CS230

Classifying YouTube Videos by Thumbnail

Ye Chen
Department of Computer Science
Stanford University
yechen9@stanford.edu

Yan Wang
Department of Computer Science
Stanford University
yan12@stanford.edu

Robert Tan
Department of Computer Science
Stanford University
rtan21@stanford.edu

Abstract

YouTube videos can be classified into different categories. Thumbnails, as one of the first things that users look at, often decide what category the video belongs to and whether the user would click on that video. To provide automatic classification based on the thumbnails, we trained 4 different Convolutional Neural Networks (VGG16, Inception, and compared their performance. We classified the thumbnail images into 4 categories. We utilized transfer learning to learn the high level features of the images and added extra layers to achieve the specific classification task. The results demonstrate that Xception's performance is the best among the four CNN archetictures.

1 Introduction

YouTube videos are often classified and tagged with certain video categories in order to be used in YouTube's recommender systems and listed when a user searches videos related to a certain category. The goal of this project is to train different deep learning architectures and compare their performance on classifying the category of the YouTube video based only on its thumbnail. This project can largely help reduce users' cognitive load by auto-classifying the YouTube video. Besides, it can also serve as an input or supplement to other classifiers that classify videos based on other features. Thumbnails are usually the first thing that come to users' eyes, and they are usually very related to the content of the video so that users would click on that video. For example, if the thumbnail is a screenshot of a game, then the video is very likely about games; if the thumbnail contains a sandwich or steak, then it is likely about Food and Drinks. Therefore, a classification algorithm based on thumbnails can help understand the content to a great extent. The input to the algorithm is an image. We then use neural networks to output a predicted category for the image. We will investigate and compare 4 different CNN architectures in this project.

2 Related work

Computer vision and CNNs are applied in various different image and video classification problems. As one of the world's largest video sharing platform, YouTube has become an important source for image and video data. In 2016, Abu-El-Haija et al. introduced YouTube-8M, the largest YouTube video multi-label classification dataset [1]. YouTube-8M is a large-scale labeled video dataset that consists of millions of YouTube video IDs, with high-quality machine-generated annotations from a diverse vocabulary of 3,800+ visual entities [1]. Automatic categorization for YouTube videos has been a challenging task due to noisy and missing labels. In 2010, Wang et al. introduced YouTubeCat to classify different YouTube videos [2]. However, the YouTubeCat algorithm requires a lot of extra information that is not easily available to the users. Considering the thumbnail is usually the first thing that the user would look at, we will simply use YouTube thumbnails to classify different YouTube videos.

The state-of-the-art technique for our task is Convolutional Neural Networks. In recent years, many different Convolutional Neural Networks (CNNS) bring a huge breakthrough in image classification. For example, in 2014, Simonayn et al. introduced a deep neural networks architecture for large-scale image recognition (VGG16) [3]. Very deep neural network started to become the mainstream solutions for many computer vision related problems. In 2016, Szegedy et al. introduced a new method to build very deep neural networks (Inception) [4]; instead of stacking many convolutional layers, they stacked modules or blocks of convolutional layers to efficiently scale up networks. However, as the networks got deeper, they became more difficult to train. To solve this problem, in the same year, He et al. presented a residual learning framework (ResNet-50) to ease the difficulty of training deep neural networks, which are easier to optimize and can gain accuracy from considerably increased depth [5]. In 2017, based on the successful Inception networks, Chollet introduced a depthwise Separable Convolutions called Xception [6]. Xie et al. introduced a homogeneous, multi-branch architecture (ResNeXt-50) that has only a few hyper-parameters to set [7]. There are many other advanced and complicated CNN architectures. As more computing power becomes available, more complicated models generally produce better results. Therefore, in our project, we expect Xception, the most complicated model we will try, to generate the best result.

3 Dataset and Features

The dataset we use is a subset of the YouTube-8M dataset described above. The original dataset is huge and contains a lot of data and information that are not useful for our project. There are 25 categories in total: Arts & Entertainment; Autos & Vehicles; Beauty & Fitness; Books & Literature; Business & Industrial; Computers & Electronics; Finance; Food & Drink; Games; Health; Hobbies & Leisure; Home & Garden; Internet & Telecom; Jobs & Education; Law & Government; News; People & Society; Pets & Animals; Real Estate; Reference; Science; Shopping; Sports; Travel; Unknown [1]. Each category can be divided into some subcategories, so there are 3862 subcategories [1].

Figure 1: A thumbnail with category `Arts & Entertainment`

The first step is to preprocess the dataset to generate labeled thumbnails. Most thumbnails are 360x240 and we will not resize the images until feeding into the model. Figure 1 is an example thumbnail in the Arts & Entertainment category. In order to do this, for each YouTube ID in the dataset, we obtained its thumbnail by querying YouTube's API. Then, given the mapping from subcategories to main categories, we converted every ID's labels to a set of main categories. We pick the most frequent category as its final label for the thumbnail. If there are multiple labels with the same frequency, we simply pick a random one. Although we are able to obtain the labeled thumbnails, it is hard to do all the data preprocessing work while training and testing our models. To save space and get it easier for training, we serialized the images and labels into bytes and uploaded the tfrecord to AWS S3 for later use.

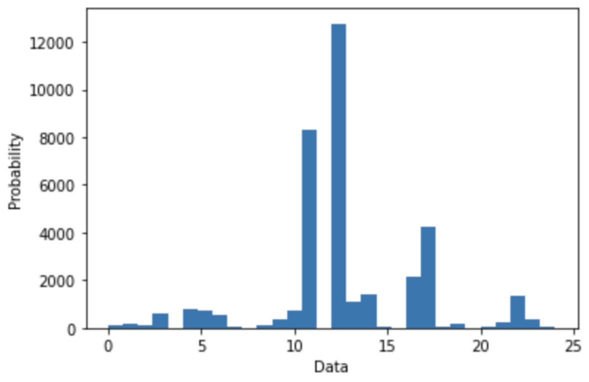

Figure 2: Histogram of 25 categories

We collected over 300,000 labeled training examples and the size of the dataset is around 150 GB. However, due to limited GPUs and computing power, we were only able to train, validate and test on 4,800 examples from our collected dataset. In addition, because the size dataset is greatly reduced, we also reduce our initial 25 categories to only 4 categories. From Figure 1, we can see the columns 11, 12 and 17 are the most frequent labels in our dataset, which represent Arts & Entertainment, Games, and Autos & Vehicles. We add a fourth category to represent all the other original categories.

4 Methods

We experiment on four different Convolutional Neural Networks (CNNs) in our project: VGG16, Inception, ResNet-50, and Xception. Due to limited size of the dataset, we apply Transfer Learning to first extract features from the pretrained layers and add final layers to do the classification task. All the pretrained model are trained on the ImageNet dataset [8] and provided by TensorFlow [9]. At the end of the pretrained model, we first add a flatten layer to reshape the tensors. Then we use a Dense layer with 1024 neurons and ReLU as its activation function. Then we use Dropout layer to reduce overfitting. The final layer is a Softmax layer with 4 categories. In the following subsections, we we introduce the pretrained models in detail.

4.1 VGG16

The input to the model is of fixed size 224 x 224 x 3 image. The image is passed through a stack of convolutional layers with 3 x 3 filters. The convolution stride is fixed to 1 pixel; the spatial padding of convolutional layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1-pixel for 3×3 convolutional layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the convolutional layers. Max-pooling is performed over a 2×2 pixel window, with stride 2. Three Fully-Connected (FC) layers follow the 5 convolutional blocks: the first two have 4096 channels each, the third performs 1000-way ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer. Since we have 25 different classes in our dataset, the final Dense layer contains 25 channels as well [3].

4.2 Inception

There exists multiple versions of Inception model [4]. We use Inception_v3 for this project. It uses Factorized Convolutions to reduce the computational efficiency as it reduces the number of parameters involved in a network. It also keeps a check on the network efficiency. By replacing bigger convolutions with smaller convolutions, it leads to faster training as well. The thir important point is asymmetric convolutions. A 3×3 convolution could be replaced by a 1×3 convolution followed by a 3×1 convolution. If a 3×3 convolution is replaced by a 2×2 convolution, the number of parameters would be slightly higher than the asymmetric convolution proposed. It also uses auxiliary classifiers for regularization and pooling operations for grid size reduction.

4.3 ResNet-50

Similar to Inception_v3, ResNet-50 is a newer version based on the original ResNet paper [5].The architecture of ResNet50 has 4 stages. The network can take the input image of size 224 x 224 x 3. It performs the initial convolution and max-pooling using 7×7 and 3×3 kernel sizes respectively. Afterward, Stage 1 of the network starts and it has 3 Residual blocks containing 3 layers each. The size of kernels used to perform the convolution operation in all 3 layers of the block of stage 1 are 64, 64 and 128 respectively. The convolution operation in the Residual Block is performed with stride 2, hence, the size of input will be reduced to half in terms of height and width but the channel width will be doubled. As we progress from one stage to another, the channel width is doubled and the size of the input is reduced to half. Finally, the network has an Average Pooling layer followed by a fully connected layer having 1000 neurons (ImageNet class output).

4.4 Xception

Xception is an extension of the inception Architecture which replaces the standard Inception modules with depthwise Separable Convolutions [6]. Cross-channel (or cross-feature map) correlations are captured by 1×1 convolutions. Spatial correlations within each channel are captured via the regular 3×3 or 5×5 convolutions. The model has 71 layers in total.

5 Experiments and Results

The first step is to resize the image from 360 x 240 x 3 to 224 x 224 x 3 so that we can directly feed the images into the model (150 x 150 x 3 for Inception). Then we normalized the pixels of the images by dividing each pixel by 255. This step makes training easier and faster. Since we have 4 classes, we applied one-hot encoding on the labels to make each label of size (4, 1). After these two steps, the shape of our training images is (3200, 224, 224, 3) and the the shape of our training labels is (3200, 4). The shape of our validation images is (800, 224, 224, 3) and the the shape of our validation labels is (800, 4).

Because our task is multi-class classification, we chose categorical_cross_entropy as the loss function. We tried multiple batch size and the results are very similar, so we just keep 32 as the batch size for all four models. For optimizer, we tried RMSProp, SGD, and Adam, each with learning rate from 0.01 to 0.00001. The results showed that Adam or RMSProp with learning rate 0.001 to 0.00001 generates the similar results for all four models. When learning rate = 0.01, the training accuracy doesn't improve at all over time. For simplicity, we chose Adam with learning = 0.00001 as our final optimizer.

Model	Training Acc	Validation Acc	Test Acc	Precision	Recall	F1 score
VGG16	81.38%	60.12%	59.38%	0.5707	0.6175	0.5729
Inception	100.00%	60.00%	59.87%	0.5752	0.6083	0.5837
ResNet50	49.78%	47.50%	47.12%	0.4657	0.4941	0.4546
Xception	100.00%	67.87%	65.13%	0.6297	0.6675	0.6391

Table 1: Results

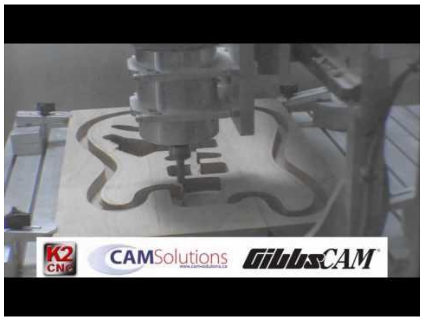

Figure 3: Misclassified example

From Table 1, we observe that training accuracy is generally a lot higher than the validation accuracy across all 4 models. The Test accuracy on 800 examples is very close to the validation accuracy. From the metrics above, we can easily conclude that Xception's performance is the best among the four CNN models. Since the dataset is small, models are prone to overfitting, so the results are within our expectation. In Figure 3, we present the confusion matrices of the test set. All the four models classify the first three categories relatively well, but did poor job in classifying the "Others" category, which makes sense because the "Others" category contains a variety of different categories, so the images don't share as much common features as in the same category. We expect the accuracy to increase if we can train on larger datasets. Figure 4 shows a misclassified example. It belongs to the "Arts & Entertainment" category while it is predicted to be "Games" category. This example is hard to classify because through human eyes, we can think of it to be either "Arts" or "Games". Both make sense.

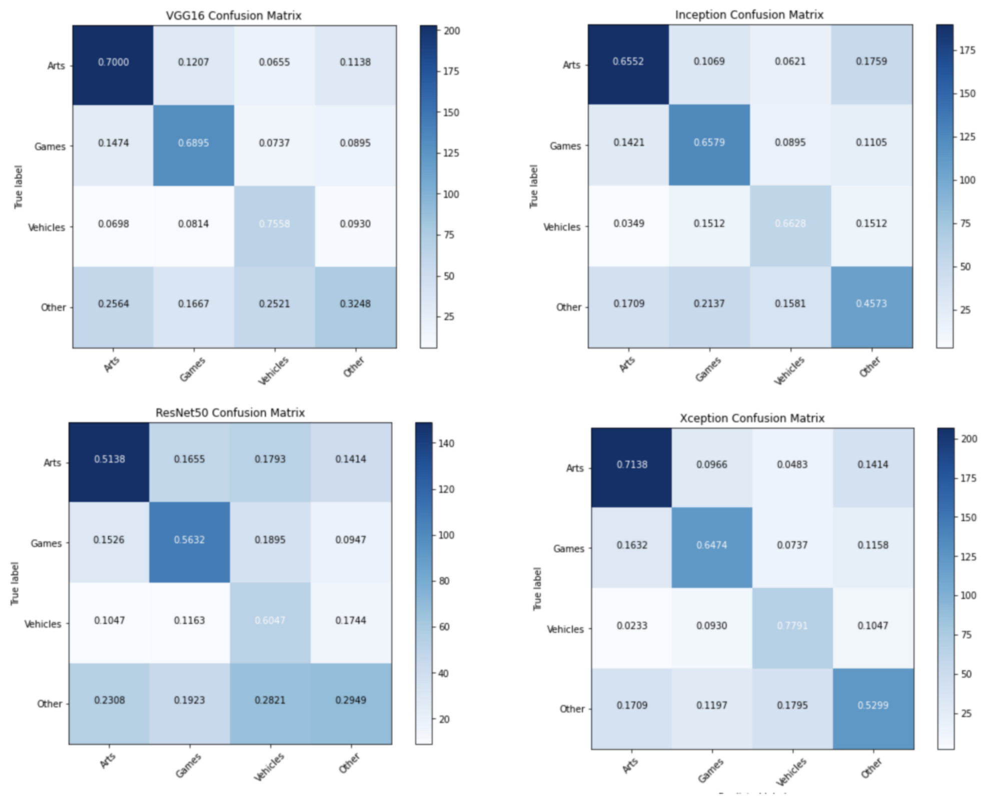

Figure 4: Confusion matrices of 4 models

6 Conclusion and Future Work

The results demonstrate that Xception works best for classifying YouTube videos based on thumbnails, which aligns with our expectation. We were only able to train on a small subset of data we have collected and divide the images into 4 major categories. There are a few things that we can try in the future to extend and improve our models. First, we can extend the model to classify the full 25 categories of images. If trained on 25 categories, then the number of misclassified "Others" category is supposed to be greatly reduced. Second, although we have collected around 300,000 examples, due to limited computing resources (AWS didn't approve our GPUs request), we were only able to train on 4,000 examples. When we get access to more computing power, we will train on a larger dataset and expect a better accuracy. Third, each YouTube videos is tagged with multiple categories and we selected the most frequent category. In the future, we can try the multi-label classification, which will make the model more useful and reasonalbe in practice.

7 Contributions

All three members of the team contributed to each aspect of the project, but each focused more on different areas. Ye and Robert focused on collecting the YouTube thumbnail datasets and recording final video presentation. Yan focused on training and testing CNN models and performing result and error analysis. All team members contributed to the literature review, proposal, milestone, and final reports.

8 Code Link

https://github.com/robert-tan/yt-thumbnail-classifier

References

[1] Abu-El-Haija, S., Kothari, N., Lee, J., Natsev, P., Toderici, G., Varadarajan, B., & Vijayanarasimhan, S. (2016). Youtube-8m: A large-scale video classification benchmark. arXiv preprint arXiv:1609.08675.

[2] Wang, Z., Zhao, M., Song, Y., Kumar, S., & Li, B. (2010, June). Youtubecat: Learning to categorize wild web videos. In 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (pp. 879-886). IEEE.

[3] Simonyan, K., Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[4] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2818-2826).

[5] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[6] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1251-1258).

[7] Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2017). Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1492-1500).

[8] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248–255).

[9] Abadi, Martin, Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., … others. (2016). Tensorflow: A system for large-scale machine learning. In 12th Symposium on Operating Systems Design and Implementation (pp. 265–283).