Automatic Labelling of Videos using Deep Neural Network

CSE 847
Project Progress Report

Tarang Chugh
chughtar@msu.edu

Rahul Dey
deyrahul@msu.edu

1. PROBLEM DESCRIPTION

In the last decade, the world wide web has witnessed a massive explosion of multimedia data due to the rise of online media-sharing services such as Youtube, Facebook, etc. While many studies have tackled the problem of analyzing and understanding static images [1], improvements in video understanding research has been lagging due to unavailability of large-scale labelled dataset. However, analysis and understanding of video content on these video-sharing services provide insights to make the interface more engaging and customized to individual needs. Relevant search results, improved video recommendations, and content filtering are some of the many desired properties, which require the understanding of video content. In this project, we aim to analyze and understand video content to accurately assign labels. Google's recent release of the Youtube-8M dataset [2] accompanied with the launch of the Kaggle competition[1] to benchmark the performance on this task, has motivated us to take up this arduous task. We plan to take a deep learning based approach to design the classifier. We will also make use of the Google Cloud Machine Learning[2] available to the participants of this competition.

2. SURVEY

Convolutional Neural Networks (CNN) have emerged as a powerful tool for image based automation tasks such as recognition, segmentation, enhancement, encoding etc. It has become the preferred choice for such tasks as compared to other machine learning algorithms such as SVM, k-NN, Multilayer Neural Networks, etc. because of their unique ability to extract useful features from two dimensional inputs. However, when it comes to spatio-temporal inputs such as videos, CNN in itself is often found to be performing poorly. In our survey, we came across several approaches adopted by researchers to perform tasks such as video classification and labelling. We will discuss some of the approaches here briefly.

In case of videos, traditional CNN architectures suffer due to three main issues. The first issue relates to the large amount of computational resources required to train a CNN for classifying videos. For instance, videos of frame size 178 × 178, takes weeks to train a CNN using powerful GPUs. Performance of a deep neural network relies on the abundance of data. Thus, in case of videos, the size of data becomes even more, because of their three dimensional (two spatial and one temporal) nature. The second main issue is that CNN doesn't have any inbuilt mechanism to learn temporal patterns in the input data. And lastly, videos are usually of varying time duration. Even if all the input videos are resized to a particular resolution, different durations make them harder to be fed to a predesigned fixed network.

Several researchers have taken different approaches to overcome some of these issues. Karpathy et al. [3] suggested an approach where the input video is divided into two separate streams of data, a *context* stream that learns low resolution features in the frames, and a *fovea* stream that learns high resolution features from the middle portion of the video. Thus, the effective resolution can be reduced substantially. This takes advantage of the camera bias since the object of interest is usually in the center of the frame. Another approach relies on treating an entire slice of video having T number of frames and feeding them into a modified first convolutional layer by extending them to be of size $rows \times columns \times 3 \times T$. When successive slices are fed into the CNN, the first layer can find out the difference in them, thus characterizing the global motions in the video. Similar attempts have been made, such as in [4], to extend CNN for multimedia input by treating space and time as equivalent dimensions of the input, and performing convolutions in both time and space.

In recent years, Recurrent Neural Networks have come up as a promising tool in learning sequential and temporal data. They have been applied successfully in many speech and text based tasks such as recognition, translation, sentiment analysis, synthesis, etc. Recently, Venugopalan et al. [5] modeled frames of videos using pre-trained CNN and sequence of words using a pre-trained RNN on images associated with the sentence captions to translate videos to natural language. We plan to combine the power of CNN to learn spatial data and RNN to learn temporal data in our goal of understanding videos. The final goal of this project is to automatically recognize objects and activities in videos to improve the performance of auto-labeling, video recommendations, content filtering, retrieval, and indexing, etc.

3. DATASET

The dataset is provided by Youtube in the Google Cloud & Youtube-8M Video Understanding Challenge website[3]. The Youtube8m dataset contains both video level and frame level (1 sec. granularity) visual as well as audio features for 8Million videos belonging to 4000+ classes. Inception-V3 image

[1]https://www.kaggle.com/c/youtube8m
[2]https://cloud.google.com/ml/
[3]https://research.google.com/youtube8m/download.html

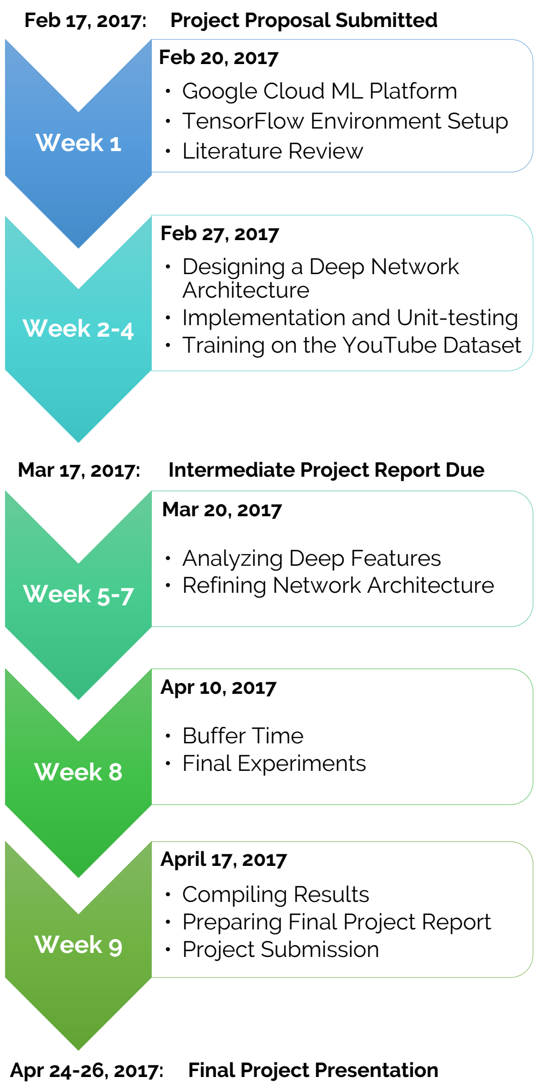
Feb 17, 2017: Project Proposal Submitted
Feb 20, 2017
Google Cloud ML Platform
TensorFlow Environment Setup
Literature Review
Week 1
Feb 27, 2017
Designing a Deep Network Architecture
Implementation and Unit-testing
Training on the YouTube Dataset
Week 2-4
Mar 17, 2017: Intermediate Project Report Due
Mar 20, 2017
Analyzing Deep Features
Refining Network Architecture
Week 5-7
Apr 10, 2017
Buffer Time
Final Experiments
Week 8
April 17, 2017
Compiling Results
Preparing Final Project Report
Project Submission
Week 9
Apr 24-26, 2017: Final Project Presentation

Figure 1: Timeline presenting the targeted milestones in the project.

annotation model [6], trained on ImageNet database [7], was used to extract the visual features from the bottleneck layer which is 2048 dimensional, and VGG based acoustic model was used to extract the audio features. In the case of frame-level features, these visual and audio representations extracted from each frame are PCAed and quantized to result in a 1024-dimensional visual feature, and 128-dimensional audio feature. For video-level features, these frame-level features have been averaged out. Fig. 2 presents the most frequent top-20 class labels in the database.

4. CURRENT PROGRESS

4.1 Environment Setup

Google provides a $300 worth credit for competition participants to use their Google Cloud ML Platform. This platform provides an environment for accessing the large training and testing files, writing code in tensorflow, submitting training/testing jobs on their GPU resources, and downloading the predictions file for submission to the competition.

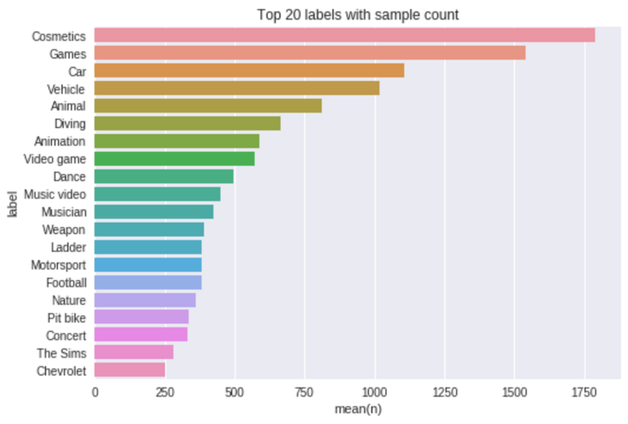

Figure 2: Top-20 frequent class labels in Youtube-8M dataset.

4.2 Understanding Deep Neural Networks (CNN & RNN)

One of the main goals of this project is to make ourselves well-acquainted with deep neural networks, specifically Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). Both CNN and RNN are evolved from ordinary Neural Networks.

4.2.1 Convolutional Neural Networks

CNNs are specialized neural networks for mapping 2-D and 3-D spatial information, such as in images and videos, to the desired outputs. A typical CNN architecture consists of several convolutional layers, pooling layers, fully-connected layers and typically a softmax layer at the end that is directly mapped to the output labels. Each convolutional layer maps a set of input 2-D feature maps to another set of feature maps using a shared kernel of weights (typical kernels are of sizes 3x3, 5x5, 7x7 etc.). The pooling layers remove redundancies and sub-sample the input feature maps. Thus, the convolutional layers learn low-level spatial features in images such as edges and arbitrary shapes and the fully-connected layer summarizes the low-level features into high-level shapes and objects in the input image.

4.2.2 Recurrent Neural Networks

Recurrent Neural Networks, on the other hand, are good for learning sequential data. As the name suggests, the output at one-time step is fed back into the network in the next time-step in a recurrent way. Thus, it tries to learn from the temporal relations in the data. As a result of the problem of vanishing gradients, a simple RNN is often unable to learn long temporal relationships in the data, concentrating on only the more recent inputs. This problem has been addressed by two prominent RNN models - the Long Short Term Memory (LSTM) and the Gated Recurrent Units (GRU). These models, in addition to the basic RNN unit, use a set of gates that control the flow of information from the current input and the previous output in the computation of the current output. They also typically use a 'forget' gate to control the rate of 'forgetting' past data.

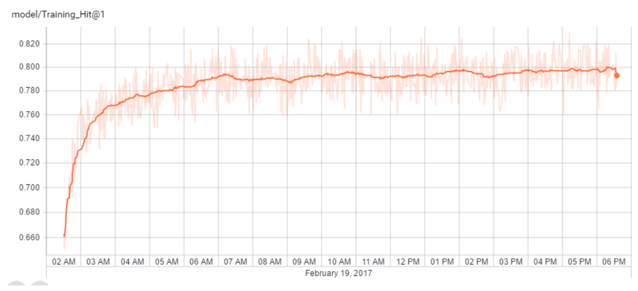

Figure 3: Tensorboard display of Rank-1 training accuracy during the training of LogisticModel.

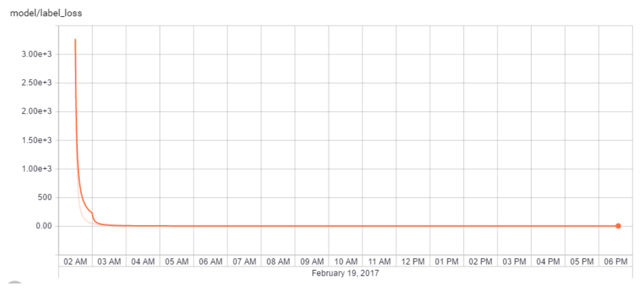

Figure 4: Tensorboard display of training loss during the training of LogisticModel.

4.3 Learning Tensorflow

TensorFlow is a python based open source library for complex numerical computations using tensors and graphs. It has been successfully applied in several deep learning applications because of the many features it offers including parallel deployment of computation to several CPUs or GPUs, data visualization toolkit etc. During the past month, we tried to learn and understand the TensorFlow library. We specifically focussed on modeling several Convolutional Neural Network architectures starting from the toy MNIST dataset to learning more diverse datasets using Inception-v3 and VGGNet architectures. We also modeled Recurrent Neural Network architectures using LSTM or GRU. In our architecture, we plan on feeding the input frame-level features from the dataset directly to a vector of RNN cells (LSTM or GRU). The last RNN cell will give the video level labels.

4.4 Performance Baseline

A starter code is provided by the competition hosts to check if the environment has been setup correctly and to understand the baseline performance. With the help of the starter code, we successfully trained a LogisticModel[4] with L2 regularization on the video-level features (See Figs. 3 and 4 presenting the training performance visualized using Tensorboard). A LogisticModel learns a linear projection of features to the class labels, which are then treated with a sigmoid function to result in class probabilities. We achieved an average Rank-1 accuracy of 72.4% when evaluated 5000 examples. We did not complete the full evaluation as this was only a test for correct setup. To utilize higher granularity frame level features and with the focus on to improve the performance, we are currently working on utilizing the RNN models which are known to be well-suited for sequence data such as video frames or audio signals.

5. REMAINING MILESTONES

5.1 Explore LSTM/GRU models

The starter code that we tried implemented a very basic logistic model using the video-level features. We plan to implement an RNN model using both video as well as frame-level features to exploit the temporal nature of the input. Since the input dataset already consists of features trained using CNN and quantized after PCA, they represent high-level spatial information from each frame. This amounts to feeding object-level information to the RNN in order to understand the class labels associated with the videos.

[4]https://github.com/google/youtube-8m/blob/master/video_level_models.py

5.2 Utilizing Audio Features

We will explore the performance of RNN models trained on the audio features independently and one concatenated with visual features, to understand the usefulness of audio features.

5.3 Model FineTuning

As with any machine learning model, we will fine-tune our model hyper-parameters using cross-validation with the provided validation set. These hyper parameters include the number of parallel RNN cells in the architecture, learning rate, dropout (regularization), loss function, etc. We plan to perform sufficient number of trials on the Google Cloud ML platform in order to optimize the hyper-parameters.

5.4 Score Level Fusion

We will explore the possibility of fusing multiple RNN models based on visual and audio features at score level to investigate if fusion can help in improving the overall performance.

6. REFERENCES

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[2] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016.

[3] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[4] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.

[5] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using

deep recurrent neural networks. *arXiv preprint arXiv:1412.4729*, 2014.

[6] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.