Imbalanced Data

Anh-Khoa Nguyen
Ph.D. Student

Quang-Vinh Dinh
Ph.D. in Computer Science

Outline

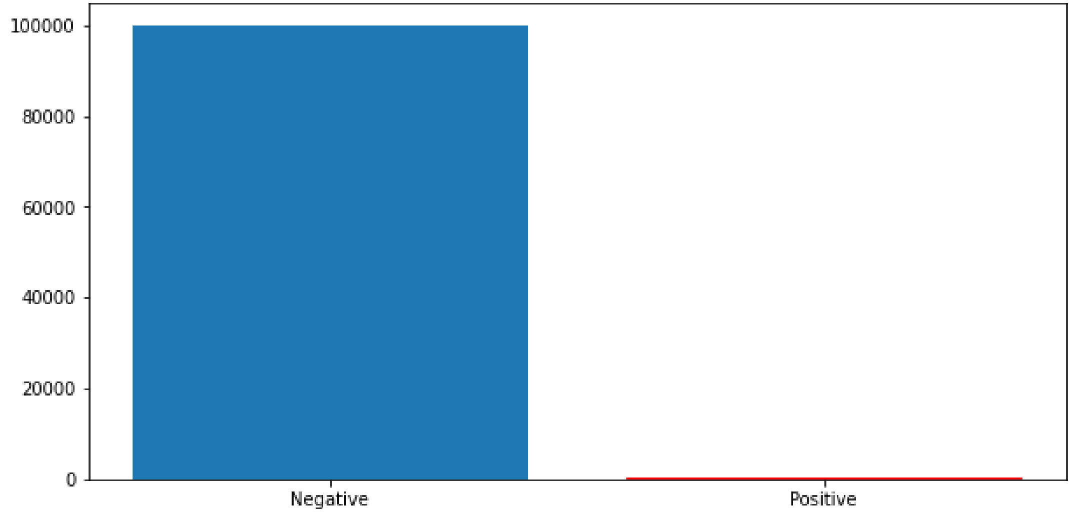
- Introduction
- Focal Loss
- Metrics
- Data Augmentation
- Experiments

Classification on Imbalanced Data

❖ Imbalanced Data vs. Balanced Data

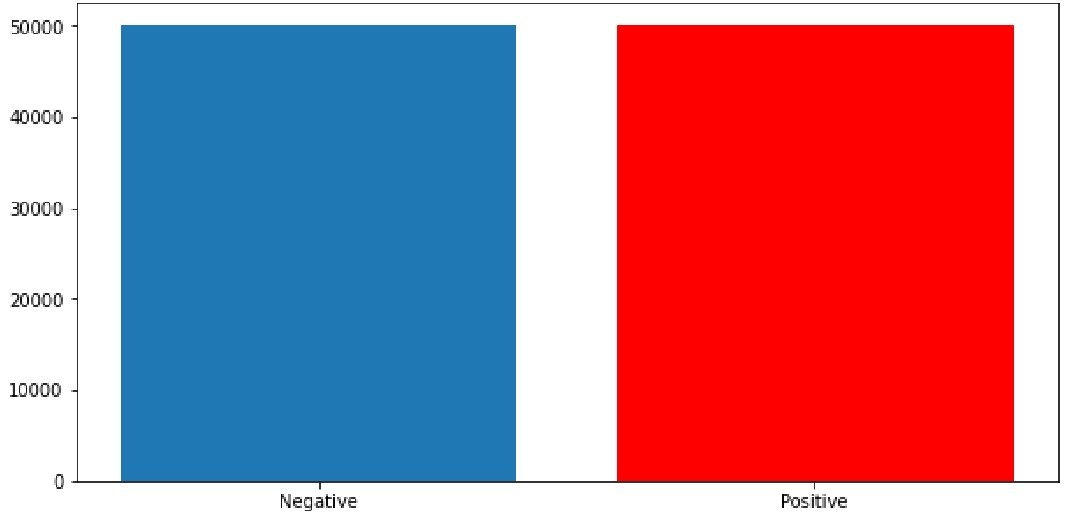
Imbalanced Data

Negative	100000
Positive	200



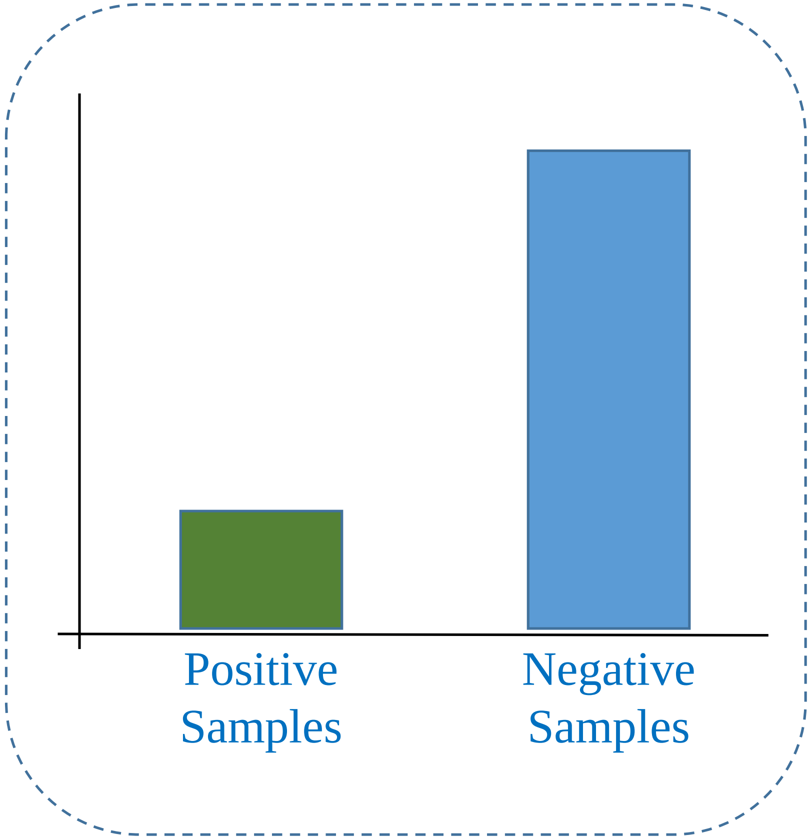
Balanced Data

Negative	50100
Postive	50100

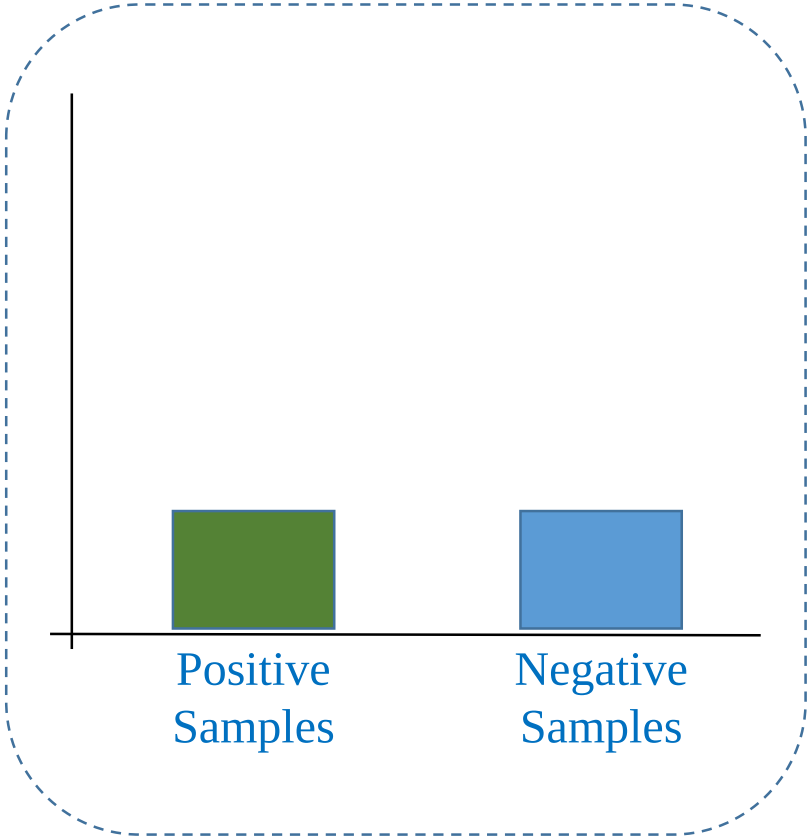


Imbalanced Data

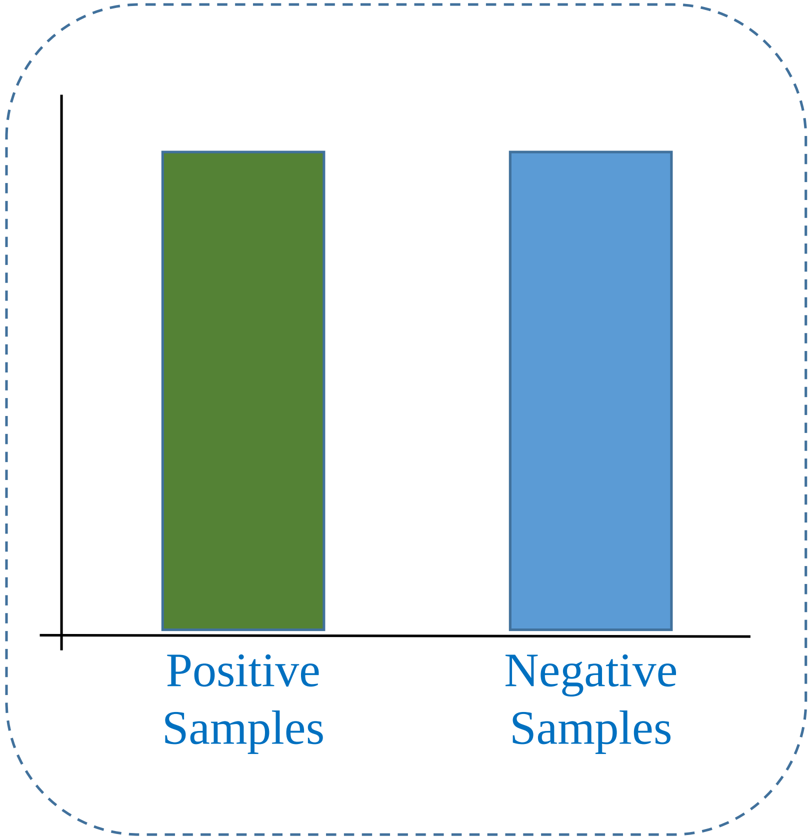
❖ Approach 1: Data manipulation



Original Data



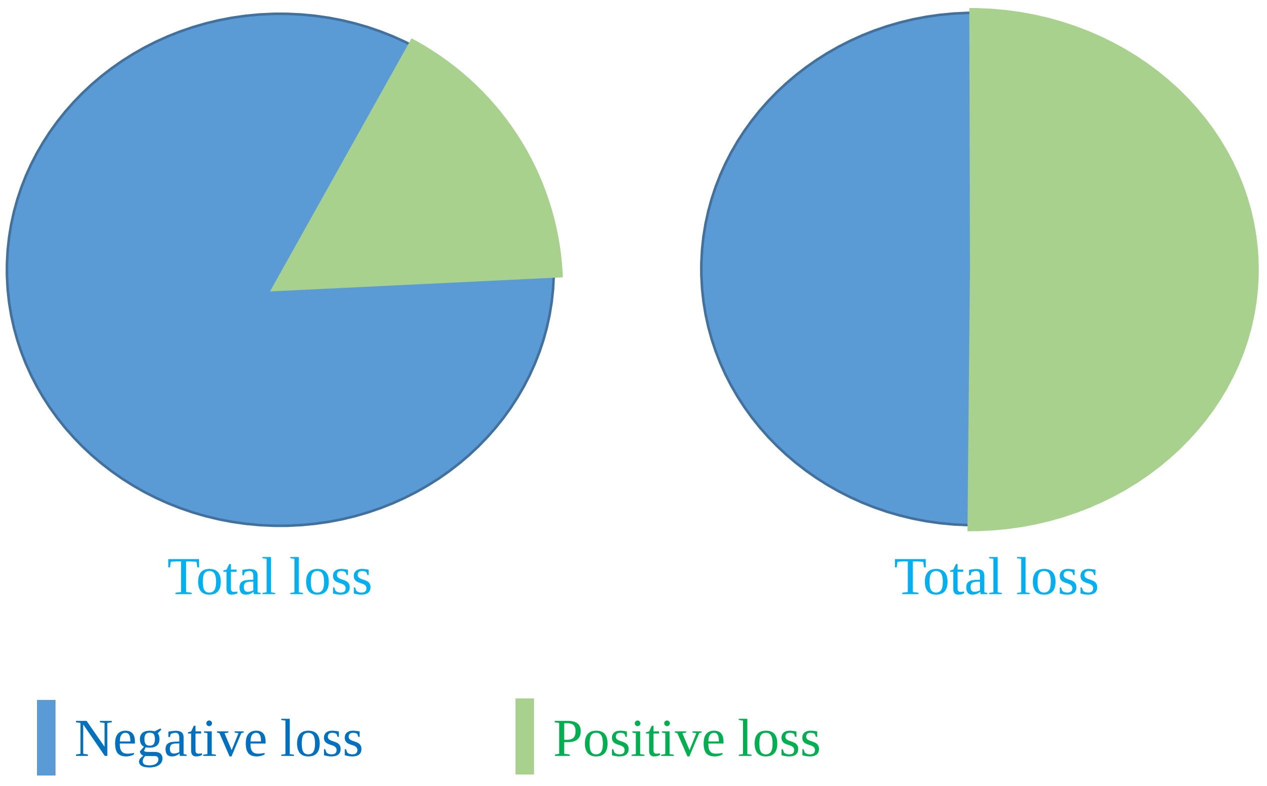
Undersampling Data



Oversampling Data

Imbalanced Data

❖ Approach 2: Loss Functions



Class weight

$$w_p = \frac{N}{2N_p}$$

Tell the model to "pay more attention" to samples from an under-represented class

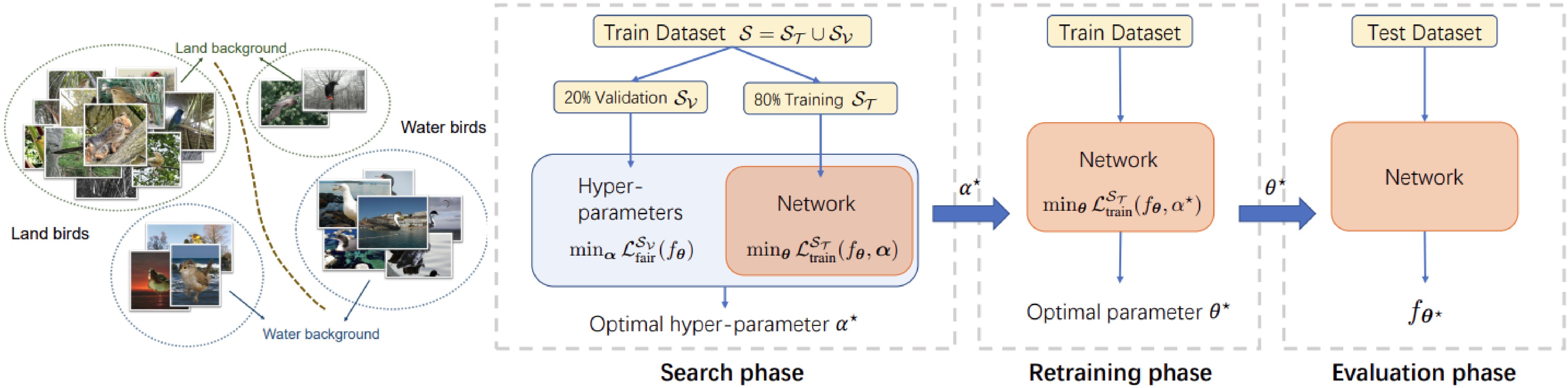
A higher loss means higher optimization which results in efficient classification

Focal loss

$$\mathrm{FL}(\mathrm{p_t}) = -(1 - p_t)^{\gamma}\log(\mathrm{p_t})$$

Imbalanced Data

❖ Approach 3: Optimization



https://arxiv.org/pdf/2201.01212.pdf

AutoBalance: Optimized Loss Functions for Imbalanced Data, 2022

Outline

- Introduction
- Focal Loss
- Metrics
- Data Augmentation
- Experiments

Binary Crossentropy

❖ Linear regression

Area-based House Price Data

Feature	Label
area	price
6.7	9.1
4.6	5.9
3.5	4.6
5.5	6.7

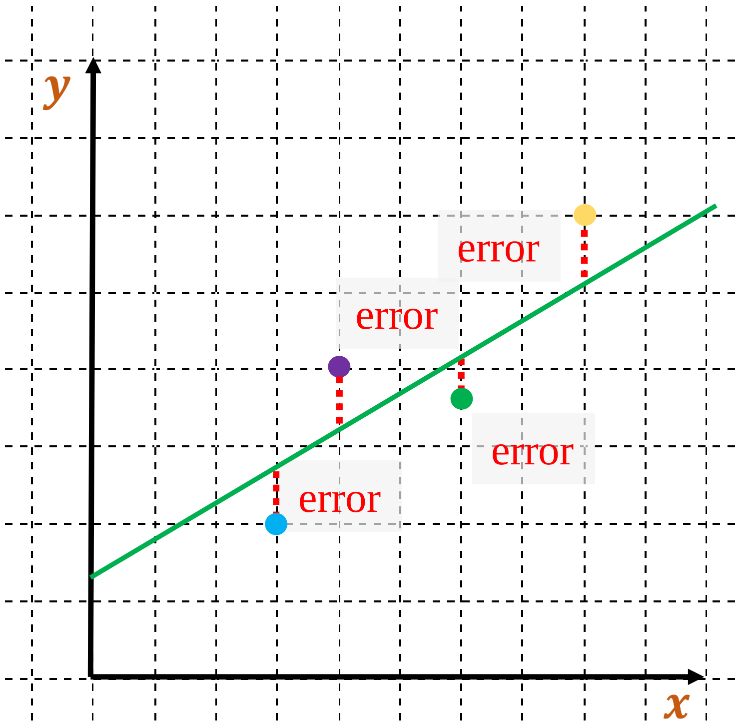
Training data

construct →

$\hat{y} = \boldsymbol{\theta}^T \boldsymbol{x} = ax + b$

$\hat{y} \in (-\infty \quad +\infty)$

Model



Find the line $\hat{y} = \boldsymbol{\theta}^T \boldsymbol{x}$ that is best fit given data, then use y to predict for new data

Binary Crossentropy

❖ Given a new kind of data

Feature **Label**

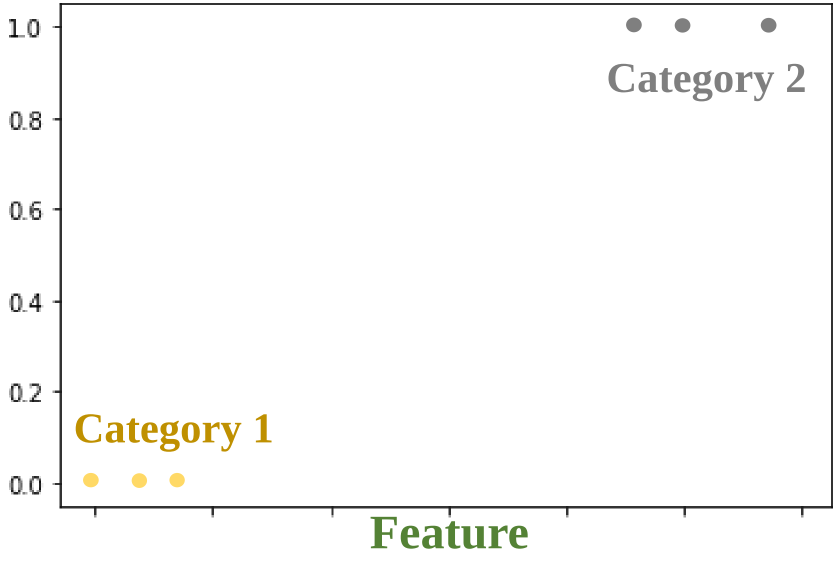
Petal_Length	Category	
1.4	Flower A	Category 1
1	Flower A	Category 1
1.5	Flower A	Category 1
3	Flower B	Category 2
3.8	Flower B	Category 2
4.1	Flower B	Category 2

Assign numbers to categories

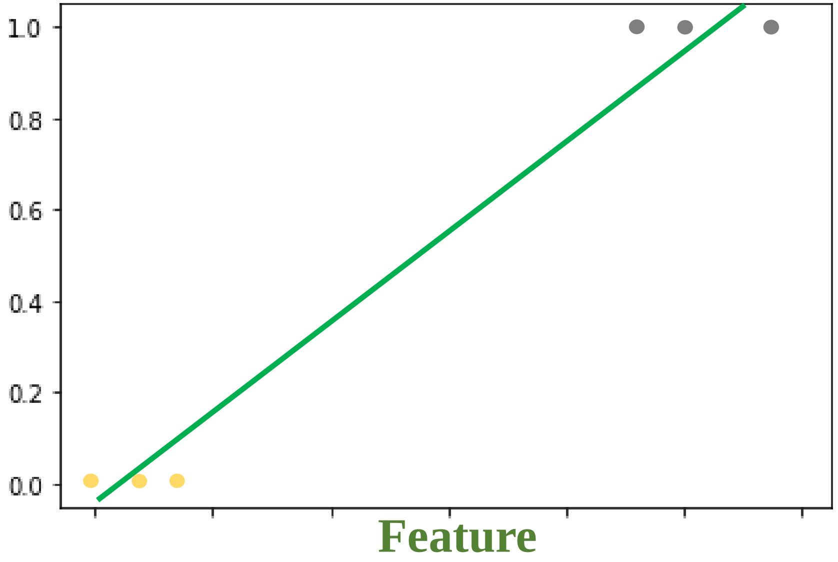
Feature **Label**

Petal_Length	Category	
1.4	0	Category 1
1	0	Category 1
1.5	0	Category 1
3	1	Category 2
3.8	1	Category 2
4.1	1	Category 2

Plot data



A line is not suitable for this data



Binary Crossentropy

❖ Given a new kind of data

Feature **Label**

Petal_Length	Category	
1.4	Flower A	Category 1
1	Flower A	Category 1
1.5	Flower A	Category 1
3	Flower B	Category 2
3.8	Flower B	Category 2
4.1	Flower B	Category 2

Assign numbers to categories

Feature **Label**

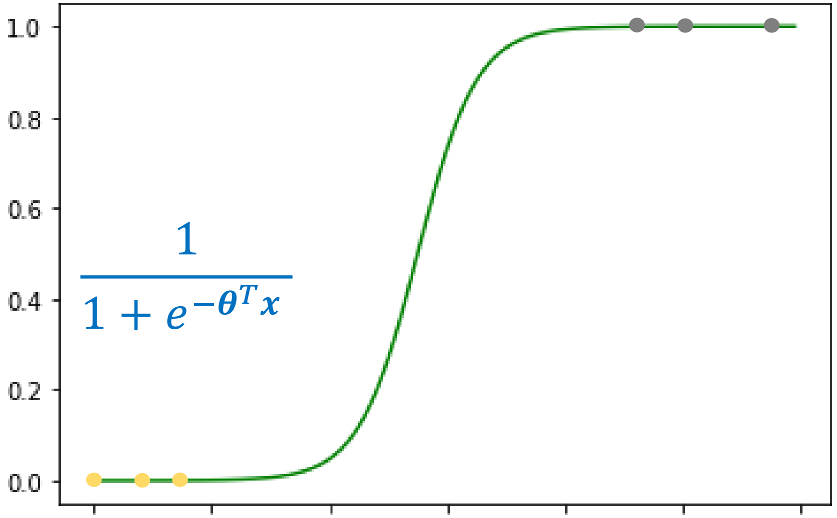
Petal_Length	Category	
1.4	0	Category 1
1	0	Category 1
1.5	0	Category 1
3	1	Category 2
3.8	1	Category 2
4.1	1	Category 2

Sigmoid function could fit the data

$$z = \boldsymbol{\theta}^T \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{\theta}$$

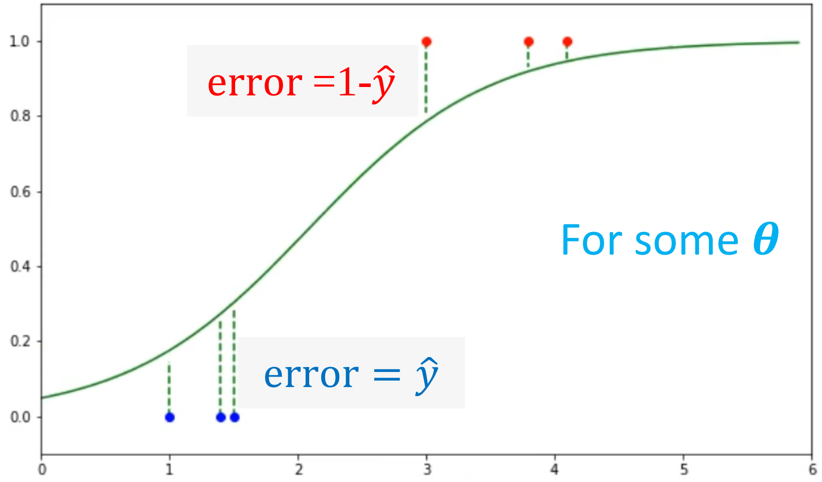
$$\hat{y} = \sigma(z) = \frac{1}{1+e^{-z}}$$

$$\hat{y} \in (0 \quad 1)$$



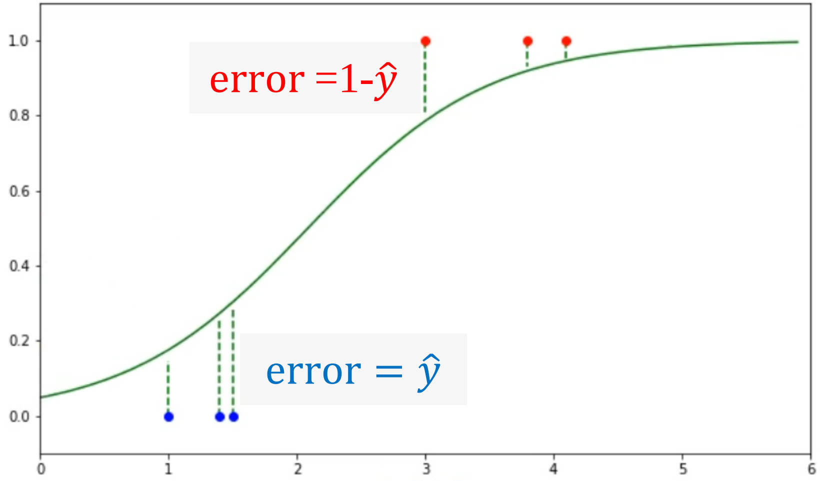
Error

$$\text{if } y = 1$$
$$\text{error} = 1 - \hat{y}$$
$$\text{if } y = 0$$
$$\text{error} = \hat{y}$$



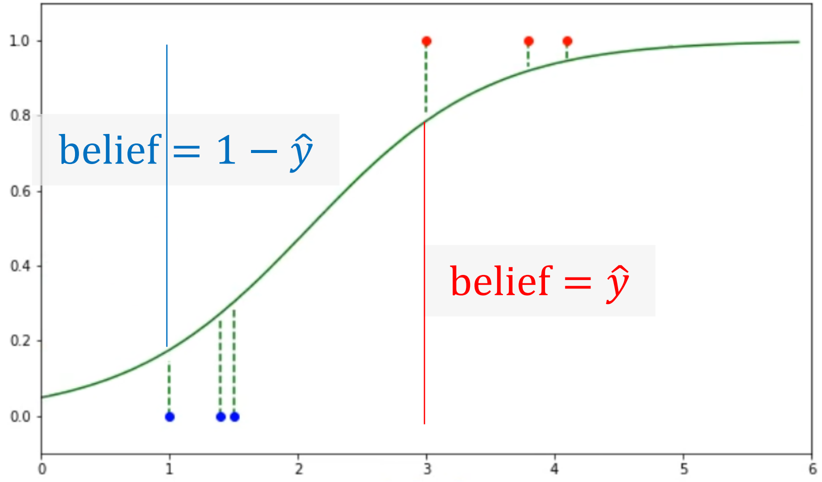
Binary Crossentropy

❖ Construct loss



Error

if $y = 1$
  $\text{error} = 1 - \hat{y}$
if $y = 0$
  $\text{error} = \hat{y}$



Belief

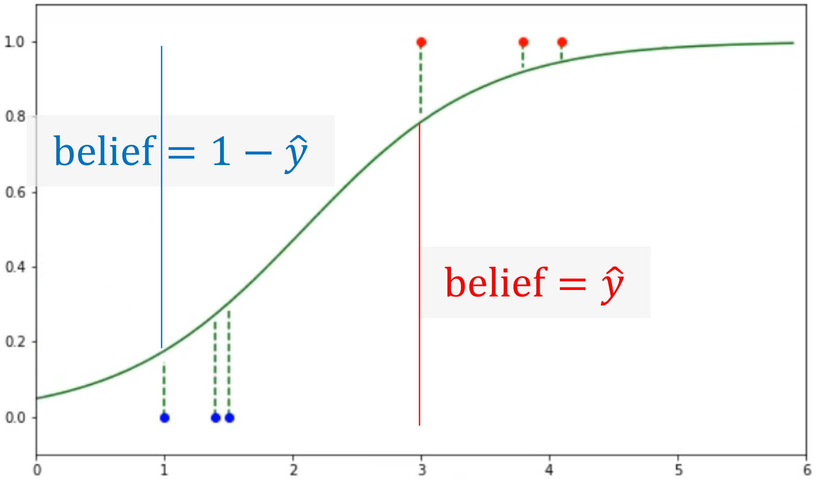
if $y = 1$
  $\text{belief} = \hat{y}$
if $y = 0$
  $\text{belief} = 1 - \hat{y}$

$$P = \hat{y}^{\,y}(1 - \hat{y})^{1-y}$$

Minimize error ~ maximize belief ~ Minimize (-belief)

Binary Crossentropy

❖ Construct loss



Belief

$$\text{if } y = 1$$
$$\text{belief} = \hat{y}$$
$$\text{if } y = 0$$
$$\text{belief} = 1 - \hat{y}$$

$$P = \hat{y}^{\,y}(1 - \hat{y})^{1-y}$$

$$\text{belief} = P$$

$$\text{log\_belief} = \log P$$

$$\text{log\_belief} = y\log\hat{y} + (1 - y)\log(1 - \hat{y})$$

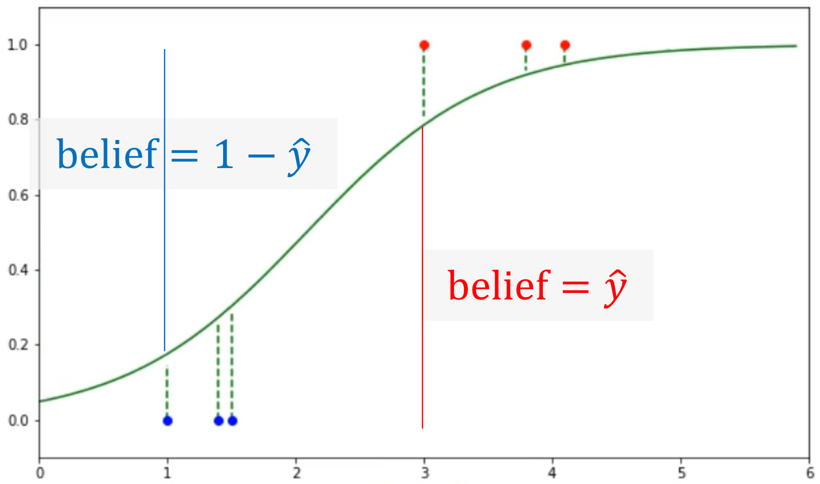
$$\text{loss} = -\text{log\_belief}$$
$$= -[y\log\hat{y} + (1 - y)\log(1 - \hat{y})]$$

$$\mathrm{L} = -y\log\hat{y} - (1 - y)\log(1 - \hat{y})$$

Binary cross-entropy

Binary Crossentropy

❖ Construct loss



Belief

$$\text{if } y_i = 1 \quad \text{belief} = \hat{y}_i$$
$$\text{if } y_i = 0 \quad \text{belief} = 1 - \hat{y}_i$$

$$P_i = \hat{y}_i^{y_i}(1-\hat{y}_i)^{1-y_i}$$

$$\text{belief} = \prod_{i=1}^{n} P_i \qquad \text{since iid}$$

$$\text{log\_belief} = \sum_{i=1}^{n} \log P_i$$

$$\text{log\_belief} = \sum_{i=1}^{n} [y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)]$$

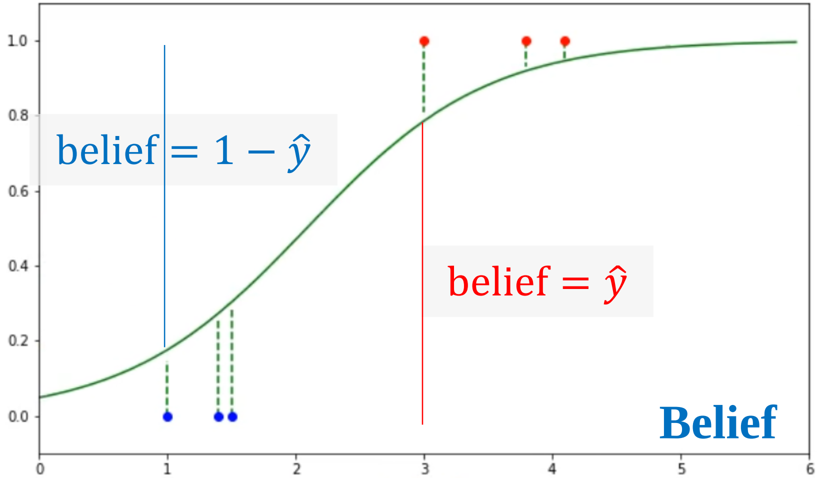
$$\text{loss} = -\text{log\_belief}$$
$$= -\sum_{i=1}^{n} [y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)]$$

$$\mathrm{L} = \frac{1}{N}\left(-\boldsymbol{y}^T log(\hat{\boldsymbol{y}}) - (1-\boldsymbol{y}^T) log(1-\hat{\boldsymbol{y}})\right)$$

Binary cross-entropy

Binary Crossentropy

❖ Construct loss

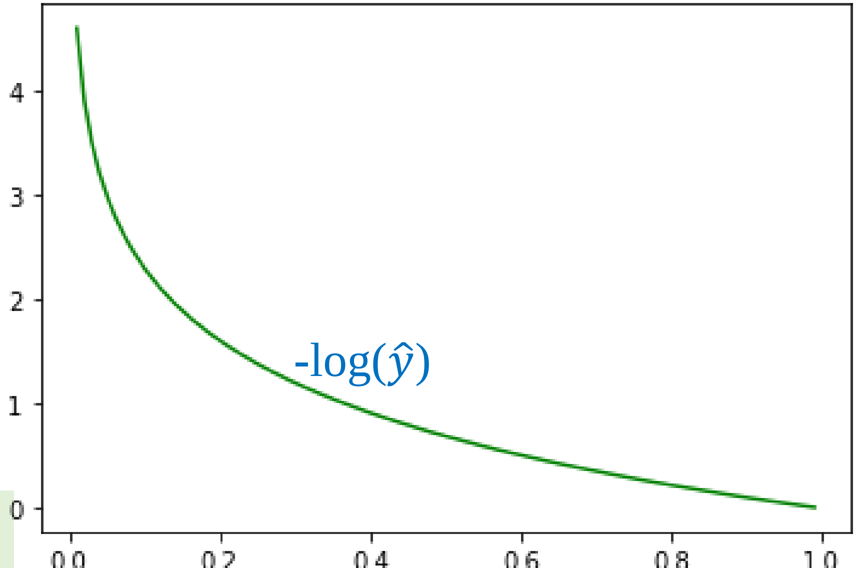


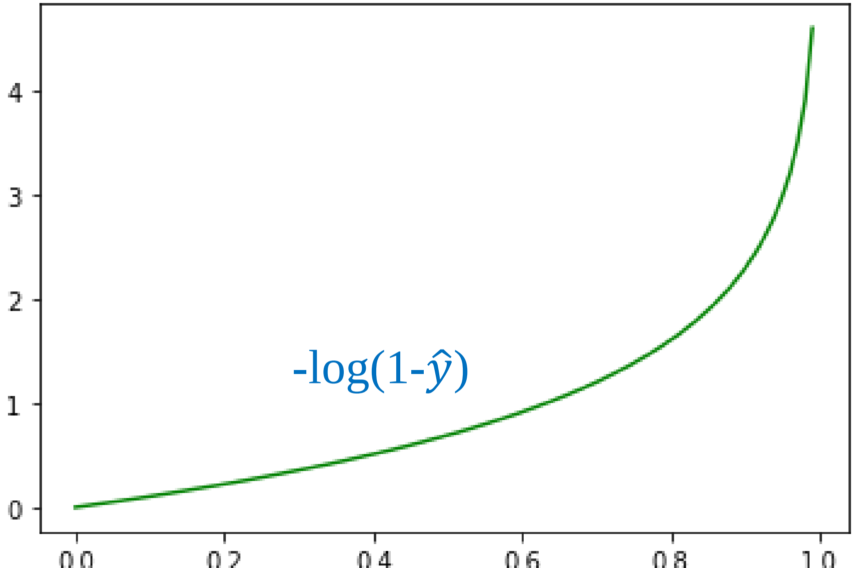
$$L = \frac{1}{N}\left(-\boldsymbol{y}^T log(\hat{\boldsymbol{y}}) - (1 - \boldsymbol{y}^T) log(1 - \hat{\boldsymbol{y}})\right)$$

Binary cross-entropy

if $y_i = 1$
belief $= \hat{y}_i$
if $y_i = 0$
belief $= 1 - \hat{y}_i$

$$P_i = \hat{y}_i^{y_i}(1 - \hat{y}_i)^{1-y_i}$$





Binary Crossentropy

❖ Construct loss

Model and Loss

$$z = \boldsymbol{\theta}^T \boldsymbol{x}$$

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}$$

$$\mathrm{L} = -y\log(\hat{y}) - (1 - y)\log(1 - \hat{y})$$

Derivative

$$\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z} \frac{\partial z}{\partial \theta}$$

$$\frac{\partial L}{\partial \hat{y}} = -\frac{y}{\hat{y}} + \frac{1 - y}{1 - \hat{y}} = \frac{\hat{y} - y}{\hat{y}(1 - \hat{y})}$$

$$\frac{\partial \hat{y}}{\partial z} = \hat{y}(1 - \hat{y})$$

$$\frac{\partial z}{\partial \theta} = x$$

$$\frac{\partial L}{\partial \theta} = \boldsymbol{x}(\hat{y} - y)$$

Focal Loss

❖ Binary Cross Entropy Loss

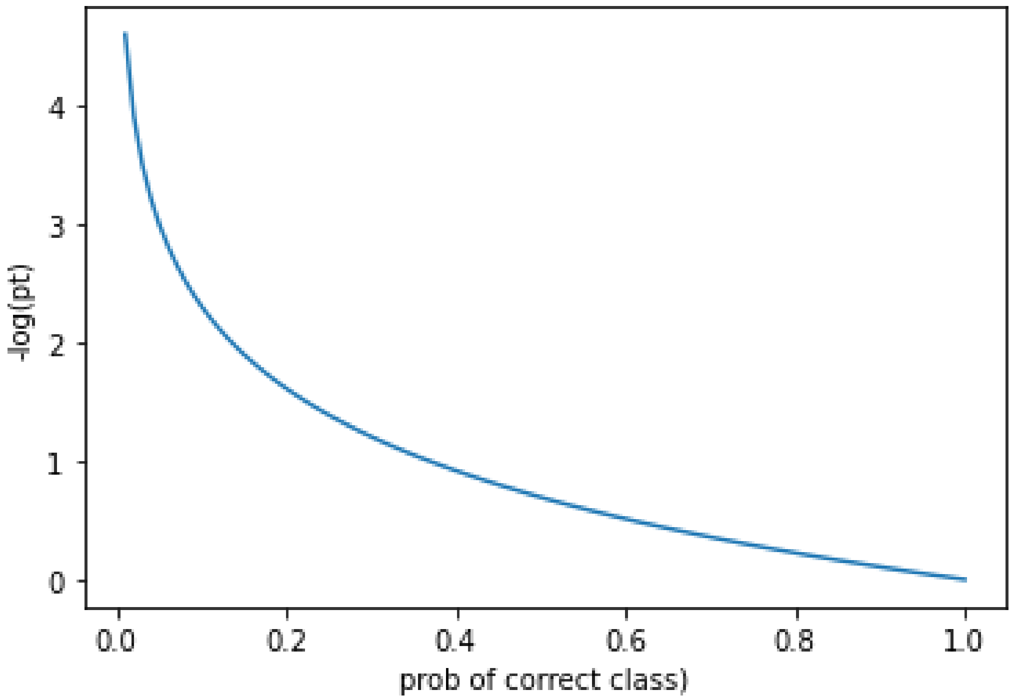
$$\mathrm{BCE}(p,y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1-p) & \text{otherwise} \end{cases}$$

$y \in \{0,1\}$ label cho negative và positive class
$p \in [0,1]$ xác suất model predict cho một class

Rewrite

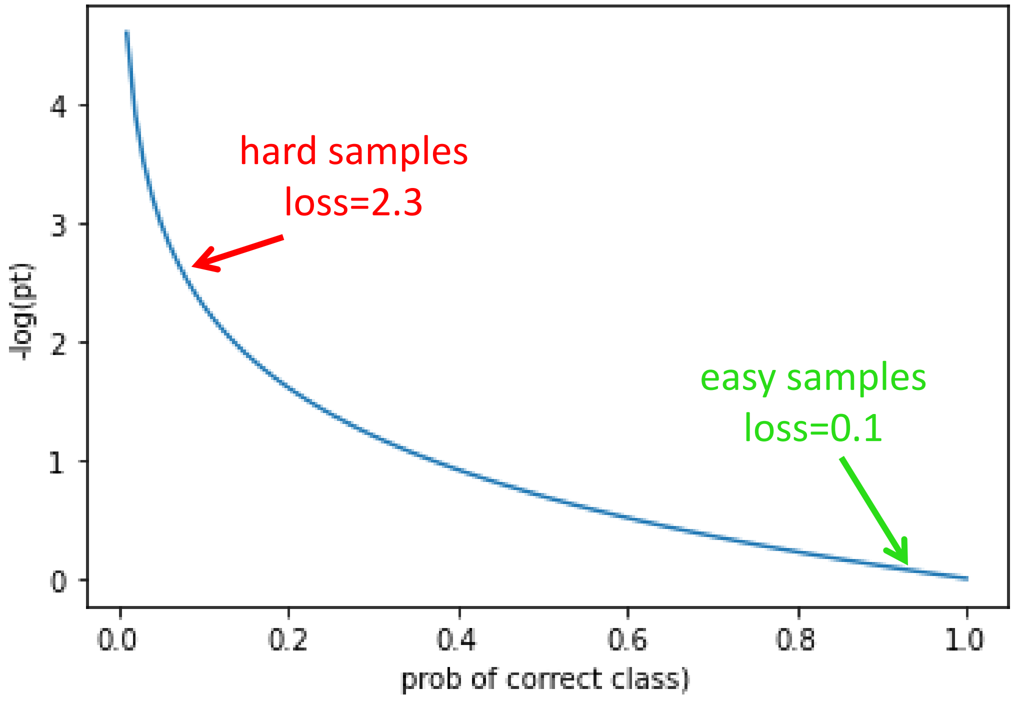
$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise} \end{cases}$$

$$\mathrm{BCE}(p,y) = \mathrm{BCE}(p_t) = -\log(p_t)$$



Focal Loss

❖ Binary Cross Entropy Loss



Imbalanced Case:
- 100000 easy samples vs 100 hard samples

Easy samples loss = 100000*0.1 = 10000

Hard samples loss = 100*2.3 = 230

Loss = Easy samples loss + Hard samples loss

Easy samples loss : Hard samples loss = 10000:230 ≈ 43

=> Loss phần lớn bị ảnh hưởng của Easy samples và model có thể bỏ qua hard samples.

=> BCE không tốt cho trường hợp data bị imbalance nặng

Focal Loss

❖ Balanced Binary Cross Entropy Loss

$$\mathrm{BCE}(\mathrm{p}, \mathrm{y}) = \mathrm{BCE}(\mathrm{p_t}) = -\log(\mathrm{p_t})$$

weighting factor

$$\text{Balanced BCE}(\mathrm{p_t}) = -\alpha_t \log(\mathrm{p_t})$$

class 1: $\alpha_t \in [0,1]$

class 0 : 1-α_t

Thêm weighting factor: thông thường là tần suất xuất hiện của class

Số lượng sample >> thì weighting factor <<

Số lượng sample << thì weighting factor >>

=> Giúp cân bằng lại đóng góp vào loss của các class

=> Điều này chỉ giúp cho thay đổi trọng số trên loss của từng class. Chưa giải quyết vấn đề tập trung vào học các class khó học

Focal Loss

❖ Focal Loss

$$\mathrm{BCE}(p, y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1-p) & \text{otherwise} \end{cases}$$

-

Rewrite

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise} \end{cases}$$

$$\mathrm{BCE}(p, y) = \mathrm{BCE}(p_t) = -\log(p_t)$$

$$\mathrm{BCE}(p, y) = \mathrm{BCE}(p_t) = -\log(p_t)$$

modulating
factor

$$\text{Focal Loss }(p_t) = -(1-p_t)^{\gamma}\log(p_t)$$

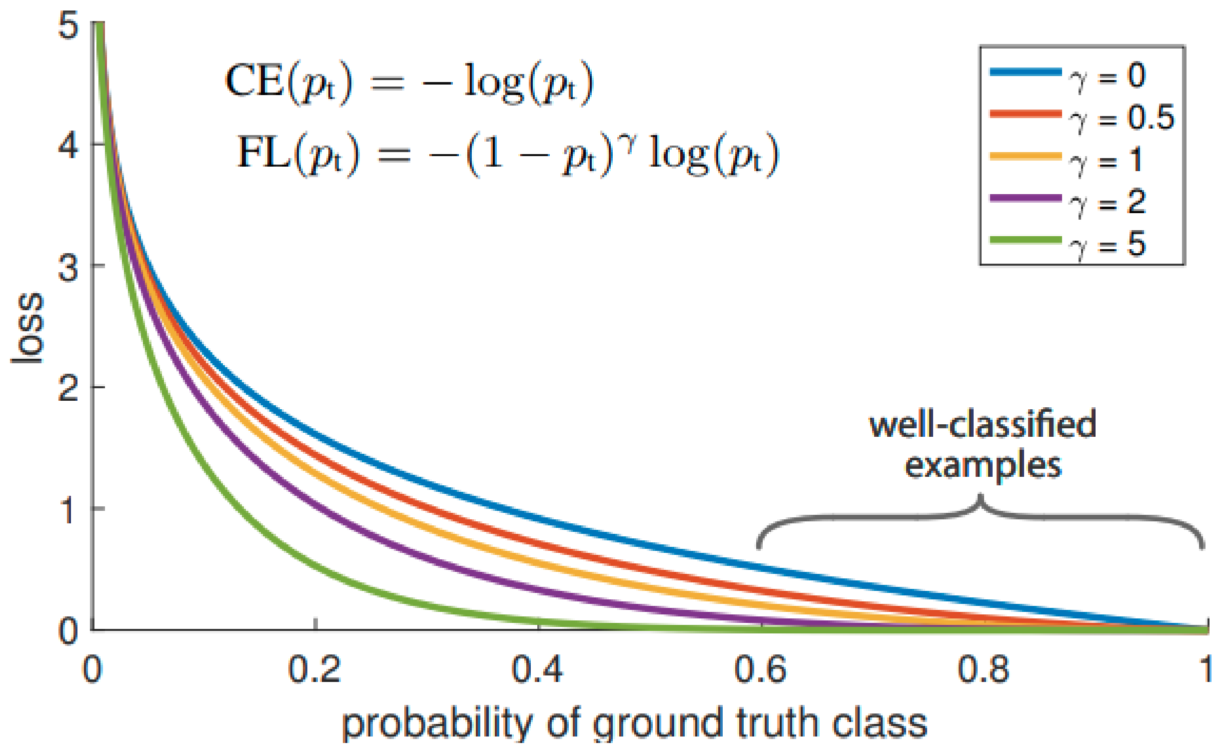
γ theo paper test [0, 5]

- Khi model predict **xác suất càng gần với label**, p_t sẽ **tiến về 1** thể hiện model đã **học tốt** cho class này và **modulating factor tiến về 0**, đóng góp vào loss giảm (easy samples)

- Khi model predict **xác suất càng xa với label** p_t sẽ **tiến về 0** thể hiện model đã **học không tốt** cho class này và **modulating factor tiến về 1**, đóng góp vào loss tăng (hard samples)

Focal Loss

❖ Focal Loss



– γ = 0 chính là cross entropy loss

Lin, T., Goyal, P., Girshick, R.B., He, K., & Dollár, P. (2017). Focal Loss for Dense Object Detection. 2017 IEEE International Conference on Computer Vision (ICCV), 2999-3007

Focal Loss

❖ α-balanced Variant of The Focal Loss

$$\boxed{\text{Focal Loss}\ (\mathrm{p_t}) = -(1-p_t)^{\gamma}\log(\mathrm{p_t})}$$

γ theo paper test [0, 5]

$\Big\downarrow$ weighting factor

$$\boxed{\text{Focal Loss}\ (\mathrm{p_t}) = -\alpha_t(1-p_t)^{\gamma}\log(\mathrm{p_t})}$$

Thực nghiệm khi thêm α_t performance có cải thiện	γ giúp loss tập trung vào hard samples
Khi sử dụng sigmoid activation để tính p_t thì kết quả stable hơn	α_t trọng số giúp cân bằng loss theo số lượng samples

Focal Loss

❖ α-balanced Variant of The Focal Loss

α	AP	AP_{50}	AP_{75}
.10	0.0	0.0	0.0
.25	10.8	16.0	11.7
.50	30.2	46.7	32.8
.75	31.1	49.4	33.0
.90	30.8	49.7	32.3
.99	28.7	47.4	29.9
.999	25.1	41.7	26.1

(a) **Varying** α **for CE loss** ($\gamma = 0$)

γ	α	AP	AP_{50}	AP_{75}
0	.75	31.1	49.4	33.0
0.1	.75	31.4	49.9	33.1
0.2	.75	31.9	50.7	33.4
0.5	.50	32.9	51.7	35.2
1.0	.25	33.7	52.0	36.2
2.0	.25	**34.0**	**52.5**	**36.5**
5.0	.25	32.2	49.6	34.8

(b) **Varying** γ **for FL** (w. optimal α)

Kết quả thực nghiệm so sánh giữa Cross Entropy và Focal Loss
trên cùng một network cho task Object Detction

Lin, T., Goyal, P., Girshick, R.B., He, K., & Dollár, P. (2017). Focal Loss for Dense Object Detection. 2017 IEEE International Conference on Computer Vision (ICCV), 2999-3007.

Outline

- Introduction
- Focal Loss
- Metrics
- Data Augmentation
- Experiments

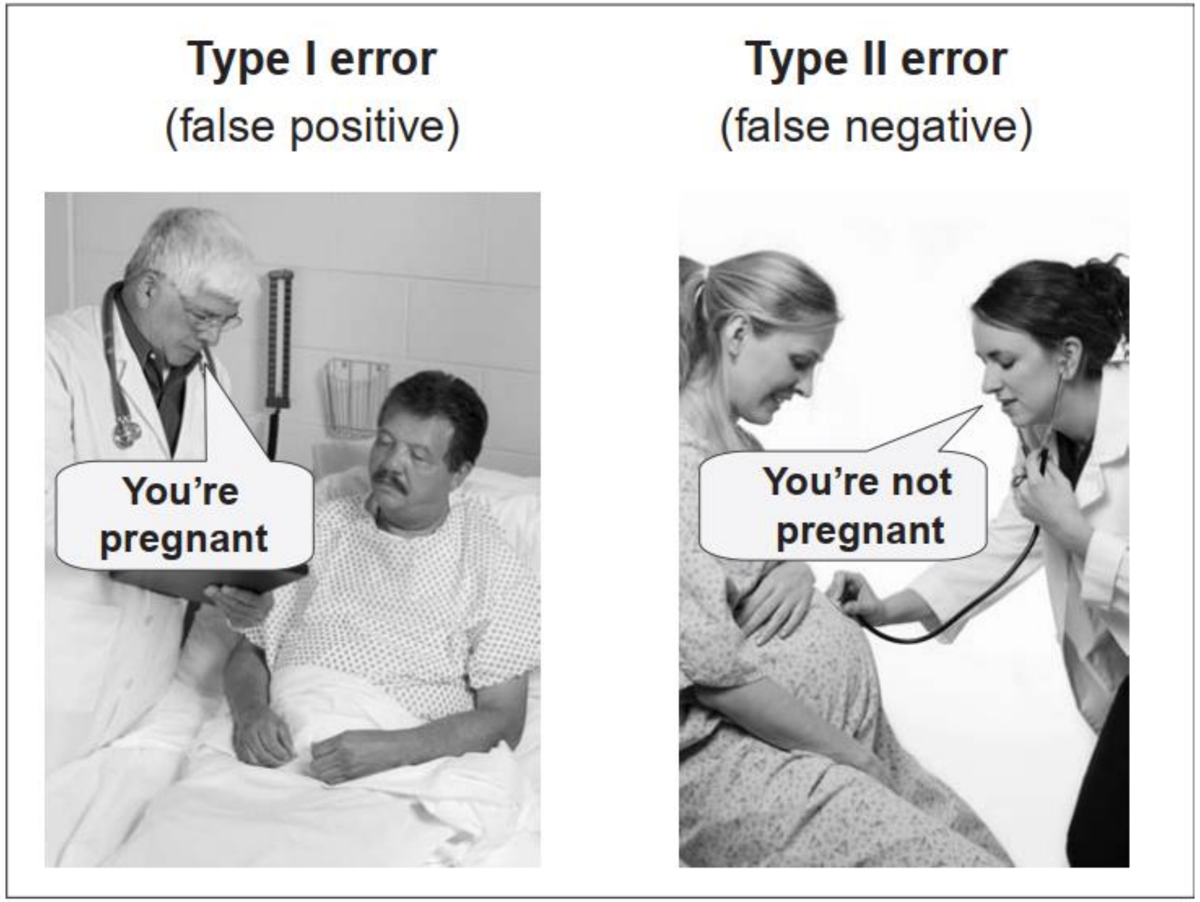
Metrics

Confusion Matrix

		Actual Value	
		Positive	Negative
Predicted Value	Positive	TP (True Positive)	FP (False Positive)
	Negative	FN (False Negative)	TN (True Negative)

- True Positive (TP) : Observation is positive, and is predicted to be positive.
- False Negative (FN) : Observation is positive, but is predicted negative.
- True Negative (TN) : Observation is negative, and is predicted to be negative
- False Positive (FP) : Observation is negative, but is predicted positive.

Result	Prediction
True Positive (TP): A correct detection.	
False Positive (FP): A wrong detection.	
False Negative (FN): A ground truth not detected	
True Negative (TN): Does not apply.	



The Essential Guide to Effect Sizes

https://www.kdnuggets.com/2020/04/performance-evaluation-metrics-classification.html

Metrics

Result	Prediction
True Positive (TP): A correct detection.	
False Positive (FP): A wrong detection.	
False Negative (FN): A ground truth not detected	
True Negative (TN): Does not apply.	

❖ Precision

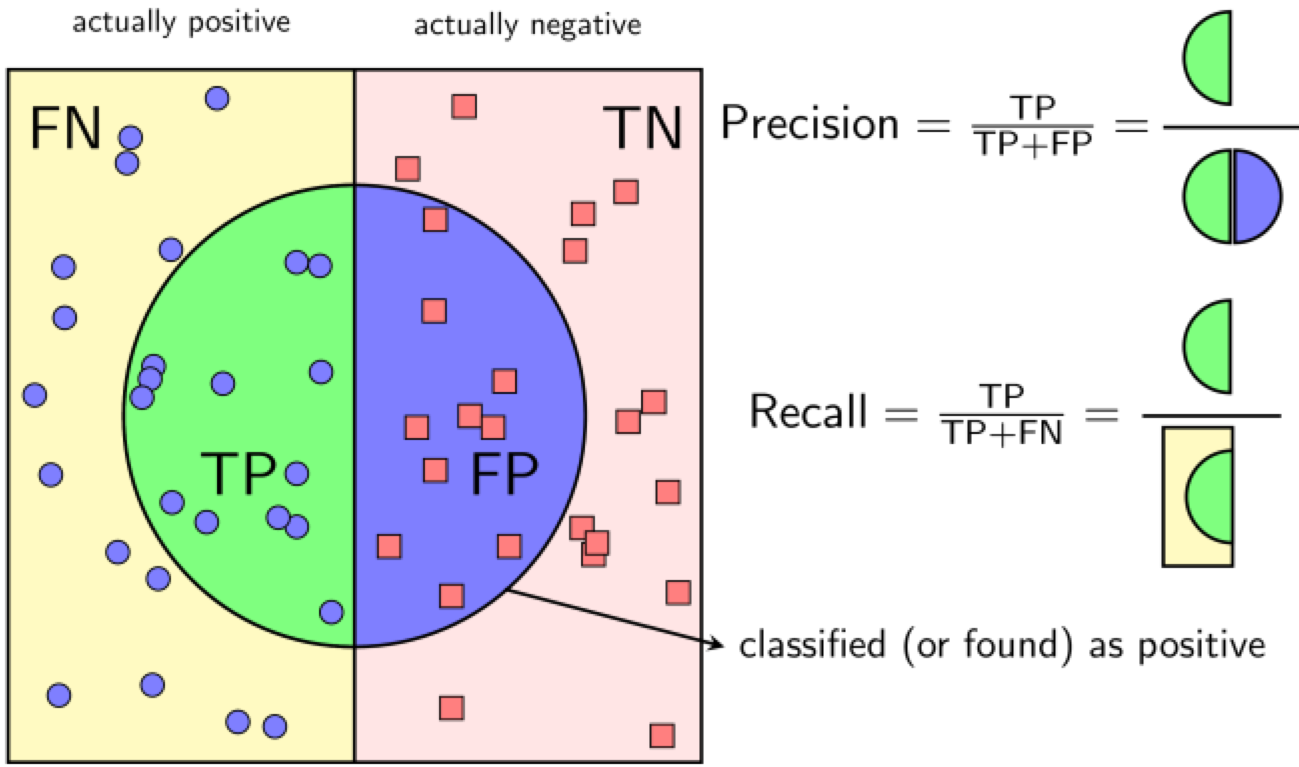
❖ Ability of a model to identify only the relevant objects

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{all\ detections}$$

❖ Recall

❖ Ability of a model to find all the relevant cases (all ground truth bounding boxes)

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{all\ ground\ truths}$$



Metrics

❖ Precision

❖ Ability of a model to identify only the relevant objects

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{all\ detections}$$

❖ Recall

❖ Ability of a model to find all the relevant cases (all ground truth bounding boxes)

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{all\ ground\ truths}$$

n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Recall: When it's actually yes, how often does it predict yes?

TP/actual yes = 100/105 = 0.95

Precision: When it predicts yes, how often is it correct?

TP/predicted yes = 100/110 = 0.91

Classification on Imbalanced Data

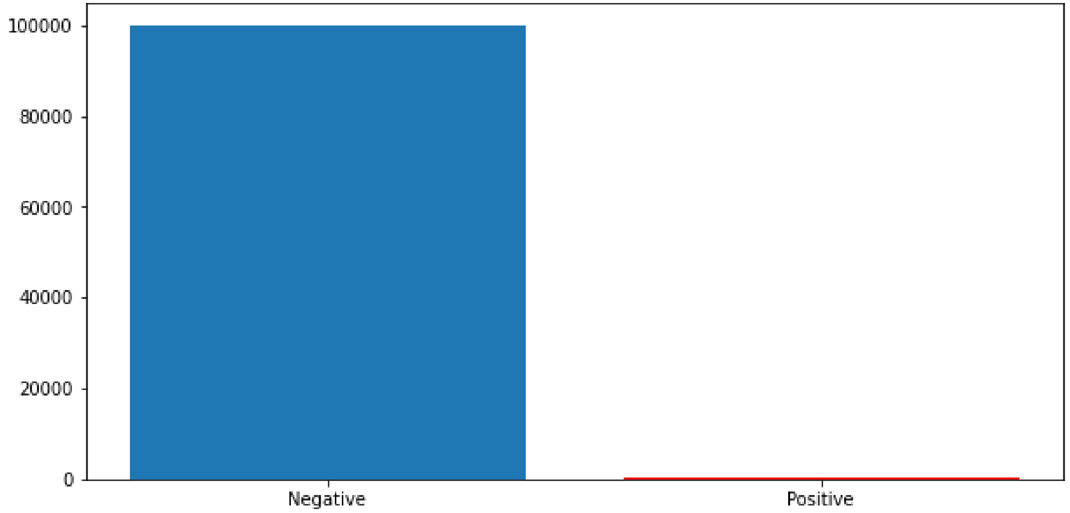
❖ Imbalanced Data vs Balanced Data

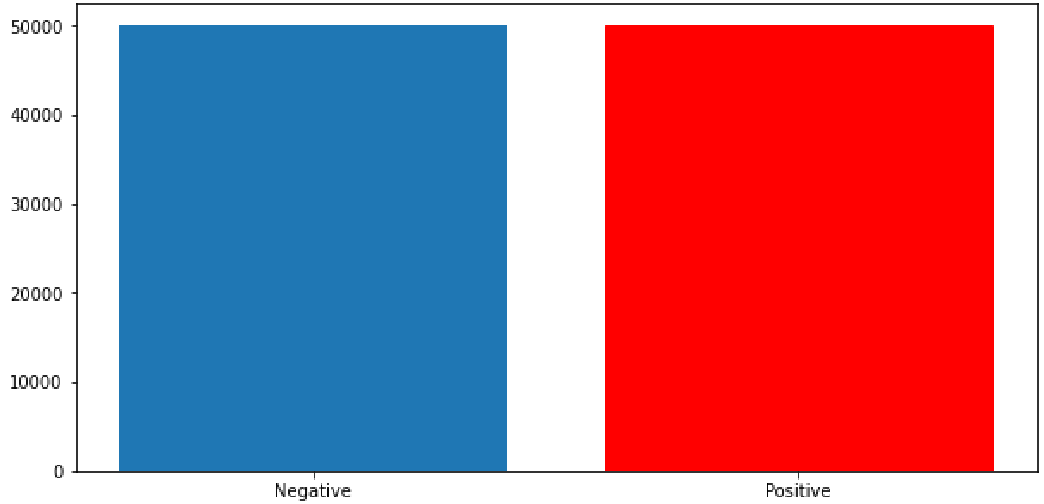
Imbalanced Data

	Labels	Model Predict Negative	Model Predict Positive
Negative	100000	99900	100
Positive	200	100	100

Balanced Data

	Labels	Model Predict Negative	Model Predict Positive
Negative	50100	50000	100
Postive	50100	100	50000





Classification on Imbalanced Data

❖ Imbalanced Data vs Balanced Data

TN FP

Imbalanced Data

	Labels	Model Predict Negative	Model Predict Positive
Negative	100000	99900	100
Positive	200	100	100

Balanced Data

	Labels	Model Predict Negative	Model Predict Positive
Negative	50100	50000	100
Postive	50100	100	50000

FN TP

True positive (TP): sample có **label** là **positive** và **model** phân loại là **positive**
True negative (TN): sample có **label** là **negative** và **model** phân loại là **negative**
False positive (FP): sample có **label** là **negative** và **model** phân loại là **positive**
False negative (FN): sample có **label** là **positive** và **model** phân loại là **negative**

Classification on Imbalanced Data

❖ Precision

- ❖ Ability of a model to identify only the relevant objects

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{all\ detections}$$

❖ Recall

- ❖ Ability of a model to find all the relevant cases (all ground truth bounding boxes)

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{all\ ground\ truths}$$

❖ Metric for Imbalanced Class

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{F1} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Precision: tỉ lệ model predict các positive sample đúng là positive
Recall: tỉ lệ model có thể nhận dạng được các positive sample
F1-score: Trung bình điều hòa giữa accuracy và precision

Outline

- Introduction
- Focal Loss
- Metrics
- Data Augmentation
- Experiments

Augmentation Library

❖ Imgaug Library

Có thể làm việc được với nhiều loại image data và groundtruth:

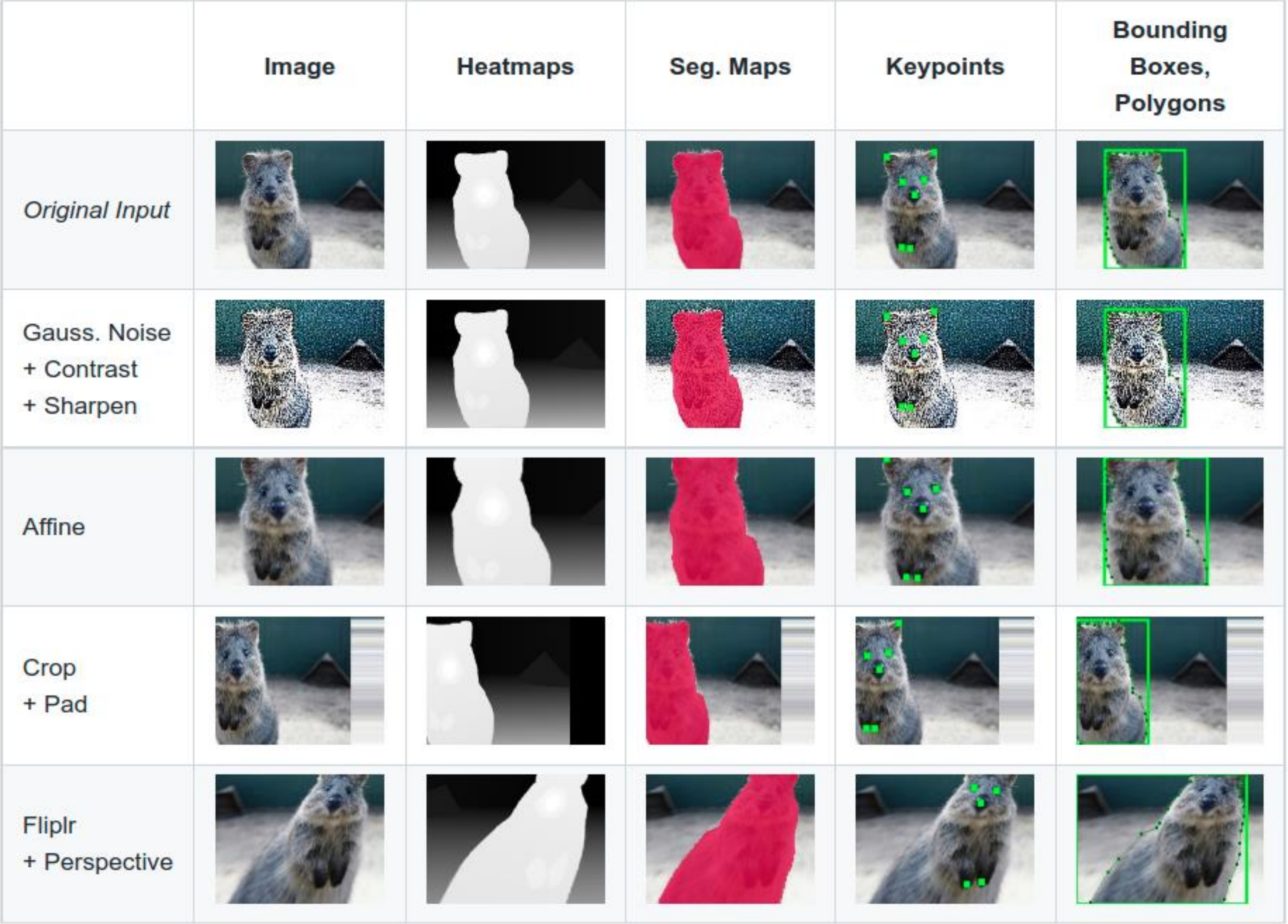
- Keypoints/Landmarks

- Bounding Boxes, Polygons

- Line Strings

- Heatmaps, Segmentation Maps

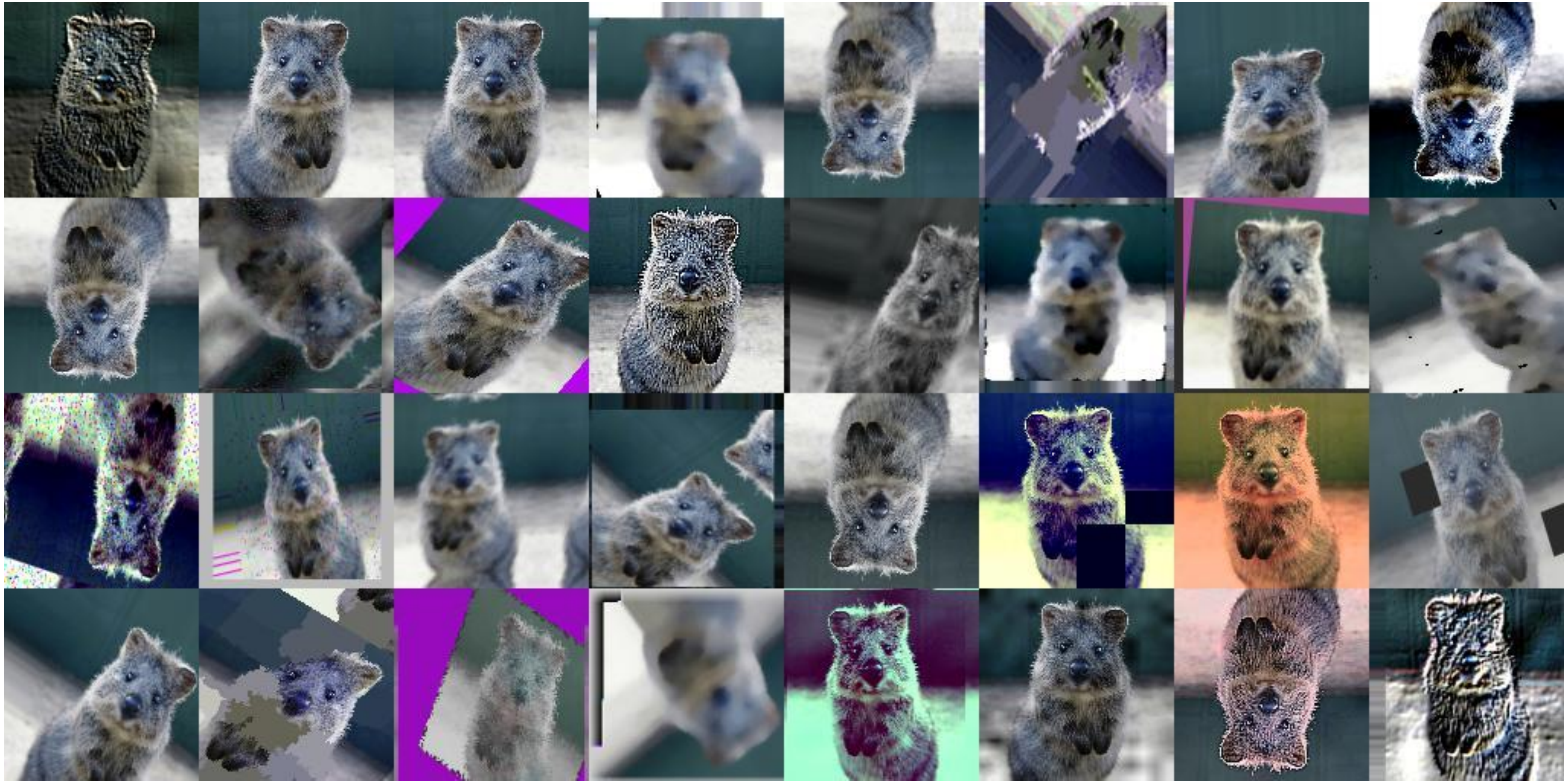
https://github.com/aleju/imgaug#documentation



Augmentation Library

❖ Image Augmentation

Ảnh gốc



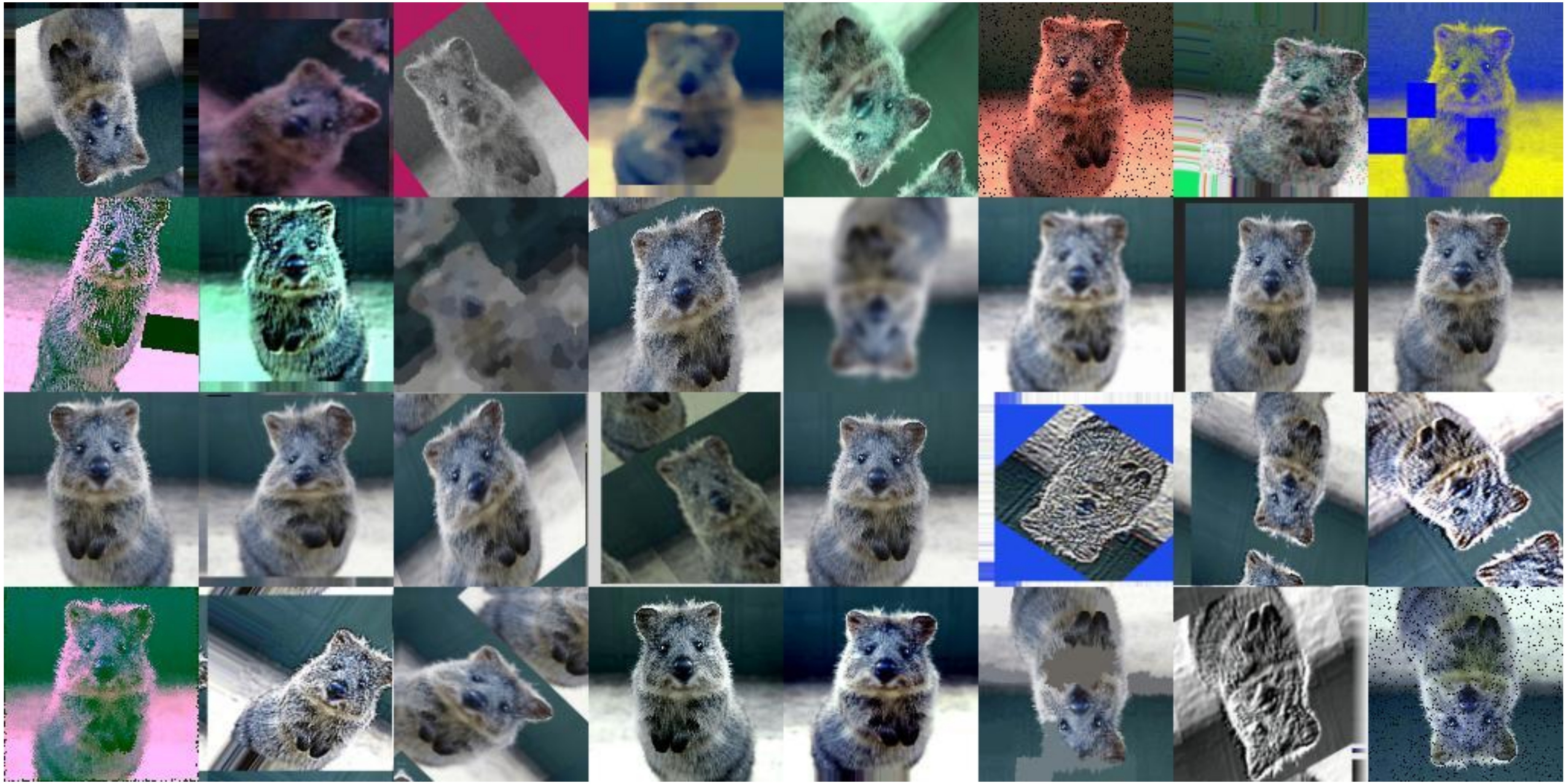
Sau Augmentation

https://github.com/aleju/imgaug#documentation

Augmentation Library

❖ Image Augmentation

Ảnh gốc



Sau Augmentation

https://github.com/aleju/imgaug#documentation

Augmentation Library

❖ Image Augmentation

- Có chức năng augmentation 1 batch ảnh (batch, height, width, channel)
- Cho phép sử dụng một chuỗi các kỹ thuật augmentation và khai báo như Sequential mode của Tensorflow

```
import imgaug.augmenters as iaa
```

import thư viện

```
# Pipeline:
# (1) Crop images from each side by 1-16px, do not resize the results
#     images back to the input size. Keep them at the cropped size.
# (2) Horizontally flip 50% of the images.
# (3) Blur images using a gaussian kernel with sigma between 0.0 and 3.0.
seq = iaa.Sequential([
    iaa.Crop(px=(1, 16), keep_size=False),
    iaa.Fliplr(0.5),
    iaa.GaussianBlur(sigma=(0, 3.0))
])
```

Khai báo các chuỗi kỹ thuật augmentations sẽ được sử dụng

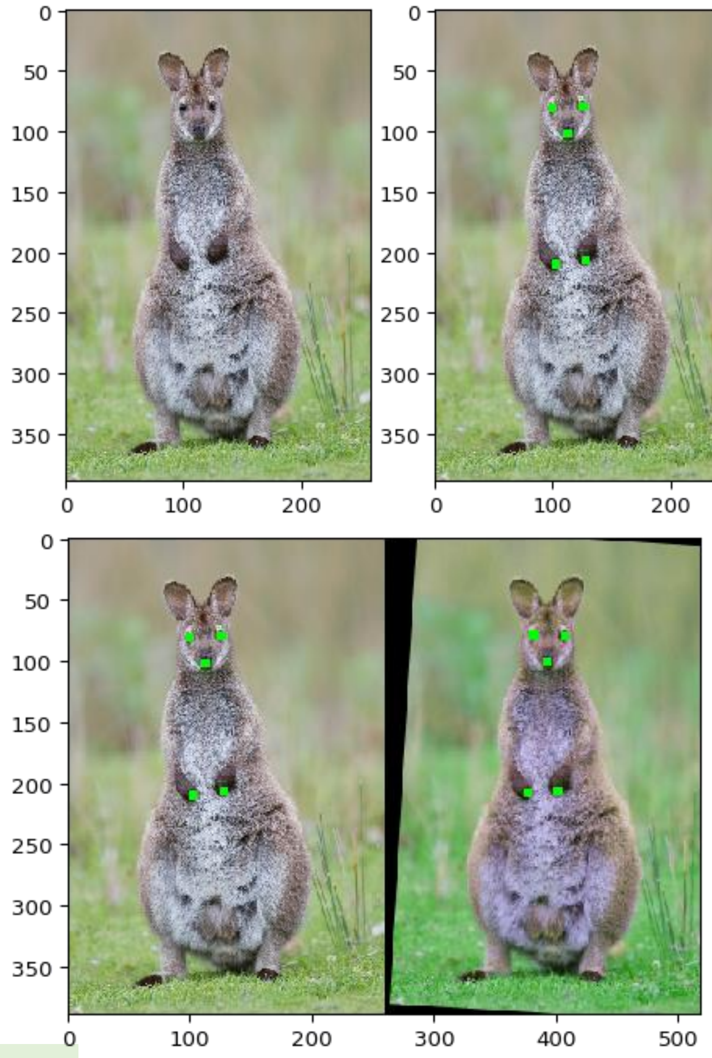
```
images_aug = seq(images=images)  # done by the library
```

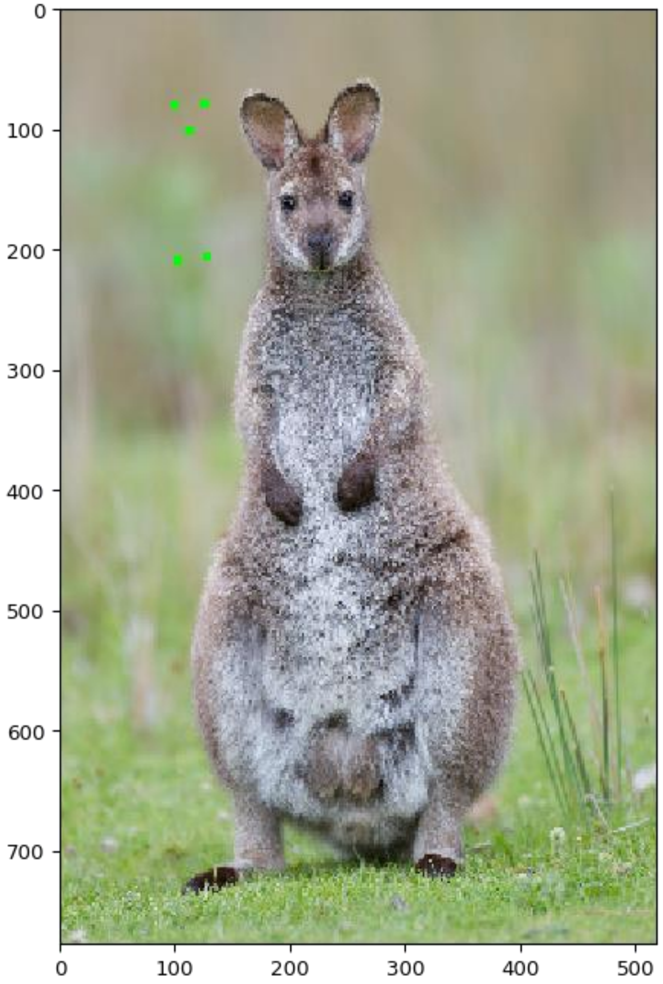
Áp dụng chuỗi các kỹ thuật augmentation cho 1 batch ảnh

https://github.com/aleju/imgaug#documentation

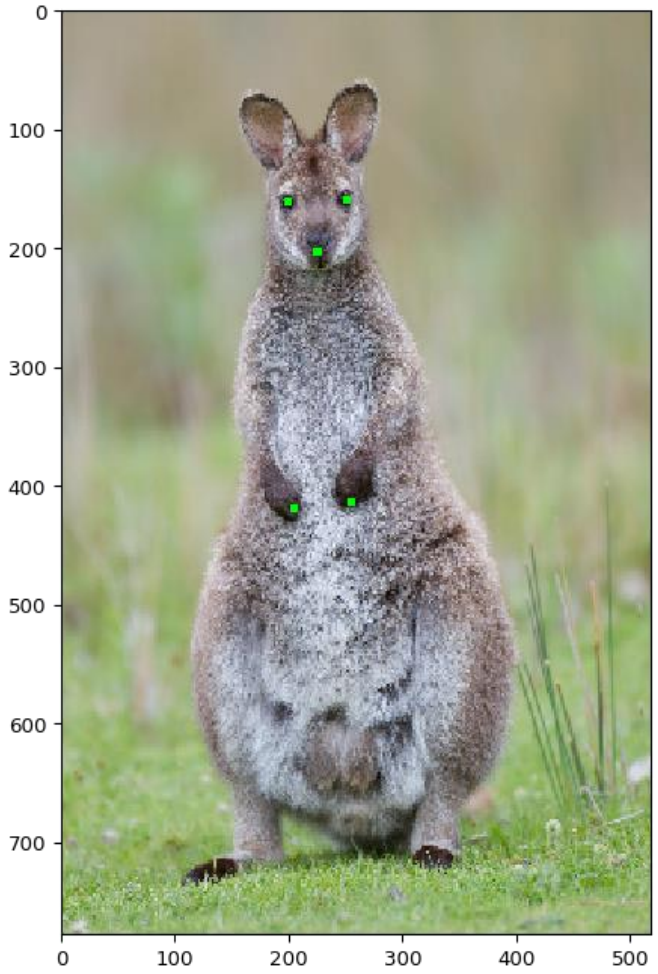
Augmentation Library

❖ Augment Keypoints/Landmarks





Resize ảnh lỗi

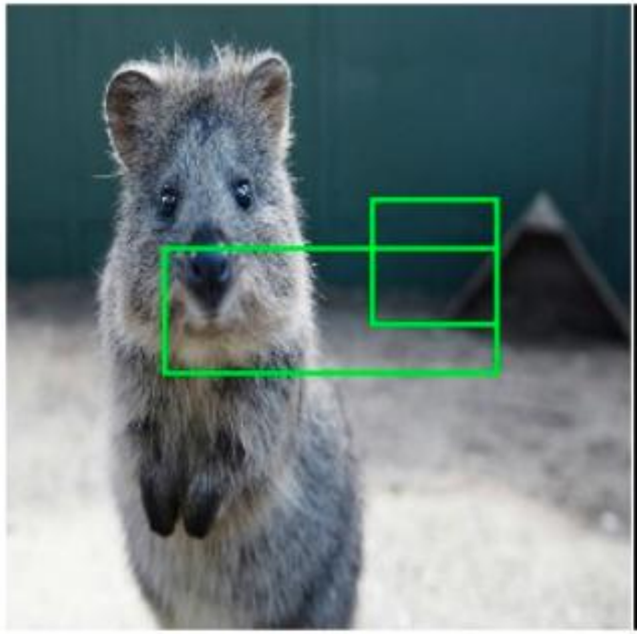


Resize ảnh đúng

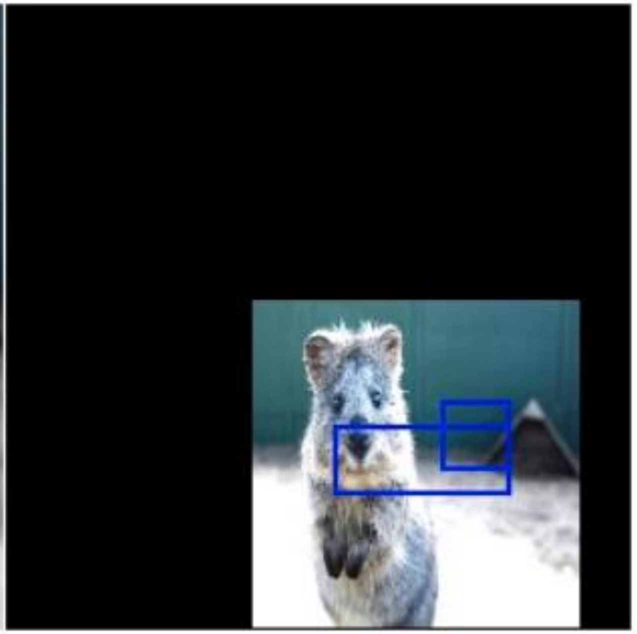
https://github.com/aleju/imgaug#documentation

Augmentation Library

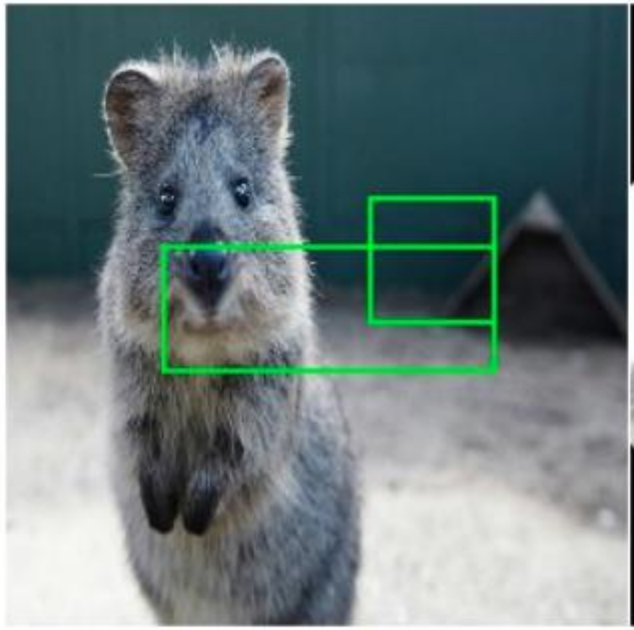
❖ Bounding Boxes



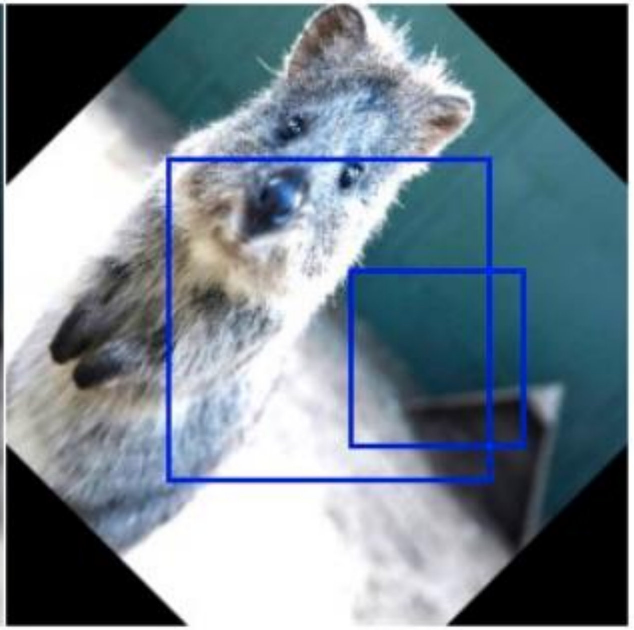
Trước
Augmentation



Sau
Augmentation



Trước
Augmentation

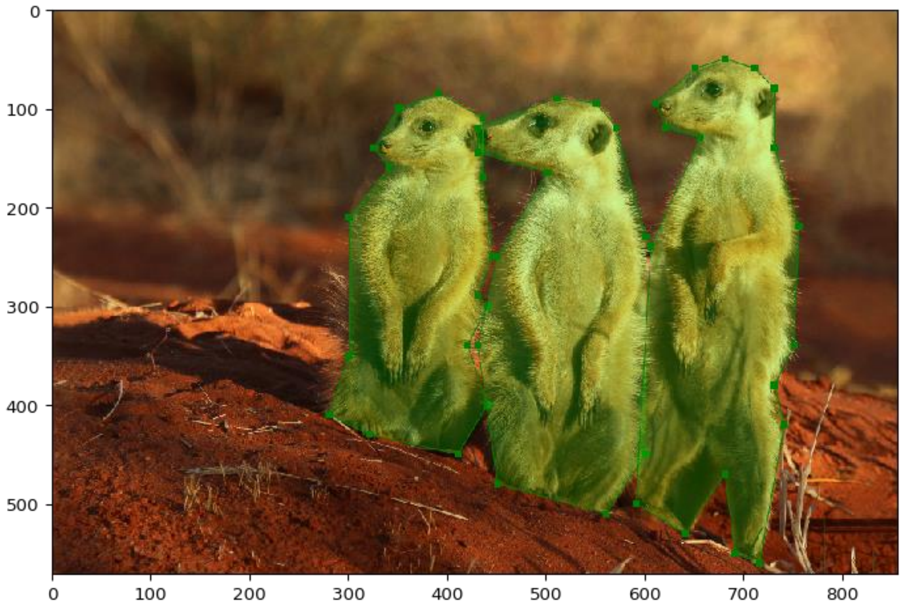


Sau
Augmentation

https://github.com/aleju/imgaug#documentation

Augmentation Library

❖ Polygon



Trước
Augmentation

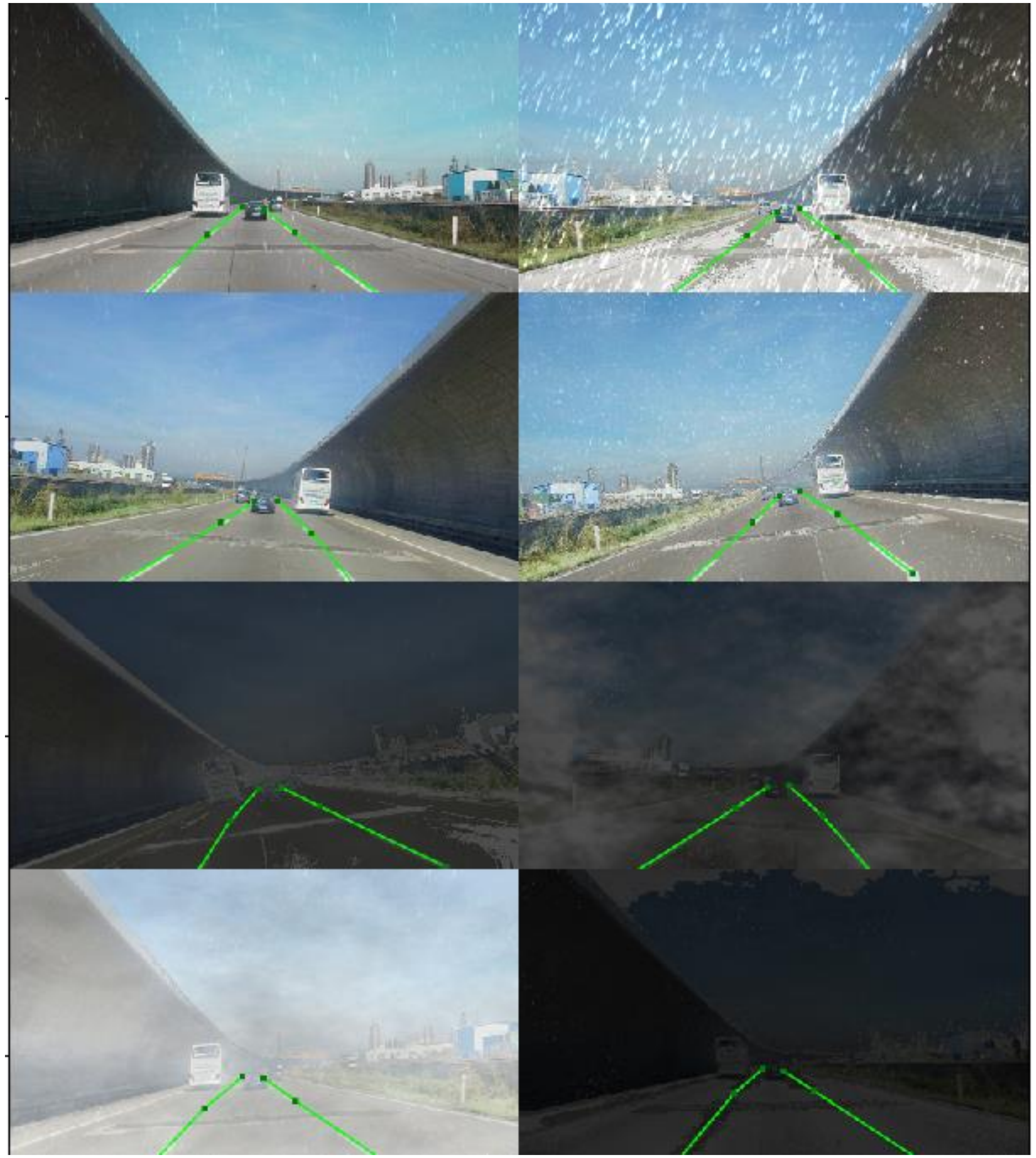
Sau Augmentation

https://github.com/aleju/imgaug#documentation

Augmentation Library

❖ Line Strings

Trước
Augmentation

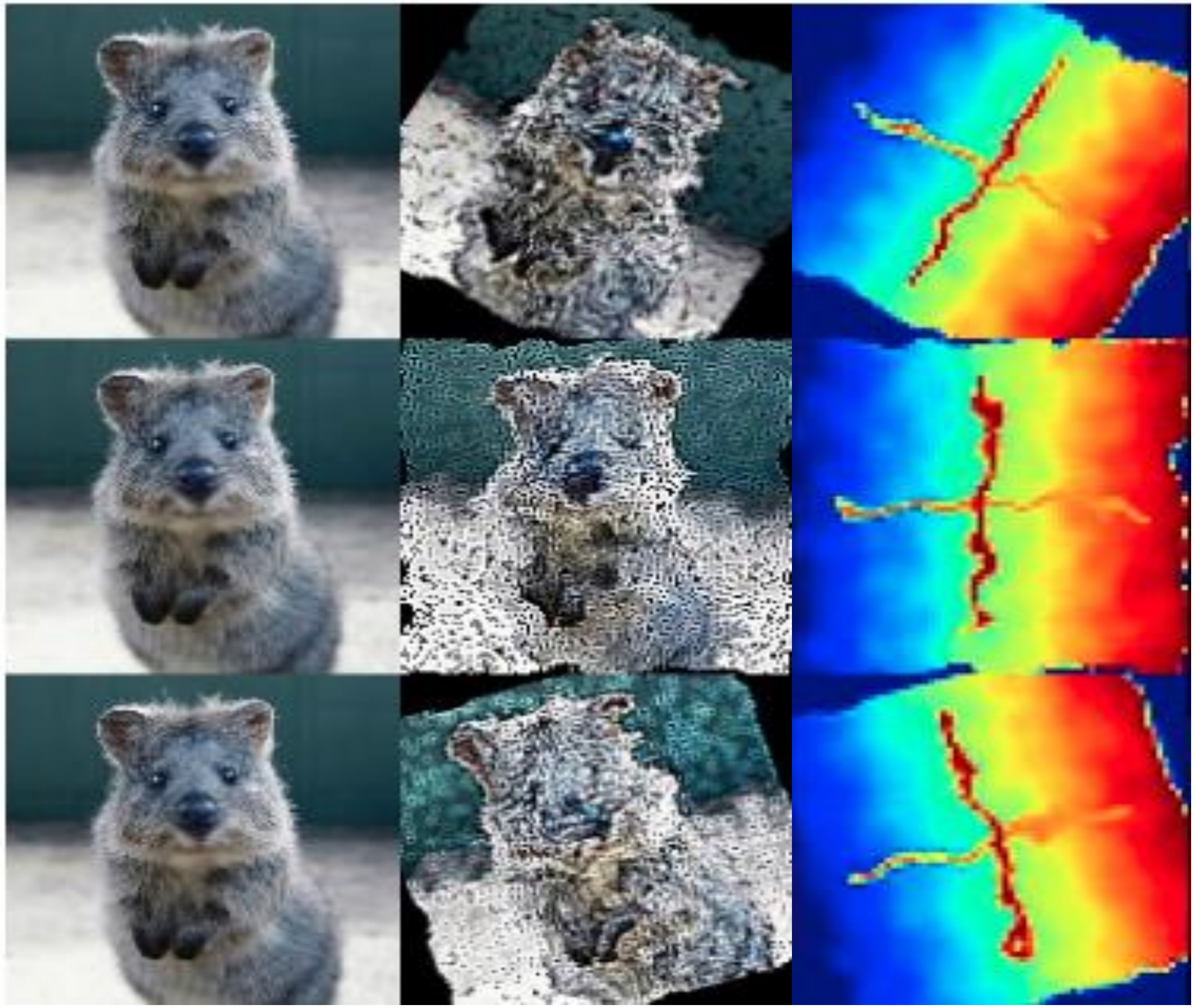


Sau Augmentation

https://github.com/aleju/imgaug#documentation

Augmentation Library

❖ Heat Map



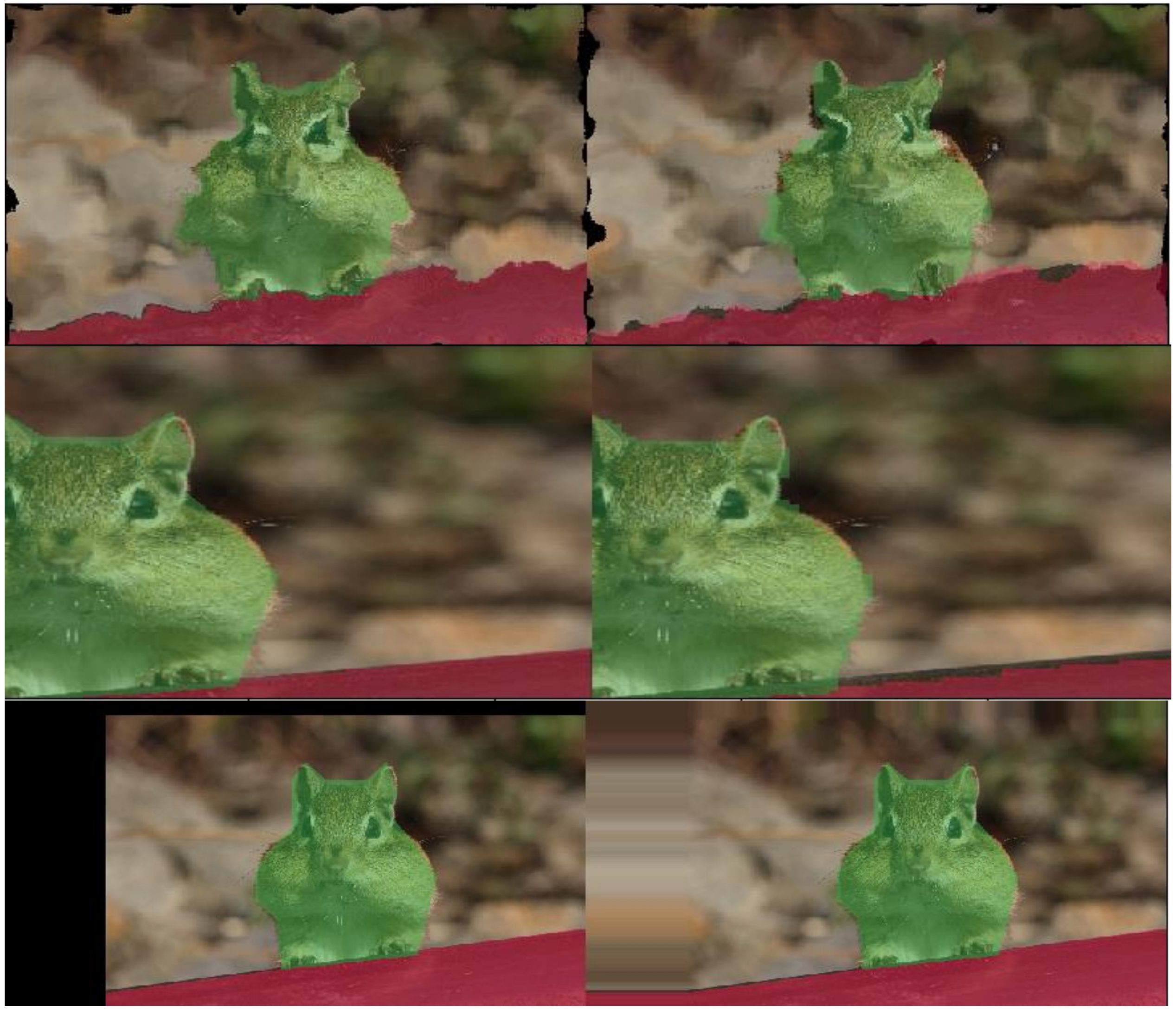
ảnh gốc | ảnh augmentation | heat map augmenation

https://github.com/aleju/imgaug#documentation

Augmentation Library

❖ Segmentation

TrướcAugmentation



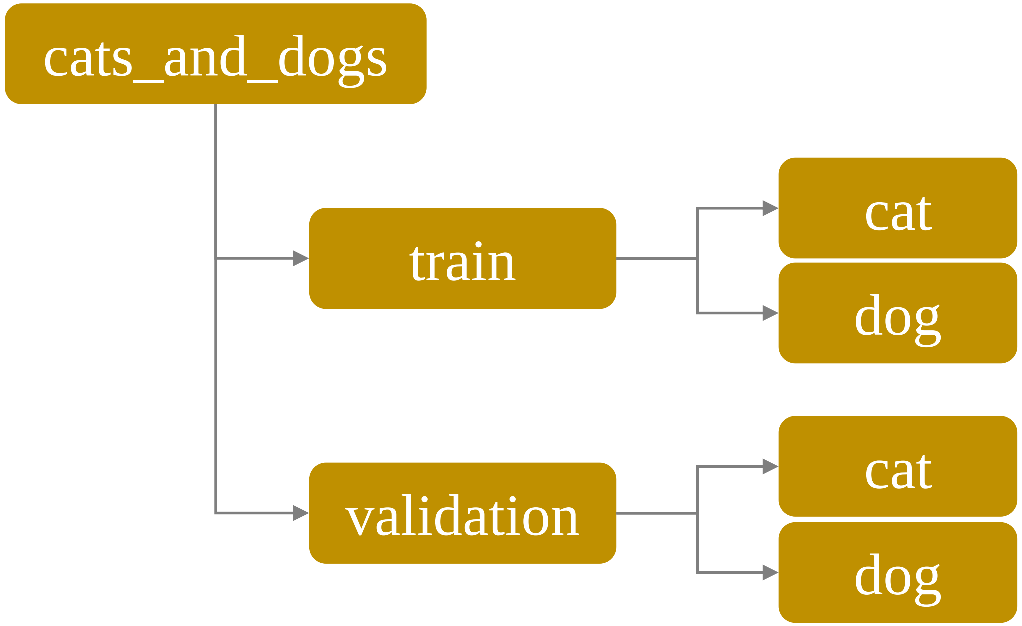
Sau Augmentation

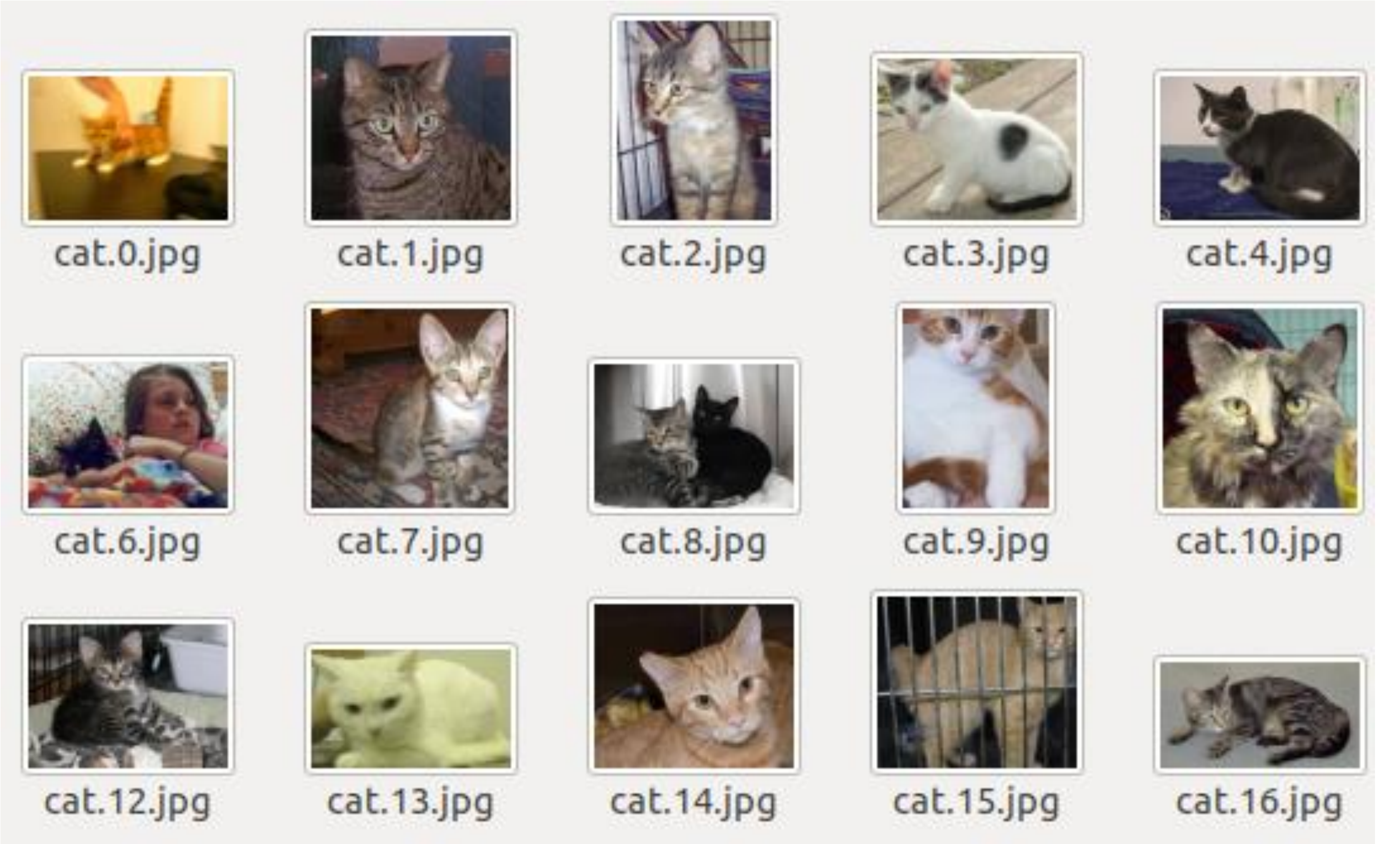
Outline

- Introduction
- Focal Loss
- Metrics
- Data Augmentation
- Experiments

Experiments

❖ Cat-Dog dataset

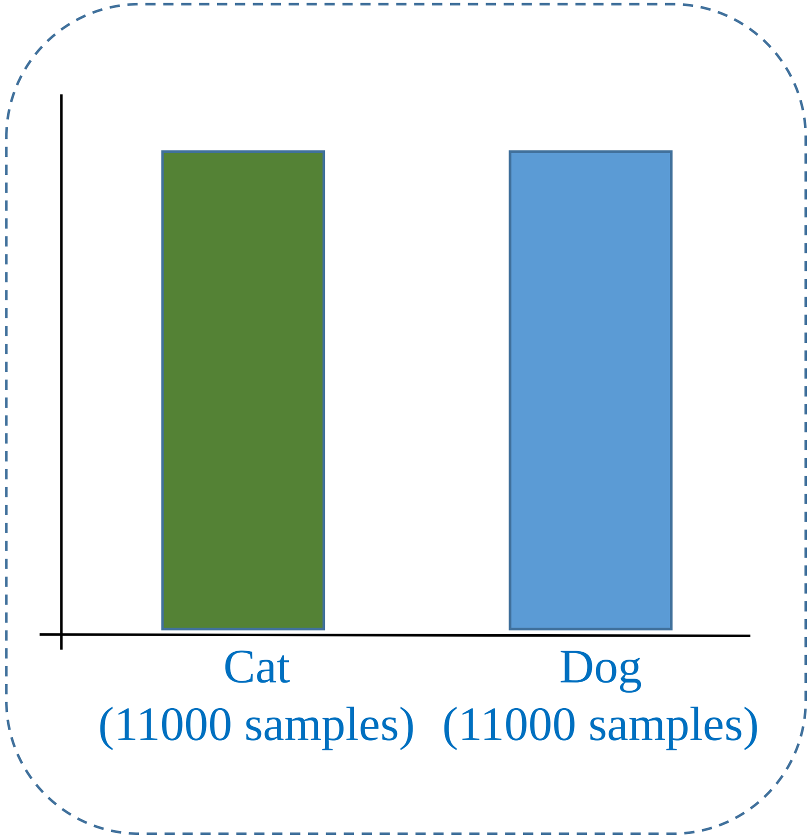




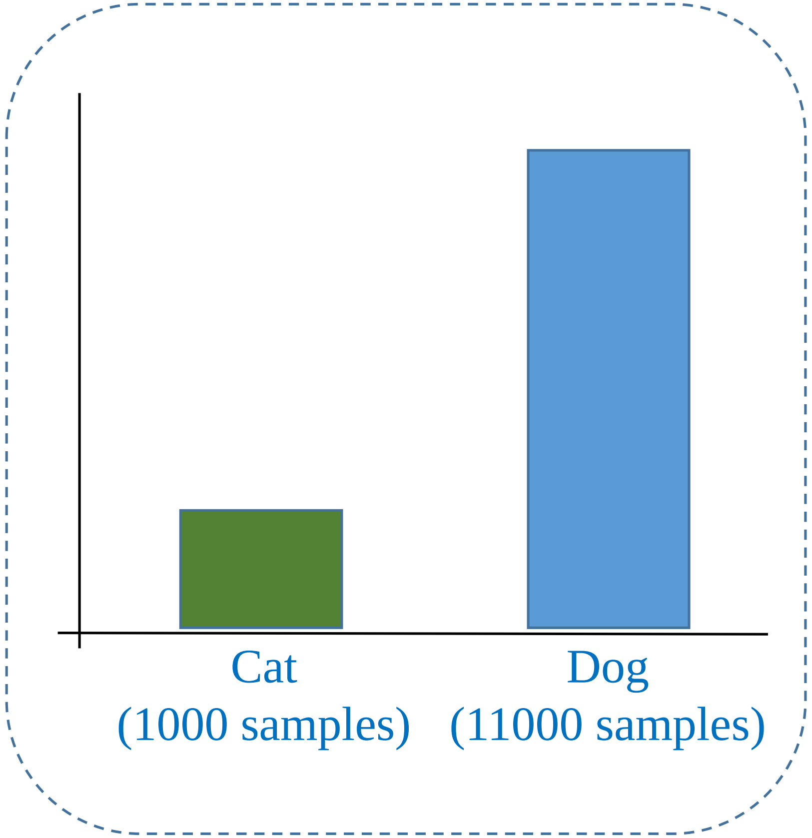
Experiments

❖ Cat-Dog dataset

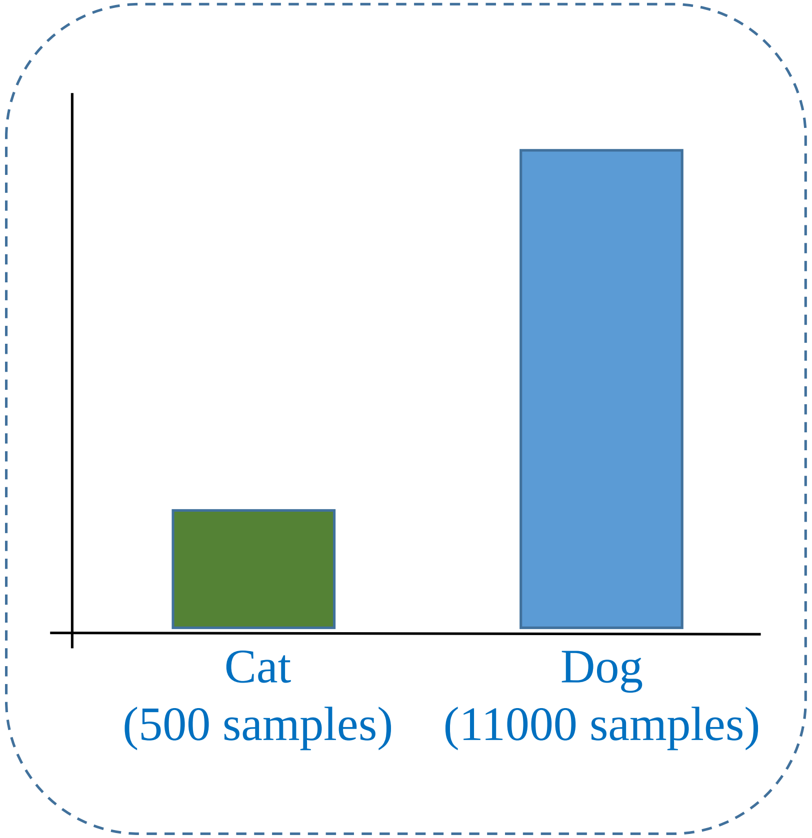
Validation data (3000 samples)



Balanced Data



Imbalanced Data 1

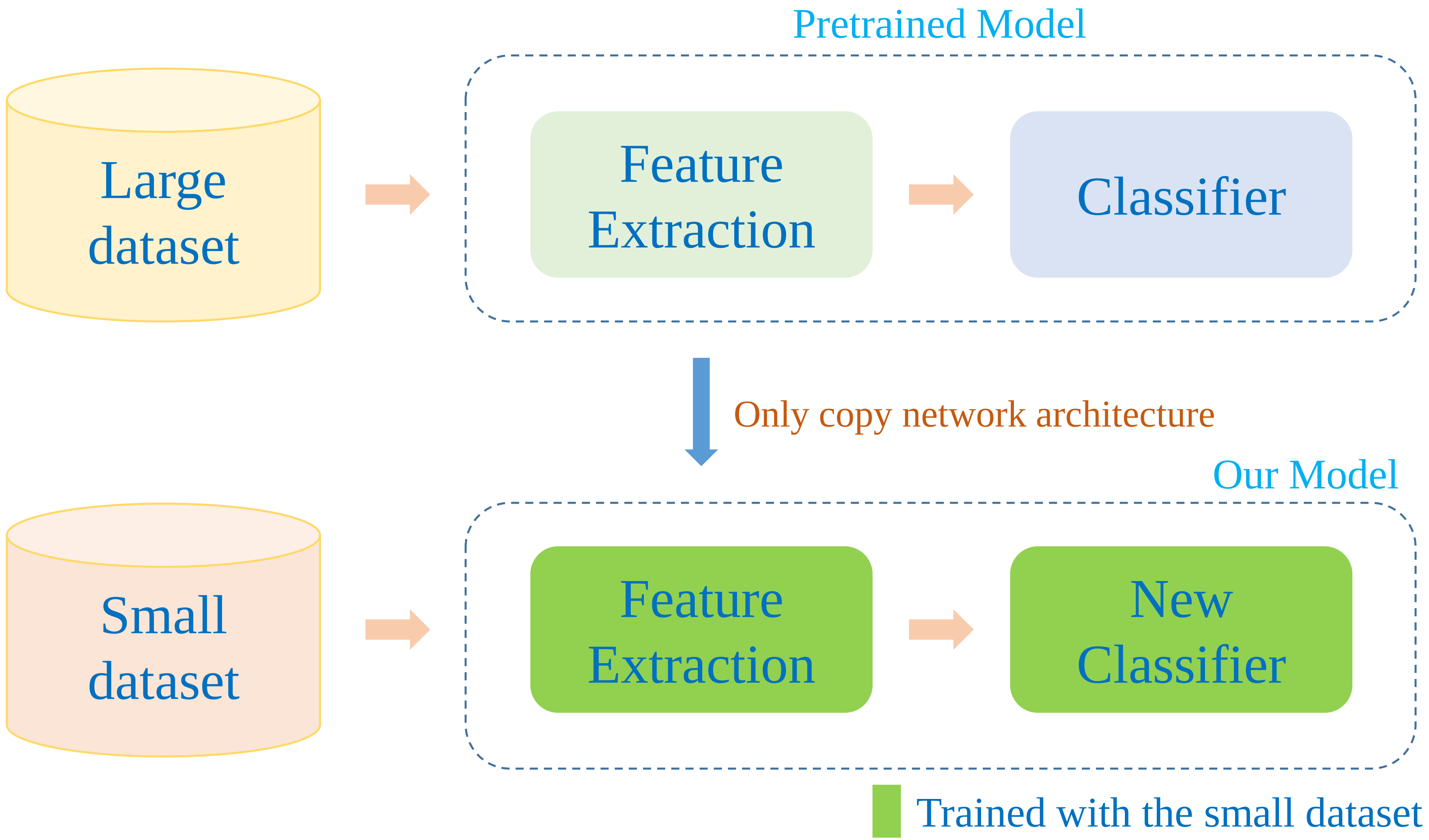


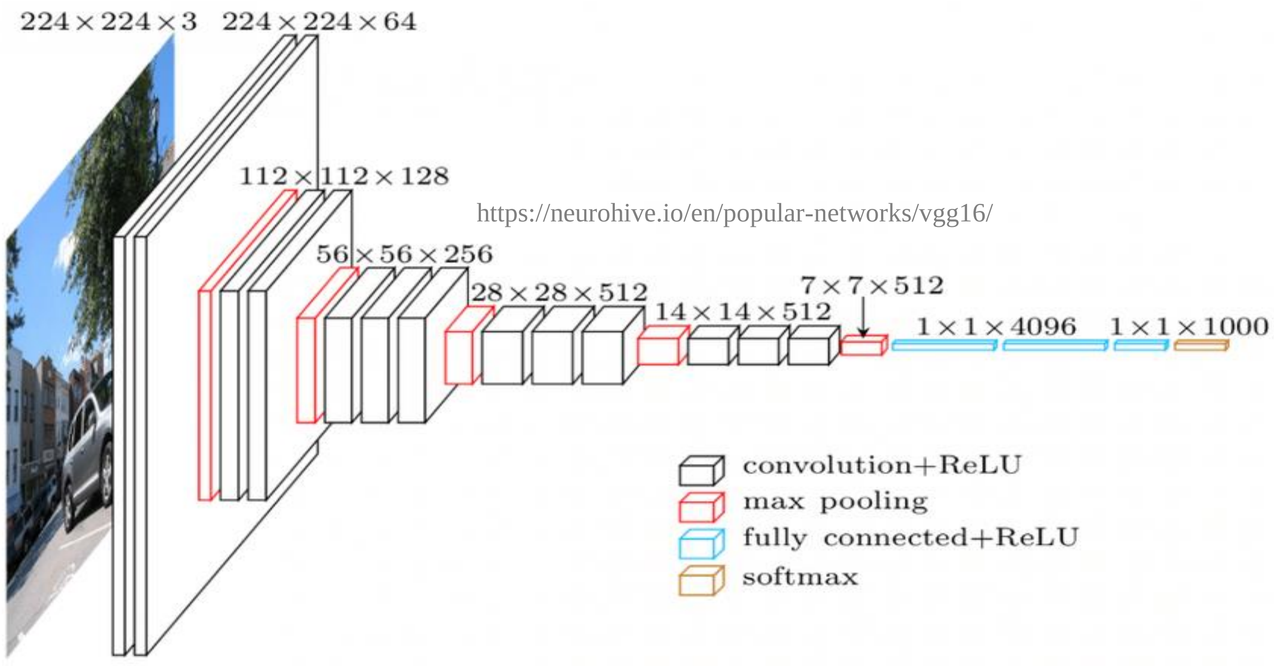
Imbalanced Data 2

Experiments

❖ Train from scratch

ImageNet dataset
(1.2 million images
of 1000 categories)





https://neurohive.io/en/popular-networks/vgg16/

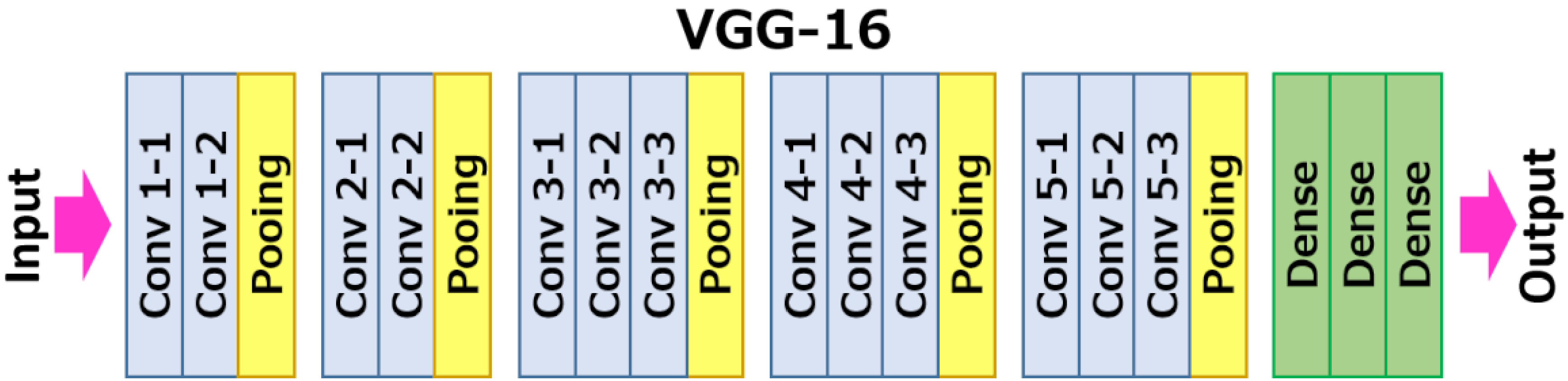
(3x3) Convolution
padding=‘same’
stride=1 + ReLU

(2x2) max pooling

Dense Layer
+ ReLU
(4096)

Dense Layer
+ Softmax
(1000)

VGG16



Experiments

❖ Cat-Dog dataset

❖ Train from scratch

```
PATH = 'cats_and_dogs_small/'

train_dir = os.path.join(PATH, 'train')
validation_dir = os.path.join(PATH, 'validation')

BATCH_SIZE = 256
IMG_SIZE = (160, 160)
BUFFER_SIZE = BATCH_SIZE*5

train_dataset = image_dataset_from_directory(train_dir,
                                             shuffle=True,
                                             batch_size=BATCH_SIZE,
                                             image_size=IMG_SIZE)
```

```
Found 2000 files belonging to 2 classes.
```

```
validation_dataset = image_dataset_from_directory(validation_dir,
                                                  shuffle=True,
                                                  batch_size=BATCH_SIZE,
                                                  image_size=IMG_SIZE)
```

```
Found 1000 files belonging to 2 classes.
```

Experiments

- ❖ **Cat-Dog dataset**
  - ❖ **Train from scratch**

```
# top=False
model = tf.keras.applications.VGG16(input_shape=(160,160,3),
                                    include_top=False,
                                    weights=None)
model.summary()
```

```
Model: "vgg16"
_________________________________________________________________
Layer (type)                Output Shape              Param #
=================================================================
input_1 (InputLayer)        [(None, 160, 160, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)       (None, 160, 160, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)       (None, 160, 160, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)  (None, 80, 80, 64)        0
_________________________________________________________________
block2_conv1 (Conv2D)       (None, 80, 80, 128)       73856
_________________________________________________________________
block2_conv2 (Conv2D)       (None, 80, 80, 128)       147584
_________________________________________________________________
block2_pool (MaxPooling2D)  (None, 40, 40, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)       (None, 40, 40, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)       (None, 40, 40, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)       (None, 40, 40, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)  (None, 20, 20, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)       (None, 20, 20, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)       (None, 20, 20, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)       (None, 20, 20, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)  (None, 10, 10, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)       (None, 10, 10, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)       (None, 10, 10, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)       (None, 10, 10, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)  (None, 5, 5, 512)         0
=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
_________________________________________________________________
```

Experiments

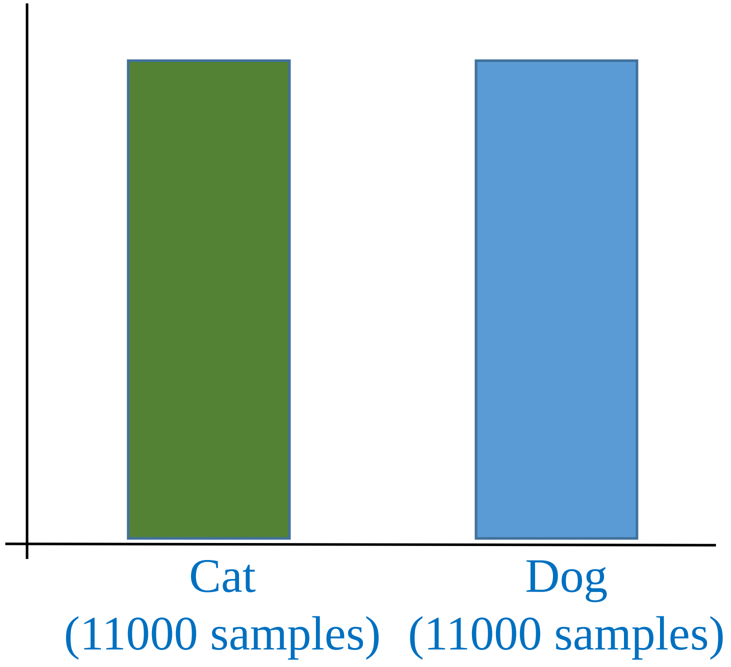
❖ Cat-Dog dataset

❖ Train from scratch

```
# process data
data_augmentation = tf.keras.Sequential([
    tf.keras.layers.experimental.preprocessing.RandomFlip('horizontal'),
    tf.keras.layers.experimental.preprocessing.RandomRotation(0.2),
    tf.keras.layers.experimental.preprocessing.Rescaling(1./127.5, offset= -1)
])

# flattening
flatten = tf.keras.layers.Flatten()

# final layer
prediction_layer = tf.keras.layers.Dense(1)

# construct a new network
inputs = tf.keras.Input(shape=(160, 160, 3))
x = data_augmentation(inputs)
x = base_model(x)
x = flatten(x)
outputs = prediction_layer(x)
model = tf.keras.Model(inputs, outputs)
```

Experiments

❖ Cat-Dog dataset



Balanced Data

Validation data (3000 samples)

Balanced Loss	Correct prediction
$L_b = L_c + L_d$	$\#_{cat} = 1445$ $\#_{dog} = 1436$ $F_1 = 0.96$

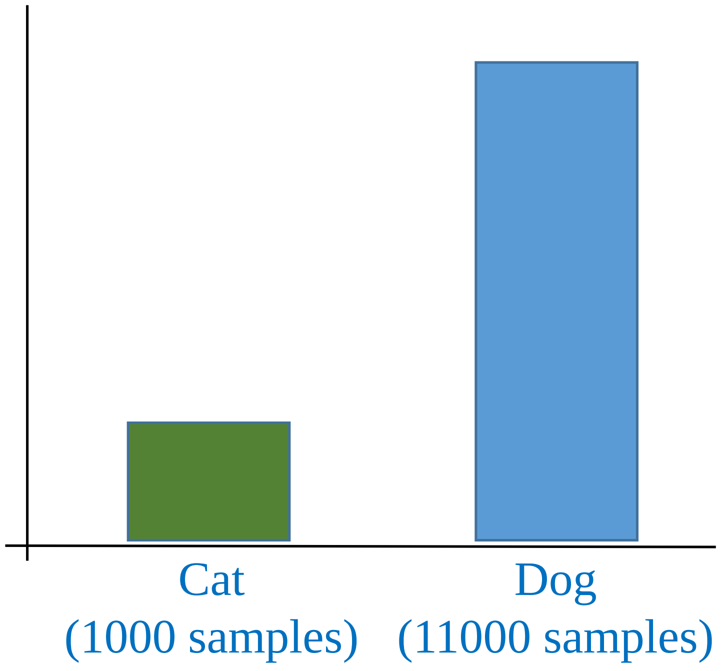
Imbalanced Loss 1	Correct prediction
$L_b = L_c + 100 \times L_d$	$\#_{cat} = 1076$ $\#_{dog} = 1499$ $F_1 = 0.835$

Imbalanced Loss 2	Correct prediction
$L_b = L_c + 1000 \times L_d$	$\#_{cat} = 670$ $\#_{dog} = 1498$ $F_1 = 0.617$

Experiments

❖ Cat-Dog dataset

Validation data (3000 samples)



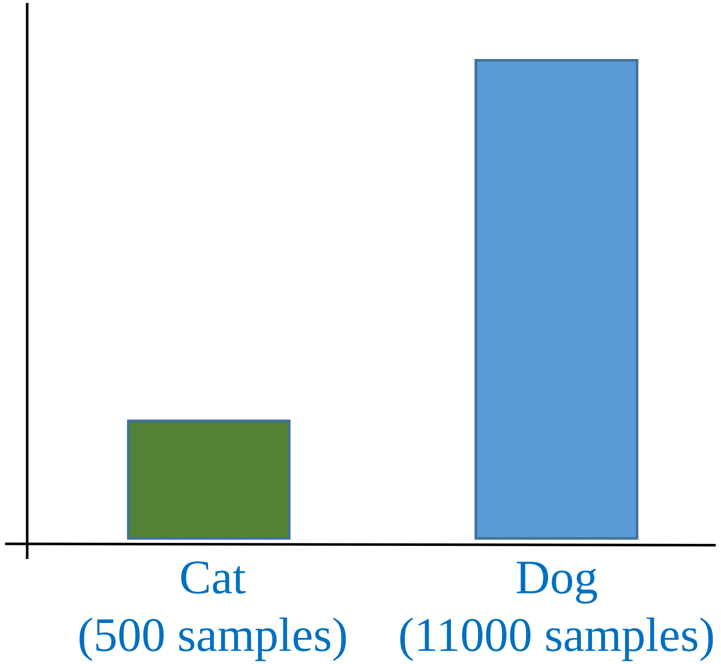
Imbalanced Data 1

Balanced Loss	Correct prediction
$L_b = L_c + L_d$	$\#_{cat} = 1082$ $\#_{dog} = 1483$ $F_1 = 0.833$

Imbalanced Loss	Correct prediction
$L_b = 6 \times L_c + 0.55 \times L_d$	$\#_{cat} = 1163$ $\#_{dog} = 1379$ $F_1 = 0.835$

Experiments

❖ Cat-Dog dataset



Imbalanced Data 2

Validation data (3000 samples)

Correct prediction

$$\#_{cat} = 821$$
$$\#_{dog} = 1489$$

n=3000	Predicted: NEGATIVE	Predicted: POSITIVE	
Actual: NEGATIVE	TN=1489	FP=11	1500
Actual: POSITIVE	FN=679	TP=821	1500
	2168	832	

$$\text{Recal} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{821}{821 + 679} \approx 0.547$$

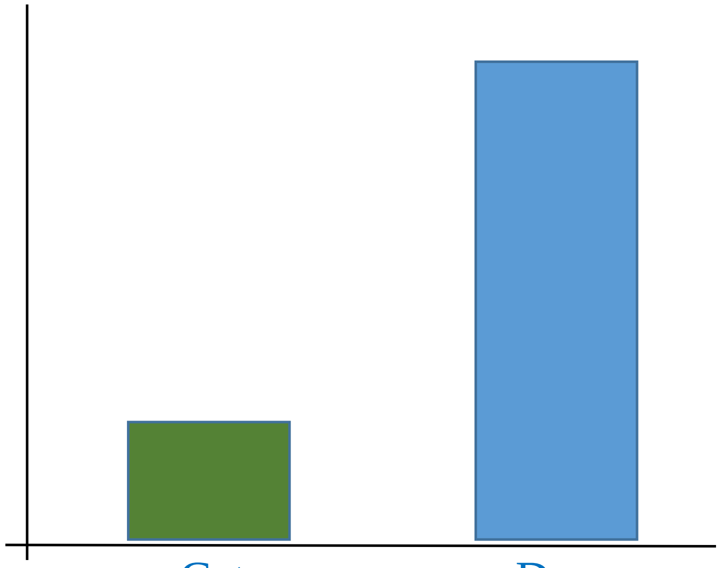
$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{821}{821 + 11} \approx 0.987$$

$$\text{F1} = \frac{2 * \text{Recal} * \text{Precision}}{\text{Recal} + \text{Precision}} = \frac{2 * 0.547 * 0.987}{0.547 * 0.987} \approx 0.704$$

Experiments

❖ Cat-Dog dataset

Validation data (3000 samples)



Imbalanced Data 2

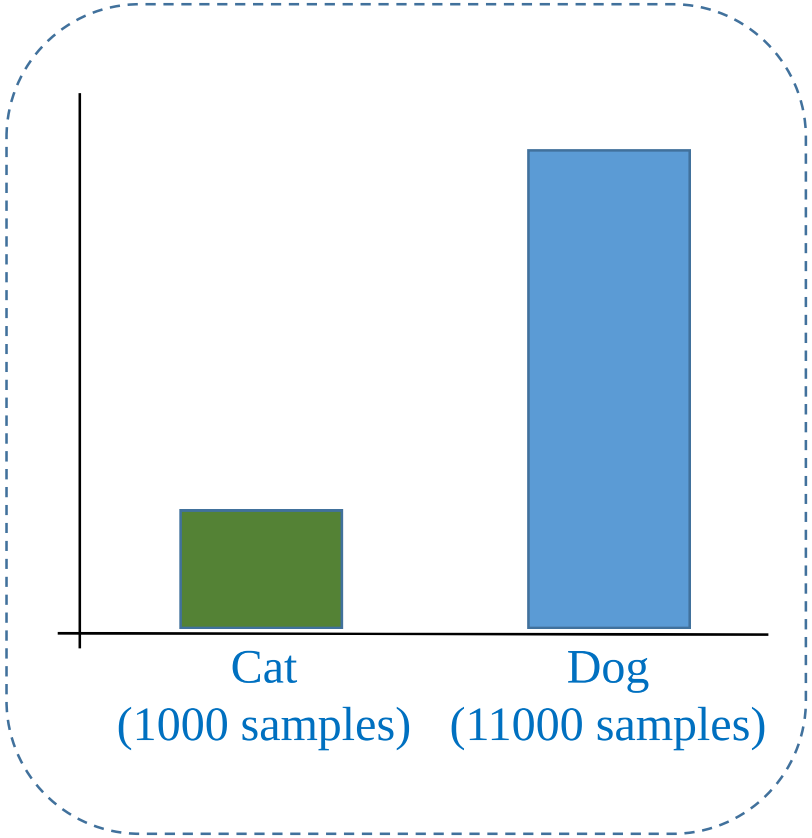
Balanced Loss	Correct prediction
$L_b = L_c + L_d$	$\#_{cat} = 821$ $\#_{dog} = 1489$ $F_1 = 0.704$

Imbalanced Loss	Correct prediction
$L_b = 11.5 \times L_c + 0.52 \times L_d$	$\#_{cat} = 1123$ $\#_{dog} = 1309$ $F_1 = 0.798$

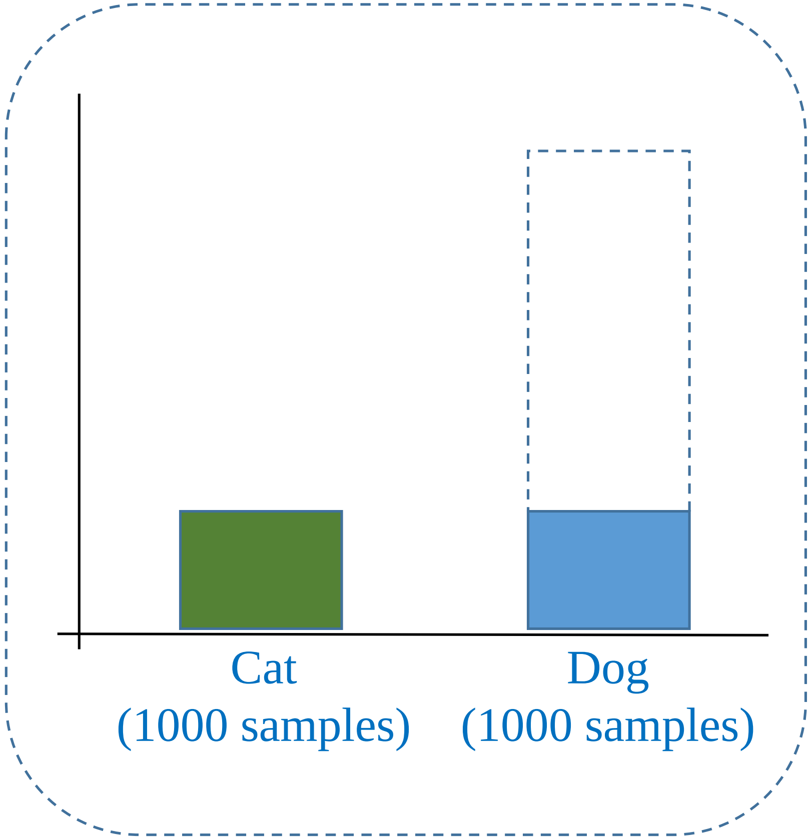
Experiments

❖ Cat-Dog dataset: Undersampling

Validation data (3000 samples)



Imbalanced Data 1



Undersampling Data

Imbalanced Data 1

$\#_{cat} = 1082$

$\#_{dog} = 1483$

$F_1 = 0.833$

Undersampling Data

$\#_{cat} = 1282$

$\#_{dog} = 1017$

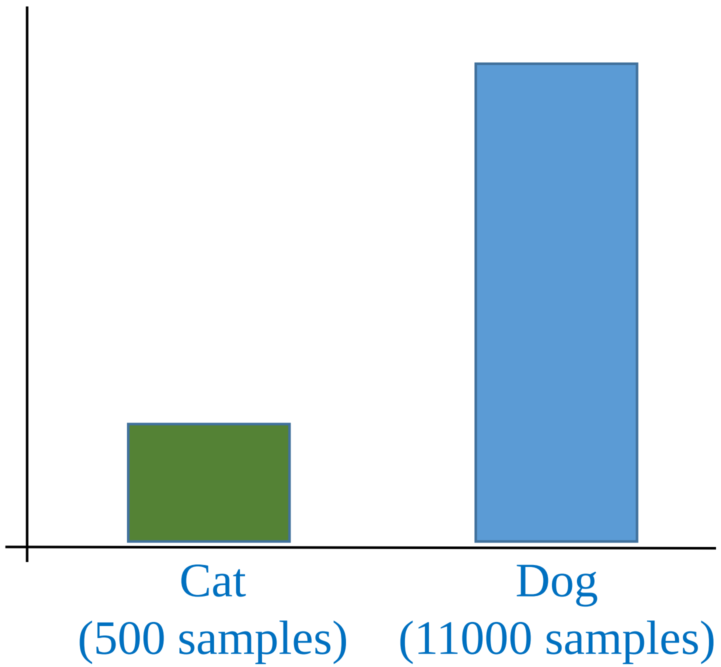
$F_1 = 0.785$!

AI VIETNAM
AI Course

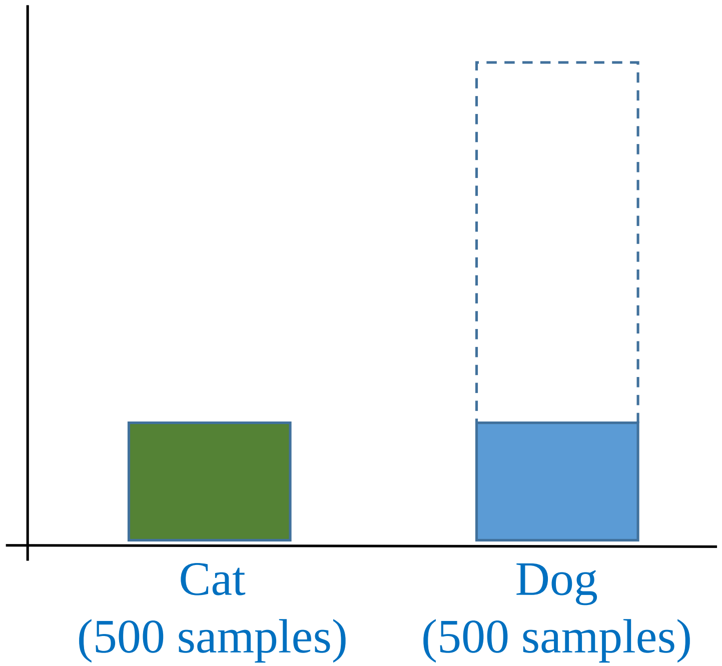
Experiments

❖ Cat-Dog dataset: Undersampling

Validation data (3000 samples)



Imbalanced Data 2



Undersampling Data

Imbalanced Data 2

$$\#_{cat} = 821$$
$$\#_{dog} = 1489$$
$$F_1 = 0.704$$

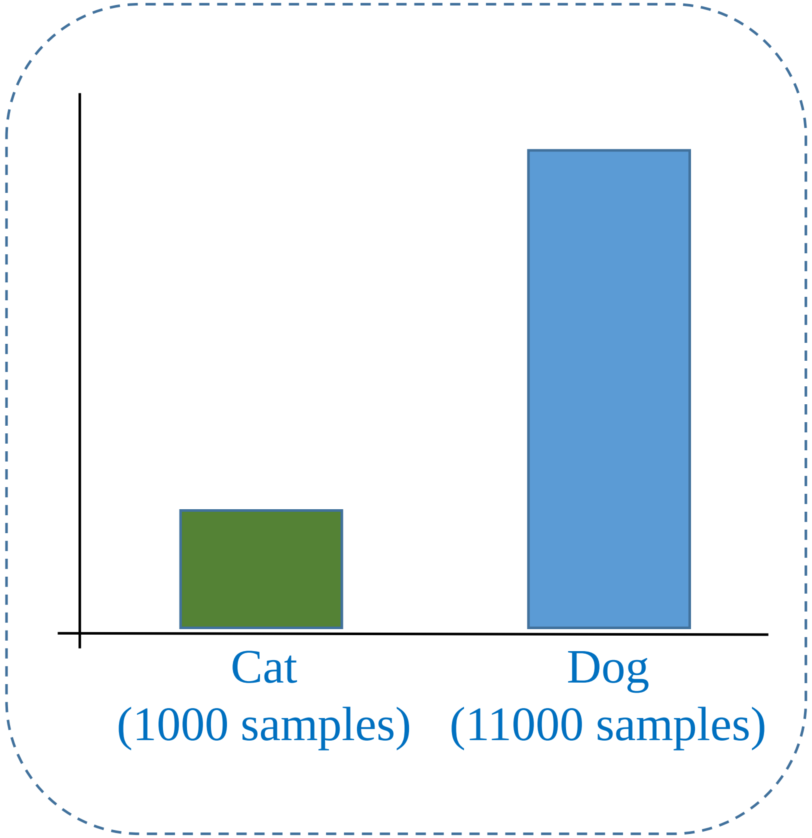
Undersampling Data

$$\#_{cat} = 1335$$
$$\#_{dog} = 734$$
$$F_1 = 0.741$$

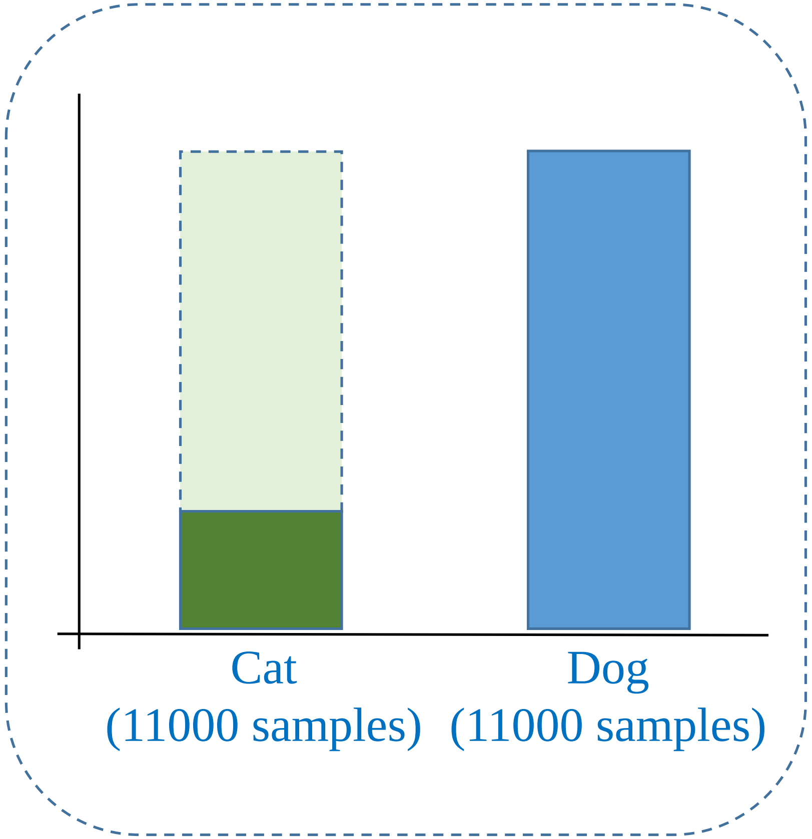
Experiments

❖ Cat-Dog dataset: Oversampling

Validation data (3000 samples)



Imbalanced Data 1



Oversampling Data

Imbalanced Data 1

$\#_{cat} = 1082$

$\#_{dog} = 1483$

$F_1 = 0.833$

Oversampling Data

$\#_{cat} = 1167$

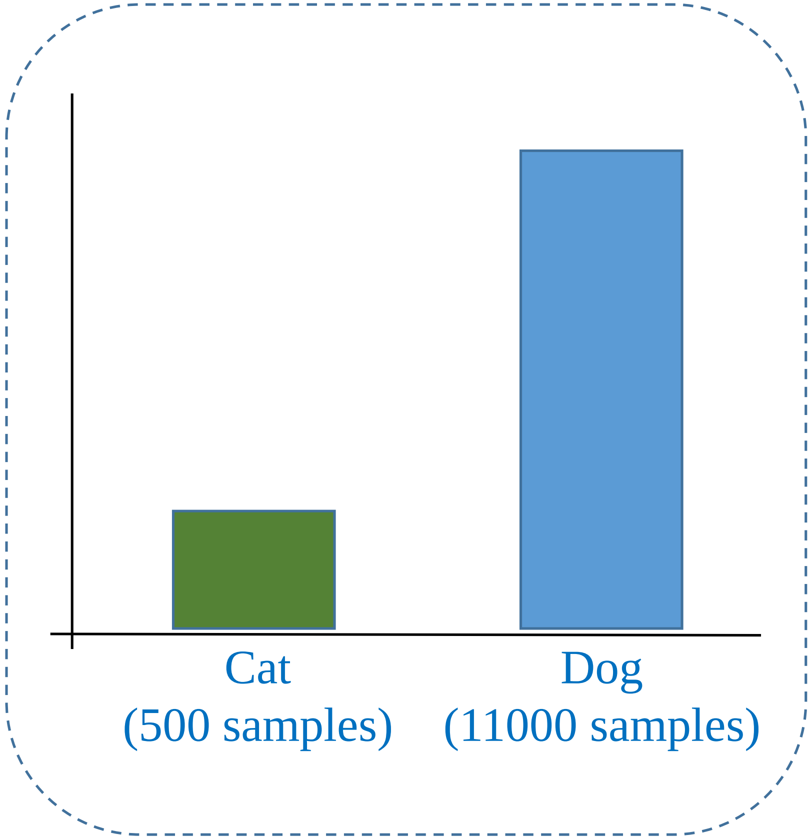
$\#_{dog} = 1438$

$F_1 = 0.855$

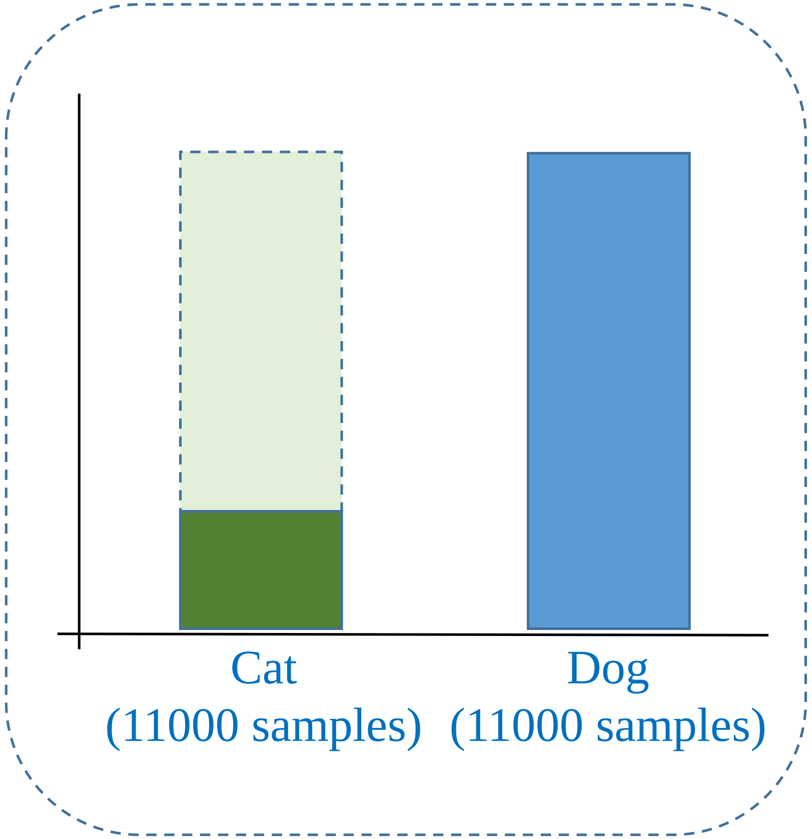
Experiments

❖ Cat-Dog dataset: Oversampling

Validation data (3000 samples)



Imbalanced Data 2



Oversampling Data

Imbalanced Data 2

$\#_{cat} = 821$

$\#_{dog} = 1489$

$F_1 = 0.704$

Oversampling Data

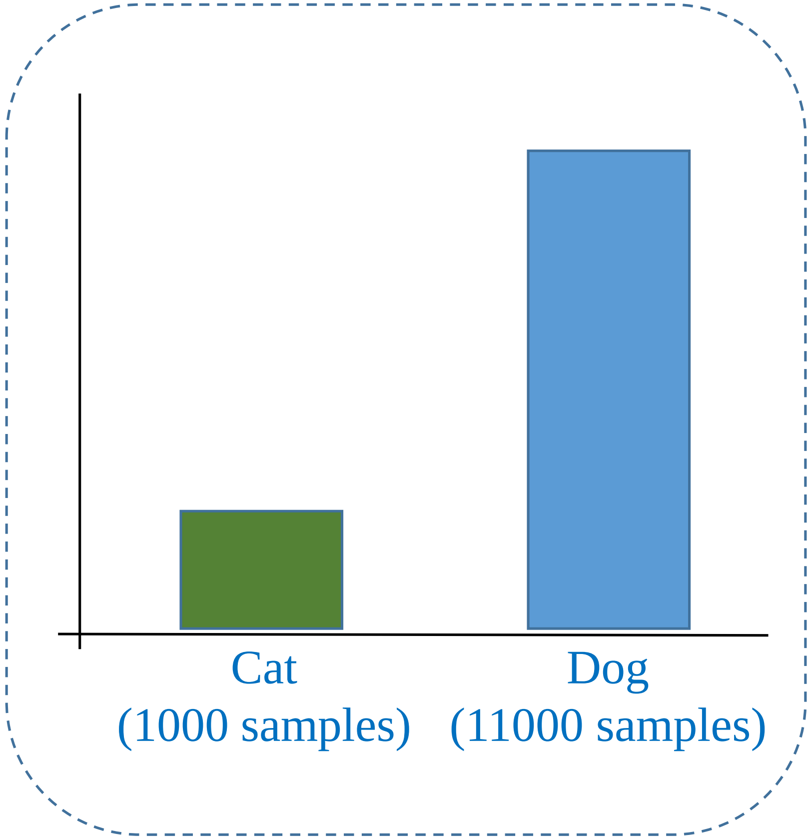
$\#_{cat} = 1159$

$\#_{dog} = 1386$

$F_1 = 0.836$

Experiments

❖ Cat-Dog dataset: Focal loss



Imbalanced Data 1

Normal crossentropy

$\#_{cat} = 1082$
$\#_{dog} = 1483$
$F_1 = 0.833$

Focal loss

$\#_{cat} = 1210$
$\#_{dog} = 1447$
$F_1 = 0.876$