final project

Video Classification with Feature Extraction: Transfer Learning in Keras Using Pre-trained Models

Student 1, Student 2

IST 652 Scripting for Data Analysis
Professor Deborah Landowski
Syracuse University // September 2018

Table of Contents

Introduction

The goal of the study is to solve a real-world business question and to enable the authors to learn about TensorFlow and Keras. In order to accelerate our learning and coming up with a solution to the business question, the authors decided to leverage “Transfer Learning” and make use of pre-trained models in Keras with TensorFlow backend. We will apply pre-trained models to three *Lip Sync Battle Shorties* videos to explore automatic classification of the videos. Our first question is whether pre-trained models is able to generate a set of features that may help predict the popularity of new Lip Sync Battle Shorties. Is this a viable solution? This report documents the process of and discoveries from our exploratory study.

What is “Lip Sync Battle Shorties”?
Nickelodeon's kid-focused version of "Lip Sync Battle" has made its debut as "Lip Sync Battle Shorties." Kids will battle each other and go head-to-head by lip-syncing their favorite songs in celebration of artists that they admire.

Data Sources

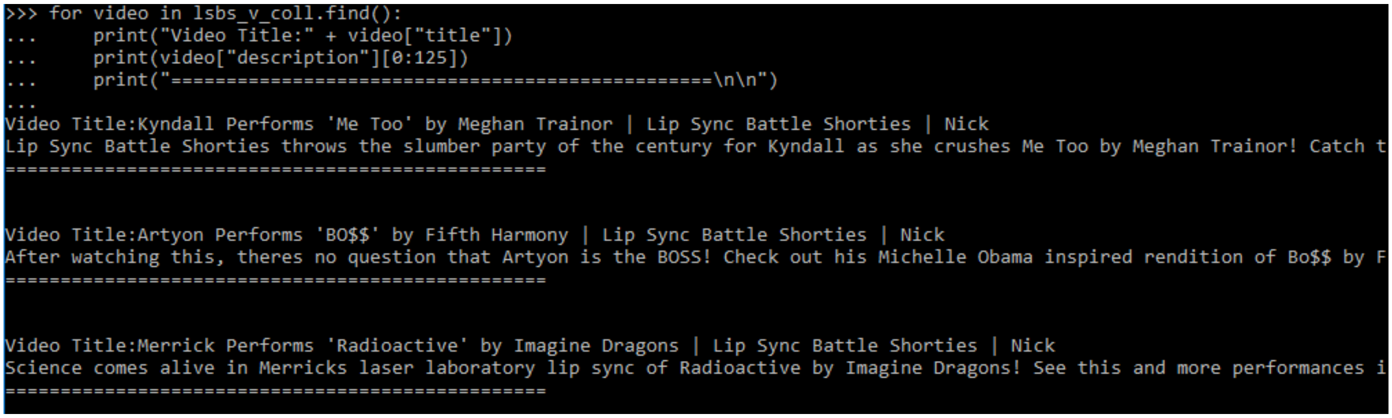
Three YouTube videos from Nickelodeon’s channel were used for analysis. The videos are performances from Nickelodeon’s *Lip Sync Battle Shorties* program.

Data source 1: Kyndall Performs 'Me Too' by Meghan Trainor | Lip Sync Battle Shorties
“Lip Sync Battle Shorties throws the slumber party of the century for Kyndall as she crushes Me Too by Meghan Trainor!”

Data source 2: Artyon Performs 'BO$$' by Fifth Harmony | Lip Sync Battle Shorties
“After watching this, there’s no question that Artyon is the BOSS! Check out his Michelle Obama inspired rendition of Bo$$ by Fifth Harmony from Lip Sync Battle Shorties”

Data source 3: Merrick Performs 'Radioactive' by Imagine Dragons | Lip Sync Battle Shorties
“Science comes alive in Merricks laser laboratory lip sync of Radioactive by Imagine Dragons”

This code output demonstrates getting data source information from MongoDB.



```
>>> for video in lsbs_v_coll.find():
...     print("Video Title:" + video["title"])
...     print(video["description"][0:125])
...     print("=================================================\n\n")
...
Video Title:Kyndall Performs 'Me Too' by Meghan Trainor | Lip Sync Battle Shorties | Nick
Lip Sync Battle Shorties throws the slumber party of the century for Kyndall as she crushes Me Too by Meghan Trainor! Catch t
=================================================


Video Title:Artyon Performs 'BO$$' by Fifth Harmony | Lip Sync Battle Shorties | Nick
After watching this, theres no question that Artyon is the BOSS! Check out his Michelle Obama inspired rendition of Bo$$ by F
=================================================


Video Title:Merrick Performs 'Radioactive' by Imagine Dragons | Lip Sync Battle Shorties | Nick
Science comes alive in Merricks laser laboratory lip sync of Radioactive by Imagine Dragons! See this and more performances i
=================================================
```

Pre-processing

Each video was downloaded from the YouTube content servers. We then proceeded to extract image frames from each video at a rate of 24 frames per second of video. YouTube video tags (title, description, “keywords” - unstructured text) were extracted via YouTube video API and stored in a MongoDB.

Method of Analysis

Two pre-trained models, Inception v3 and ResNet50, were used to classify the three Lip Sync Battle Shorties videos. Both models were convolutional network, pre-trained on ImageNet, a large-scale image database containing 1.4 million images with 1000 categories.

At various points of the video analysis pipeline, analyses were conducted; analytic results were propagated forward to later pipeline stages.

Analysis method 1	Data input	Output
Run the image set through pre-trained Inception v3 and ResNet50 models in the Keras package to generate features for video	Processed image fran extracted from video	Raw feature list

Analysis method 2	Data input	Output
Aggregate raw features and compute the sum of feature count across each video	Raw feature list	Ranked features

Analysis method 3	Data input	Output
Run the image set through pre-trained Inception v3/ResNet50 models in the Keras package to generate features for video	Ranked features	Top n features

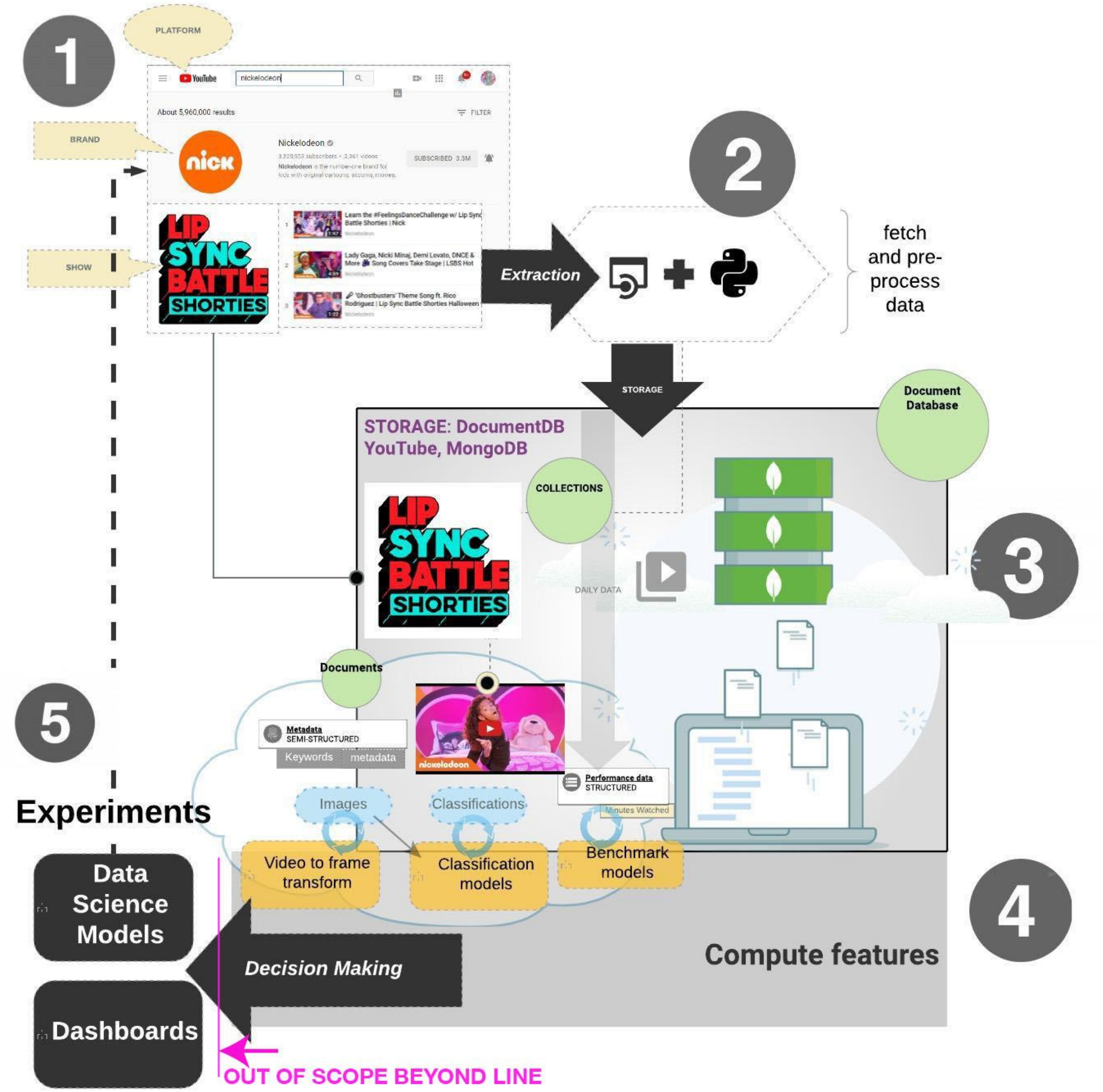
Analysis method 4	Data input	Output
Run regression model to find coefficients of features extracted by Inception v3 and ResNet50 models.	Top n features	Ranked features

The code from step 4 will be used for future study with a large sample size. The outcome will provide a set of features that can be used to train models to predict the popularity of new videos (a form of supervised machine learning).

Lastly, as a sanity check/tuning procedure, the accuracy of the Convolutional Neural Network models were evaluated by assessing the mean confidence scores.

Notable omission: *In addition to CNN models, YouTube video tags (title, description, "keywords" - unstructured text) extracted via YouTube video API were compared. It was discovered that the tags are mostly the same for all three videos. Therefore, they are not good predictors of popularity of videos. Furthermore, the pre-trained categories that inception v3 uses are highly unlikely to be found in any of the ~50 tags allocated to video. This analysis was made, but not presented due to its lack of compelling information.*

Description of the Pipeline



1. **Acquire data & pre-process data**
    a. For each video, we will acquire statistical data and perform exporatory data analysis to derive the "performance measure."
    b. Video assets will be downloaded as MP4 files from YouTube's Creator Studio.
        i. We wanted to write the MP4 assets to MongoDB. We spent a bit of time learning how to write videos to MongoDB using GridFS, and though we understood the procedures, we felt it was not a good use of our time, so we kept the videos in our shared repository.
2. **Pre-processing**
    a. Performance data, video metadata, and other data pertaining to a video will be written to a MongoDB node on cloud atlas
    b. Images will be extracted from videos: 24 image per second of video (e.g., a video with duration 2:00 produces 2,880 images). This was accomplished using OpenCV.
    c. We will install and configure pre-trained models in Keras.

(Author's note: Ian Aliman accomplished the load of YouTube metadata in Homework 2).

3. **Script data analysis of pipeline output:**
    a. Use the two pre-trained models (Inception v3 model and ResNet50 model) for unsupervised machine-learning process to classify the three videos.
    b. Compare the classification/features classified by both models. Specifically, review top features and perform regression analysis to identify a set of features that predict the popularity of new videos.
    c. Classify videos with image frames using Keras:
        i. Remove generic program title frames from each video. When testing the pre-trained ResNet50 model, it was discovered that the beginning and the ending program slides might have skewed the classification. For example, 'web_site' came up as a top category. Therefore, About 40 title frames and 500+ ending screens were removed to prevent irrelevant classification.
        ii. Code the pre-trained ResNet50 model in the Keras Python package. The code was tested with one image to ensure the model runs.
        iii. Use the ResNet50 model to classify each video.
        iv. Code the pre-trained Inception v3 model provided in the Keras Python package. The code was tested with one image to ensure the model runs.
        v. Use the Inception v3 model to classify each video.

4. **Interpret the results of features extracted by applying the Inception v3 and ResNet50 models:**
    i. Compare top features extracted by the two models.
    ii. Create a wordcloud for features classified by each model for each video and compare the outcome between two models.
    iii. Run the multiple regression model to evaluate the coefficients of features and identify the set of features that predict the popularity of videos. (A video with about 20M views is considered popular; a video with about 10M views is considered unpopular.)

5. **Save enriched model results for business stakeholders' review and further modeling.**

Analysis & Interpretation

The single most important outcome of this assignment was to learn about how to implement feature generation for image/video content using TensorFlow as part of a data science pipeline. Going into this project, we knew that, due to the extremely low sample size chosen, we would not make scalable analysis. The goal of this project was to 1) build a data pipeline for ingesting videos and creating features, 2) explore transfer learning in Keras with pre-trained models.

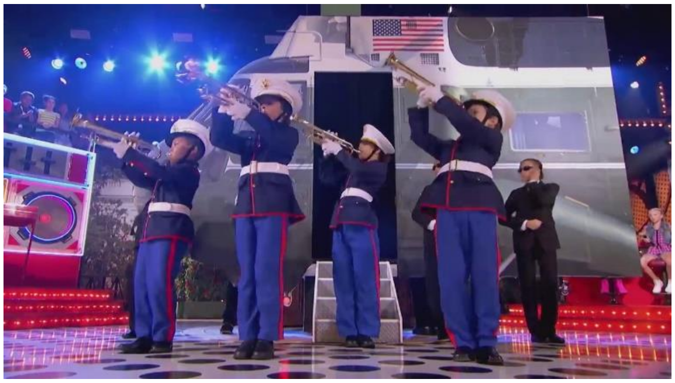
Assessing the viability of pre-trained models

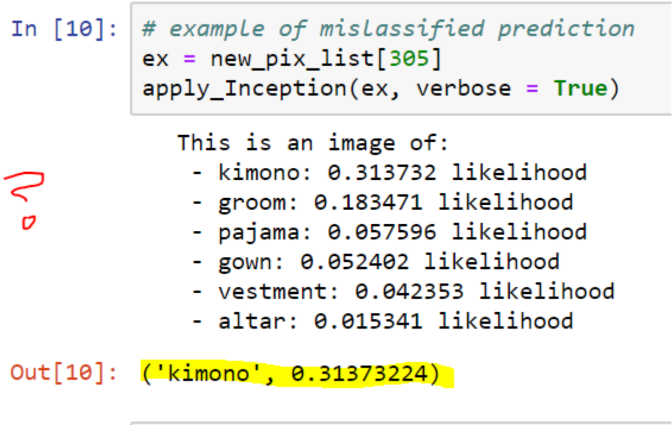
Early assessments of the ImageNet predictions for Nickelodeon videos showed that something was slightly off (for example, 4 kids playing trumpet is not a kimono). The accuracy of the object recognition algorithm is extremely important, as we want to use these generated features for additional modeling and business insights. In the later stages of this project, we thought it was a good idea to write a script to report on the veracity of the pre-trained models.

Central Limit Theorem and Law of Large Numbers. *The Central limit Theorem states that when sample size tends to infinity, the sample mean will be normally distributed. The Law of Large Number states that when sample size tends to infinity, the sample mean equals to population mean. source:www.mathcentral.uregina.ca/QQ/database/QQ.09.99/yam1.html*

We did this by estimating the mean confidence of a video based on all the predictions an image classifier makes. We did this by estimating the population mean, AKA, the expected confidence of a classification result. Below are the normal distributions created.

- **Takeaway: The 1000 classes in the** pre-**trained** models don't really work for Lip Sync Battle Shorties, and transfer learning models require non-trivial amounts of time to set up.





```
In [10]: # example of mislassified prediction
         ex = new_pix_list[305]
         apply_Inception(ex, verbose = True)

            This is an image of:
             - kimono: 0.313732 likelihood
             - groom: 0.183471 likelihood
             - pajama: 0.057596 likelihood
             - gown: 0.052402 likelihood
             - vestment: 0.042353 likelihood
             - altar: 0.015341 likelihood

Out[10]: ('kimono', 0.31373224)
```

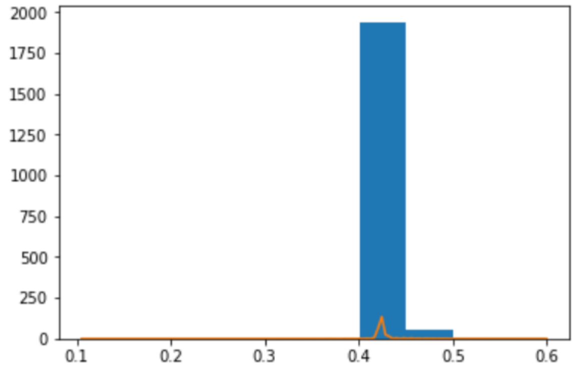
?

Left: a group of people playing trumpets. Right: our guess is kimono?

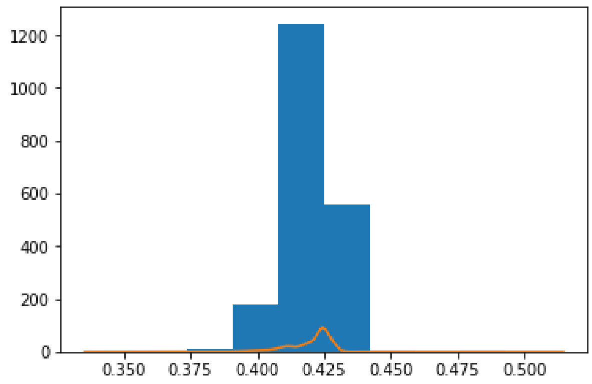
All models had an expected mean confidence level of ~0.42 (ranging from 0 to 1). In the future, we think spending more time on strategic blending of the pre-trained model with new stimuli.

Merrick performs radioactive (Inception v3)

```
mean of all top predictions:  0.42711303
Mean of sample means -- 0.42545828
```

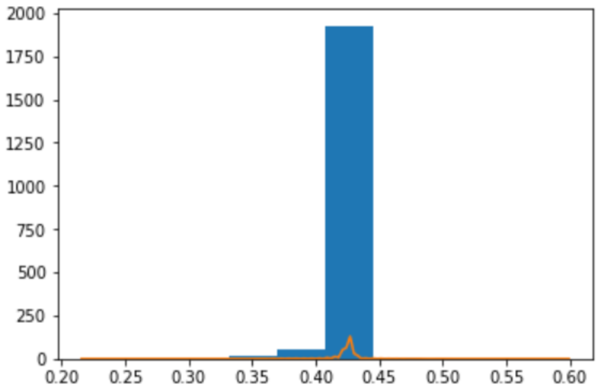


Kyndall performs Me Too (Inception v3)



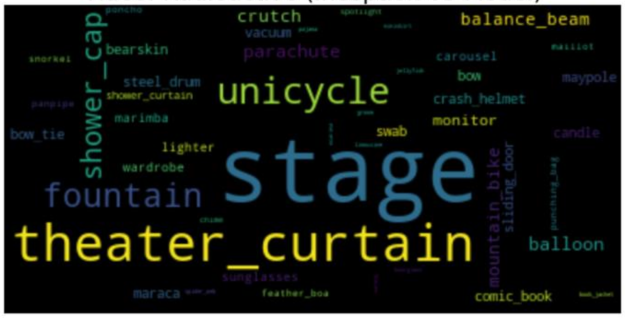
Artyon performs Boss (Inception v3)

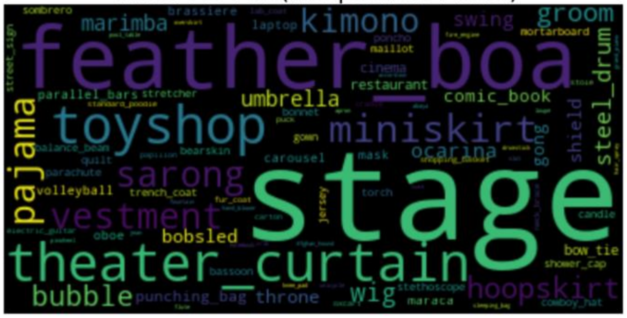
```
mean of all top predictions:  0.4275622
Mean of sample means -- 0.4238689
```

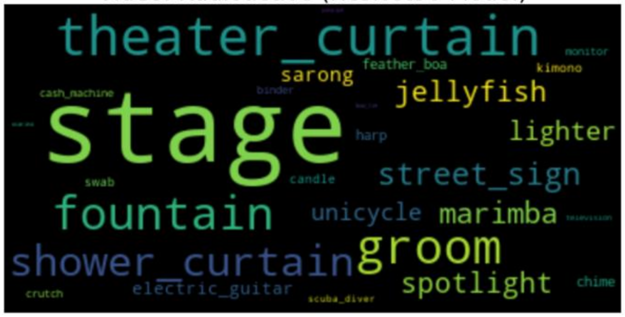


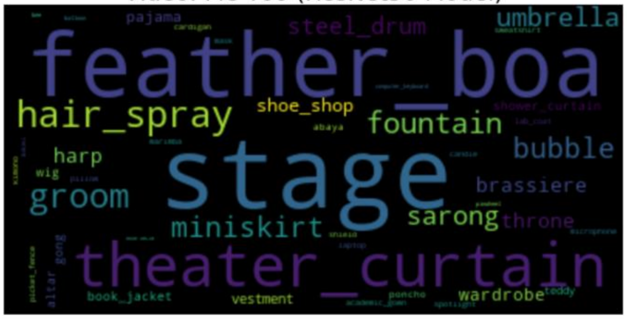
Top Features

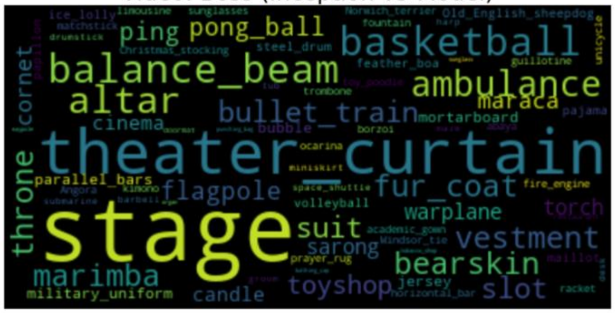
Six wordclouds were created to compare top features extracted from each video by Inception v3 model vs. ResNet50 model.

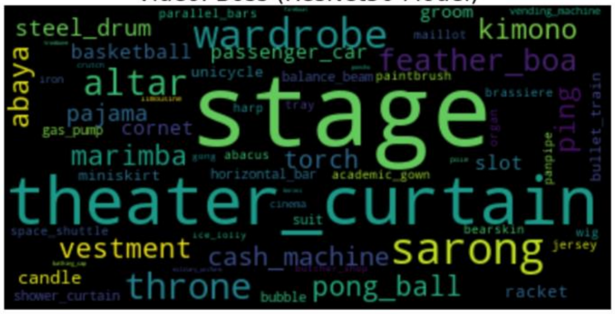

Video: Radioactive (Inception v3 Model)


Video: Me Too (Inception v3 Model)


Video: Radioactive (ResNet50 Model)


Video: Me Too (ResNet50 Model)

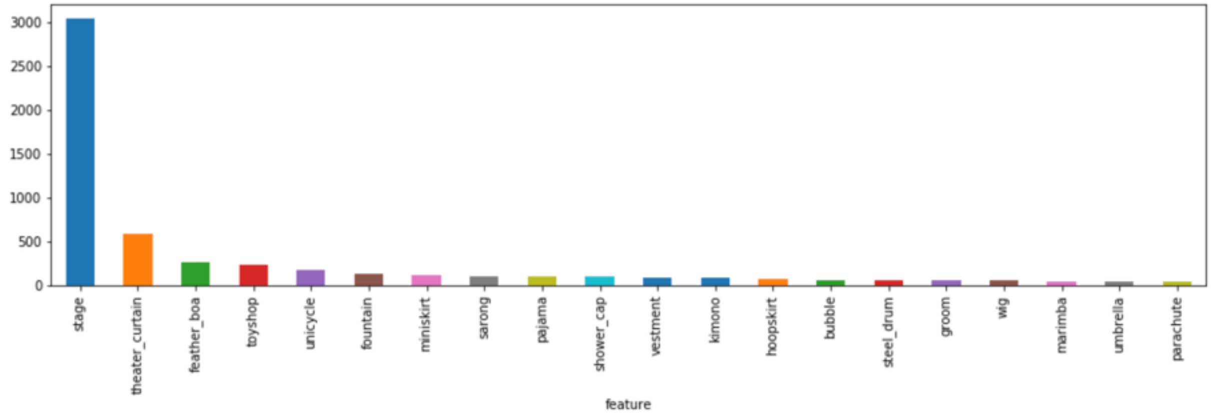

Video: Boss (Inception v3 Model)


Video: Boss (ResNet50 Model)

We separate the 3 videos into two groups — popular vs. unpopular. A video with about 20M views is considered popular; a video with about 10M views is considered unpopular. Then explored the top 20 features for each group and compared them between two models. Bar charts were created to visualize the frequency distribution.

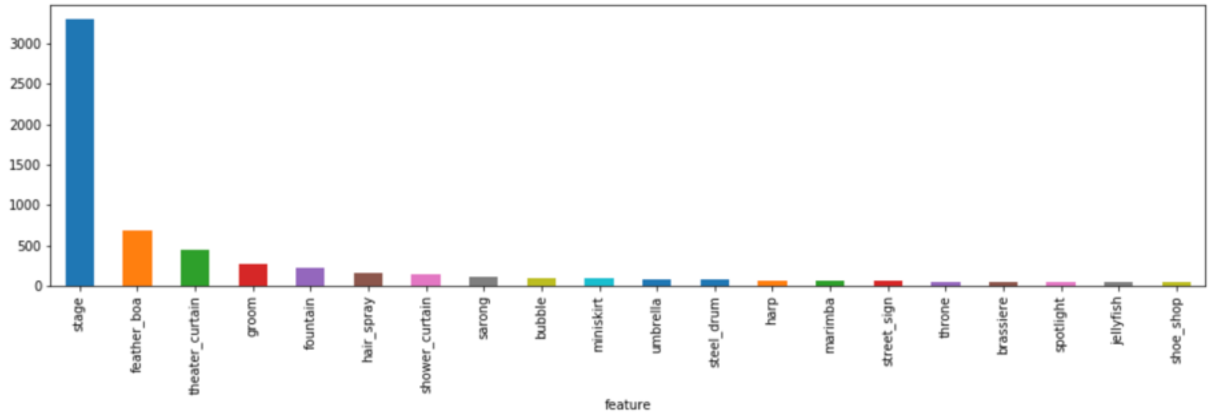
Top 20 features extracted from popular videos (with about 20M+ views) by Inception v3 model

stage	3042	vestment	92
theater_curtain	587	kimono	81
feather_boa	270	hoopskirt	69
toyshop	227	bubble	64
unicycle	174	steel_drum	63
fountain	127	groom	63
miniskirt	112	wig	52
sarong	98	marimba	50
pajama	97	umbrella	43
shower_cap	95	parachute	3



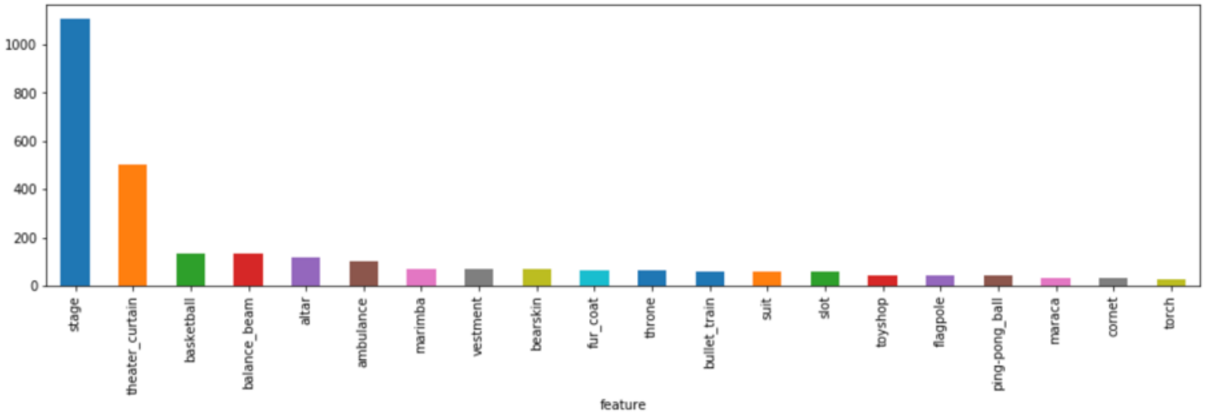
Top 20 features extracted from popular videos (with about 20M+ views) by ResNet50 model

stage	3303	umbrella	85
feather_boa	688	steel_drum	72
theater_curtain	438	harp	62
groom	271	marimba	56
fountain	229	street_sign	55
hair_spray	160	throne	54
shower_curtain	137	brassiere	52
sarong	109	spotlight	51
bubble	95	jellyfish	47
miniskirt	95	shoe_shop	46



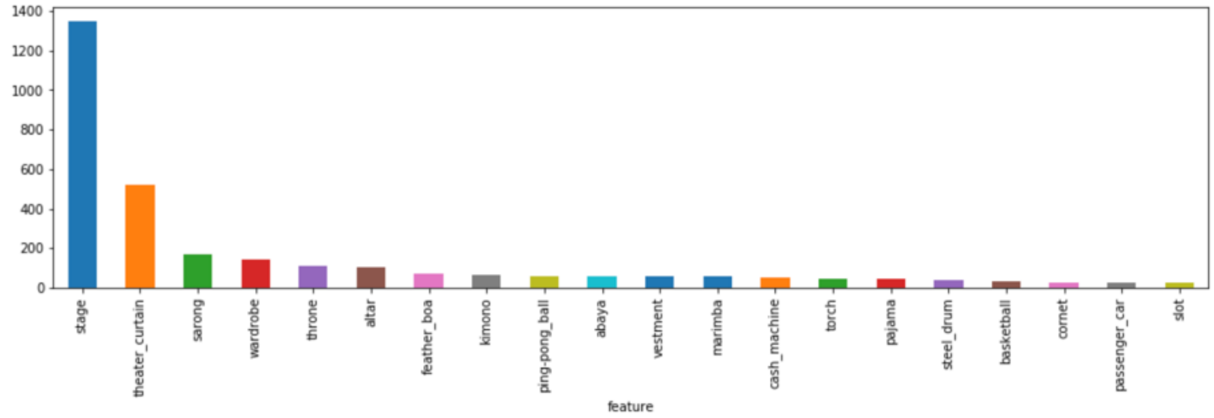
Top 20 features extracted from unpopular videos (with about 10M views) by Inception v3 model

stage	1108	throne	65
theater_curtain	503	bullet_train	59
basketball	135	suit	57
balance_beam	133	slot	56
altar	119	toyshop	43
ambulance	99	flagpole	40
marimba	69	ping-pong_ball	40
vestment	67	maraca	34
bearskin	67	cornet	33
fur_coat	66	torch	29



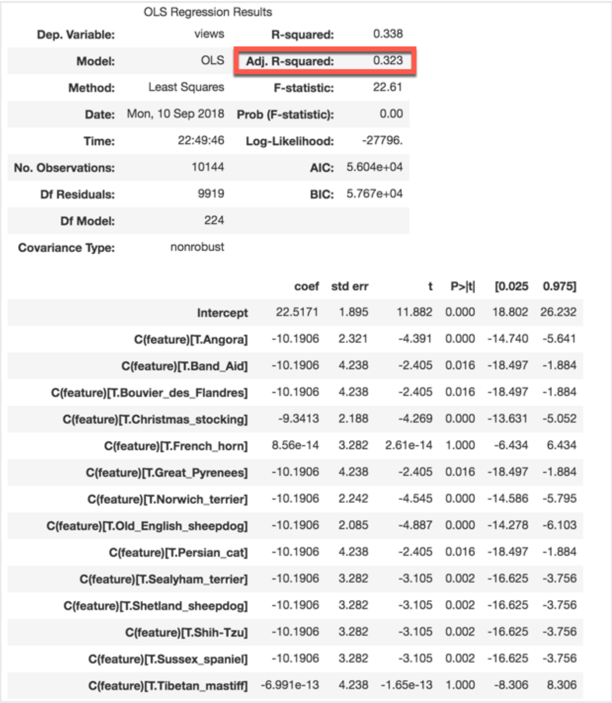
Top 20 features extracted from unpopular videos (with about 10M views) by RestNet50 model

stage	1349	vestment	58
theater_curtain	521	marimba	57
sarong	169	cash_machine	54
wardrobe	141	torch	47
throne	112	pajama	43
altar	105	steel_drum	41
feather_boa	74	basketball	29
kimono	62	cornet	28
ping-pong_ball	61	passenger_car	27
abaya	60	slot	27



Regression Analysis

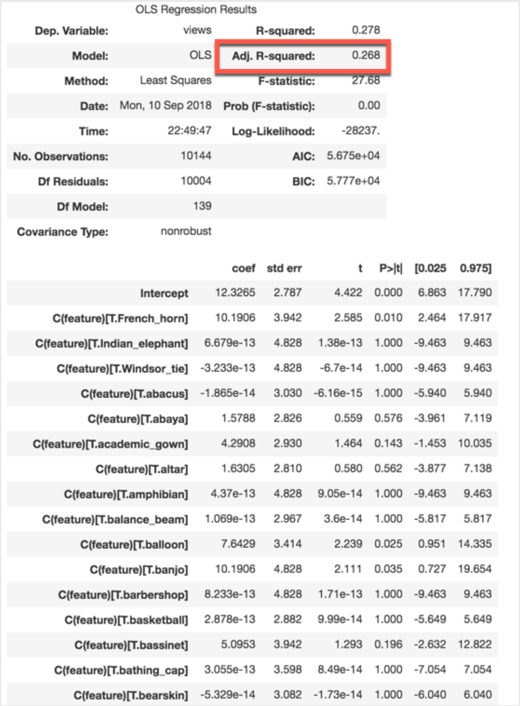
The regression was exploratory in nature. With only 3 video samples, we really don't know whether the X and Y have a linear relationship. The results showed several p values below 0.05, so we know that these features (i.e. words) do have a correlation with 'views'. However, the low adjusted R square shows that only 32.3% (Inception v3 model, image below on the left) and 26.8% (ResNet50 model, image below on the right) of the variations of Y can be explained by the model. This is understandable because our sample size is too small.



OLS Regression Results

Dep. Variable:	views	**R-squared:**	0.338
Model:	OLS	**Adj. R-squared:**	0.323
Method:	Least Squares	**F-statistic:**	22.61
Date:	Mon, 10 Sep 2018	**Prob (F-statistic):**	0.00
Time:	22:49:46	**Log-Likelihood:**	-27796.
No. Observations:	10144	**AIC:**	5.604e+04
Df Residuals:	9919	**BIC:**	5.767e+04
Df Model:	224		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	22.5171	1.895	11.882	0.000	18.802	26.232
C(feature)[T.Angora]	-10.1906	2.321	-4.391	0.000	-14.740	-5.641
C(feature)[T.Band_Aid]	-10.1906	4.238	-2.405	0.016	-18.497	-1.884
C(feature)[T.Bouvier_des_Flandres]	-10.1906	4.238	-2.405	0.016	-18.497	-1.884
C(feature)[T.Christmas_stocking]	-9.3413	2.188	-4.269	0.000	-13.631	-5.052
C(feature)[T.French_horn]	8.56e-14	3.282	2.61e-14	1.000	-6.434	6.434
C(feature)[T.Great_Pyrenees]	-10.1906	4.238	-2.405	0.016	-18.497	-1.884
C(feature)[T.Norwich_terrier]	-10.1906	2.242	-4.545	0.000	-14.586	-5.795
C(feature)[T.Old_English_sheepdog]	-10.1906	2.085	-4.887	0.000	-14.278	-6.103
C(feature)[T.Persian_cat]	-10.1906	4.238	-2.405	0.016	-18.497	-1.884
C(feature)[T.Sealyham_terrier]	-10.1906	3.282	-3.105	0.002	-16.625	-3.756
C(feature)[T.Shetland_sheepdog]	-10.1906	3.282	-3.105	0.002	-16.625	-3.756
C(feature)[T.Shih-Tzu]	-10.1906	3.282	-3.105	0.002	-16.625	-3.756
C(feature)[T.Sussex_spaniel]	-10.1906	3.282	-3.105	0.002	-16.625	-3.756
C(feature)[T.Tibetan_mastiff]	-6.991e-13	4.238	-1.65e-13	1.000	-8.306	8.306



OLS Regression Results

Dep. Variable:	views	**R-squared:**	0.278
Model:	OLS	**Adj. R-squared:**	0.268
Method:	Least Squares	**F-statistic:**	27.68
Date:	Mon, 10 Sep 2018	**Prob (F-statistic):**	0.00
Time:	22:49:47	**Log-Likelihood:**	-28237.
No. Observations:	10144	**AIC:**	5.675e+04
Df Residuals:	10004	**BIC:**	5.777e+04
Df Model:	139		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	12.3265	2.787	4.422	0.000	6.863	17.790
C(feature)[T.French_horn]	10.1906	3.942	2.585	0.010	2.464	17.917
C(feature)[T.Indian_elephant]	6.679e-13	4.828	1.38e-13	1.000	-9.463	9.463
C(feature)[T.Windsor_tie]	-3.233e-13	4.828	-6.7e-14	1.000	-9.463	9.463
C(feature)[T.abacus]	-1.865e-14	3.030	-6.16e-15	1.000	-5.940	5.940
C(feature)[T.abaya]	1.5788	2.826	0.559	0.576	-3.961	7.119
C(feature)[T.academic_gown]	4.2908	2.930	1.464	0.143	-1.453	10.035
C(feature)[T.altar]	1.6305	2.810	0.580	0.562	-3.877	7.138
C(feature)[T.amphibian]	4.37e-13	4.828	9.05e-14	1.000	-9.463	9.463
C(feature)[T.balance_beam]	1.069e-13	2.967	3.6e-14	1.000	-5.817	5.817
C(feature)[T.balloon]	7.6429	3.414	2.239	0.025	0.951	14.335
C(feature)[T.banjo]	10.1906	4.828	2.111	0.035	0.727	19.654
C(feature)[T.barbershop]	8.233e-13	4.828	1.71e-13	1.000	-9.463	9.463
C(feature)[T.basketball]	2.878e-13	2.882	9.99e-14	1.000	-5.649	5.649
C(feature)[T.bassinet]	5.0953	3.942	1.293	0.196	-2.632	12.822
C(feature)[T.bathing_cap]	3.055e-13	3.598	8.49e-14	1.000	-7.054	7.054
C(feature)[T.bearskin]	-5.329e-14	3.082	-1.73e-14	1.000	-6.040	6.040

We suggest the models to be repeated in a future study to try out much larger sample sizes and fine-tuned to fit our datasets. Then we can compare the goodness of fit for each linear model with a different sample size to determine whether there is a linear relationship between features and views.

Conclusions

Going into this project, we knew that, due to the extremely low sample size chosen, we would not make scalable analysis. The goal of this project was to 1) build a data pipeline for ingesting videos and creating features, 2) explore transfer learning in Keras with pre-trained models.

After the exploration, we have a big question mark on the veracity of ImageNet for using this set for object recognition, at least for our objects. When manually evaluating/eyeballing the results, we cannot accurately match many objects to what we perceived as the story of the videos. This was supported by the regression model's low adjusted R-squared value.

Fine-tune pre-trained models — it is very likely that we need to fine tune the pre-trained model for our dataset. According to the article by Gupta, there are three ways of fine-tuning we can consider: 1) Remove the output layer, which gives the probabilities for being in each of the 1000 classes of the model and then use the entire network as a fixed feature extractor for our dataset; 2) Use architecture of the model but randomly initialize all the weights to train the model on our dataset; 3) Partially train the model by freezing the weights of initial layers of the model and only re-train higher layers.

Limitation of the Study

Due to the small sample size and the exploration nature of the study, the authors attempt to create re-usable scripts and a pipeline for the future with a large sample size. Hence, the features presented in this study should not be considered as the final conclusion. The study only serves as a proof of concept and it should be repeated with a large-scale exploratory study to come up with a valid set of features that can be used as a training set for a supervised machine-learning study in the next phase.

Appendix A: References

Building an Image Classifier Using Pretrained Models With Keras
https://innolitics.com/articles/pretrained-models-with-keras/

Dog Breed Classification with Keras
http://machinememos.com/python/keras/artificial%20intelligence/machine%20learning/transfer%20learning/dog%20breed/neural%20networks/convolutional%20neural%20network/tensorflow/image%20classification/imagenet/2017/07/11/dog-breed-image-classification.html

ImageNet: VGGNet, ResNet, Inception, and Xception with Keras
https://www.pyimagesearch.com/2017/03/20/imagenet-vggnet-resnet-inception-xception-keras/

Keras Applications: Available Models
https://keras.io/applications/

Keras Tutorial : Transfer Learning using pre-trained models
https://www.learnopencv.com/keras-tutorial-transfer-learning-using-pre-trained-models/

Pretrained Convolutional Neural Networks
https://www.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html

Transfer learning & The art of using Pre-trained Models in Deep Learning
https://www.analyticsvidhya.com/blog/2017/06/transfer-learning-the-art-of-fine-tuning-a-pre-trained-model/

Using a pre-trained convnet
https://github.com/fchollet/deep-learning-with-python-notebooks/blob/master/5.3-using-a-pretrained-convnet.ipynb

Using Keras' Pre-trained Models for Feature Extraction in Image Clustering
https://medium.com/@franky07724_57962/using-keras-pre-trained-models-for-feature-extraction-in-image-clustering-a142c6cdf5b1

Appendix B: Program and Output Files

The attached program files are:

1. video to frames.ipynb
2. Video_Frames_Classification_Inceptionv3-Boss-rev.ipynb
3. Video_Frames_Classification_Inceptionv3-MeToo-rev.ipynb
4. Video_Frames_Classification_Inceptionv3-Radioactive-rev.ipynb
5. Video_Frames_Classification_ResNet50-Boss-rev.ipynb
6. Video_Frames_Classification_ResNet50-MeToo-rev.ipynb
7. Video_Frames_Classification_ResNet50-Radioactive-rev.ipynb
8. Evaluate_feature_predictions.ipynb
9. Classification_Output_Analysis.ipynb
10. Classification_Output_Analysis_Regression-rev.ipynb
11. Classification_Output_Analysis_WordClouds-rev.ipynb

The attached output files are:

- wordcloud images
    1. Boss_Inceptionv3.png
    2. Boss_ResNet50.png
    3. Me_Too_Inceptionv3.png
    4. Me_Too_ResNet50.png
    5. Radioactive_Inceptionv3.png
    6. Radioactive_ResNet50.png
- csv files
    1. For Inception v3 model
        1. boss_raw.csv
        2. boss_csv
        3. me_too_raw.csv
        4. me_too.csv
        5. radioactive_raw.csv
        6. radioactive.csv
        7. reatures_3videos_inception_raw.csv
        8. features_3videos_inception.csv
    2. For ResNet50 model
        1. boss0_raw.csv
        2. boss0_csv
        3. me0_too_raw.csv
        4. me0_too.csv
        5. radioactive0_raw.csv
        6. Radioactive0.csv
        7. features_3videos_resnet50_raw.csv
        8. features_3videos_resnet50.csv

Appendix C: Group Member Tasks & Roles

A breakdown of roles and responsibilities is found below. However, most components were developed in close collaboration with our teammates.

- Domain Knowledge — Ian Aliman
- Data Collection — Ian Aliman
- Feature Frames Extraction — Ian Aliman
- YouTube Tags Extraction and Analysis — Ian Aliman
- Keras Pre-trained Models Coding — Daphne Chang
- Features Comparison, Regression Analysis, and Wordclouds — Daphne Chang
- Pipeline Diagram — Ian Aliman
- Paper Write-up — Ian Aliman, Daphne Chang