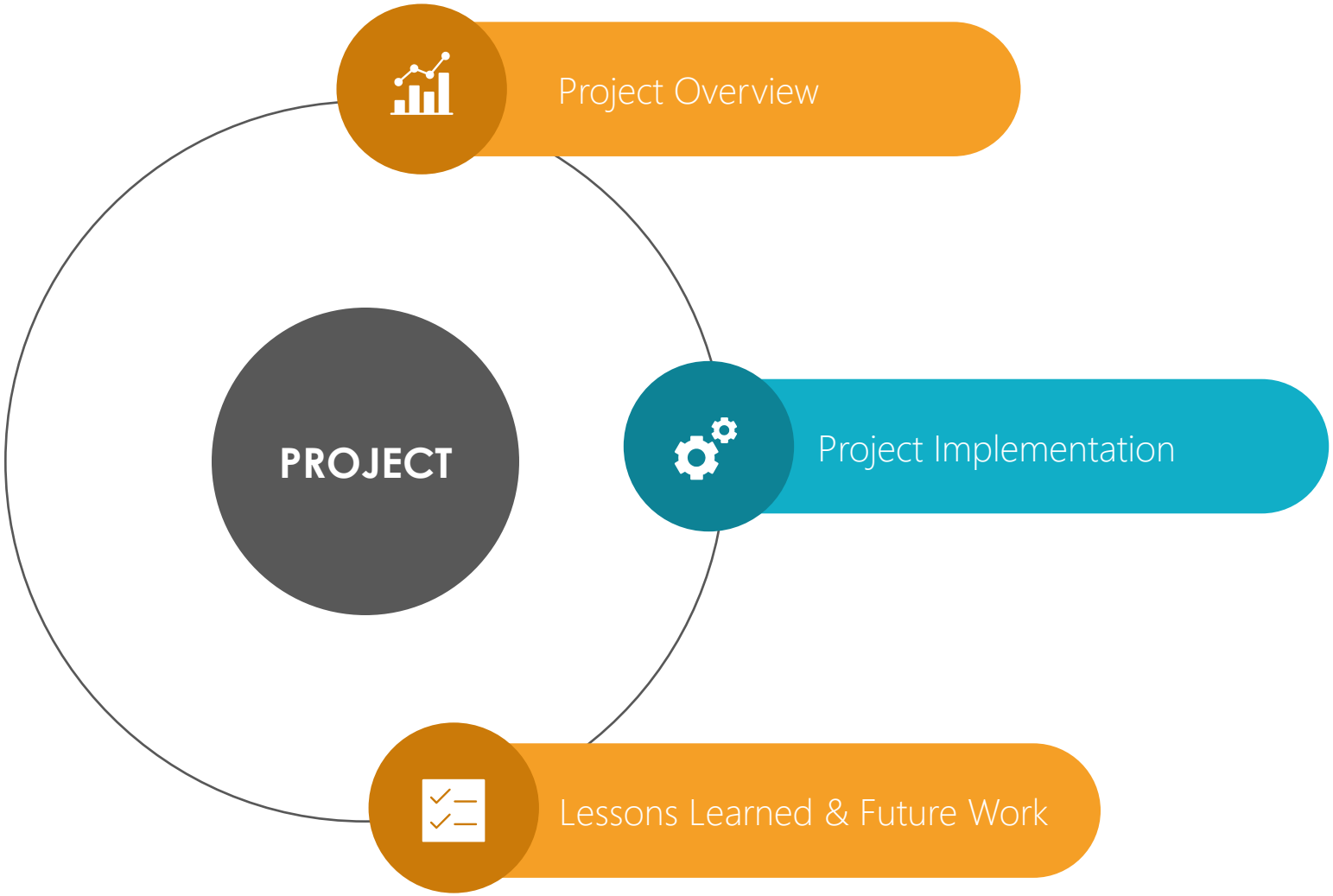
Distracted Driver Detection with Convolution Neural Network

Name: Liu, Shijie
Professor: Liu, Xinlian
Date: 11/19/2020

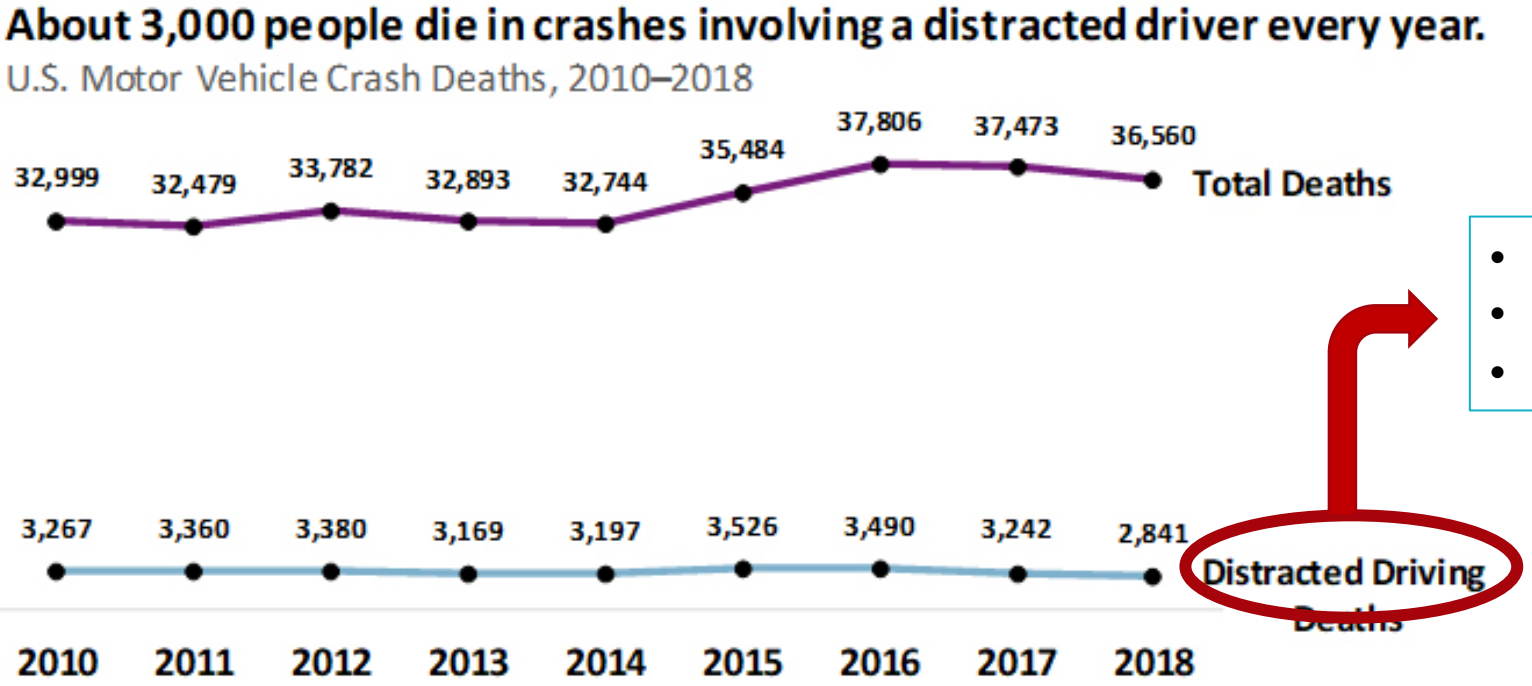
Computer Science Department
Hood College of Frederick Maryland

Project Outline



Project Overview

Background



- Visual Distraction: Eyes off the road
- Manual Distraction: Hands off the wheel
- Cognitive Distraction: Minds off the driving

Objective

A Convolution Neural Network that can detect distracted driver from a 2D dash-cam image.

Implementation

Dataset

- Dataset Collection
- Dataset Exploration
- Dataset Spilt
- Dataset pre-processing

Architectures

- Convolution Neural Network from Scratch
- Transfer learning with VGG-19 via fine-tuning

Techniques

- Image Data Generator
- Transfer Learning
- Image Data Augmentation
- Dropout

Evaluation

- Loss
- Accuracy
- Confusion Matrix

Data Description

Source

State Farm Distracted Driver Detection dataset on Kaggle

https://www.kaggle.com/c/state-farm-distracted-driver-detection/data

Description

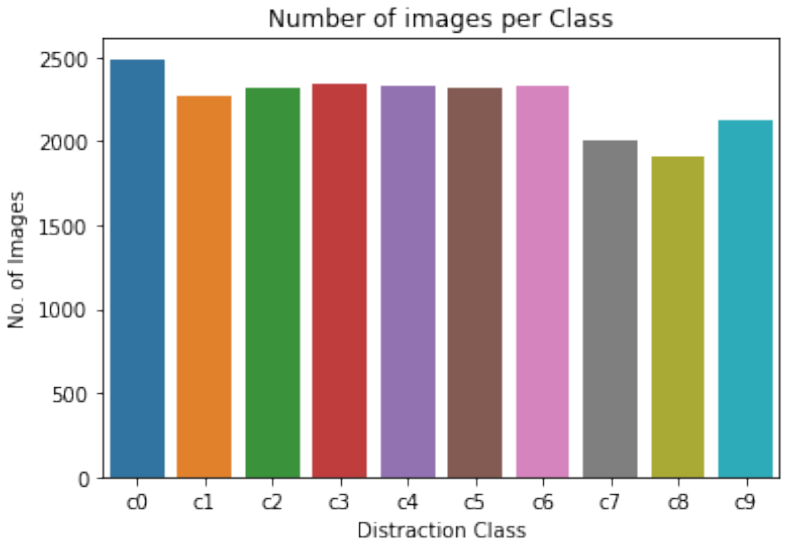
```
├── imgs
│   ├── train                          → Labeled 22,424 files with the resolution of 480x640
│   │       ├── c0                     → Class id, see the class id mapping on the right
│   │       │   ├── img_103.jpg
│   │       │   ├── .....
│   │       │   └── img_352.jpg
│   │       ├── c1
│   │       │   ├── img_421.jpg
│   │       │   ├── .....
│   │       │   └── img_782.jpg
│   │       ├── c2
│   │       ├── .....
│   │       └── c9
│   └── test                           → Unlabeled 79,726 files with the resolution of 480x640
│           ├── img_1028.jpg
│           ├── img_1029.jpg
│           └── .....
├── driver_imgs_list.csv               → Captures filename, subject id, class id
└── sample_submission.csv              → Sample format for Kaggle submission, NOT APPLICABE
```

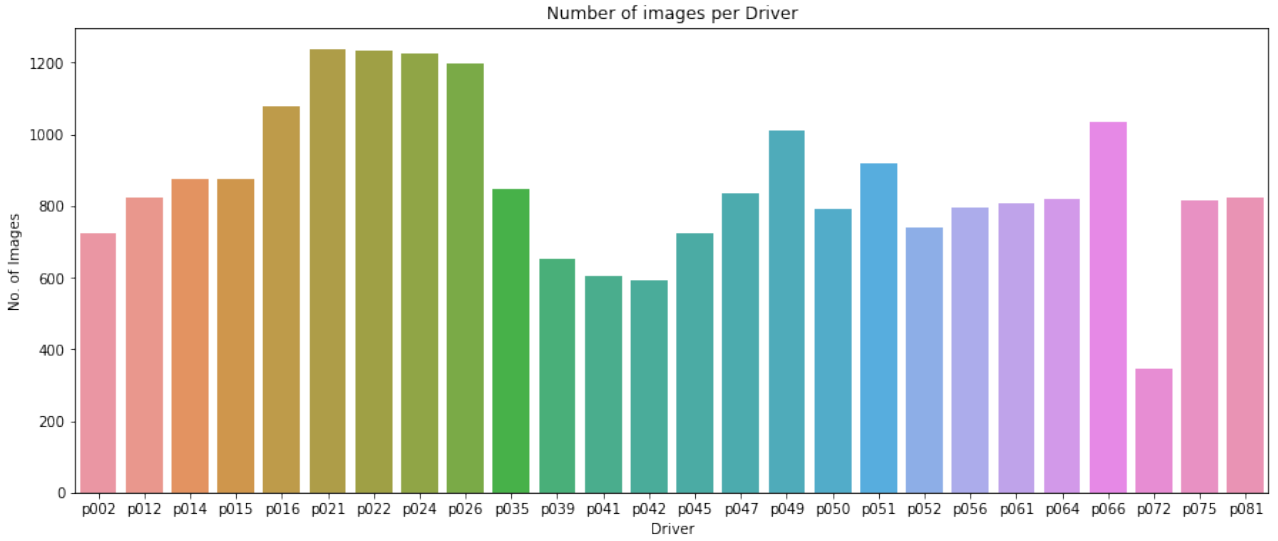
Distraction Classes:

- c0: safe driving
- c1: texting - right
- c2: talking on the phone - right
- c3: texting - left
- c4: talking on the phone - left
- c5: operating the radio
- c6: drinking
- c7: reaching behind
- c8: hair and makeup
- c9: talking to passenger

Data Exploration

Sample image per distraction class





Data Pre-Processing

Image Augmentation

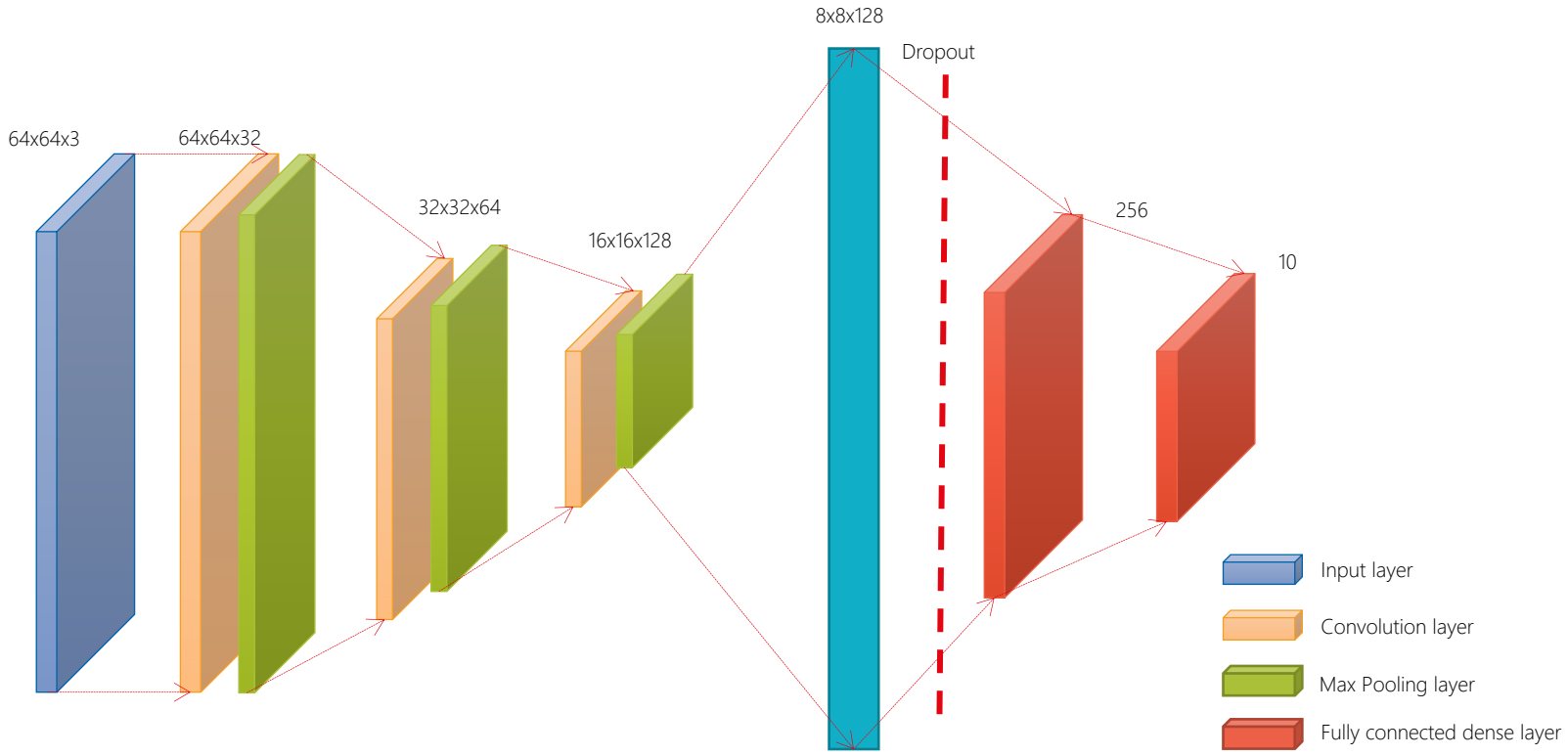
- Generates more data from existing data
- Exposes more aspects of same data
- Supports random on-the-fly transformation

Image Resizing

- Reduces memory load by reducing the image size
- Allows the batch size to be increased to speed up model training time
- Keeps the original aspect ratio and preserves the feature information

Architecture

Convolution Neural Network from Scratch

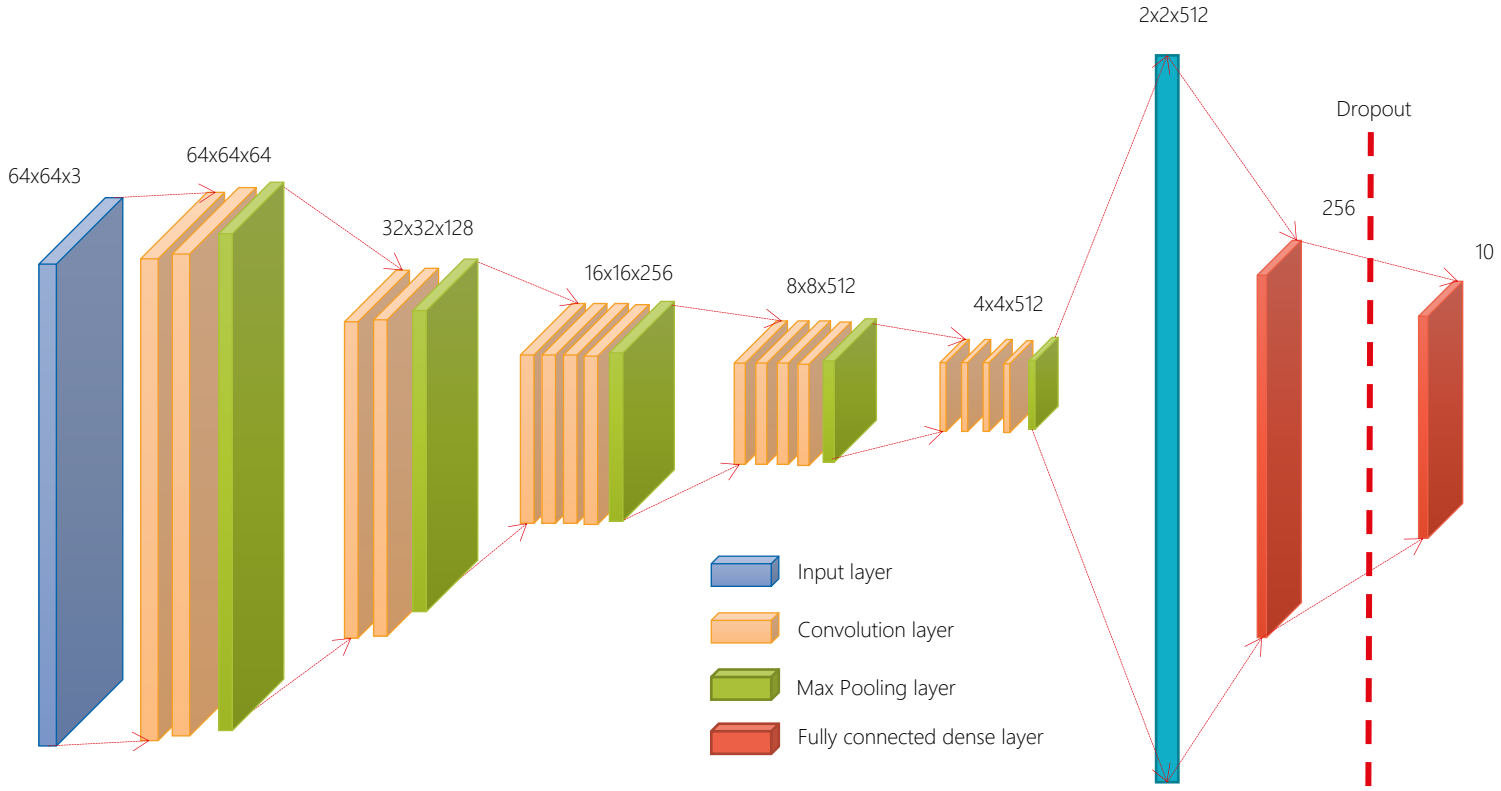


- Three common layers are used, convolution, pooling, and fully connected
- Input layers accepts 64x64x3 input (RGB images)
- Three convolution layers, with various number of filters. E.g. 32, 64, and 128; All convolution layers use *relu* activation function and 5x5 stride.
- Each convolution layer has a following max pooling layer, which uses 2x2 pooling window.
- Two fully connected dense layers are following after flattening out, the last layer has 10 as number of output, which corresponds to number of classes to classify.

Architecture

Fine-tuned VGG-19 Convolution Neural Network

- VGG network architecture includes two variants, VGG-16 and VGG-19.
- Original VGG-19 takes 224x224x3 input shape
- Includes 16 convolution layers(3x3 stride), 5 max pooling layers(2x2 pooling window), and 3 fully connected layers(first two layers have 4096 outputs, while the last has 1000 outputs)



Fine-tuning:

- Replace three fully connected dense layers with two, which use 256 and 10 as output
- Keep convolution and pooling layers as convnet base, but later layers will be kept trainable.

Techniques

ImageDataGenerator

- Supports loading data via batches, which overcomes the challenge with limited memory.
- Offers an easy way to load dataset, categorizing classes and train/validation spilt.
- Supports image pre-processing, such as rescaling, resizing, and image augmentation.

```
train_datagen = ImageDataGenerator(
    rescale=1.0 / 255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    brightness_range=[0.4, 1.2],
    shear_range=0.2,
    zoom_range=0.3,
    validation_split=0.2)
```

```
train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=(INPUT_HEIGHT, INPUT_WIDTH),
    classes=None,
    class_mode="categorical",
    batch_size=BATCH_SIZE,
    shuffle=True,
    subset="training")
```

Transfer Learning

- A common and highly effective approach for deep learning with small image dataset.
- Pre-trained models were trained with large scale dataset that local features for generic objects have been learned. E.g. ImageNet, 1000 classes trained from 1.4 million labeled images.
- Offers two approaches, feature extraction(Add new classifier on top of convolutional base) and fine-tuning(Replace fully connected layers and optionally fine-tune all or part of convolutional base via backpropagation).

```
# load the convnet base
conv_base = VGG19(
             weights="imagenet",
             include_top=False,
             input_shape=(INPUT_HEIGHT, INPUT_WIDTH, 3))

x = conv_base.output
# add flatten, dropout, dense, etc
predictions=layers.Dense(10, activation="softmax")(x)

model=models.Model(inputs=conv_base.input,
                   outputs=predictions)

# freeze the training
for layer in model.layers[:10]:
    layer.trainable = False

model.compile(…)
model.fit(…)
```

Dropout

- One of the most effective and most commonly used regularization techniques for neural networks, which is an efficient way of reducing overfitting by randomly drooping out some neurons during training phase.

```
model.add(layers.Dropout(0.3))
```

Model Evaluation

- D: Dropout
- BN: Batch Normalization
- DA: Data Augmentation
- LR: Learning Rate
- LF: Layers frozen

Convolution Neural Network from Scratch

	Training		Validation		Test	
	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy
Base	0.022	99.33%	0.085	97.83%	2.69	66.00%
Base + D(0.3)	0.009	99.72%	0.022	99.46%	2.85	62.99%
Base + D(0.3) + BN	0.051	99.81%	0.041	99.62%	0.903	75.00%
Base + D(0.3) + BN + DA	0.207	94.98%	0.422	87.03%	0.664	81.00%
Base + D(0.2) + BN + DA	0.148	96.50%	0.221	94.11%	0.858	75.00%
Base + D(0.4) + BN + DA	0.343	91.52%	0.487	85.69%	0.762	74.00%
Base + D(0.3) + BN + DA + LR(0.01)	0.087	97.55%	0.074	97.88%	0.683	84.00%
Base + D(0.3) + BN + DA + LR(0.001)	0.199	95.12%	0.32	90.02%	0.804	75.00%
Base + D(0.3) + BN + DA + LR(0.0001)	0.59	88.65%	0.55	89.49%	0.96	70.00%
Base + BN + DA + LR(0.01)	0.0626	98.32%	0.078	97.63%	0.792	78.00%

- Experimented dropout, batch normalization, data augmentation, and learning rate.
- 0.3 seems to be the best choice for dropout rate.
- 0.01 seems to be the best choice for learning rate.

Fine-tuned VGG-19 Convolution Neural Network

- Experimented dropout, data augmentation, learning rate, and number of layers for re-training.
- 0.3 dropout rate was kept for consistency.
- 0.0001 seems to be the best choice for learning rate.
- More layers to be retrained leads to better result.

	Training		Validation		Test	
	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy
Base	0.013	99.88%	0.083	97.46%	2.42	54.00%
Base + D(0.3)	0.092	97.11%	0.071	98.12%	2.11	46.99%
Base + D(0.3) + DA	1.23	56.95%	1.072	63.59%	1.536	46.00%
Base + D(0.2) + DA	1.186	58.52%	1.073	63.39%	1.735	39.00%
Base + D(0.4) + DA	1.378	51.39%	1.207	59.29%	1.546	48.00%
Base + D(0.3) + DA + LR(0.01)	1.807	33.25%	1.745	35.07%	1.678	30.00%
Base + D(0.3) + DA + LR(0.001)	1.303	53.69%	1.13	61.92%	1.66	44.00%
Base + D(0.3) + DA + LR(0.0001)	1.613	43.54%	1.514	47.92%	1.467	47.00%
Base + D(0.3) + DA + LR(0.0001) + LF(15)	0.123	96.03%	0.157	95.02%	0.635	82.00%
Base + D(0.3) + DA + LR(0.00001) + LF(15)	0.177	94.33%	0.207	93.06%	0.9843	70.00%
Base + D(0.3) + DA + LR(0.0001) + LF(10)	0.092	97.32%	0.115	96.43%	0.623	87.00%

Model Evaluation

- D: Dropout
- BN: Batch Normalization
- DA: Data Augmentation
- LR: Learning Rate
- LF: Layers frozen

Convolution Neural Network from Scratch

	Training		Validation		Test	
	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy
Base	0.022	99.33%	0.085	97.83%	2.69	66.00%
Base + D(0.3)	0.009	99.72%	0.022	99.46%	2.85	62.99%
Base + D(0.3) + BN	0.051	99.81%	0.041	99.62%	0.903	75.00%
Base + D(0.3) + BN + DA	0.207	94.98%	0.422	87.03%	0.664	81.00%
Base + D(0.2) + BN + DA	0.148	96.50%	0.221	94.11%	0.858	75.00%
Base + D(0.4) + BN + DA	0.343	91.52%	0.487	85.69%	0.762	74.00%
Base + D(0.3) + BN + DA + LR(0.01)	0.087	97.55%	0.074	97.88%	0.683	84.00%
Base + D(0.3) + BN + DA + LR(0.001)	0.199	95.12%	0.32	90.02%	0.804	75.00%
Base + D(0.3) + BN + DA + LR(0.0001)	0.59	88.65%	0.55	89.49%	0.96	70.00%
Base + BN + DA + LR(0.01)	0.0626	98.32%	0.078	97.63%	0.792	78.00%

- Experimented dropout, batch normalization, data augmentation, and learning rate.
- 0.3 seems to be the best choice for dropout rate.
- 0.01 seems to be the best choice for learning rate.

Fine-tuned VGG-19 Convolution Neural Network

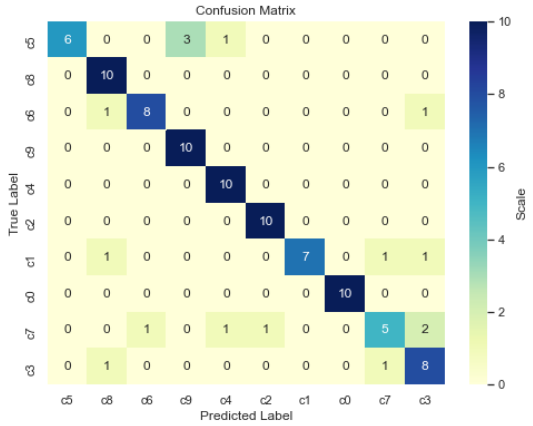
- Experimented dropout, data augmentation, learning rate, and number of layers for re-training.
- 0.3 dropout rate was kept for consistency.
- 0.0001 seems to be the best choice for learning rate.
- More layers to be retrained leads to better result.

	Training		Validation		Test	
	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy
Base	0.013	99.88%	0.083	97.46%	2.42	54.00%
Base + D(0.3)	0.092	97.11%	0.071	98.12%	2.11	46.99%
Base + D(0.3) + DA	1.23	56.95%	1.072	63.59%	1.536	46.00%
Base + D(0.2) + DA	1.186	58.52%	1.073	63.39%	1.735	39.00%
Base + D(0.4) + DA	1.378	51.39%	1.207	59.29%	1.546	48.00%
Base + D(0.3) + DA + LR(0.01)	1.807	33.25%	1.745	35.07%	1.678	30.00%
Base + D(0.3) + DA + LR(0.001)	1.303	53.69%	1.13	61.92%	1.66	44.00%
Base + D(0.3) + DA + LR(0.0001)	1.613	43.54%	1.514	47.92%	1.467	47.00%
Base + D(0.3) + DA + LR(0.0001) + LF(15)	0.123	96.03%	0.157	95.02%	0.635	82.00%
Base + D(0.3) + DA + LR(0.00001) + LF(15)	0.177	94.33%	0.207	93.06%	0.9843	70.00%
Base + D(0.3) + DA + LR(0.0001) + LF(10)	0.092	97.32%	0.115	96.43%	0.623	87.00%

Model Evaluation

Convolution Neural Network from Scratch

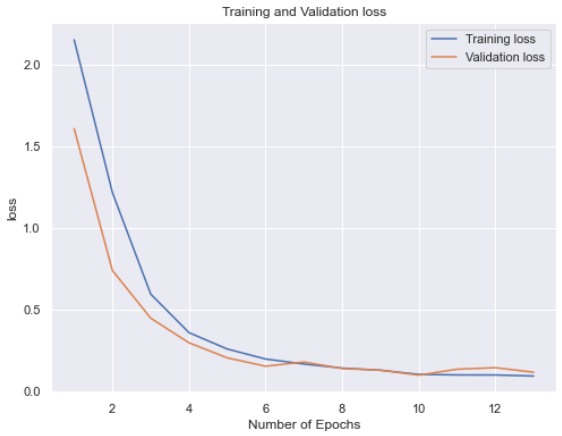
Dropout 0.3, Batch Normalization, Data Augmentation, Learning Rate 0.01

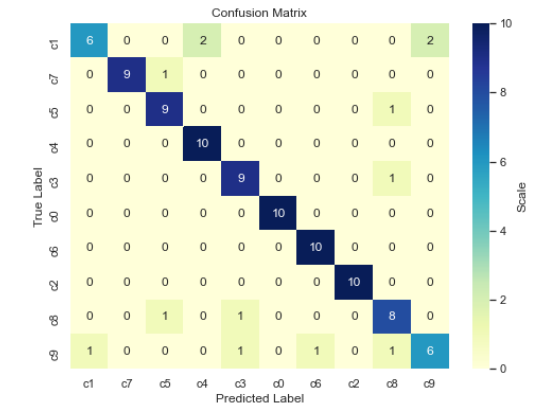


	precision	recall	f1-score	support
c5	1.00	0.60	0.75	10
c8	0.77	1.00	0.87	10
c6	0.89	0.80	0.84	10
c9	0.77	1.00	0.87	10
c4	0.83	1.00	0.91	10
c2	0.91	1.00	0.95	10
c1	1.00	0.70	0.82	10
c0	1.00	1.00	1.00	10
c7	0.71	0.50	0.59	10
c3	0.67	0.80	0.73	10
accuracy			0.84	100
macro avg	0.86	0.84	0.83	100
weighted avg	0.86	0.84	0.83	100

Fine-tuned VGG-19 Convolution Neural Network

Dropout 0.3, Data Augmentation, Learning Rate 0.0001, Frozen Layers: 10

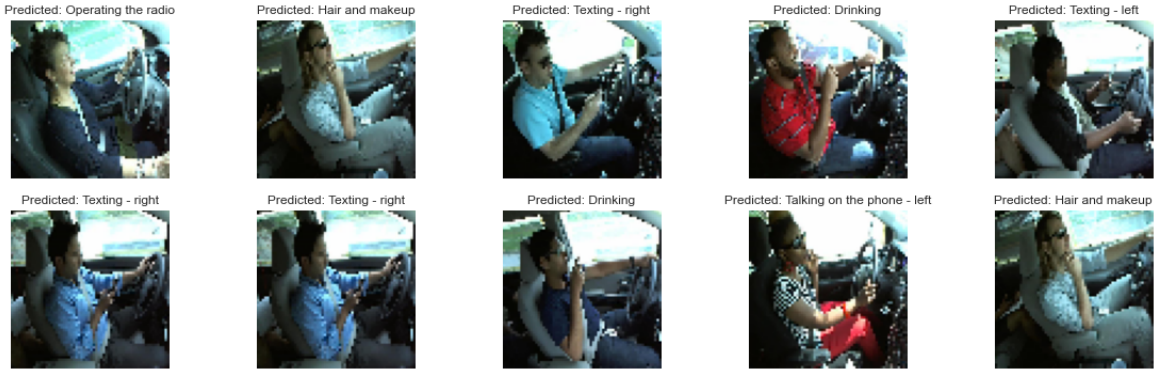




	precision	recall	f1-score	support
c1	0.86	0.60	0.71	10
c7	1.00	0.90	0.95	10
c5	0.82	0.90	0.86	10
c4	0.83	1.00	0.91	10
c3	0.82	0.90	0.86	10
c0	1.00	1.00	1.00	10
c6	0.91	1.00	0.95	10
c2	1.00	1.00	1.00	10
c8	0.73	0.80	0.76	10
c9	0.75	0.60	0.67	10
accuracy			0.87	100
macro avg	0.87	0.87	0.87	100
weighted avg	0.87	0.87	0.87	100

Conclusion & Lessons Learned

Do we have a Convolution Neural Network that can detect distracted driver?



- Convolution Neural Network is an efficient approach for image classification problem.
- Fine-tuned VGG-19 network outperforms the convolution neural network from scratch eventually with an accuracy of 87%.
- Convnet from scratch always beats VGG-19 if conv base is not re-trained. (training time is much less)
- Image Data Augmentation, Batch Normalization and Dropout can be used to minimize the impact of overfitting.

What I have learned?

- Mistakenly used horizontal flip for image augmentation with *ImageDataGenerator* leads to worse performance
- The *shuffle* needs to be set to False when generating confusion matrix while using *ImageDataGenerator*.
- Fine-tuning a pre-trained network without re-train the convnet base does not guarantee good result for image classification that has complex input.
- Dropout rate does not show liner impact on the network performance.
- Learning rate needs to be set accordingly by experiments, there is no BEST rate for all networks.

Future Work

Image Augmentation with Deep Learning Approach

- Adversarial training
- Neural style transfer
- GAN (Generative adversarial networks) data augmentation

Training Platform

- Bridges-2 system at Pittsburgh Supercomputing Center (PSC)

Model Deployment

- Integration with cameras for Real-time detection

Reference

[1] Danger – Distracted Driving! https://dekalbcountyonline.com/danger-distracted-driving/
[2] Centers Disease Control and Prevention (CDC) Transportation Safety. *Distracted Driving*. https://www.cdc.gov/transportationsafety/Distracted_Driving/index.html
[3] State Farm Distracted Driver Detection. https://www.kaggle.com/c/state-farm-distracted-driver-detection/data
[4] National Highway Traffic Safety Administration. (April 2020) Traffic Safety Facts Research Note: Distracted Driving 2018external icon. Department of Transportation, Washington, DC: NHTSA. Accessed 18 August 2020.
[5] "Deep Learning for Computer Vision." Deep Learning with Python, by Chollet François, Manning Publications Co., 2018.
[6] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. Journal of Big Data, 6(1). https://doi.org/10.1186/s40537-019-0197-0
[7] Image data preprocessing. Keras. https://keras.io/api/preprocessing/image/
[8] Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., Hazlewood, V., Lathrop, S., Lifka, D., Peterson, G.D., Roskies, R., Scott, J.R. and Wilkens-Diehr, N. 2014. XSEDE: Accelerating Scientific Discovery. Computing in Science & Engineering. 16(5):62-74. http://doi.ieeecomputersociety.org/10.1109/MCSE.2014.80.
[9] Nystrom, N. A., Levine, M. J., Roskies, R. Z., and Scott, J. R. 2015. Bridges: A Uniquely Flexible HPC Resource for New Communities and Data Analytics. In Proceedings of the 2015 Annual Conference on Extreme Science and Engineering Discovery Environment (St. Louis, MO, July 26-30, 2015).

Q & A

Thank You !