Deep Learning YouTube Video Tags

Travis Addair
Stanford University
taddair@stanford.edu

Abstract

Video tagging is a complex problem that combines single-image feature extraction with arbitrarily long sequence understanding. By improving at the task of tagging videos with useful metadata labels, we necessarily improve our ability to understand the content and context of video data. Until recently, however, there hasn't been a large corpus of labelled video data for researchers to study. With Google's release of the YouTube-8M[1] [1] dataset, academic researchers now have access to 7 million video URLs, 450 000 hours of video, 3.2 billion features, and 4716 label classes. In conjunction with Kaggle's announcement of their Video Understanding Challenge[2], this project seeks to combine state-of-the-art deep learning methods to the problem of automatically labelling video frame data. We propose a hybrid CNN-RNN architecture that takes the image features generated for each video, and combines them with an LSTM model run over the word embeddings of the label set, to generate label predictions that take label correlation and dependency into account. We show that this model outperforms baseline models that operate only on raw image features without accounting for structural label similarity.

1. Introduction

The goal for this project was to generate a classifier that most accurately labels a collection of YouTube videos with up to 20 tags that denote the genre and context of the video.

1.1. Dataset

The YouTube-8M dataset consists of 7 million videos sampled uniformly at random from the entire collection of YouTube videos available publicly online. Every video sampled has at least 1000 views, is between 120 and 500 seconds long, is associated with at least one tag in the vocabulary, and is not considered to have adult content.

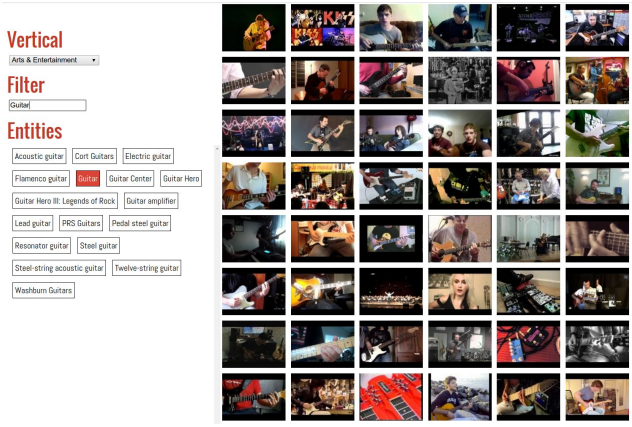

Figure 1. Example images tagged with the label *guitar* from the YouTube-8M dataset.

Due to the volume of data in the collection, pre-computed features have been derived from the source videos. 1.6 billion video features were extracted using Google's Inception-V3 image annotation model[3] [2]. 1.6 billion audio features were extracted using a VGG acoustic model. Both sets of features were run through PCA and quantized such that the combined set of all features is less than 2TB.

Both video level and frame level features are provided for each video. 1024 8-bit quantized features are provided per second of video (frame), up to 300 seconds. 128 8-bit audio features are provided per second of video as well, up to 300 seconds.

Every video is annotated with 1 to 31 tags that identify the themes of each video. The tags were generated using a combination of content, metadata, context, and user signals, and were generated by a neural network as a first pass, then curated by 3 human raters as a second pass. The tags are drawn from a vocabulary set of 4716 Knowledge Graph[4] entities. Each tag is represented by at

[1] https://research.googleblog.com/2016/09/announcing-youtube-8m-large-and-diverse.html

[2] https://www.kaggle.com/c/youtube8m

[3] https://www.tensorflow.org/tutorials/image_recognition

[4] https://www.google.com/intl/es419/insidesearch/features/search/knowled

least 101 videos in the dataset. Tags are grouped into one of 24 different "verticals" or high level categories.

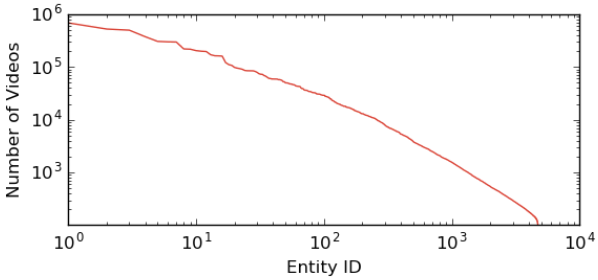

Figure 2. The distribution of videos tagged with each label roughly follows a power law distribution.

In addition to providing the Knowledge Graph entity ID associated with each tag, a Wikipedia URL and summary description of each tag is provided for context.

1.2. Metrics

In evaluating the performance of our tagging system, we use Kaggle's preferred evaluation metric of the Global Average Precision (GAP) defined as:

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

Where p(i) is the precision of prediction i, r(i) is the recall of prediction i, N is the number of predictions (label/confidence pairs). For the purpose of the Kaggle competition, we limit our submission to the top k predictions per video, where k = 20 for the competition. The total number of predictions N = k * m where m is the number of videos in the test set.

2. Related Work

Research in the area of mutli-label classification for rich video data has been limited in the past by the lack of an accurately labeled, large scale database of such videos. The release of the YouTube-8M dataset marks the beginning of a new set of opportunities to explore this nascent space. That said, the problem of multi-label video classification can be compared to image labelling and image captioning, both of which have been heavily studied in recent years.

Investigation into the problem of multi-label image classification has taken off in recent years due to the creation of the ImageNet [3] database. Wang et al [4] created a hybrid CNN-RNN architecture that attempted to learn a join image-label embedding that could be used to generate a set of distinct labels for a given image, as well as learn the semantic label dependency structure. Their approach provided the intuition behind our own approach to the problem of video classification, as did their decision to use beam search during inference.

Hu et al [5] also applied a CNN-RNN architecture to the problem of multi-label image classification, but instead of using word embeddings as input to the recurrent sub-network, they instead use a word similarity matrix constructed from the WordNet [6] taxonomy. One limitation of Hu et al's approach is that it relies heavily on the labels having a pre-existing, rich hierarchy of attributes that can be used to generate a structure from "coarse" attributes to "fine" labels. The Google Knowledge Graph taxonomy, in contrast to WordNet, uses Schema.org[5] entities, which lack rich attribute associations.

Liu et al [7] expanded upon the work done by Wang et al [4] but separated the problem of learning the visual concepts (tags) from learning the concept similarity structure. They proposed using a semantic regularization embedding between the CNN image features and the RNN label features. The concept of separating this two subproblems for more efficient training is interested, but there wasn't sufficient time in this study to explore this idea fully, but we certainly believe it could potentially be used to improve training performance on our model.

There has also been work done directly on the problem of multi-label video classification, but often with the addition of certain metadata features or raw visual/audio signals we didn't have access to in this study. Yang and Toderici [8] combined metadata with raw video, but their metadata consisted of per user watching statistics, which we lacked in the YouTube-8M dataset.

Jang et al [9] proposed a model called rDNN (Regularized Deep Neural Network) that extracts visual, audio, and trajectory features for each video and combines them using a series of deep fully connected layers. The label space is associated by concatenating label-level predictions into another series of fully connected layers for final prediction. In our work, we do something similar to fuse LSTM outputs together for final prediction, which goes one step further than the feed forward model

ge.html

[5] http://schema.org/

proposed by Jang et al.

Karpathy et al [10] used CNNs on the problem of video tagging, but focused primarily on improvements to the upstream CNN architecture to enhance predictive capability. Because our dataset provides pre-computed CNN features, our focus is primarily on synthesizing these rich features with label structure, and constructing a robust ensemble model for final prediction.

Vinyals et al [11] produced an open source image captioning architecture[6] similar to the multi-label image classification model proposed by Wang et al [4], including the use of beam search during inference. The problem of image captioning is very similar to the problem of image labelling, with the key difference being that in the former the order of the outputted labels matters, and in the latter they do not. Their approach was one of the first to create a holistic model that learns visual and label features in contrast to previous models [12, 13] that stitched together those subproblems using disjoint model architectures.

Other approaches that do not employ the CNN-RNN architecture for exploiting label structure in multi-label image classification include [14], which trains a series of binary classifiers that predict whether the given label is present in the image or not given the image features and the previous predictions. Graphical models have also been employed to model label similarity, including Conditional Random Fields [15], Dependency Networks [16], and co-occurrence matrices [17], and Label Augment Models [18]. However, these models fall short in only capturing pairwise label correlations, whereas the RNN can efficiently capture more complex probability distributions, as Wang et al [4] demonstrated. As such, our solution also went with a CNN and RNN formulation.

3. Methods

Our model uses the video-level visual features generated by the Inception-V3 network in conjunction with an LSTM to generate per label confidence scores, roughly probabilities that the given video is tagged with the given label. Concretely, we attempt to minimize the cross-entropy loss of the predicted label probabilities from the true label probabilities, where a true label has probability 1 if the video is tagged with the label, and 0 if it is not.

The model architecture described by Figure 3 is our version of the CNN-RNN hybrid architecture popularized for image classification and captioning by [4, 11]. It begins by taking the robust visual features C generated by the Inception-V3 network as inputs to a multi-layer perceptron (MLP) shown in the figure as purple denoted Video Projection. Given a sequence of input features of length N, the Video Projection layer non-linearly combines these features using a sequence of ReLU activated fully connected layers to generate a vector in the concept embedding space. Each concept vector has dimensionality D, so we project the final output vector I from this layer to have length D as well.

At training time, we select from the set of ground truth labels the set of K labels with the highest confidence for the given video. We break ties by selecting the labels that most frequently occur in the dataset, so as to optimize our prior probability of choosing a valid label for the video. We then perform a lookup into our label embedding matrix to obtain the concept vectors of length D corresponding to the top K labels for the current video. We talk more about the concept embeddings in section 3.2 below.

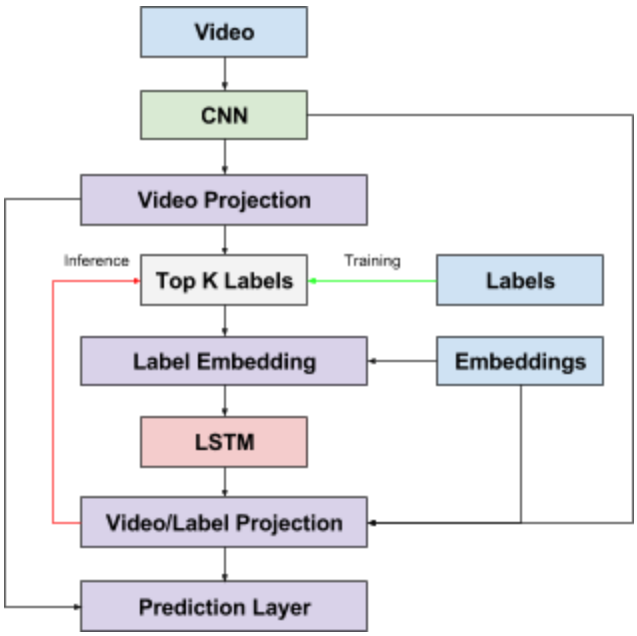

Figure 3. The network architecture used in our model. The video features (generated by a CNN) and concept embeddings are provided as inputs. Several dense layers are used to combine features and project their representations into the appropriate dimensions. An LSTM is used to learn the label dependencies. The video and label features are combined into a concatenated feature vector for prediction.

Now that we've embedded the ground truth labels L to

[6] https://github.com/tensorflow/models/tree/master/im2txt

represent our desired predictions at each time step in the RNN, we begin the process to learning the label dependencies. We use a Long Short-Term Memory (LSTM) cell for this step. The LSTM cell is given as its initial input the output vector from the Video Projection layer *I*. At each time step *t*, the LSTM outputs an M dimensional vector of output features. The M output features at timestep *t* denoted O_t is finally concatenated with the raw CNN input features *C* to produce the final feature vector:

$$O_2 = [C,\ O]$$

Here, *O* is the t by M matrix where $O[t] = O_t$. This matrix acts as our Video/Label Projection output from Figure 3. Finally, we apply a multi-layer perceptron to O_2 to produce our final V dimensional output of label probabilities, given our label vocabulary of length V. Similar to the Video Projection layer, we apply a series of ReLU dense layers to the input to produce our final output confidence scores, which are normalized with a sigmoid function to produce the final probabilities.

3.1. Inference

At the time of inference, we no longer have access to the ground truth label vectors for the purpose of generating the input features to our LSTM cell at time *t*. Instead we must synthesize these features from our previous predictions.

One approach would be to either sample or select the argmax label from the output probabilities of the model at time *t-1*, but this creates the problem that if our first guess is wrong, the entire subsequent chain of predictions is likely to be wrong as well.

As an alternative, we use a technique known as Beam Search where we select some number of beams B, and at each time step we retain the B highest scoring labels.

To generate our a priori estimates of the label scores independent of their semantic structure, we use the same dense MLP architecture from the Video Projection layer to generate a vector of features projected into the label vocabulary space. Given V possible labels in our vocabulary, the resulting vector is now of length V, and corresponds to rough confidence scores for each label in our vocabulary with respect to the current image. At inference time, we use these coarse confidence values to make our first set of selections for Beam Searching.

3.2. Concept Embeddings

It's been shown [19] that word embeddings generated by methods such as *word2vec*[7] are effective at encoding semantic relationships between words in a language. Because our labels are entities or concepts that span multiple words, we needed vectors trained to learn the context of holistic concepts.

We explored using entity vectors generated by Google using *word2vec* and the Freebase open source knowledge base, but it was found that 845 of the tags in our vocabulary were missing from the pretrained embeddings. Furthermore, the embeddings had dimensionality 1000, which was prohibitively expensive for training. We decided to explore other embeddings.

As an alternative, we used concept embeddings described in more detail in [20] that attempt to encode entity similarity with higher accuracy than *word2vec* with dimensionality 300. Of the 4716 label classes in the vocabulary, 2226 had direct concept vector mappings, 2237 were approximated from the Wikipedia descriptions provided in the CSV provided with the YouTube-8M dataset, and 37 were missing and had to be generated with random noise.

To approximate concept vectors from Wikipedia descriptions, we extracted the noun phrases from the descriptions, and queried the concept vectors corresponding to each noun phrase. Then we appealed to the geospatial properties of word vectors and simply calculated the center of mass of all these concept embeddings to create approximate concept vectors.

4. Results

Our experiments were run on Google Compute Engine using the full set of 1024 visual features aggregated for each video and a 4716 vocabulary set of tags.

4.1. Baselines

In evaluating the performance of our models, we began by investigating several baseline models. All of these baselines, except for the dense model described in more detail below, were provided by Google as part of the YouTube-8M starter code[8].

The first and simplest baseline model we examined was a Logistic Model that applies a fully connected layer over

[7] https://code.google.com/archive/p/word2vec/

[8] https://github.com/google/youtube-8m

video level features in the input, using a sigmoid activation function and L2 regularization penalty of 1e-8.

The second model, and the one that performed the best in evaluation, is the Mixture of Experts model. This is an ensemble method where multiple classifiers (experts) that divide the feature space into homogeneous regions, such that one classifiers predicts on one set of features and another classifier on other features. An additional gating network is used to determine which classifier to use for which region of the input. We chose for our baseline a model consisting of two experts and one "dummy" network that always predicts 0.

The third video level model is a dense multi-layer perceptron that attempts to mirror the MLP layers from our CNN-RNN network to determine how much gain we achieve by introducing the semantic concept embeddings. The dense model consists of two hidden layers with ReLU activations and 1024 hidden units. A small regularization penalty is applied to prevent overfitting.

The next two baseline models were selected to operate on the frame level features, to see if using these more granular features would result in greater predictive capability.

The first of these models was a Deep Bag of Frames model that projected the features for each frame into a higher dimensional clustering space and pooled across the frames within that space. It used a configurable video-level model to classify the newly aggregated features, and randomly sampled either frames or sequences of frames during training to speed up convergence.

The final baseline used a stack of Long Short-Term Memory (LSTM) networks to represent each video. It used a forget bias of 1.0 to improve performance. The input to the LSTM was the frame at each time step, up to 300 frames in total. Each frame consisted of 1024 RGB features. A learning rate of 0.001 was used in place of the 0.01 used in other baseline models. The LSTM was very slow to train overall, and surprisingly performed worse than other models along most metrics.

An interesting observation from the baseline experimentation was that video level models tended to significantly outperform frame level models in both time to convergence and overall GAP score. The conclusion we drew was that there was little headroom to be found in simply finding a better way to generate image features from the frames of the video, and that more significant performance gains could be found by focusing instead on the semantic similarity between the labels.

4.2. Hyperparameters

We chose as our learning rate 0.001, which helped in preventing the LSTM from overstepping and consistently predicting 0 scores for all labels. In the Top K layers, we selected K=10. In practice, we choose the top 20 labels when computing the GAP score, but we found empirically that increasing K to 20 did not significantly improve performance, but severely slowed down time to convergence.

For the LSTM layers, we chose M -- the number of units in the output -- to be 512, and stacked the LSTMs into three separate layers.

In beam search, we select B=3, as a means of reducing the computational overhead of using a larger number of paths.

4.3. Evaluation

For evaluation, we ran both the video and frame level training sets for 20 epochs of training.

In addition to the GAP score described above, we also looked at Hit@1, which computes the average number of "hits" where the top scoring prediction is a valid label in the truth label sets. PERR, or prevision at equal recall rate, gives us the average precision at the point where the precision and recall are equivalent. In practice, we found that some classifiers performed better one some metrics than others, but should not diverge too heavily on one over the other.

Model	**GAP**	**Hit@1**	**PERR**
Logistic (video)	.720	.800	.660
MoE (video)	.760	.805	.679
Dense (video)	.708	.811	.669
Deep Bag (frame)	.692	.781	.633
LSTM (frame)	.674	.773	.611
CNN-RNN (video)	.890	.962	.865

Table 1. Performance of each classifier with respect to each of the evaluation metrics described. We make note of which models use video level features and which use frame level

features.

As can be seen from the results above, the CNN-RNN architecture we've laid out significant outperforms baseline models on the evaluation dataset.

5. Conclusions

In summary, we've presented a variation of the CNN-RNN architecture recently popularized as state-of-the-art to the task of image captioning, and applied it to the problem of multi-label video classification. By learning the structural similarity between tags in the label space, we were able to significantly improve classification performance over baseline models.

In contrast to other approaches to this problem, we didn't focus on directly improving the quality of the RGB features by passing them through very deep networks or attempting to create a better aggregator of the frame level features, finding instead that the base video features with a simple model significantly outperformed a power LSTM model on the frame features. Instead, we focused on the label similarity and learning the structure of the label space, and achieved strong improvements as a result of this alternative approach.

Future work should examine ways to combine the beam search generated labels into a unified confidence score for each label that can be compared directly with other classifiers. In this model, we were unable to get an entirely fair comparison against baselines because the nature of our model as sequence generating diverged from the logistic classification models used in the baselines.

We further suspect that with with additional work integrating semantic regularization, this architecture could prove to be a component of state-of-the-art video tagging systems.

References

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan: "YouTube-8M: A Large-Scale Video Classification Benchmark", 2016; arXiv:1609.08675.

[2] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens: "Rethinking the Inception Architecture for Computer Vision", 2015; arXiv:1512.00567.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. FeiFei. Imagenet: A large-scale hierarchical image database. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pages 248–255. IEEE, 2009.

[4] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang: "CNN-RNN: A Unified Framework for Multi-label Image Classification", 2016; [http://arxiv.org/abs/1604.04573 arXiv:1604.04573].

[5] Hexiang Hu, Guang-Tong Zhou, Zhiwei Deng, Zicheng Liao: "Learning Structured Inference Neural Networks with Label Relations", 2015; [http://arxiv.org/abs/1511.05616 arXiv:1511.05616].

[6] G. A. Miller. Wordnet: a lexical database for english. Communications of the ACM (CACM), 38(11):39–41, 1995.

[7] Feng Liu, Tao Xiang, Timothy M. Hospedales, Wankou Yang: "Semantic Regularisation for Recurrent Image Annotation", 2016; [http://arxiv.org/abs/1611.05490 arXiv:1611.05490].

[8] W. Yang and G. Toderici. Discriminative tag learning on youtube videos with latent sub-tags. In CVPR, 2011.

[9] Y.-G. Jiang, Z. Wu, J. Wang, X. Xue, and S.-F. Chang. Exploiting feature and class relationships in video categorization with regularized deep neural networks. arXiv preprint arXiv:1502.07209, 2015.

[10] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1725–1732, Columbus, Ohio, USA, 2014.

[11] Oriol Vinyals, Alexander Toshev, Samy Bengio: "Show and Tell: Lessons learned from the 2015 MSCOCO Image Captioning Challenge", 2016, IEEE Transactions on Pattern Analysis and Machine Intelligence (Volume: PP, Issue: 99 , July 2016); [http://arxiv.org/abs/1609.06647 arXiv:1609.06647]. DOI: [http://dx.doi.org/10.1109/TPAMI.2016.2587640 10.1109/TPAMI.2016.2587640].

[12] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth, "Every picture tells a story: Generating sentences from images," in ECCV, 2010.

[13] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg, "Baby talk: Understanding and generating simple image descriptions," in CVPR, 2011.

[14] H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. In Computer Vision, 2009 IEEE 12th International Conference on, pages 237–244. IEEE, 2009.

[15] N. Ghamrawi and A. McCallum. Collective multi-label classification. In Proceedings of the 14th ACM international conference on Information and knowledge management, pages 195–200. ACM, 2005.

[16] Y. Guo and S. Gu. Multi-label classification using conditional dependency networks. In IJCAI Proceedings International Joint Conference on Artificial Intelligence, volume 22, page 1300, 2011.

[17] X. Xue, W. Zhang, J. Zhang, B. Wu, J. Fan, and Y. Lu. Correlative multi-label multi-instance image annotation. In Computer Vision (ICCV), 2011 IEEE International Conference on, pages 651–658. IEEE, 2011.

[18] X. Li, F. Zhao, and Y. Guo. Multi-label image classification with a probabilistic label enhancement model. UAI, 2014.
[19] Tomas Mikolov, Kai Chen, Greg Corrado: “Efficient Estimation of Word Representations in Vector Space”, 2013; [http://arxiv.org/abs/1301.3781 arXiv:1301.3781].
[20] Robert Speer and Catherine Havasi. "Representing General Relational Knowledge in ConceptNet 5." LREC 2012; [http://lrec-conf.org/proceedings/lrec2012/pdf/1072_Paper.pdf]