Univ. Al

Final Project: Real-time facial expression prediction

Arnab Chattopadhyay, Kunal Vartia, Siddharth D P, Neeraj K Udupa

Motivation and About the Project

- The field of Computer Vision is one of the most popular fields in AI today
- One of it's key highlights is that it allows us to automatically extract knowledge from complex raw data structures such as images.
- The goal of this project is building a face expression recognition system with Deep Learning
- We built a custom Deep CNN, as well as experimented with transfer learning.
- We trained the model from scratch using our facial expression dataset
- Finally, we fed real time images to the model to see if it could classify the expressions successfully.

Data and Labels

- We used a dataset from a Kaggle challenge. (Link below)
- The data consists of 48x48 pixel grayscale images of faces.
- Each image corresponds to a facial expression in one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).
- The dataset contains approximately 36,000 images.

References

- https://www.kaggle.com/c/challengesin-representation-learning-facialexpression-recognition-challenge
- https://arxiv.org/pdf/1409.1556
- https://docs.opencv.org/master/
- https://keras.io/

Model

	yer (type) Outp	out Shape P	aram #			
				*****		*******
	out_1 (inputLayer)	(None, 48, 48, 3)]	0			
	ock1_conv1 (Conv2D)	(None, 48, 48, 6	4)	1792		
	ock1_conv2 (Conv2D)	(None, 48, 48, 6	4)	36928		
	ock1_pool (MaxPooling	2D) (None, 24, 24	,64)	0		
	ack2_conv1 (Conv2D)	(None, 24, 24, 1	28)	73856	U.	
	ock2_conv2 (Conv2D)	(None, 24, 24, 1	28)	147584		
	ock2_pool (MaxPooling	2D) (None, 12, 12	, 128)	0		
Blocks_com/s (Com/2D) (None, 12, 12, 756) 500080	ock3_conv1 (Conv2D)	(None, 12, 12, 2	56)	295168		
Slock3_pool (Markhoding2D) (Moon, 6, 6, 256) 0	ock3_conv2 (Conv2D)	(None, 12, 12, 2	56)	590080	8 II	
Block4_com4 (Com210) Ploons, 6, 6, 512) 1180100	ock3_conv3 (Conv2D)	(None, 12, 12, 2	56)	590080		
Block4_comd (Com20) Plome, 6, 6, 512] 2359868	ock3_pool (MaxPooling	2D) (None, 6, 6, 2	56)	0		
Blocks_com3 (Com/2D) Blome, 6, 6, 512) 2359958	ock4_conv1 (Conv2D)	(None, 6, 6, 512) 1	180160		
Slocks_pool (March/orling20) (Picone, 3, 3, 522) 0	ock4_conv2 (Conv2D)	(None, 6, 6, 512) 2	359808		
Blocks_cond_(Com/20) Plome, 3, 3, 512] 2359908	ock4_conv3 (Conv2D)	(None, 6, 6, 512) 2	359808		
BookS_comd (Conv210) (Nover, 3, 3, 512) 2359868 BookS_comd (Conv210) (Nover, 5, 3, 512) 2359868 BookS_comd (Conv210) (Nover, 5, 3, 512) 2359868 BookS_pool (Makerhoding20) (Nover, 1, 1, 512) 0 BookS_pool (Makerhoding20) (Nover, 1, 1, 512) 0 BookS_pool (Makerhoding20) (Nover, 1, 1, 512) 0 BookS_pool (Makerhoding20) (Nover, 256) 111328 BookS_pool (Decon) (Nover, 256) 1524 BookS_pool (Decon, 256) 131344 BookS_pool (Decon, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,	ock4_pool (MaxPooling	2D) (None, 3, 3, 5	12)	0		
	ock5_conv1 (Conv2D)	(None, 3, 3, 512) 2	359808	-	
	ock5_conv2 (Conv2D)	(None, 3, 3, 512) 2	359808		1 2
global_morrage_pooling2d (SI (None, 132)	ock5_conv3 (Conv2D)	(None, 3, 3, 512) 2	359808		
Senier (Denier) (Hone, 256) 131328 Janth, Jonnalization (Batchilor (Breun, 256) 1024 dropout (Diregount) (None, 256) 1024 dropout (Diregount) (None, 256) 0 Johns, 1 (Denier) (Denier, 132) 131584 bestch, normalization, 1 (Batch (Bions, 512) 2068 dropout, 1 (Denier) (Hone, 512) 0 dropout, 1 (Denier) (Hone, 513) 0 dropout, 1 (Denier) (Hone, 514) 0 dropout, 2 (Denier) (Hone, 514) 0 dropout, 3 (Hone, 514) 0 dropout, 4 (Hone, 514) 0 dropout, 5 (Hon	ock5_pool (MaxPooling	2D) (None, 1, 1, 5	12)	0		
batch_normalization (BatchNo (Noon, 256) 1024 dropout (Dropout) (Noon, 256) 0 denies_(Denies) (Noon, 256) 113584 batch_normalization_1 (Batch (Noon, 512) 2048 dropout_(Dropout) (Noon, 512) 0 dropout_(Dropout) (Noon, 512) 0 flooring_normalization_1 (Batch (Noon, 512) 3591 flooring_normalization_1 (Noon, 7) 3591	obal_average_pooling2	d (GI (None, 512)		0		
dropout (Propout) (None, 256) 0	ense (Dense) (Ne	one, 256) 1	31328			
Sense_1 (Dense) (Noon, 512) 131584 batch_normalization_1 (Batch (Moon, 512) 2048 dropout_1 (Dropout) (Moon, 512) 0 dropout_1 (Dropout) (Moon, 512) 0 floating_1000000000000000000000000000000000000	tch_normalization (Bat	chNo (None, 256)		1024		
Datch, normalization, 1 (Batch (Mone, 512) 2048 dropout, 1 (Dropout) (Mone, 512) 0 dense, 2 (Dense) (Mone, 7) 3591 Total params: 14,984,263	opout (Dropout) (None, 256)	0			
dropout 1 (Dropout) (None, 512) 0 dense, 2 (Dense) (None, 7) 3591 Total params: 14,984,263	nse_1 (Dense) (f	Vane, 512)	13158	4		
dense_2 (Dense) (None, 7) 3591 Total params: 14,984,263	tch_normalization_1 (8	latch (None, 512)		2048		
Total params: 14,984,263	opout_1 (Dropout)	(None, 512)	0			
	nse_2 (Dense) (f	Vone, 7) 3	591			
Trainable params: 13,247,239		220				
Von-trainable params: 1,737,024						

Results











The model was successfully able to detect the facial expression of the live-fed images.

Conclusion and Future Work In conclusion:

- We used transfer learning, using the VGG16 model, and added some of our own custom layers on top to suit our problem.
- There were 467 angry instances in test set. We could classify 214 angry items correctly. On the other hand, the model classified 9 items as disgust but these items are actual angry ones.
- Clearly, accuracy should not express right impression for multi class classification problems.
- To help in interpreting the results better, we have included the confusion matrix of the model in our presentation.

Future Work:

- For this project, we trained all the models from scratch using CNN packages in Keras.
- In the future, we would like to implement a Facial Action Coding System (FACS) for determining facial expressions.
- This posits that expressions result from the change of facial parts and each muscular movement is tracked.
- For this, we could investigate the efficacy of pretrained SOTA such as the GoogLeNet