2020 embedded VISION summit

Practical Image Data Augmentation Methods for Training Deep Learning Object Detection Models

Evan Juras
EJ Technology Consultants

September 2020

EJ Technology Consultants

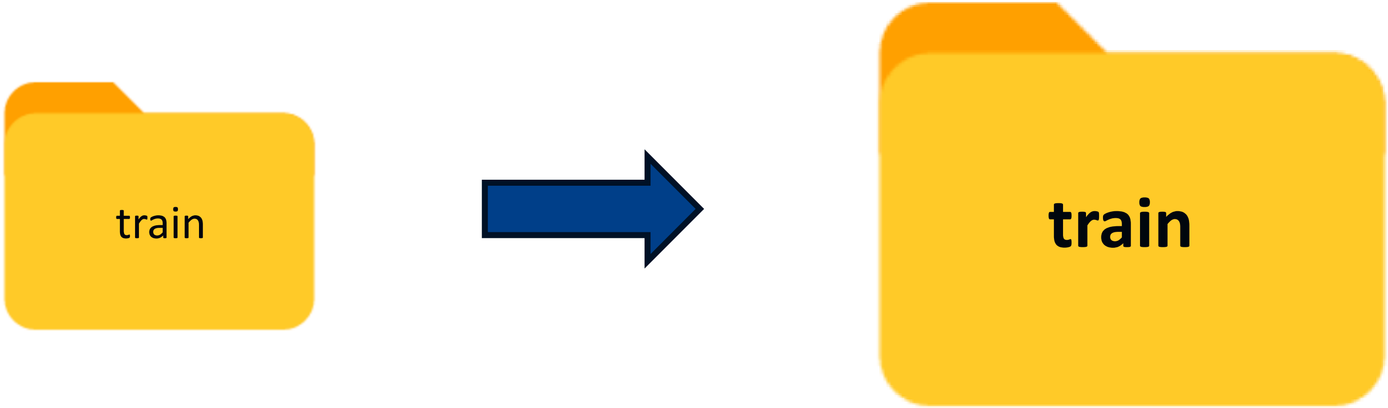
Presentation Goals

Introduce **practical** image data augmentation methods for increasing dataset size

- **Effective**
- **Low-effort**

Improve your model’s robustness and accuracy

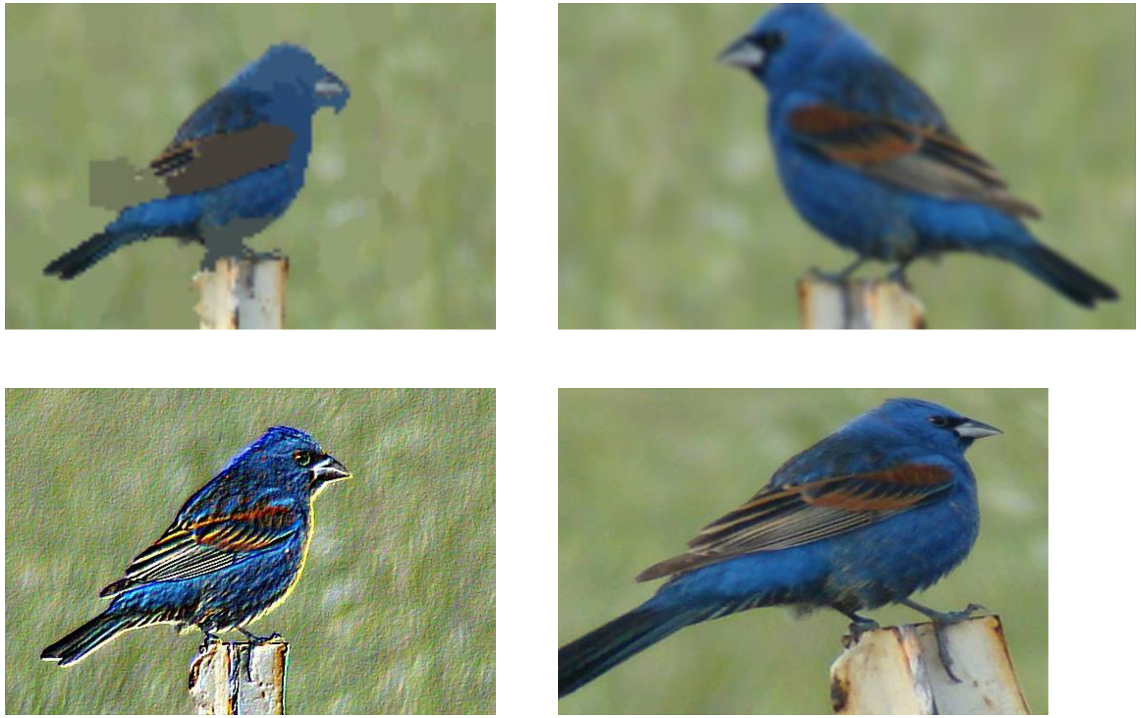
Discuss considerations for applying data augmentation



What is Data Augmentation?

Creating new training dataset images by modifying existing ones

Original image



Augmented images

Why use Data Augmentation?

Increase dataset size

- Prevents overfitting
- Increases robustness of features
- Increase variety of visual environments

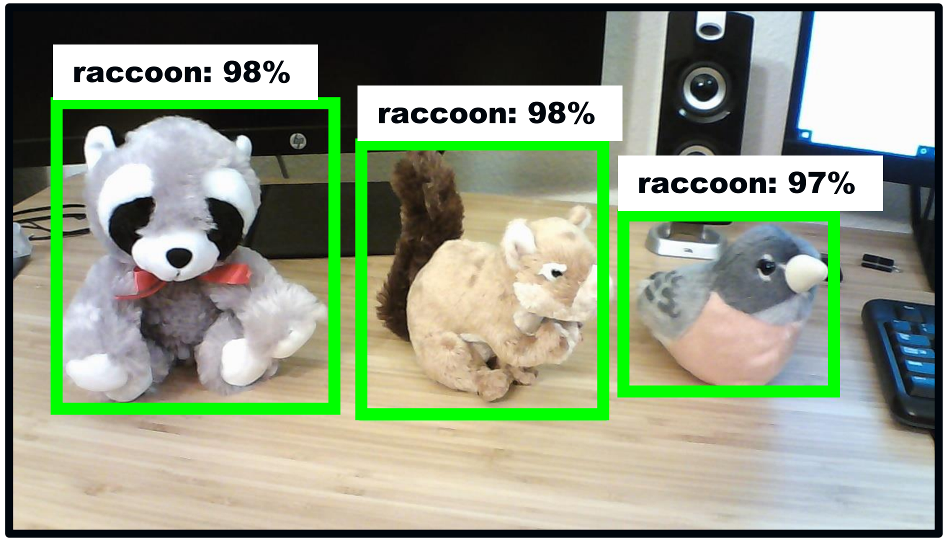
Resolve class imbalances in dataset

- Class imbalances causes model to bias towards majority class

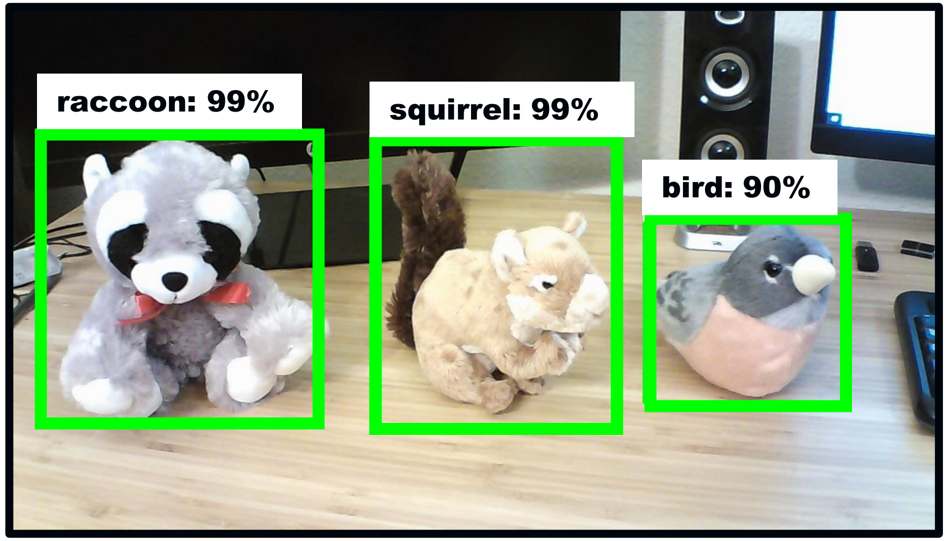
Save time over manually acquiring and labeling images

- 10 s to label one image * 50,000 images = 139 hours!

Model biased towards racoons

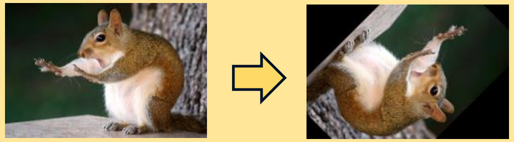


Unbiased model

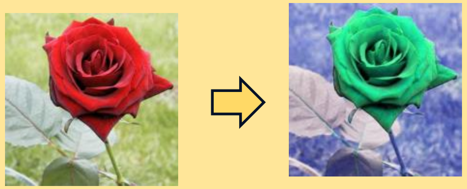


Methods for Data Augmentation

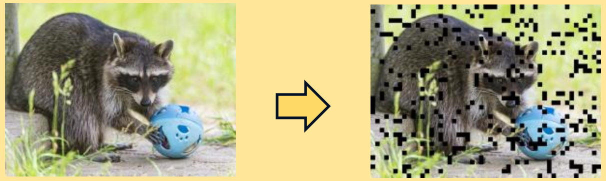
Basic image manipulation
(geometric, color space, kernel filters)



Rotation



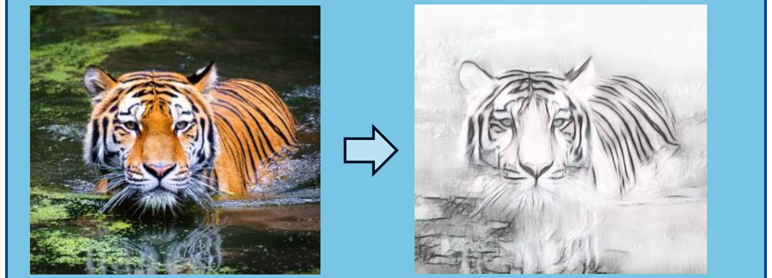
HSV shift



Dropout filter

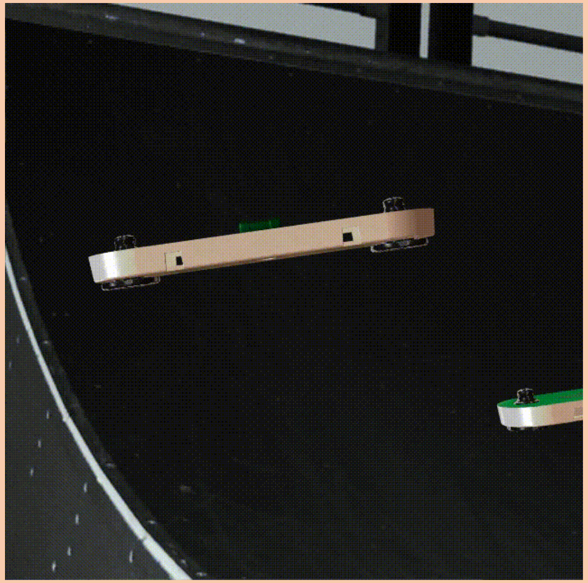
Neural networks
(GANs, Neural Style Transfer)

GAN examples



Neural Style Transfer

Synthetic image generation

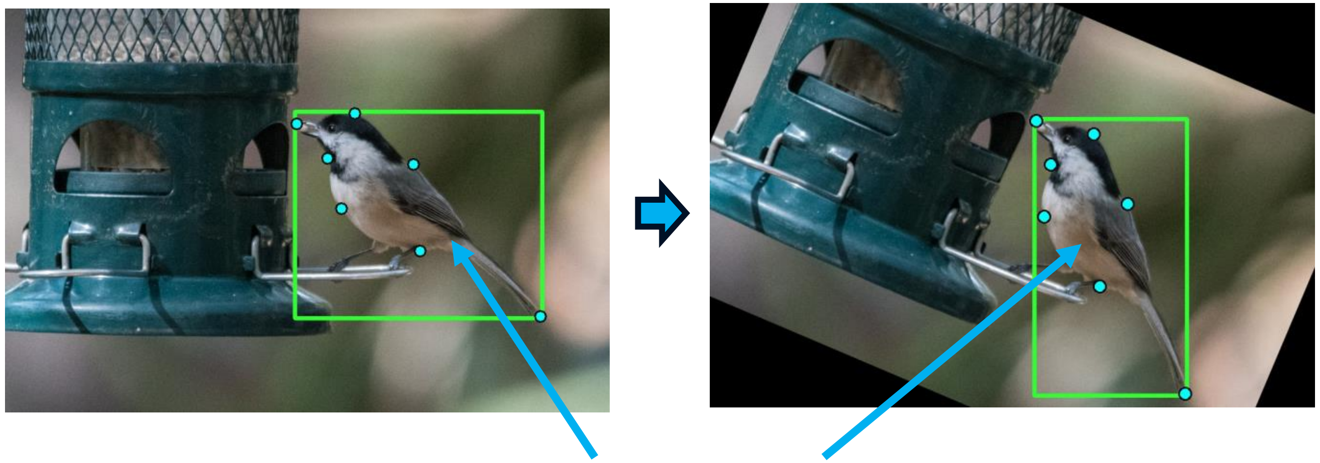


Animation courtesy of Synapse Product Development, Inc.

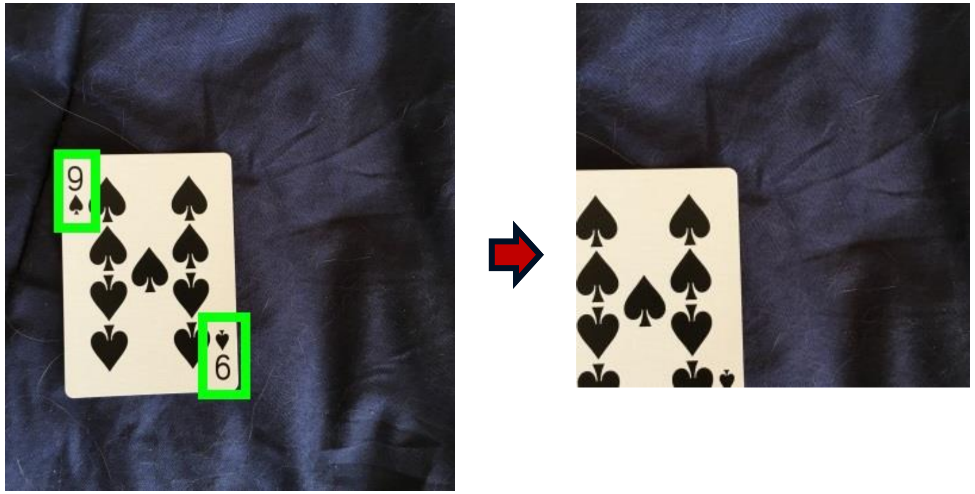
Data Augmentation Applied to Object Detection

Augmenting object detection datasets is **more challenging** than with image classification

- Not just transforming image, must also transform object keypoints and/or bounding box coordinates
- Need to make sure augmentation doesn't destroy label information



Keypoints and bounding boxes rotate with image



Destroying bounding boxes with cropping

Data Augmentation Toolsets

Open-source GitHub libraries: Imgaug, Augmentator, Albumentations

- Pros: Free, well-documented, good examples
- Cons: May become unavailable or stop being maintained

OpenCV has image manipulation functions for manually creating augmentations

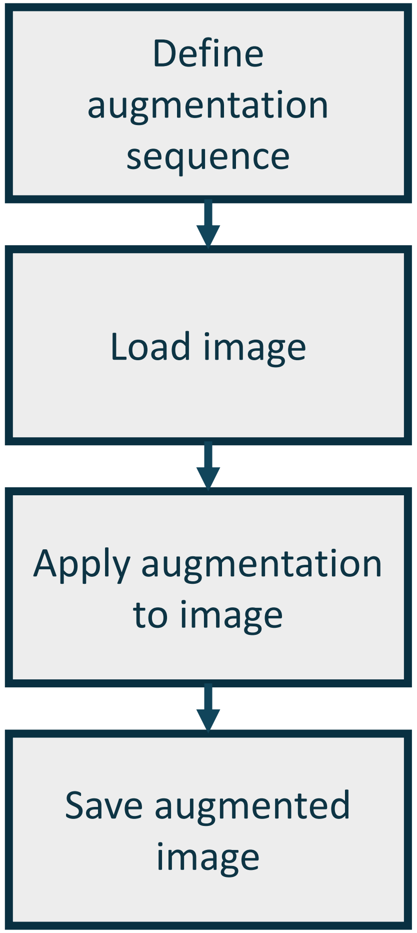
- Pros: Free, tight and flexible control over augmentation parameters
- Cons: Have to write everything yourself

TensorFlow Object Detection API provides augmentation options to automatically use during preprocessing

- Pros: Free, easy to use (zero-effort)
- Cons: Not many operations available, may stack poorly with existing augmentations in dataset

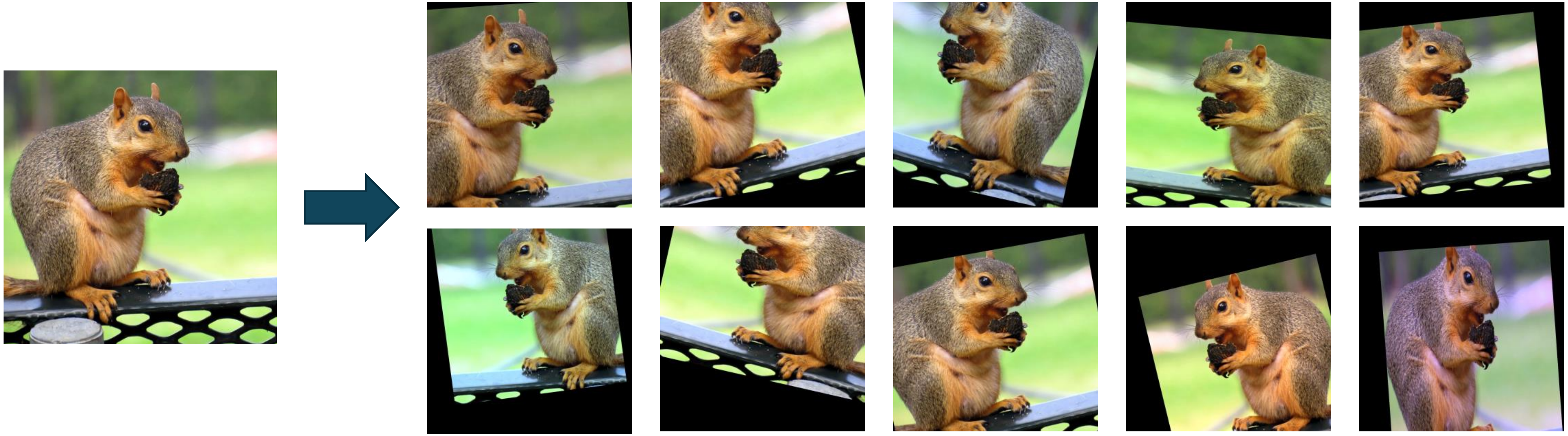
Code Example Using Imgaug

Basic augmentation process



```
import imgaug as ia
from imgaug import augmenters as iaa
import cv2

#---- Define augmentation ----#
seq1 = iaa.Sequential([
    iaa.Fliplr(0.5),                            # Horizontal flip 50% of images
    iaa.Crop(percent=(0, 0.10)),                # Crop all images between 0% to 10%
    iaa.GaussianBlur(sigma=(0, 0.5)),           # Add slight blur to images
    iaa.Multiply((0.8, 1.2), per_channel=0.2),  # Slightly brighten or darken images
    iaa.Affine(
        scale={"x": (0.8, 1.2), "y": (0.8,1.2)},               # Resize image
        translate_percent={"x": (-0.2, 0.2), "y": (-0.2, 0.2)}, # Translate image
        rotate=(-15, 15)                                        # Rotate image
        )
    ])

#---- Load image ----#
img1_bgr = cv2.imread('images/squirrel-780.jpg')  # Load image with OpenCV
img1 = cv2.cvtColor(img1_bgr, cv2.COLOR_BGR2RGB)    # Re-color to RGB from BGR

#---- Augment image ----#
img_aug1 = seq1(images=[img1])[0]   # Apply augmentation to image

#---- Save image ----#
img_aug_bgr1 = cv2.cvtColor(img_aug1, cv2.COLOR_RGB2BGR)    # Re-color to BGR from RGB
cv2.imwrite('outputs/basic1.jpg',img_aug_bgr1)              # Save image to disk
```

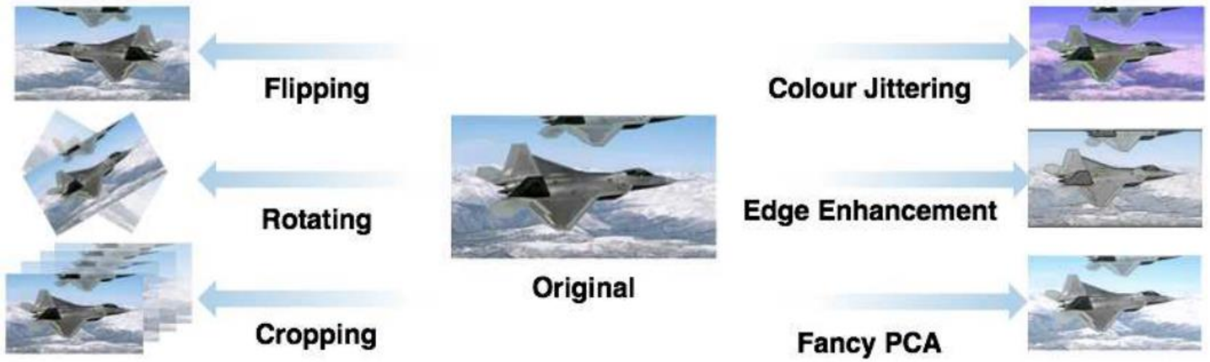
Basic Example Results



Note: Link to full code example including keypoint augmentation is available in Resources slide

Effectiveness of Basic Augmentations

L. Taylor and G. Nitschke [Reference 2] experiment on Caltech101 dataset shows accuracy improvement for various augmentation methods

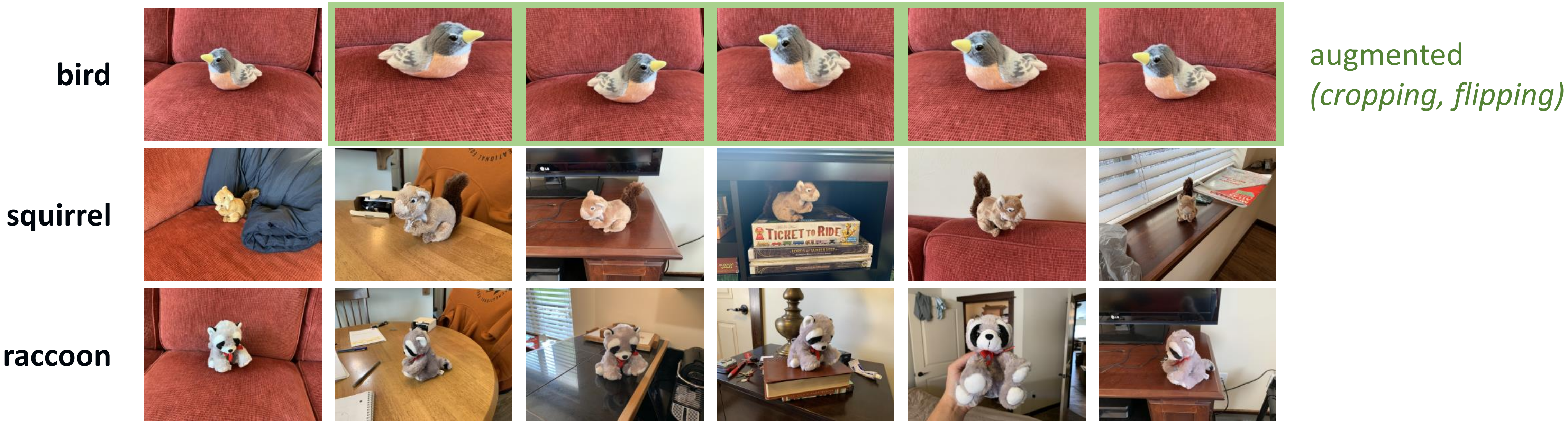


Augmentation Method	Top-1 Accuracy (%)	Top-5 Accuracy (%)
Baseline	48.13 ± 0.42	64.50 ± 0.65
Flipping	49.73 ± 1.13	67.36 ± 1.38
Rotating	50.80 ± 0.63	69.41 ± 0.48
Cropping	**61.95 ± 1.01**	**79.10 ± 0.80**
Color Jittering	49.57 ± 0.53	67.18 ± 0.42
Edge Enhancement	49.29 ± 1.16	66.49 ± 0.84
Fancy PCA	49.41 ± 0.84	67.54 ± 1.01

Augmentation Experiment – Unbalanced Dataset

Experiment to demonstrate how augmentation can improve performance

- 3-class model: Bird, squirrel, and raccoon (stuffed animals)
- Start with unbalanced dataset: 5 bird images, 30 squirrel images, 30 raccoon images
- Use augmentation to create 25 more bird images
- Train models using unbalanced and augmented dataset, compare performance



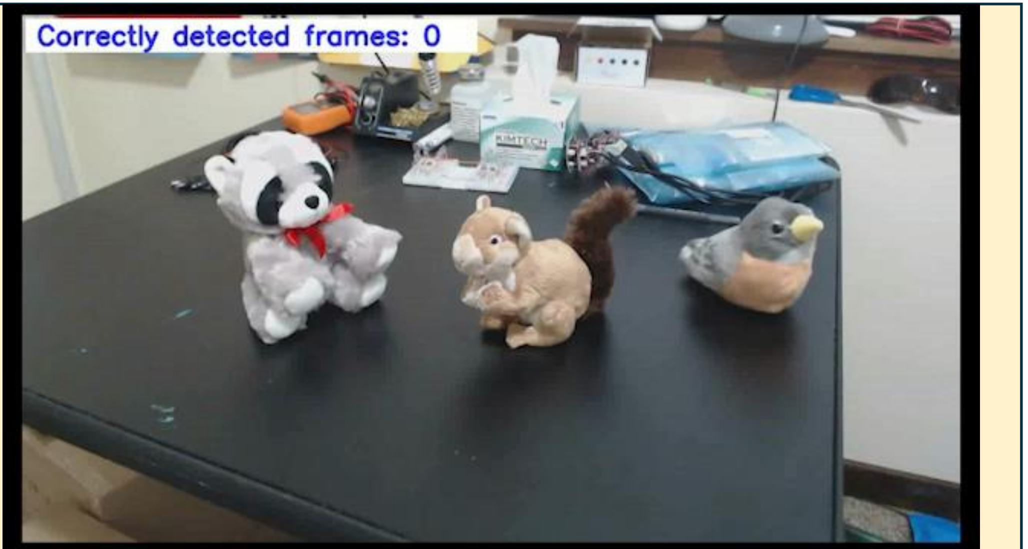
Results

Model	mAP Score (MSCOCO AP50)	Correct Frames
Base	77.87%	18
Augmented	89.47%	67

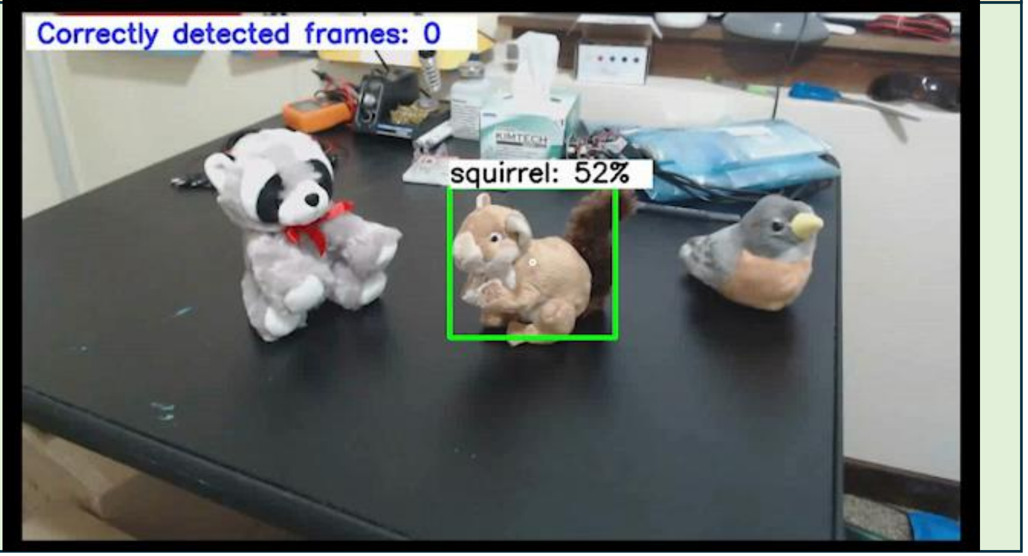
Model: yolov3-tiny
Framework: darknet
Epochs: 6000

Augmented model is better at detecting birds!

Base Model



Augmented Model



Random Background Augmentation Technique

Generate images by imposing objects on random backgrounds

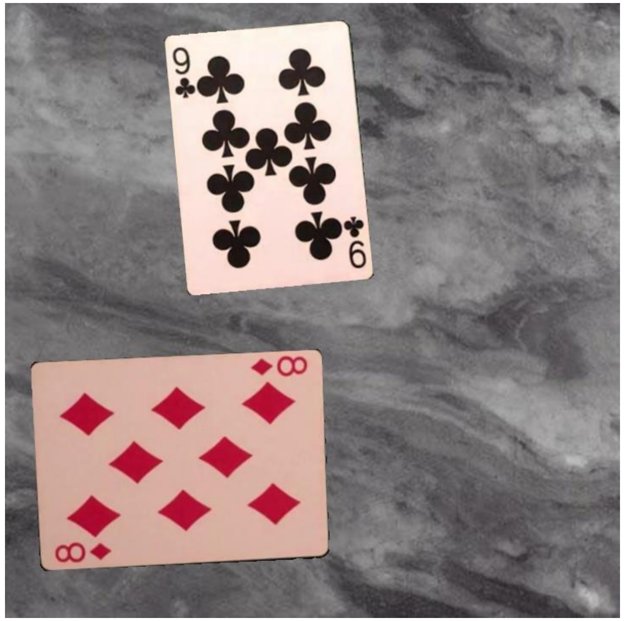
- Vary scale, perspective, position, rotation

Reduces false positives, forces model to build robust feature space

Can generate thousands of variations from a single image

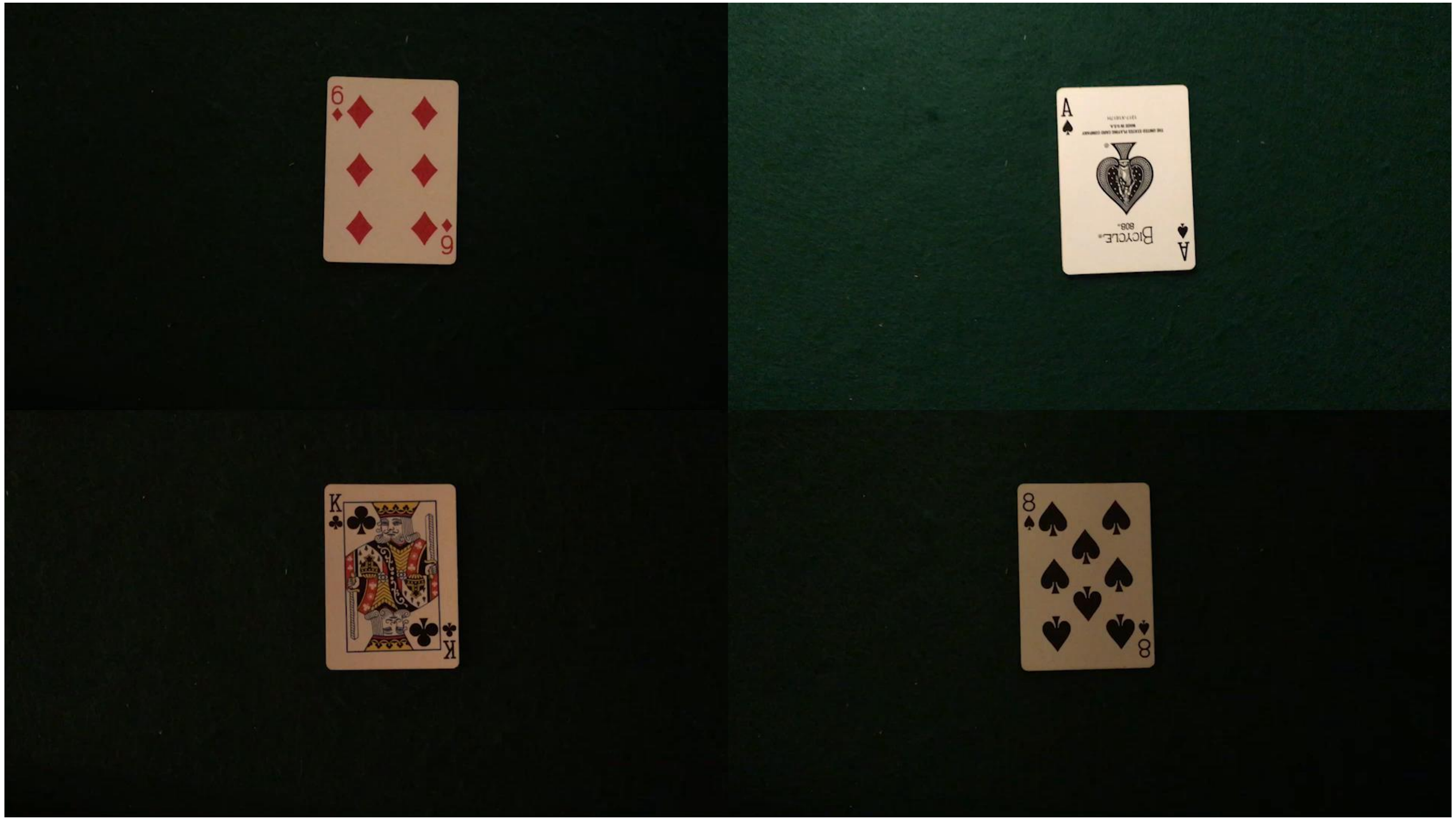
Great for 2D static objects

- Playing cards, street signs, book covers, decals, etc.

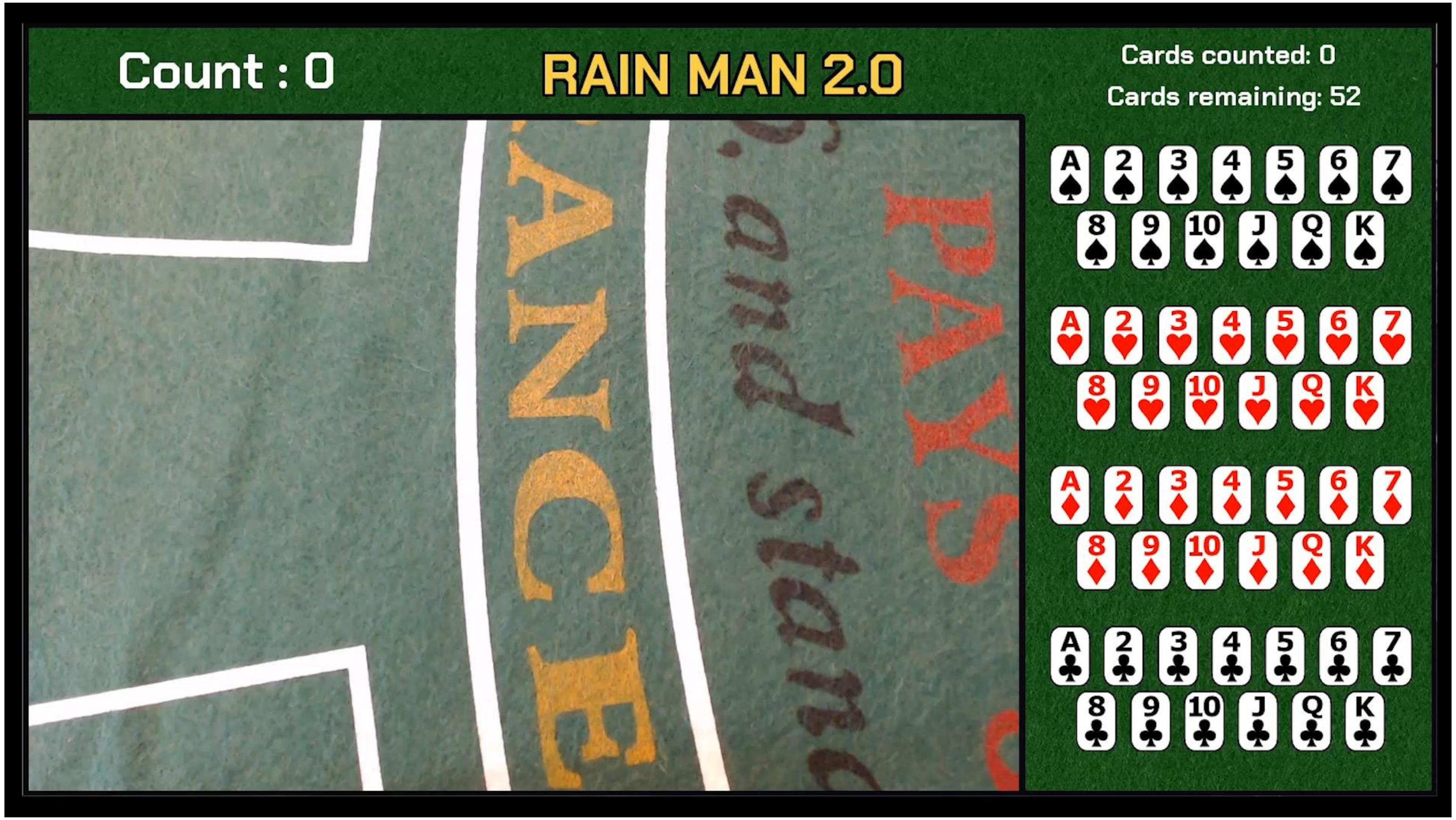


My Workflow with Playing Cards

1. Record cards in various lighting conditions
2. Isolate card images from video frames (automated)
3. Impose card images on random backgrounds (automated)
4. Save bounding box and label data in XML files (automated)



Result: Highly Accurate Playing Card Detection!



Count : 0

RAIN MAN 2.0

Cards counted: 0

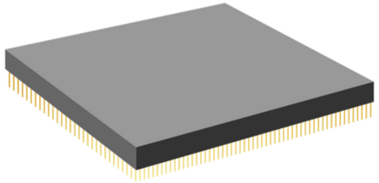
Cards remaining: 52

Augmentation Considerations

When to Perform Augmentation?

- Augment images before training
  - Reduces training pipeline overhead (speeds up training)
  - Can take considerable storage space
- Augment images during training
  - Requires more processing power (slows down training)
  - Reduces and simplifies storage requirements
  - Increases statistical variation, better improvement in generalization

VS.



Which Dataset to Augment?

- Train: Yes – allows network to learn robust features
- Validation: No – Unless planning to tweak hyperparameters during training
- Test: No – Test images should be from real-world conditions

Dataset Considerations

Overall number of images

- Generally, more is better, but there’s an upper limit to effective dataset sizes
- Dependent on application – if used in wide variety of visual conditions, more images are needed
- Pete Warden rule of thumb: 1000 images per class if training from scratch

Ratio of augmented to original images

- If only 50 images are used to create 5000 augmented images, model will be heavily biased towards 50 original images
- Ratio between 5:1 and 10:1 works well

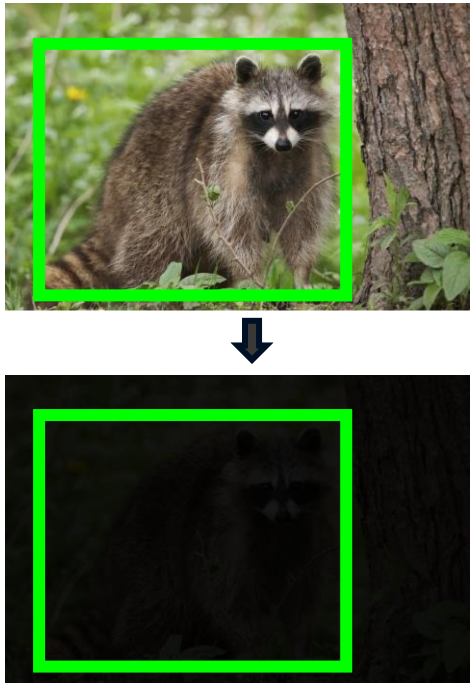
Resolution of images

- Resolution should be similar to resolution of camera/video/images that will be used in application
- Higher resolution requires much more memory during training

Limitations of Data Augmentation

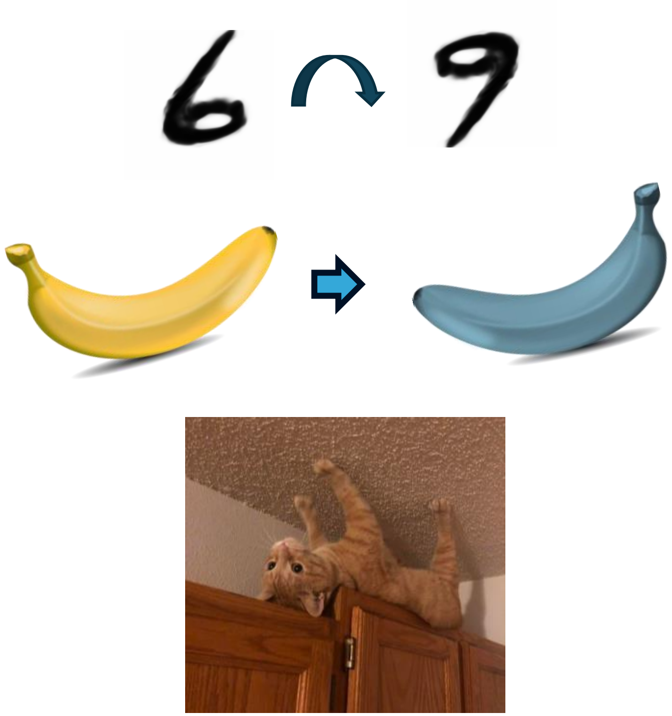
Augmentations must be label-preserving

- Unsafe transformations can destroy label data



Augmentations must be dataset aware

- Transformation can change label or be nonsensical



Can't overcome a limited dataset

- Can't "create" new classes from other classes

Conclusions

1. Data augmentation can increase model accuracy with minimal effort
2. Random background augmentation technique is very effective for static 2D objects
3. Need to consider when and where to apply augmentation, and how much
4. Data augmentation has some limitations

Any questions?

Resources

Resource Links

Basic data augmentation example code:
https://github.com/EdjeElectronics/Image-Augmentation-Examples-for-Machine-Learning

Playing card image generation:
https://github.com/geaxgx/playing-card-detection

Contact Information

Website: www.ejtech.io

Email: evan.juras@ejtech.io

References

[1]: Shorten, Connor & Khoshgoftaar, Taghi. (2019). A Survey on Image Data Augmentation for Deep Learning. Journal of Big Data. 6. 10.1186/s40537-019-0197-0.

[2]: L. Taylor, G. Nitchske. (2017). Improving Deep Learning using Generic Data Augmentation. arXiv:1708.06020

[3]: L. Perez, J. Wang. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. arXiv:1712.04621