Google Cloud and YouTube-8M Video Understanding Challenge

Hyun Sik Kim
Stanford University
hsik@stanford.edu

Ryan Wong
Stanford University
rawong@stanford.edu

Abstract

In this project, we implemented and evaluated a range of deep learning architectures (both frame-level models and video-level models) to tackle the Google Cloud and Youtube-8M video understanding challenge ('YT-8M challenge'), a multi-label video classification challenge based on the recently released Youtube-8M V2 dataset. Our best performing architecture was a novel residual network trained on video-level features that achieves a test set GAP of 0.801, which placed in the top 20% in the YT-8M challenge. The residual network outperformed our LSTM model, suggesting that well-designed video-level models can outperform frame-level ones and / or learning temporal cues is difficult. In addition, audio features were found to be important to video classification, materially improving model classification performance.

1. Introduction

Encouraged by positive results in the image and speech domain, the application of deep learning to understanding the semantic content of videos is an active area of research that has broad applications, including search and summarization. However, the mix of spatial, temporal and acoustic cues makes off-the-shelf deep learning models insufficient for understanding video semantics.

Deep learning has demonstrated impressive results in single label classification but achieving similar success in the multi-label domain is still an open research problem. Multi-label classification is a more general and practical problem since many real-world objects, such as videos, have a variable number of labels [1]. One of the key reasons for this discrepancy is arguably the lack of large-scale video classification benchmarks like ImageNet.

Hence, in late 2016, Youtube released the Youtube-8M ('YT-8M') dataset, the first large-scale video classification benchmark [5], and in early 2017, launched the Google Cloud & Youtube-8M Video Understanding Challenge ('YT-8M challenge'), a multi-label video classification challenge, to promote advancements in video semantics.

This project was focused on tackling the YT-8M challenge and details our implementation and empirical results of different deep learning networks applied to the YT-8M dataset.

2. Related work

There are two main types of architectures that researchers have applied to multi-label video classification problems – convolutional neural network and recurrent neural network architectures.

Karpathy et al. explored a variety of convolutional neural network (CNN) architectures that were extended to the time domain for video classification. They discovered that the slow fusion model (balanced approach that slowly fuses temporal information throughout the network) consistently performed better than the early and late fusion alternatives. However, the best spatio-temporal networks only surprisingly exhibited a modest improvement over single-frame models, suggesting local motion cues may not be critically important [2].

More recently, recurrent neural networks (RNN) have achieved state-of-the-art performance in multi-label image and video classification due to their ability to learn temporal cues and label dependencies. Wang et al.'s CNN-RNN framework used RNNs to model the label dependencies in multi-label image classification [3]. Ng. et al.'s Long Short-Term Memory (LSTM, a type of RNN) architectures, which explicitly modelled the video as an ordered sequence of frames, outperformed CNN temporal pooling models on the Sports-1M dataset [4].

Given the recent release of the YT-8M dataset, the only research detailing empirical performance of neural networks trained on the YT-8M dataset is the paper written by Google researchers that accompanied the release of the dataset. LSTM and Mixture of Experts (MoE) models reportedly exhibited the best performance on the dataset [5].

Residual networks are considered a state-of-the-art technique to train very deep architectures. He et al. provided empirical evidence that residual learning frameworks, which were adopted in the ILSVRC 2015 winner ResNet, are easier to optimize and gain accuracy from considerably increased depth [6].

In this project, we implemented and evaluated a range of

different multi-label video classification models on the YT-8M dataset and to the authors' best knowledge, is the first to present a novel residual network that achieves strong performance on the YT-8M dataset.

3. YT-8M dataset

The latest version of the YT-8M dataset is version 2. YT-8M is a large-scale video dataset of over 7 million labelled Youtube video IDs (totaling 450,000 hours) across a diverse vocabulary of 4,716 labels (3.4 labels per video on average). Every label contains at least 101 training examples with an average of 3,552 training examples. Table 1 below summarizes the dataset.

Total no. of videos	7,009,128
Training set	4,906,660 (~70%)
Validation set	1,401,828 (~20%)
Test set	700,640 (~10%)
Total no. of labels	4,716
Avg. no. of labels	3.4 per video
Original video length	120-500 seconds long
No. of encoded frames	Up to 360 frames per video
Visual features	1,024 dimensional (8-bit each)
Audio features	128 dimensional (8-bit each)

Table 1: YT-8M V2 dataset statistics

Figure 1 below shows the dataset is heavily skewed with the top 40 labels (out of 4,716) accounting for about 45% of total ground truth labels across the training set.

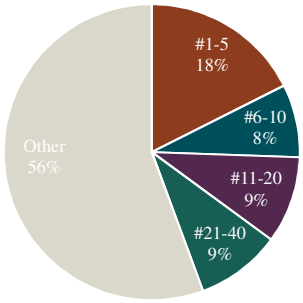

Figure 1: Percentage of ground truth labels within top 40 labels

Dataset includes 3.2 billion pre-processed visual / audio features that were PCA'ed and quantized (refer to Figure 2). On a frame level basis, visual features (extracted using the Inception Network) are 1,024 dimensional per second and audio features (extracted using a VGG-inspired acoustic model) are 128 dimensional per second. Frame level data is provided at 1-second resolution up to the first 6 min of each video. Video level data is the simple mean of visual and audio features across frames.

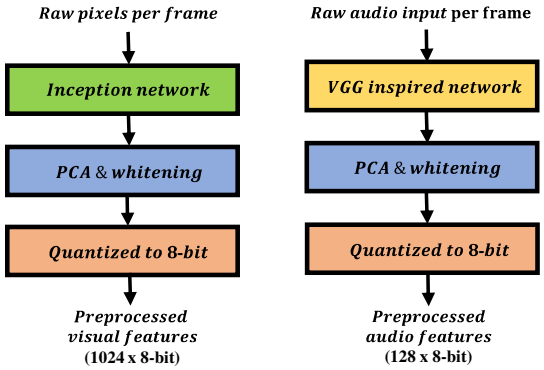

Figure 2: Visual and audio features pre-processing

Although the pre-processed features make the large-scale dataset more manageable, it places two key restrictions on possible model architectures that can be applied to the dataset: 1) raw pixels are not accessible and so an end-to-end model from pixels to predictions cannot be learnt; and 2) motion cues (optical flow) cannot be learnt given the low frame rate (i.e. 1 frame per second).

The dataset is split into three partitions: training (~70%), validation (~20%) and test (~10%).

4. Methods

A mix of models trained on frame level features only ('frame-level models') and video-level features only ('video level models') were implemented and empirically tested. Models include:

- *Independent classifiers* (video-level model): comprises of 4,716 one-vs-all binary logistic regression classifiers for each label.
- *Mixture of experts* ('MoE') (video-level model): model comprising of *k* experts, each being a version of the above independent classifiers model.
- *LSTM* (frame-level model): multi-layer LSTM network based on frame-level features.
- *Fully-connected block network* (FC network) (video-level model): multi-layer feed-forward network comprising of repeating fully-connected (FC) with ReLU and batch normalization (BN) layers.
- *Residual network* (video-level model): multi-layer feed-forward network comprising of residual learning blocks that have FC with ReLU and BN layers.

The Youtube-8M Tensorflow starter code in [7] was used as a base for training, validation and inference of the various model architectures.

Each model (except LSTM) was trained over 2-3 epochs with an initial learning rate of 5e-5 to 5e-4 with learning rate decay of 0.95 every 4 million examples, and batch size of 1,024. The Adam optimizer was chosen for training as it adaptively anneals the learning rate in each

dimension, thus reducing dependence on initial learning rate selection and improving the speed of convergence.

Sections 4.1 to 4.5 describe each model architecture in further detail and section 4.6 outlines the metrics used to assess each models' performance.

4.1. Independent classifiers

The model comprises of 4,716 one-vs-all binary logistic regression classifiers for each label trained on video-level features. Figure 3 below details its architecture.

Independent classifiers was selected as the baseline model given its relative ease of implementation and its simple intuitive strategy of resolving a multi-label classification problem into multiple single-label classification problems.

During training, the classifiers were independently trained on video-level features with L2 regularization penalty of 1e-8.

During inference, each video is scored by each classifier. Video-level label scores are then obtained by a simple concatenation of all scores across classifiers.

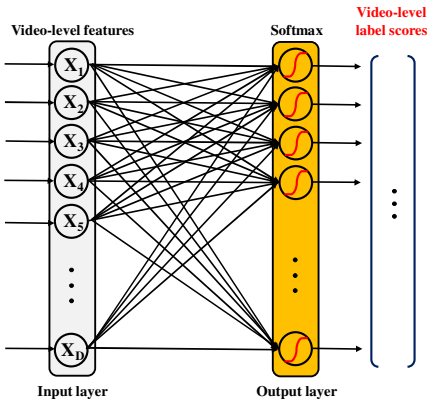

Figure 3: Independent classifiers architecture

4.2. Mixture of experts (MoE)

The MoE model comprises of k experts where the final video-level prediction is a mix (weighted sum) of the predictions from each expert. Figure 4 below details the architecture.

Each expert is a set of independent classifiers (akin to section 4.1). In addition to learning the parameters for each expert, the model learns the parameters for a gating network layer (a fully-connected layer followed by softmax), which controls the contribution of each expert to the final video-level prediction. This essentially results in the model learning the optimal mix of different experts over different inputs. The model is a classifier of classifiers and the intuition is that a combination of experts, particularly ones with negatively correlated errors, will improve model generalization.

MoE was chosen for implementation given [5] reported it to be one of the highest performing models on the YT-8M dataset and the prevalent (and often successful) use of "kitchen-sink approaches" in similar data challenges, like the Netflix challenge.

A MoE with $k = 2$ was trained and tested.

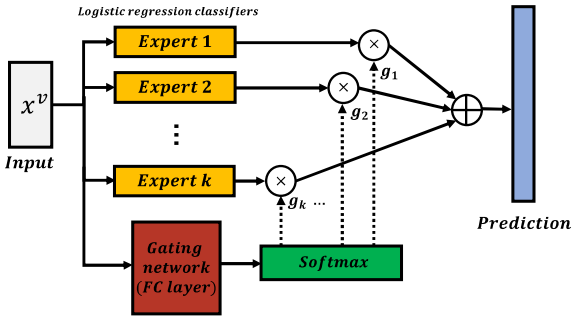

Figure 4: Mixture of experts architecture

4.3. LSTM

The LSTM model is a frame-level model comprising of 2-layers of LSTM units with hidden state dimension of 1,024. The number of layers and hidden dimension size were selected based on [5]. Figure 5 below shows the architecture.

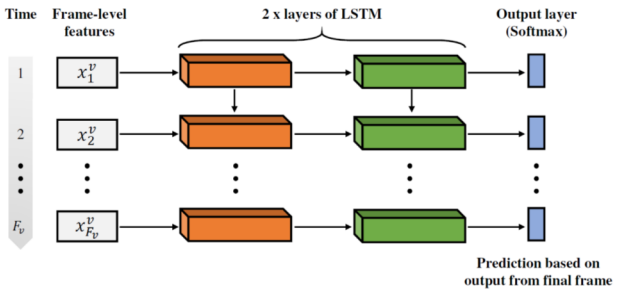

Figure 5: LSTM architecture

Given state-of-the-art results achieved by recurrent neural networks in precedent research [3, 4, 5], we decided to implement one for the YT-8M challenge. In particular, LSTMs were selected since they avoid the vanishing and exploding gradient issues of vanilla RNNs.

All frames of a video are passed through two LSTM layers. The input to the second LSTM layer is simply the output of the preceding layer. Label predictions are performed at each time step based on a softmax layer on top of the last (second) LSTM layer.

Training was attempted over one epoch with a batch size of 128 (sized based on memory constraints) and gradient horizon of 60 seconds as adopted in [5]. i.e. gradients are back-propagated every 60 frames. However, as discussed in section 5.3, even just one epoch proved to be computationally intractable.

During inference, video-level predictions were based on the softmax output at the last time step $t = F_v$.

4.4. FC network

The FC network is a video-level model with a modular design made up of seven fully connected blocks, each comprising of a fully connected layer with ReLU and batch normalization. Figure 6 below details its architecture.

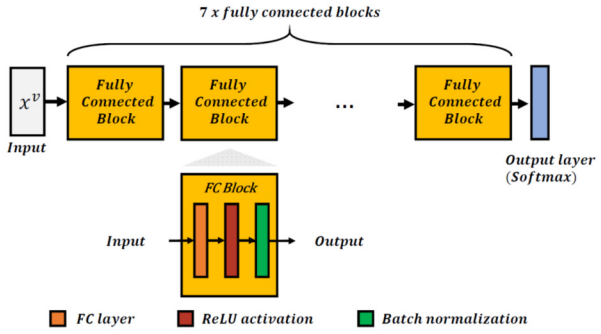

Figure 6: FC architecture

The FC network is one of two deep multi-layer video-level feed-forward designs that were investigated (the other being the residual network in section 4.5). These designs were examined to determine whether strong performance could be obtained based on video-level features (as opposed to frame-level features) – a significantly computationally less intensive process but the key drawback being the inability to learn temporal cues.

The design makes heavy use of batch normalization to provide robustness to parameter initializations and its modularity is inspired by the Inception Network.

4.5. Residual network

The residual network is the second deep multi-layer video-level feed-forward design made up of three residual learning blocks, each comprising of 3 x fully-connected, 2 x ReLU and 2 x batch normalization layers. Figure 7 below shows the architecture.

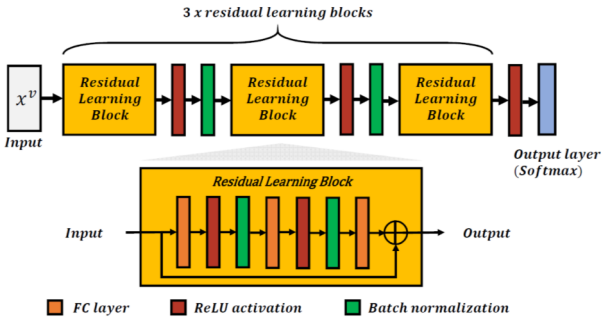

Figure 7: Residual network architecture

The residual network design is inspired by ResNet, the winner of the ILSVRC 2015, which is widely regarded as the state-of-the-art convolutional network in image classification. Given the features are pre-processed (not raw pixels), spatial convolutions cannot be performed and hence were replaced by fully connected layers. Similar to ResNet, the design makes heavy use of batch normalization to provide robustness to weight initializations.

4.6. Evaluation

The performance of each model was evaluated using the following metrics (commonly used in multi-label classification).

Mean Average Precision (mAP) is the unweighted mean of all per-label average precisions. It is computed using the formulae below, where AP_e is the average precision for label e in the set of all labels E. AP_e is an approximation of the area under the precision-recall curve, where the label scores are rounded into buckets of 10^{-4} [5]:

$$mAP = \sum_{e \in E} AP_e \qquad AP_e = \sum_{j=1}^{10000} P(\tau_j)[R(\tau_j) - R(\tau_{j-1})]$$

The precision $P(\tau)$ and recall $R(\tau)$ of each label at a given threshold τ is calculated using the formulae below, where $\mathrm{II}(.)$ is the indicator function, y_t is the rounded score of label t in the set T of all labels for each video and $g_t \in \{0, 1\}$ denotes the ground truth of label t [5].

$$P(\tau) = \frac{\sum_{t \in T} \mathrm{II}(y_t \geq \tau) g_t}{\sum_{t \in T} \mathrm{II}(y_t \geq \tau)} \qquad R(\tau) = \frac{\sum_{t \in T} \mathrm{II}(y_t \geq \tau) g_t}{\sum_{t \in T} g_t}$$

Hit@k is the fraction of test examples that contain at least one of the ground truth labels in the top k predictions. This is computed via the formula below, where $rank_{v,e} \leq k$ is the rank of label e for video v in the set of examples V, G_v is the set of ground-truth labels for video v and $\vee$ is the logical OR operation [5].

$$\frac{1}{|V|} \sum_{v \in V} \vee_{e \in G_v} \mathrm{II}\,(rank_{v,e} \leq k)$$

Precision at Equal Recall Rate (PERR) is the video-level annotation precision when the same number of labels per video are retrieved as there are in the ground-truth. This is computed using the formula below (using the same notation as Hit@k) [5]:

$$\frac{1}{|V : |G_v| > 0|} \sum_{v \in V : |G_v| > 0|} \left[\frac{1}{|G_v|} \sum_{e \in G_v} \mathrm{II}\,(rank_{v,e} \leq |G_v|) \right]$$

Global Average Precision (GAP) is the average precision based on the top 20 predictions per example. GAP was computed on the validation set and the test set. The latter was used for leaderboard ranking in the YT-8M challenge. It is computed as follows, where N is the number of final predictions (number of test examples x 20), $p(i)$ is the precision and $r(i)$ is the recall. The key difference between GAP and mAP is that GAP calculates the average precision only over the top 20 predictions per video.

$$GAP = \sum_{i=1}^{N} p(i)\Delta r(i)$$

5. Results and discussion

The following section discusses the results and observations of implementing and testing each of the models previously discussed in section 4.

5.1. Summary of results

Table 2 and Figure 8 below summarize the performance of the models on the validation and test sets.

Model	Hit@1	PERR	mAP	GAP
Ind. classifiers (w/out audio)	0.789	0.646	0.376	0.707
MoE (w/out / with audio)	0.728 / 0.772	0.562 / 0.611	0.110 / 0.125	0.611 / 0.665
FC – 2 x layers (w/out / with audio)	0.792 / 0.826	0.646 / 0.687	0.244 / 0.283	0.717 / 0.758
FC – 5 x layers (w/out / with audio)	0.756 / 0.844	0.595 / 0.712	0.111 / 0.346	0.657 / 0.785
FC – 7 x layers (w/out / with audio)	0.772 / 0.807	0.613 / 0.653	0.125 / 0.144	0.675 / 0.719
Residual network (with audio)	0.853	0.725	0.399	0.800
LSTM (w/out audio)	0.841	0.708	-	0.784

Table 2: Model performance on the validation set. Table entries with two entries ([x] / [y]) denote the performance without audio features [x] and with audio features [y].

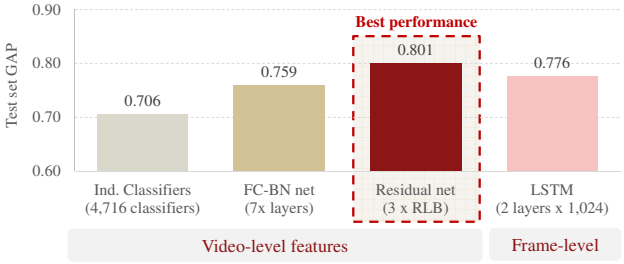

Figure 8: Model performance on the test set (GAP). Note this is test set GAP (not validation set GAP) and hence metrics are slightly different to Table 2 above.

Overall, the residual network (video-level model) exhibited the best accuracy with a test set GAP of 0.801, outperforming the frame-level model LSTM. Sections 5.2 to 5.6 describe our methodologies and observations leading towards our best performing model, the residual network.

5.2. Independent classifiers – a baseline model

Although a naive approach to multi-label classification, the independent classifiers model performed reasonably well, achieving an overall test set GAP of 0.707.

The MoE model did not exhibit an improvement over the independent classifiers model. However, in hindsight, the MoE model may have required a larger number of training epochs for a fairer comparison given the larger number of model parameters.

Despite being the least computationally intensive to train (given its simplistic structure), a key disadvantage of the independent classifiers is that it ignores label dependencies that could have been exploited to improve classification performance. Figure 9 is a sparsity graph which shows that there are underlying relationships in the data between groups of examples and groups of labels. The graph on the left hand side shows the original data – 1,000 randomly sampled training examples (rows) with their ground truth labels represented by each blue dot (columns). The graph on the right hand side shows the same data after performing co-clustering of both examples and labels. The dense block-like structures in the co-clustered graph represent regions where strong associative relationships exist between groups of examples and groups of labels.

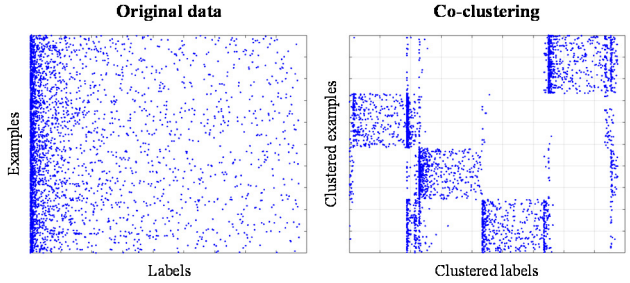

Figure 9: Label dependencies in the data for 1,000 random examples. LHS shows original data. RHS shows co-clustered data.

In addition, training a model based on video-level features ignores temporal cues as video-level features are a simple mean of frames.

These two key disadvantages prompted us to search for model architectures that could exploit these label dependencies and temporal cues to achieve higher performance. Hence, we decided to implement a LSTM network given they are able to learn both such relationships by storing and passing information between cells via a hidden state.

5.3. Learning temporal cues challenging

However, due to computational resource constraints on Google Cloud ML, training a LSTM on frame-level features over such a large-scale dataset (over 7 million examples with up to 360 frames each) ended up being a computationally intractable task, particularly over the relatively short timeframe of this project. Despite utilizing four GPUs, Google Cloud ML was unable to complete a single epoch of LSTM training over even one-week of training.

It is believed that one of the key reasons for its computational intractability is the significantly smaller batch-size during training due to memory constraints (128 in LSTM vs. 1,024 in video-level models), resulting in higher computational overhead per example. In addition, training over frame-level features required up to six pairs of forward and backward passes (as opposed to a single pair of passes in video-level models) for each block of 60 frames per video.

Hence, we decided to investigate more sophisticated models based on video-level features (as opposed to frame-level features) that were more computationally tractable over this project's timeframe.

5.4. Video-level models – FC network

Although they are unable to learn temporal cues (since trained on video-level features not frame-level features), video-level models were significantly more computationally efficient and exhibited impressive performance relative to the frame-level model, LSTM.

The first model that was implemented was the FC network made up of seven fully-connected blocks (referenced as 'layers' here for simplicity), each comprising of repeating fully-connected layers with ReLU and batch normalization.

FC network performance for varying hidden dimension sizes and number of fully connected blocks was empirically tested. Highest test set GAP was achieved using seven fully connected blocks. Beyond which, there were diminishing returns on performance improvement for an increased number of layers and complexity. The optimal hidden dimension sizes were empirically found to be (4, 8, 4, 2, 4, 4, 2), where each number represents a multiple of the input feature dimension size of the respective layer of the FC network. e.g. the first layer had a hidden dimension size of 4 x input feature dimension size.

The promising results of the FC network encouraged us to explore options for improving performance of video-level models, in particular, feature selection (incorporation of audio features in section 5.5) and model architecture (residual networks in section 5.6).

5.5. Improved performance with audio features

Incorporation of video-level audio features in addition to visual features materially improved performance of the FC network.

Figure 10 shows that the addition of audio features resulted in a 5-7% increase in Hit@1, PERR and GAP and about a 15% increase in mAP for the FC network with 7 x layers.

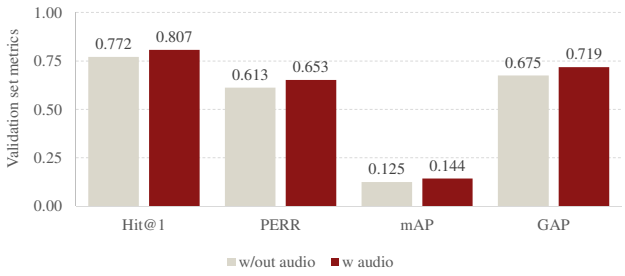

Figure 10: FC 7 x layers network with and without audio features

This marked improvement suggests that acoustic cues are material in video classification, noting their inclusion only increased the total feature dimension size by about 13% (feature dimension size increase from 1,024 to 1,024 + 128).

5.6. Outperformance of residual networks

Our best performing model was the residual network, which achieved a test set GAP of 0.801, which placed us in the top 20% of competitors in the YT-8M challenge. Despite ignoring temporal cues, well-designed video-level models like the residual network can outperform frame-level ones, like LSTM.

This outperformance is attributable to the residual network's unique structure, which makes it easier to optimize and more robust to hyperparameter selection. Learning residual functions referenced with respect to input layers is easier than learning unreferenced ones in traditional feed-forward networks [6]. The references with respect to input layers inject gradient directly from output to input layers during back-propagation, thus improving gradient flow, particularly in deep networks.

Figure 11 below shows the learning curve (loss vs. training steps) of the residual network vs. the FC network (7 x layers). It demonstrates the considerably faster rate of convergence of the residual network, taking only about one-third the total steps of the FC network to approach the same level of total loss.

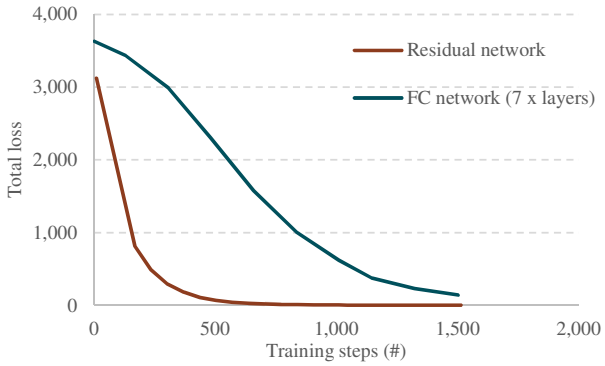

Figure 11: Learning curve of residual network vs. FC network

6. Conclusion and future work

We have explored a mix of both frame-level and video-level models to tackle the YT-8M challenge, a multi-label video classification problem.

Our best performing model was a novel residual network, inspired by ResNet, which achieved a GAP of 0.801 on the test set and placed us in the top 20% of competitors in the challenge. The strong performance of the residual network is due to its unique structure that makes it easier to optimize and more robust to hyperparameter selection.

Despite ignoring temporal cues, well-designed video-level models (like the residual network) can outperform frame-level ones, such as LSTM. The incorporation of audio features significantly improved performance, suggesting acoustic cues have material predictive power in video classification.

LSTM was computationally intractable over the timescale of this project due to Google Cloud ML resource constraints. Hence, we were not able to achieve competitive results using LSTM on this dataset despite state-of-the-art results reported by precedent research on other similar datasets.

We plan on performing further work in a number of areas. Given the strong results exhibited by models trained on video-level features, which are a simple mean of frames, we would like to explore whether there are more sophisticated frame aggregation methods that can generate features with greater predictive power. In addition, we would like to explore other recurrent neural network architectures that may be more computationally tractable, such as training on a subsampling of frames per example (as opposed to all frames). In particular, methods for determining the optimal subsample from a collection of frames, such as bloom filters, autocorrelation or locality sensitive hashing.

References

[1] Wei, Yunchao, et al. "CNN: Single-label to multi-label." arXiv preprint arXiv:1406.5726 (2014).

[2] Karpathy, Andrej, et al. "Large-scale video classification with convolutional neural networks." Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. 2014.

[3] Wang, Jiang, et al. "CNN-RNN: A unified framework for multi-label image classification." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

[4] Yue-Hei Ng, Joe, et al. "Beyond short snippets: Deep networks for video classification." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[5] Abu-El-Haija, Sami, et al. "Youtube-8M: A large-scale video classification benchmark." arXiv preprint arXiv:1609.08675 (2016).

[6] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

[7] Youtube-8M Tensorflow Starter Code (2017), Github repository, https://github.com/google/youtube-8m