multiple auto encoder models were designed to achieve compression on these I3D features. `Table 1.1` describes the number of neurons in the fully connected multi layer perceptron architecture.

Name	AE Architecture
AE1	1024-512-256-512-1024
AE2	1024-512-1024
AE3	1024-256-1024

Table 1.1: Autoencoder Architectures

Training parameters after experimentation:

- Activation function : ReLU(Rectified Linear Unit)
- Reconstruction loss : MSE loss(Mean Square Error)
- validation set : Test set(due to data scarcity)
- Learning rate: 1e-3
- Dropout: 0.03
- Embedding extraction: before and after ReLU layer following the fully connected layer with 512 dimension for AE2 and 256 dimension for AE1, AE3.
- Data: I3D features of original data + I3d features of all nine augmentations
- Cross Validation: 3-fold(67:33 Train-test split)

1.7.1 Results and Inference

The analysis of the performance of 1-NN and distance weights 1-NN on the compressed features extracted after the encoder module indicated higher performance of AE1 over the other auto encoders. Although there was no significant difference in F1 score, AE1 consistently gave higher accuracy.

Despite of having large number of weights for AE1 compared to AE2 and AE3, the gradual decrease in dimension of fully connected layers compensated the lack of training data. In AE1 compressed features extracted after ReLU layer displayed better results(0.02).

1.8 Building the Classifier

The feature embeddings(compressed features) from auto encoder are generated with the motive of regenerating back the original data. Thus the model was modified to generate the embedding in a class-aware environment as the ultimate goal is classification rather than data reconstruction. This aims to train the model for better performance on classification task than finding 1-NN of compressed features.

Several variations of architecture were proposed to achieve this.`Table 1.2` summarizes these variations.

Classifier architecture: m $->$ m/4 $->$ number of classes
where m is the output dimension from encoder module of auto encoder(m=256 for AE1).

Name	Classifier Architecture
I	Pretrained encoder(Freeze) + Classifier
Ia	Encoder + Classifier
II	Pretrained encoder + Classifier
III	Encoder + Decoder + Classifier(branch from encoder)
IV	Pretrained (Encoder + decoder) + Classifier(branch from encoder)

Table 1.2: Classifier Variations

1.8.1 Variation I

Evaluating the effect of adding a classifier to encoded features of auto encoder. Pretrained weights of encoder part from the auto encoder are loaded and only the classifier part is trained

with cross entropy loss as loss function. Thus classifying the action from the compressed feature representing the video.

1.8.2 Variation Ia

It has same architecture as I. It attempts to infer the effect of directly trying to encode I3D features with respect to classification, both encoder module and classifier are trained from scratch with cross entropy loss as the loss function.

1.8.3 Variation II

It has same architecture as Ia. Instead of training from scratch the model tries to fine tune the Encoder part with pre trained weights loaded from auto encoder. This gives an initial base for the model to start training from rather than random weights. Both encoder and classifier are trained with cross entropy loss function.

1.8.4 Variation III

The classifier part branches from the 256 dimension neuron layer of AE1 auto encoder. This variation has two losses, MSE loss for the output of decoder and cross entropy loss for the output of the classifier. The decoder module acts as a regularizer on the features that convolute with respect to classifier.
Classification loss is given higher priority, thus the total loss function is defined as

$$\text{Loss} = 0.2 * \text{MSE Loss} + 0.8 * \text{CE Loss}$$

where both MSE loss and CE loss are normalized initially.

1.8.5 Variation IV

The Architecture is same as III. The encoder and decoder modules have pre trained weights loaded and thus are fine-tuned further with lower learning rate. On the other hand, the classifier module is trained from scratch with higher learning rate.

1.8.6 Results

The above five variations were performed with encoders AE1, AE2, AE3 resulting in 5*3 = 15 trained models with 3-fold cross validation. After evaluating with performance metrics precision, recall and F1 Score AE1 encoder with variation Ia resulted in best accuracy. This indicates that feature compression with respect to classification loss is more effective even with low data.

1.8.7 Refining the model

- **Class balanced loss:** The weight given to loss from each class varies based on inverse ratios of number of samples per class. This ensures that the effect of low sample classes on the delta of model weights is comparable to that of high sample classes.
  As expected, using class balanced loss has improved performance for low sample classes.

- **Batch resampling with or without class balanced loss:** Batch resampling is a method of oversampling low sample classes. During training, based on the percent of samples present in each class the probability of selection of instance of that class proportionately varies. That is, low sample class instances would be selected more times. This assists to overcome class imbalance.
  Batch resampling without class-balanced loss gave significant improvement in the performance.

- **Relevance of Temporal augmentation:** The data consists of actions which has duration as short as 15-30 frames. Having a temporal skip of 20% might potentially remove the major part of the action. Thus, temporal augmentations for samples with ¡30 frames were removed. This resulted in slight improvement in the performance.

- **Train-Test Split:** A 5-fold analysis was carried out with 80:20 train-test split in order to increase the training data. No notable difference was observed.

- **Creating a meta class:** An attempt was made to create a meta class called "other actions" by combining all the classes having <20 samples. However, this decreased the performance as the variability in the meta class is very high which effected the classification of other classes having >=20 samples.

- **Max or avg pooled prediction:** The probability vectors for prediction generated for original and all the 8 augmentations are max pooled or average pooled to get the final prediction. Pooling helps to capture features coherently from all the augmentations of the same action. This modification significantly improved the performance.

1.9 Aggregation of classes: redistribution

The initial 76 classes had actions that could be grouped, for instance, similar action performed on different objects. Thus the 76 classes were aggregated into 19 classes based on this action similarity.
The new 19 classes describe the 'verb' action taking place in the original 76 classes. This aggregation increased samples per class, with minimum being 15 samples. But the data was still skewed for distribution of samples among classes.
The classifier modelled was then trained for this 19 class set and the results were evaluated.

1.10 Final Results

Final model: Video Frames $->$ Pretrained I3D features $->$ 1024 $->$ 512 $->$ 256 $->$ 64 $->$ number of classes.

After refining the model by

- Batch resampling without class balanced loss
- Removing temporal skip augmentations for <30 frame instances
- Average pooling of prediction probabilities from all augmentations

the following results were obtained:

- Results for >=20 sample classes(original set of classes):
  - The final classifier model(`Table 1.4`) improved the performance by about 0.3 from that of baseline 1-NN(`Table 1.3`).
  - Total number of classes = 25

3-fold Cross validation	Median precision	Median Recall	Median F1 Score
Fold I	0.478	0.493	0.5
Fold II	0.428	0.429	**0.429**
Fold III	0.5	0.478	0.485

Table 1.3: 1-NN Performance for 25 classes

3-fold Cross validation	Median precision	Median Recall	Median F1 Score
Fold I	0.714	0.714	0.727
Fold II	0.75	0.72	**0.733**
Fold III	0.8	0.714	0.714

TABLE 1.4: Final Classifier Performance for 25 classes

- Aggregate Class results:
  - Final classifier(`Table 1.4`) showed a performance improvement of about 0.45 from that of baseline 1-NN(`Table 1.3`).
  - Total number of classes = 19

3-fold Cross validation	Median precision	Median Recall	Median F1 Score
Fold I	0.444	0.389	0.421
Fold II	0.4	0.39	**0.377**
Fold III	0.33	0.33	0.364

TABLE 1.5: 1-NN Performance for 19 classes

3-fold Cross validation	Median precision	Median Recall	Median F1 Score
Fold I	0.823	0.75	0.793
Fold II	0.843	0.776	**0.79**
Fold III	0.72	0.727	0.75

TABLE 1.6: Final Classifier Performance for 19 classes

1.10.1 Methods used for further analysis

- **Correlation plot:** Samples vs Class wise F1score, Frames vs Class wise F1score. To interpret the relationship of number of samples and total frames to the performance.
- **Confusion Matrix:** Inputs information on individual misclassified samples and provides an overview of inter class relationship. This assisted in final class aggregation
- **Median and deviation of evaluation metrics:** As mean is largely affected by extreme values, median was a better evaluation metric to consider for a coherent review.
- **Misclassification Plot:** A better visualization tool to understand the degree of similarity and confusion among classes.

- **Spearman and Pearson correlation:** A formal correlation rank of number of samples, frames and aggregations with performance. This metric indicated that number of frames was predominantly correlated with the performance for a class.

1.11 Conclusion

The final end-to-end architecture proposed for action recognition from RGB-only videos of an automobile manufacturing setup achieved an accuracy of 80%. Multiple experiments were conducted to enhance the recognition by addressing the challenges of limited and skewed training data, RGB-only modality.

Data augmentation was found to be more effective for testing(by pooling) than for training. Efficient auto encoder and classifier model was proposed with potential refinements using batch re-sampling and average pooling which resulted in significant performance improvement.

For future work, an automated sequential recognition model could be developed by exploiting contextual information of the given setup and by using LSTM(Long short-term memory) networks.

Written by —
Apoorva Gondimalla

Bibliography

[1] Joao Carreira and Andrew Zisserman. "Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset". In: *arXiv e-prints*, arXiv:1705.07750 (2017), arXiv:1705.07750. arXiv: 1705.07750 [cs.CV].

[2] Yu Kong and Yun Fu. "Human Action Recognition and Prediction: A Survey". In: *arXiv e-prints*, arXiv:1806.11230 (2018), arXiv:1806.11230. arXiv: 1806.11230 [cs.CV].

[3] Nuria Oliver, Eric Horvitz, and Ashutosh Garg. "Layered Representations for Human Activity Recognition". In: ICMI '02 (2002), p. 3. DOI: 10.1109/ICMI.2002.1166960. URL: https://doi.org/10.1109/ICMI.2002.1166960.

[4] Ronald Poppe. "A Survey on Vision-based Human Action Recognition". In: *Image Vision Comput.* 28.6 (June 2010), pp. 976–990. ISSN: 0262-8856. DOI: 10.1016/j.imavis.2009.11.014. URL: http://dx.doi.org/10.1016/j.imavis.2009.11.014.

[5] H. Rahmani and M. Bennamoun. "Learning Action Recognition Model from Depth and Skeleton Videos". In: (2017), pp. 5833–5842. DOI: 10.1109/ICCV.2017.621.

[6] Du Tran et al. "Learning Spatiotemporal Features with 3D Convolutional Networks". In: *arXiv e-prints*, arXiv:1412.0767 (2014), arXiv:1412.0767. arXiv: 1412.0767 [cs.CV].

[7] Limin Wang et al. "Temporal Segment Networks: Towards Good Practices for Deep Action Recognition". In: *arXiv e-prints*, arXiv:1608.00859 (2016), arXiv:1608.00859. arXiv: 1608.00859 [cs.CV].

[8] Pichao Wang et al. "Action Recognition Based on Joint Trajectory Maps Using Convolutional Neural Networks". In: MM '16 (2016), 102–106. DOI: 10.1145/2964284.2967191. URL: https://doi.org/10.1145/2964284.2967191.

[9] D. Zarpalas et al. "Activity Recognition from Silhouettes using Linear Systems and Model (In)validation Techniques". In: 1 (2006), pp. 347–350. ISSN: 1051-4651. DOI: 10.1109/ICPR.2006.210. URL: https://doi.ieeecomputersociety.org/10.1109/ICPR.2006.210.

[10] Bolei Zhou et al. "Temporal Relational Reasoning in Videos". In: *arXiv e-prints*, arXiv:1711.08496 (2017), arXiv:1711.08496. arXiv: 1711.08496 [cs.CV].