Item Removal Detection for Retail Environments with Neural Networks

Lingjie Kong
Stanford University
Department of Mechanical Engineering
ljkong@stanford.edu

Xingchen Fan
Stanford University
Department of Mechanical Engineering
xcfan@stanford.edu

Abstract

Inspired by the recent success of Recurrent Neural Networks (RNN) and Convolutional Neural Network (CNN) in classification, localization, and segmentation, we propose a neural network application for item removal detection in retail environments. In contrast to Amazon Go which relies on both sensor fusion and deep learning algorithms, we focus on using only deep learning to enable customers to explore and shop more efficiently. Unlike traditional image classification, the input into our network is a stream of video while the output is the prediction of class of removed items as well as the number of removed items. We will implement popular video classification algorithms, compare the performance, and explore other methods to enhance the prediction accuracy. Specifically, image-based 2D convolutional networks, 3D convolutional networks (C3D), two-stream convolutional networks and long-term recurrent convolutional network (LRCN) will be investigated. Eventually, we will explore the pros and cons of each networks and try to find the best technique to solve this item removal detection problem.

1. Introduction

Neural network models have been successfully applied to recognize human actions from images and videos. This paper explores how deep neural networks with computer vision can be used for action recognition in a very specific setting, namely item removal detection in retail environments. The most related technology in the market today is Amazon Go, where computer vision is combined with weight measurements from scales embedded in shelves to detect item removal in grocery stores. Our approach differs from Amazon Go such that we only use visual information to classify item removal and addition.

The input to our deep neural network model is video of people interacting with items on shelves in front of a vending machine. The camera is mounted at the top of the machine and triggered to record video only when the door is open. We will then use several different deep neural network architectures to classify whether items have been removed or added to the shelf within the time frame of the video. Since our own dataset of videos is small, we will use pretrained models available online and implement transfer learning to avoid overfitting. Different approaches will be investigated and compared in terms of classification accuracy as well as computational efficiency. We start with image-based approaches and gradually increase the complexity of our models. We will gradually transition from image-based approaches to video-based approaches as we incorporate more temporal information into our models.

2. Related work

So far, Convolutional Neural Network (CNN) [1] has outperformed most other algorithms in understanding image contents and shows the state-of-the-art performance in image classification, localization, segmentation and detection [2] [3] [4] [5] [6] [7]. The main reason is that CNNs are extremely power in extracting useful image features for specific tasks [8].

However, currently there is no single video classification benchmark dataset. Firstly, compared to images, videos are significantly more difficult to annotate. It takes large amount of time to collect a large enough dataset to train CNNs. Secondly, videos contain more information compared to images: in addition to spatial and appearance information in each isolated frame, videos also contain temporal information.

Several approaches have been developed to solve video classification problem. One method is to use pretrained image classification models to extract features from each frame and assemble image information with various fusion strategies like late fusion and slow fusion [11]. That paper also combines a low-resolution context stream and a high-resolution fovea steam to increase computational efficiency without sacrifice in accuracy. C3D stands for deep three-dimensional convolutional networks (3D ConvNet) [9]. Compared to image-based CNNs which apply a series of 2D convolutions, C3D simply stacks each frame of video together into a 3D tensor and apply a 3D convolutional filter in all intermediate hidden layers and some fully connected layers at the end. Two-steam convolutional networks explicitly incorporate temporal relationships into the network [10]. Unlike C3D which feeds stacked images extracted from the video, two-steam convolutional networks run a conventional 2D convolutional network to

extract spatial information and a separate optical flow-based network to extract temporal information. The results from both networks will be fused together and fed into fully connected layers for classification. Inspired by RNNs which are widely used in natural language processing (NLP) [12], long-term recurrent convolutional networks are also developed for visual recognition and description [13]. LRCNs use long short-term memory (LSTM) structure. At each time step, it feeds in both the hidden information from last time step as well as a new frame from the video.

3. Methods

We will mainly retrain pre-trained TensorFlow models for video classification on our dataset.

We first implement the C3D because it is easy to implement by only adding a depth dimension for 3D convolution with filter size 3x3x3 which has been shown to yield the best performance [9]. Stochastic gradient decent will be used to update the parameters for the last fully connected layer while keeping the hidden layer parameters fixed. Random search is used to find the best regularization to prevent overfitting.

We also plan to implement two-steam networks with pretrained models. We will form two steam convolutional networks for spatial steam from single frame and temporal steam from multi frame optical flow. Eventually, two frame results will be either concatenated together through fully connected layer and apply stochastic gradient decent to minimize the Softmax loss.

If time permits, we also wish to implement long-term recurrent convolutional networks. Each RNN layer will use an LSTM structure to prevent gradient vanishing. Hidden layer parameters will be from pre-trained model and only the fully connected layer parameters will be tuned on our dataset.

4. Dataset and features

Despite the fact that the pretrained models have been trained on some popular video datasets like UCF-101 and HMDB-51, we will retrain the models on our own dataset, which currently consists of 14 manually recorded and labeled videos (101 useful snippets) but will expand in the future. An example of our raw video frame is shown in Figure 1.

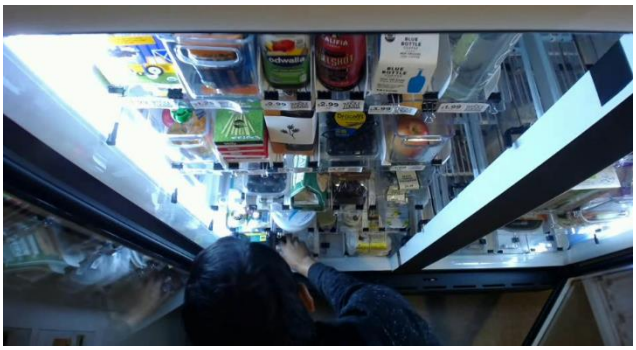

Figure 1: Raw video frame example

The raw videos are recorded at 24 frames per second with 960×540 pixels. For each video, how the person interacts with items on the shelves could vary a lot. This could change the lengths of videos significantly. In order to simplify our task, all videos in training and validation sets will be split into several snippets that consist of only one direction of hand motion: either moving towards or away from the shelf. When the hand is moving closer to the shelf, we will classify the action as either one item added or nothing added; when the hand is moving away, we will classify it as either one item removed or nothing removed. During test or real-world implementation, we assume that some hand motion detection algorithm will split the video automatically before the classification by our neural network. Details of hand detection and more variations on addition and removal are beyond the scope of this paper and will be interesting to study in the future. The sampled video snippets of interest will have 20 to 40 frames. Depending on the method, we will either use all frames or further extract a subset of frames for classification. The final frames of interest will be cropped at margins and downsampled spatially to give a size of no more than 128× 128 pixels per frame.

Furthermore, we will explore whether certain handcrafted features or data augmentation procedures could improve classification. Ideally, convolutional layers will learn to extract features from frames at various scales automatically throughout training. However, some preprocessing of data could speed up training and reduce the complexity of the network. Common techniques to implement include principal component analysis (PCA) and histogram of oriented gradients (HOG).

5. Experiments/Results/Discussion

Before running C3D, two-steam, and LRCN, we first built up our data pipeline and implemented a dummy model as the baseline.

In data preprocessing, we sample 5 frames per video snippet and down sample its dimension to 64×64×3. As a result, our input data have a dimension of 101×5×64×64×3 with dimensions as batch size, input depth, input height, input width, and input channels. In total, we have 101 data

samples, we split the data into 81 for training, 10 for validation, and 10 for testing.

For the first approach, we are inspired by the CNNs for image classification, we first convert three color channels into one gray-scale channel and reshape the data to 81×64×64×5. The later steps are the same as any classical image classification algorithm. We coded a two-layer neural network: a convolutional layer with 16 filters of size 7×7×5, followed by a fully connected layer with an input dimension of 13456 and an output of 4 classes. After training the model by using stochastic gradient decent for 20 epochs, we get a final training accuracy of 0.988, validation accuracy of 0.5, and test accuracy of 0.6 which outperforms random guess.

Furthermore, we implement a C3D two-layer neural network model. The first layer is a 3D convolutional layer with input size 101×5×64×64×3, filter size 3×3×3×3×4, while each dimension is represented by filter depth, filter height, filter width, input channels, and output channels. We pass the result from the first layer into a fully connected layer with a dimension of 7688 and an output dimension of 4. We also add a L2 regularization with a regularization weight of 0.05 to prevent overfitting. We obtain a training accuracy of 1, validation accuracy of 0.4 and test accuracy of 0.5, which also outperforms random guess.

For the next step, we will collect more data and apply transfer learning to multiple methods mentioned above.

References

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[2] A. Krizhevsky, I. Sutskever, and H. Geoffrey E., "ImageNet Classification with Deep Convolutional Neural Networks," *Adv. Neural Inf. Process. Syst. 25*, pp. 1–9, 2012.

[3] C. Farabet, C. Couprie, L. Najman, and Y. Lecun, "Learning hierarchical features for scene labeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1915–1929, 2013.

[4] D. Ciresan, A. Giusti, L. Gambardella, and J. Schmidhuber, "Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images," *Nips*, pp. 1–9, 2012.

[5] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks," *arXiv Prepr. arXiv*, p. 1312.6229, 2013.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[7] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 512–519.

[8] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks arXiv:1311.2901v3 [cs.CV] 28 Nov 2013," *Comput. Vision–ECCV 2014*, vol. 8689, pp. 818–833, 2014.

[9] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2016, vol. 11–18–Dece, pp. 4489–4497.

[10] K. Simonyan and A. Zisserman, "Two-Stream Convolutional Networks for Action Recognition in Videos," *arXiv Prepr. arXiv1406.2199*, pp. 1–11, 2014.

[11] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. F. Li, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[12] A. Karpathy and F. F. Li, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07–12–June, pp. 3128–3137.

[13] J. Donahue *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07–12–June, pp. 2625–2634.