Classification of YouTube Videos using Deep Learning Methods

Jayaram Kuchibhotla
Department of Computer Science
University at Buffalo
Person #:50208766
jayaramk@buffalo.edu

Abstract

This project report describes the methods implemented for YouTube video recognition task. The problem statement is presented by Google Cloud & YouTube-8m video understanding challenge. Apart from the Video Level logistic regression, Frame Level Logistic regression, LSTM (Long Short Term Memory) models provided in the starter code, the newer models such as GRU (Gated Recurrent Unit), BI-LSTM (Bidirectional LSTM), and BI-GRU (Bidirectional GRU) are trained and validated on the Google Cloud Platform. The report has the following steps 1) Introduction 2) Description of the dataset 3) Explanation of above-mentioned methods 4) Discussion of results 5) Problems and Possible Improvements 6) Conclusion

1 Introduction

The deep learning methods are applied successfully in the field of Computer Vision for classifying large-scale image and video datasets such as ImageNet. The objective of automating the task to analyze the content of the videos could possibly be used in applications such as filtering and categorization. As the size of dataset increases, the complexity in recognizing the content of the videos increases. The availability of large video dataset and this public challenge from Google & YouTube helped people to come up with efficient methods in short period. The top scorers in this competition mainly obtained good results using the ensemble of different deep learning methods

2 Description of the dataset

The dataset consists of video data related to areas such as sports, activities, animals, foods, products, tourist attractions, games etc. The aim of the dataset is to understand what is present in each frame of the video just by using the topic label. The number of topics or labels in this dataset is 4716 from 24 diverse categories. The average number of labels per video is 3.4

The dataset is organized as follows:

Partition	Train	Validate	Test	Total
Number of videos	4,906,660	1,401,828	700,640	7,009,128

Each video has both frame level features and video level features.

Frame-level features are stored as tensorflow.SequenceExample protocol buffers

```
context: {
  feature: {
    key  : "video_id"
    value: {
      bytes_list: {
        value: [YouTube video id string]
      }
    }
  }
  feature: {
    key  : "labels"
```

```
      value: {
        int64_list: {
          value: [1, 522, 11, 172] # The meaning of the labels can be found here.
        }
      }
    }
}

feature_lists: {
  feature_list: {
    key  : "rgb"
    value: {
      feature: {
        bytes_list: {
          value: [1024 8bit quantized features]
        }
      }
      feature: {
        bytes_list: {
          value: [1024 8bit quantized features]
        }
      }
      ... # Repeated for every second of the video, up to 300
  }
  feature_list: {
    key  : "audio"
    value: {
      feature: {
        bytes_list: {
          value: [128 8bit quantized features]
        }
      }
      feature: {
        bytes_list: {
          value: [128 8bit quantized features]
        }
      }
    }
    ... # Repeated for every second of the video, up to 300
  }

}
```

The total size of the frame-level features is 1.71 Terabytes. They are divided into 4096 shards

Video-level features are also stored as tensorflow.Example protocol buffers

```
features: {
  feature: {
    key  : "video_id"
    value: {
      bytes_list: {
        value: [YouTube video id string]
      }
    }
  }
  feature: {
    key  : "labels"
    value: {
      int64_list: {
        value: [1, 522, 11, 172]
      }
    }
  }
  feature: {
    key  : "mean_rgb" # Average of all 'rgb' features for the video
    value: {
      float_list: {
        value: [1024 float features]
      }
    }
  }
  feature: {
    key  : "mean_audio" # Average of all 'audio' features for the video
    value: {
      float_list: {
        value: [128 float features]
      }
    }
  }
}
```

The total size of the video-level features is 31 Gigabytes. They are divided into 4096 shards

The visual features were extracted by the organizers using Inception-V3 image annotation model, trained on ImageNet. The audio features were extracted using a VGG-inspired acoustic model described in Hershey et. al. on a preliminary version of YouTube-8M. Both the visual and audio features were PCA-ed and quantized to fit on a single hard disk

3 Explanation of methods

3.1 Video Level Methods

a) **Video level Logistic Regression**

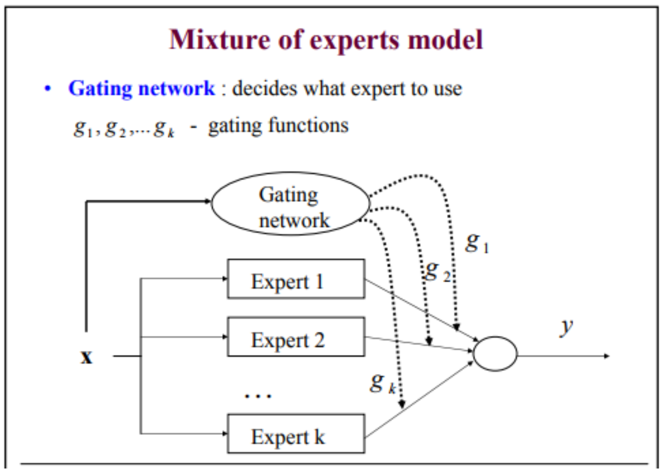
The input video–level features of each video which is a tensor of dimension (batch_size = 128 or 256 , num_features = 1024 or 128 or 1152)) .User can choose to input the video features(length = 1024) or audio features(length = 128) or concatenation of both (length = 1152). Each batch is given as input to the fully connected layer which has sigmoid activation, average pooling, L2 weight regularization with number of classes (vocab size) = 4716 to generate the predictions

3.2 Frame Level methods:

a) Frame level Logistic Regression

For all the frame level models described below the input is a tensor of dimension (batch_size = 128 or 256, num_frames = 300 , num_features = 1024 or 128 or 1152))) .The same strategy as mentioned for the video level logistic regression is used here as well

All the frame level models described below(except the above one) would use 'Mixture of experts' (Moe)(provided in the starter code) in the final/output stage. Mixture of experts is based on the divide-and-conquer principle in which the problem space is divided between a few neural network experts, supervised by a gating network



Mixture of experts model

- **Gating network** : decides what expert to use

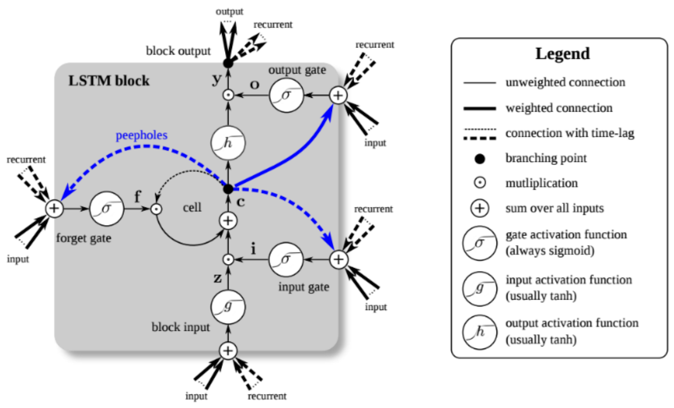
  $g_1, g_2, \dots g_k$ - gating functions

Learning Moe involves learning the parameters of individual expert networks and parameters of the gating network. Based on the probability values obtained from the gating network the space is divided–partitions belongs to different experts

In this project, the number of mixtures used = 2, gate activations(size = vocab_size * (num_mixtures + 1) and expert activations (size = vocab_size * num_mixtures) are created using the fully connected layers (with l2 regualrization) . Gating distribution is formed by using softmax activation of gate activations .Expert distribution is formed by using sigmoid activation of expert activations
Gating and Expert distributions are used to form the final probabilities or prediction values

b) Long short-term memory (LSTM)
LSTM is a recurrent neural network (RNN) architecture that remembers values and used for series classification. Videos consists of variable length images and LSTM is best suited for classifying them

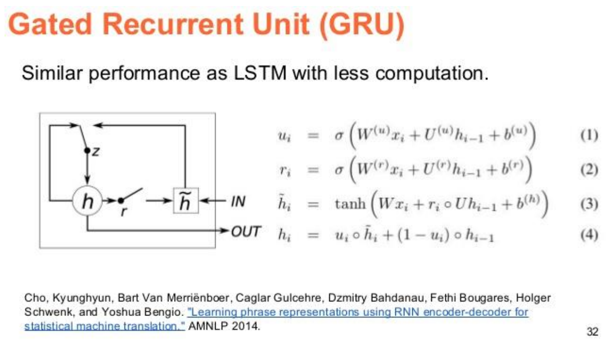
Diagram of a single LSTM cell



In this project (in the starter code),the number of layers are used are two and each layer consists of 1024 cells .

c) Gated Recurrent Unit(GRU)

GRU is also recurrent neural network (RNN) architecture that remembers values and used for series classification. It has fewer parameters compared to LSTM and lacks an output cell. It can be trained relatively fast compared to LSTM newtork



Gated Recurrent Unit (GRU)

Similar performance as LSTM with less computation.

$$u_i = \sigma\left(W^{(u)} x_i + U^{(u)} h_{i-1} + b^{(u)}\right) \quad (1)$$

$$r_i = \sigma\left(W^{(r)} x_i + U^{(r)} h_{i-1} + b^{(r)}\right) \quad (2)$$

$$\tilde{h}_i = \tanh\left(W x_i + r_i \circ U h_{i-1} + b^{(h)}\right) \quad (3)$$

$$h_i = u_i \circ \tilde{h}_i + (1 - u_i) \circ h_{i-1} \quad (4)$$

Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." AMNLP 2014.

32

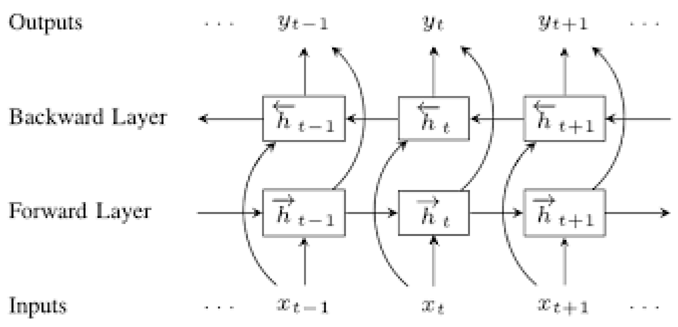
In this project (code developed by self),the number of layers are used are two and each layer consists of 1024 cells .

d) Bidirectional Recurrent Neural Networks

1) Bidirectional LSTM(BI-LSTM)
2) Bidirectional GRU(BI-GRU)

Recurrent neural network (RNNs) also have restrictions, as the future input information cannot be reached from the current state. On the contrary, BRNNs do not require their input data to be fixed. Moreover, their future input information is reachable from the current state. The basic idea of BRNNs is to connect two hidden layers of opposite directions to the same output. By this structure, the output layer can get information from past and future states.

BRNN are especially useful when the context of the input is needed. If the information from either ends of video is used, video context can be easily inferred



In the current project for both BIRNNs (code developed by self), one forward layer and backward layer are used, The states of both forward layer and backward layer are concatenated together to form an output state.

5 Discussion of results

The accuracy of the results can be inferred from the following parameters

1. Hit @1
   This is the fraction of test samples that contain at least one of the ground truth labels in the top k(k=1) predictions

2. Precision at Equal Recall Rate
   It is video level annotation precision when we retrieve the same number of entities per video as there are in the ground-truth.

3. Global average precision:
   The average precision, approximating the area under the precision-recall curve, can then be computed as where P is Precision and R is recall

$$\mathrm{AP} = \sum_{j=1}^{10000} P(\tau_j)[R(\tau_j) - R(\tau_{j+1})],$$

   The mean/global average precision is computed as the unweighted mean of all the per-class average precisions.

All the performance values range between [0,1] (higher values indicate better performance)

These performance measures are calculated for training and validation datasets only

The results of testing dataset is meant for judgement in the competition .Top 20 label and confidence values are generated in a .csv file for each video or row. It is not feasible to calculate these measures using the starter code for the test dataset.

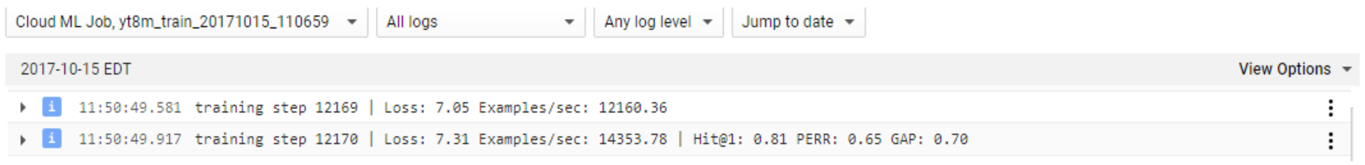
The snapshots of logs for each one of the 9 variants is as follows

(Owing to the huge size of the dataset,large training duration, Google cloud credit expense) training and validation is restricted to limited number of samples and limited duration

The last line in the log of each snapshot is the final performance measure for that method.

1. Video Level logistic Model (training duration: 42 min 53 sec)

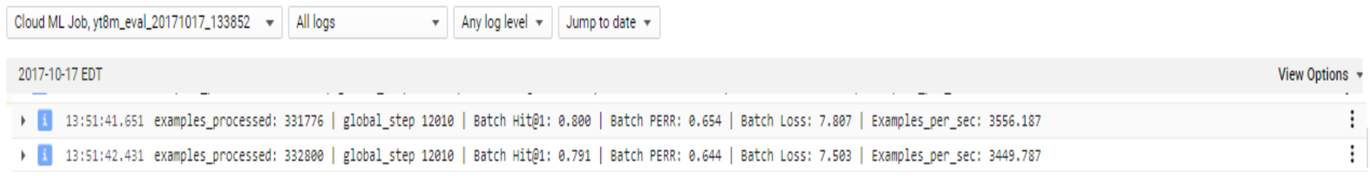
Training:



```
11:50:49.581  training step 12169 | Loss: 7.05 Examples/sec: 12160.36
11:50:49.917  training step 12170 | Loss: 7.31 Examples/sec: 14353.78 | Hit@1: 0.81 PERR: 0.65 GAP: 0.70
```

Training accuracy (Hit @1) = 81%

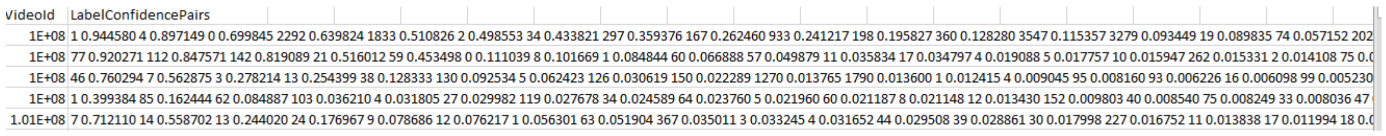
Validation:



```
13:51:41.651  examples_processed: 331776 | global_step 12010 | Batch Hit@1: 0.800 | Batch PERR: 0.654 | Batch Loss: 7.807 | Examples_per_sec: 3556.187
13:51:42.431  examples_processed: 332800 | global_step 12010 | Batch Hit@1: 0.791 | Batch PERR: 0.644 | Batch Loss: 7.503 | Examples_per_sec: 3449.787
```

Validation accuracy (Hit @1) = 79.1%

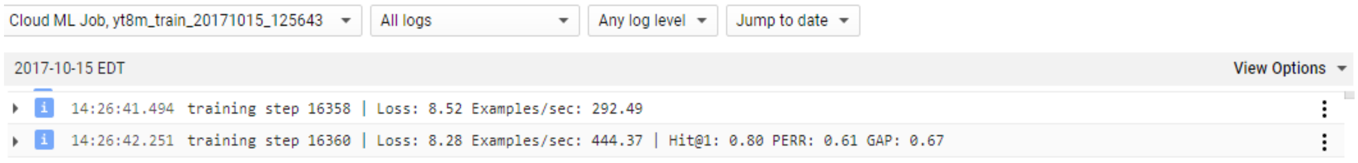
Testing(predictions.csv ,being shown only for this method) :



VideoId	LabelConfidencePairs
1E+08	1 0.944580 4 0.897149 0 0.699845 2292 0.639824 1833 0.510826 2 0.498553 34 0.433821 297 0.359376 167 0.262460 933 0.241217 198 0.195827 360 0.128280 3547 0.115357 3279 0.093449 19 0.089835 74 0.057152 202
1E+08	77 0.920271 112 0.847571 142 0.819089 21 0.516012 59 0.453498 0 0.111039 8 0.101669 1 0.084844 60 0.066888 57 0.049879 11 0.035834 17 0.034797 4 0.019088 5 0.017757 10 0.015947 262 0.015331 2 0.014108 75 0.0
1E+08	46 0.760294 7 0.562875 3 0.278214 13 0.254399 38 0.128333 130 0.092534 5 0.062423 126 0.030619 150 0.022289 1270 0.013765 1790 0.013600 1 0.012415 4 0.009045 95 0.008160 93 0.006226 16 0.006098 99 0.005230
1E+08	1 0.399384 85 0.162444 62 0.084887 103 0.036210 4 0.031805 27 0.029982 119 0.027678 34 0.024589 64 0.023760 5 0.021960 60 0.021187 8 0.021148 12 0.013430 152 0.009803 40 0.008540 75 0.008249 33 0.008036 47
1.01E+08	7 0.712110 14 0.558702 13 0.244020 24 0.176967 9 0.078686 12 0.076217 1 0.056301 63 0.051904 367 0.035011 3 0.033245 4 0.031652 44 0.029508 39 0.028861 30 0.017998 227 0.016752 11 0.013838 17 0.011994 18 0.0

2. Frame Level Logistic Model with only video features (train duration: 1 hr 34 min)

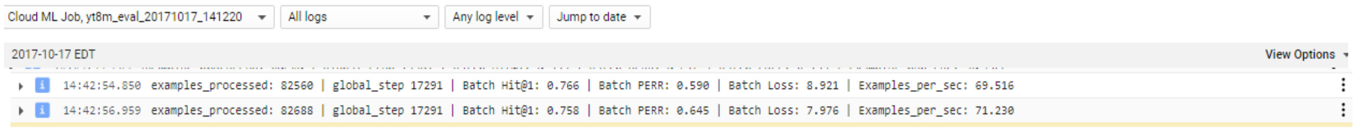
Training:



```
2017-10-15 EDT
14:26:41.494 training step 16358 | Loss: 8.52 Examples/sec: 292.49
14:26:42.251 training step 16360 | Loss: 8.28 Examples/sec: 444.37 | Hit@1: 0.80 PERR: 0.61 GAP: 0.67
```

Training accuracy (Hit @1) = 80%

Validation:

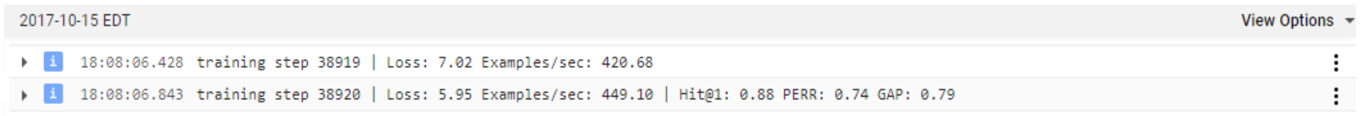


```
2017-10-17 EDT
14:42:54.850 examples_processed: 82560 | global_step 17291 | Batch Hit@1: 0.766 | Batch PERR: 0.590 | Batch Loss: 8.921 | Examples_per_sec: 69.516
14:42:56.959 examples_processed: 82688 | global_step 17291 | Batch Hit@1: 0.758 | Batch PERR: 0.645 | Batch Loss: 7.976 | Examples_per_sec: 71.230
```

Validation accuracy (Hit @1) = 64.5%

3. Frame Level Logistic Model with video and audio features (4 hr 6 min)

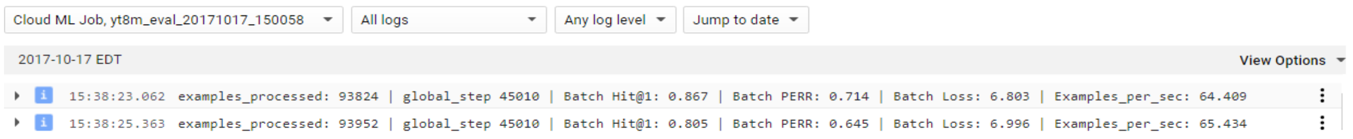
Training:



```
2017-10-15 EDT
18:08:06.428 training step 38919 | Loss: 7.02 Examples/sec: 420.68
18:08:06.843 training step 38920 | Loss: 5.95 Examples/sec: 449.10 | Hit@1: 0.88 PERR: 0.74 GAP: 0.79
```

Training accuracy (Hit @1) = 88%

Validation:

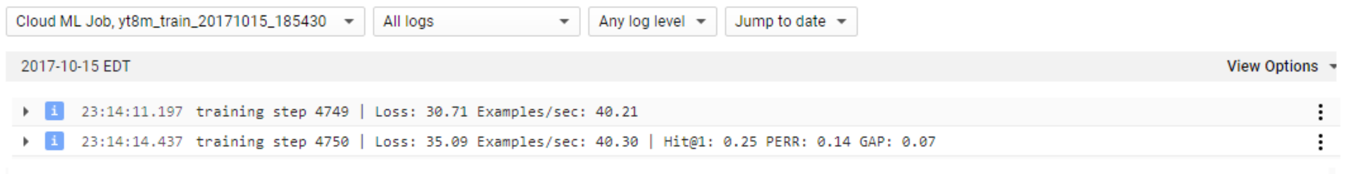


```
2017-10-17 EDT
15:38:23.062 examples_processed: 93824 | global_step 45010 | Batch Hit@1: 0.867 | Batch PERR: 0.714 | Batch Loss: 6.803 | Examples_per_sec: 64.409
15:38:25.363 examples_processed: 93952 | global_step 45010 | Batch Hit@1: 0.805 | Batch PERR: 0.645 | Batch Loss: 6.996 | Examples_per_sec: 65.434
```

Validation accuracy (Hit @1) = 80.5%

4. LSTM with only video features (4 hr 46 min)

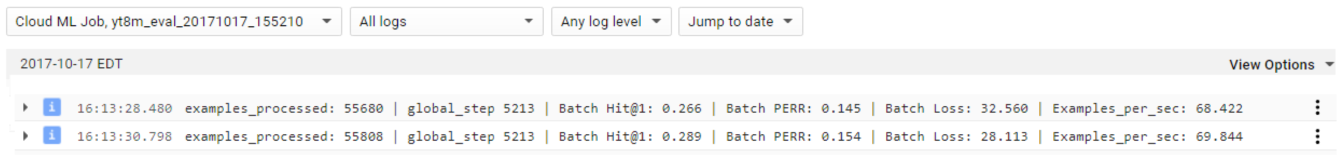
Training :



```
Cloud ML Job, yt8m_train_20171015_185430    All logs    Any log level    Jump to date
2017-10-15 EDT                                                                View Options
23:14:11.197 training step 4749 | Loss: 30.71 Examples/sec: 40.21
23:14:14.437 training step 4750 | Loss: 35.09 Examples/sec: 40.30 | Hit@1: 0.25 PERR: 0.14 GAP: 0.07
```

Training accuracy(Hit @1) = 25%

Validation:

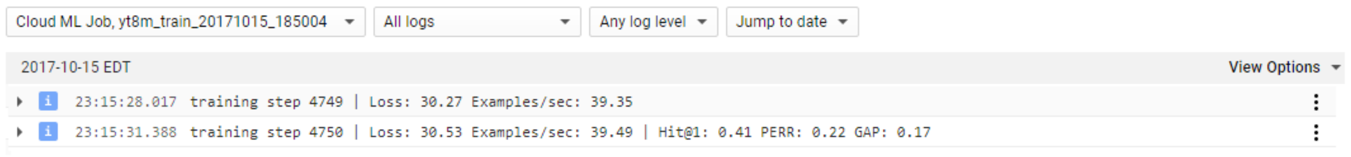


```
Cloud ML Job, yt8m_eval_20171017_155210    All logs    Any log level    Jump to date
2017-10-17 EDT                                                                View Options
16:13:28.480 examples_processed: 55680 | global_step 5213 | Batch Hit@1: 0.266 | Batch PERR: 0.145 | Batch Loss: 32.560 | Examples_per_sec: 68.422
16:13:30.798 examples_processed: 55808 | global_step 5213 | Batch Hit@1: 0.289 | Batch PERR: 0.154 | Batch Loss: 28.113 | Examples_per_sec: 69.844
```

Validation accuracy(Hit @1) = 28.9%

5. LSTM with video and audio features (4 hr 50 min)

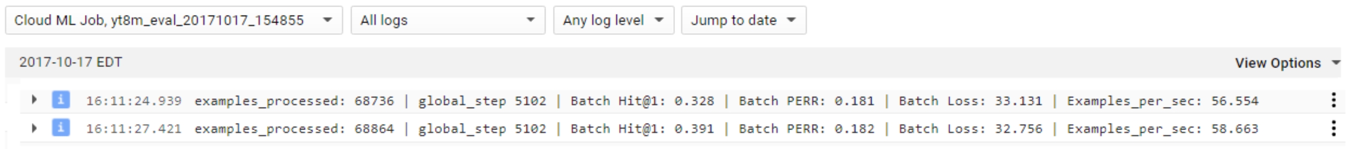
Training:



```
Cloud ML Job, yt8m_train_20171015_185004    All logs    Any log level    Jump to date
2017-10-15 EDT                                                                View Options
23:15:28.017 training step 4749 | Loss: 30.27 Examples/sec: 39.35
23:15:31.388 training step 4750 | Loss: 30.53 Examples/sec: 39.49 | Hit@1: 0.41 PERR: 0.22 GAP: 0.17
```

Training accuracy(Hit @1) = 41%

Validation:

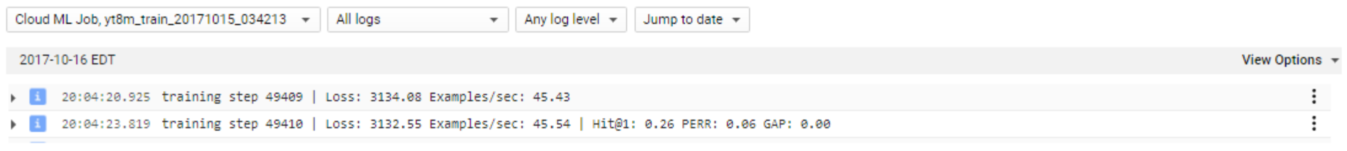


```
Cloud ML Job, yt8m_eval_20171017_154855    All logs    Any log level    Jump to date
2017-10-17 EDT                                                                View Options
16:11:24.939 examples_processed: 68736 | global_step 5102 | Batch Hit@1: 0.328 | Batch PERR: 0.181 | Batch Loss: 33.131 | Examples_per_sec: 56.554
16:11:27.421 examples_processed: 68864 | global_step 5102 | Batch Hit@1: 0.391 | Batch PERR: 0.182 | Batch Loss: 32.756 | Examples_per_sec: 58.663
```

Validation accuracy(Hit @1) = 39.1%

6. GRU with only video features (train duration:1 day 16hrs)

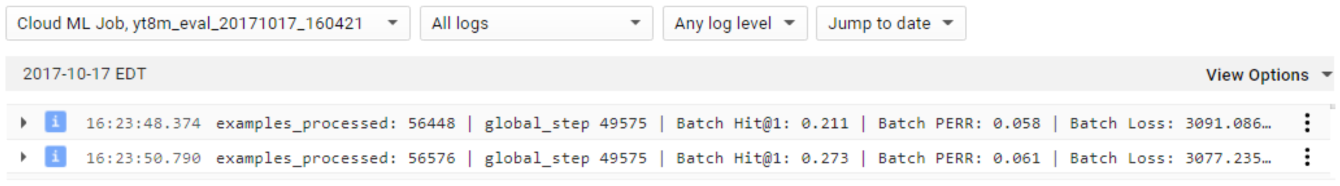
Training:



```
Cloud ML Job, yt8m_train_20171015_034213    All logs    Any log level    Jump to date
2017-10-16 EDT                                                                View Options
20:04:20.925 training step 49409 | Loss: 3134.08 Examples/sec: 45.43
20:04:23.819 training step 49410 | Loss: 3132.55 Examples/sec: 45.54 | Hit@1: 0.26 PERR: 0.06 GAP: 0.00
```

Training accuracy (Hit @1) = 26%

Validation:



Cloud ML Job, yt8m_eval_20171017_160421 | All logs | Any log level | Jump to date

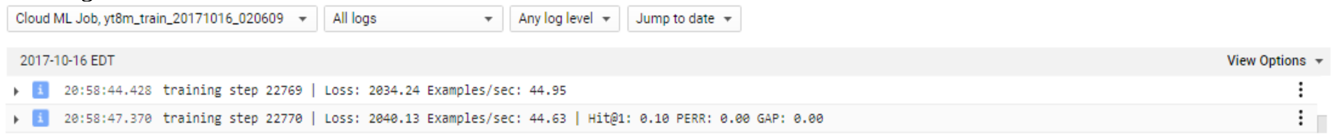
2017-10-17 EDT | View Options

```
16:23:48.374 examples_processed: 56448 | global_step 49575 | Batch Hit@1: 0.211 | Batch PERR: 0.058 | Batch Loss: 3091.086…
16:23:50.790 examples_processed: 56576 | global_step 49575 | Batch Hit@1: 0.273 | Batch PERR: 0.061 | Batch Loss: 3077.235…
```

Validation accuracy (Hit @1) = 27..3%

7. GRU with video and audio features (train duration:18 hr 51 min)

Training:



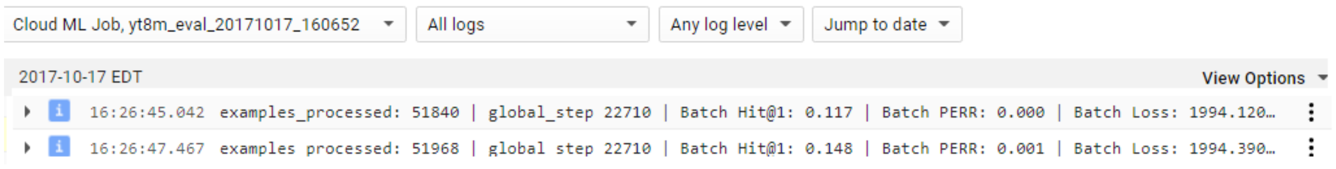
Cloud ML Job, yt8m_train_20171016_020609 | All logs | Any log level | Jump to date

2017-10-16 EDT | View Options

```
20:58:44.428 training step 22769 | Loss: 2034.24 Examples/sec: 44.95
20:58:47.370 training step 22770 | Loss: 2040.13 Examples/sec: 44.63 | Hit@1: 0.10 PERR: 0.00 GAP: 0.00
```

Training accuracy(Hit @1) = 10%

Validation:



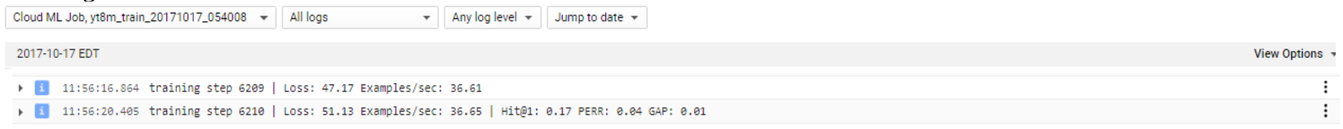
Cloud ML Job, yt8m_eval_20171017_160652 | All logs | Any log level | Jump to date

2017-10-17 EDT | View Options

```
16:26:45.042 examples_processed: 51840 | global_step 22710 | Batch Hit@1: 0.117 | Batch PERR: 0.000 | Batch Loss: 1994.120…
16:26:47.467 examples processed: 51968 | global step 22710 | Batch Hit@1: 0.148 | Batch PERR: 0.001 | Batch Loss: 1994.390…
```

Validation accuracy (Hit @1) = 14.8%

8. BiLSTM with audio and video features (train duration :6 hr 15 min)
Training:



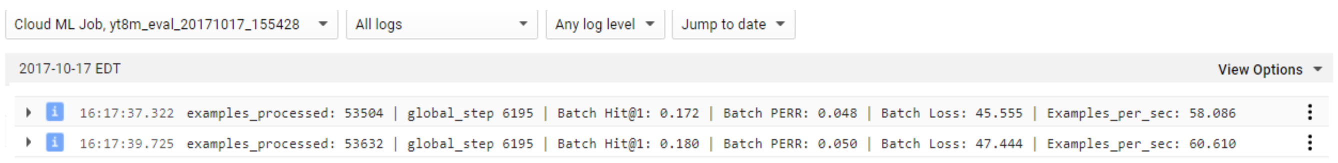
Cloud ML Job, yt8m_train_20171017_054008 | All logs | Any log level | Jump to date

2017-10-17 EDT | View Options

```
11:56:16.864 training step 6209 | Loss: 47.17 Examples/sec: 36.61
11:56:20.405 training step 6210 | Loss: 51.13 Examples/sec: 36.65 | Hit@1: 0.17 PERR: 0.04 GAP: 0.01
```

Training accuracy (Hit @1) = 17%

Validation:



Cloud ML Job, yt8m_eval_20171017_155428 | All logs | Any log level | Jump to date

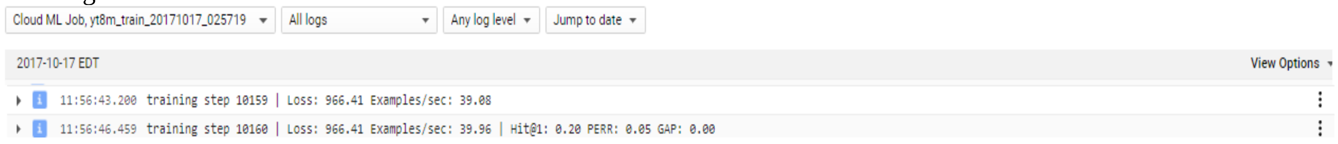
2017-10-17 EDT | View Options

```
16:17:37.322 examples_processed: 53504 | global_step 6195 | Batch Hit@1: 0.172 | Batch PERR: 0.048 | Batch Loss: 45.555 | Examples_per_sec: 58.086
16:17:39.725 examples_processed: 53632 | global_step 6195 | Batch Hit@1: 0.180 | Batch PERR: 0.050 | Batch Loss: 47.444 | Examples_per_sec: 60.610
```

Validation accuracy (Hit @1) = 18%

9. BiGRU with audio and video features (train duration:8 hr 59 min)

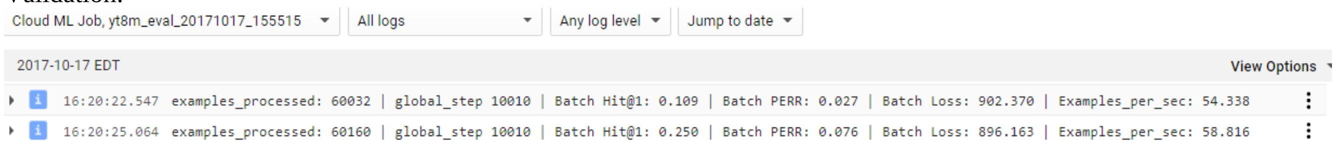
Training:



```
11:56:43.200 training step 10159 | Loss: 966.41 Examples/sec: 39.08
11:56:46.459 training step 10160 | Loss: 966.41 Examples/sec: 39.96 | Hit@1: 0.20 PERR: 0.05 GAP: 0.00
```

Training accuracy(Hit @1) = 20%

Validation:



```
16:20:22.547 examples_processed: 60032 | global_step 10010 | Batch Hit@1: 0.109 | Batch PERR: 0.027 | Batch Loss: 902.370 | Examples_per_sec: 54.338
16:20:25.064 examples_processed: 60160 | global_step 10010 | Batch Hit@1: 0.250 | Batch PERR: 0.076 | Batch Loss: 896.163 | Examples_per_sec: 58.816
```

Validation accuracy (Hit @1) = 25%

5.1 Comparison

From the facts observed from the various research papers for this problem, better accuracies can be obtained if
1. Training is done longer i.e for 3 to 5 days using multiple clusters
2. Ensemble of five or seven different models are used

6 Problems and Possible improvements:

1. GRU with two layers and layer size = 2048 would give better results but when this method was attempted , it resulted in resource exhaustion error on the GCP

2. BI-LSTM when coupled with batch normalization could possibly improve the accuracy from the present value

3. During the implementation of BILSTM if ‘state_is_tuple’ is not set with false , it results in tensor shape mismatch error . This fact and the detail for concatenating the forward and backward states was referred from the discussion thread of YouTube-8m competition in Kaggle

4. Stacked BILSTM tensorflow could not be used as the 3D tensor could not be split in list of 2D tensors .This list has to be supplied as input to this API .Instead of this method dynamic bidirectional rnn was used

7 Conclusion

The consideration of this problem statement helped to learn about newer paradigms in deep learning and tensorflow APIs apart from latest deep learning network architectures

8 References

[1] https://research.google.com/pubs/pub45611.html
[2] https://people.cs.pitt.edu/~milos/courses/cs2750-Spring04/lectures/class22.pdf\
[3] https://www.kaggle.com/c/youtube8m/discussion
[4] https://research.google.com/youtube8m/workshop.html
[5] https://static.googleusercontent.com/media/research.google.com/en//youtube8m/workshop2017/c11.pdf
[6] https://static.googleusercontent.com/media/research.google.com/en//youtube8m/workshop2017/c04.pdf
[7] https://static.googleusercontent.com/media/research.google.com/en//youtube8m/workshop2017/c14.pdf
[8] https://static.googleusercontent.com/media/research.google.com/en//youtube8m/workshop2017/c03.pdf