Practices in visual computing 1

Lab7: Image Classification 2

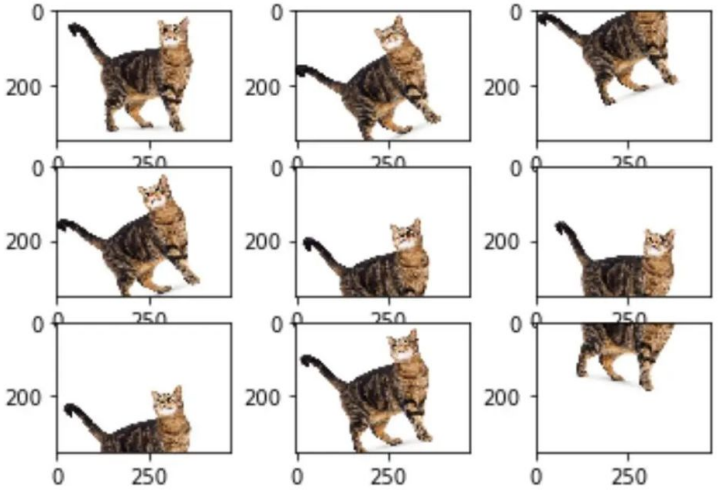
Simon Fraser University
Fall 2024

What is Data Augmentation?

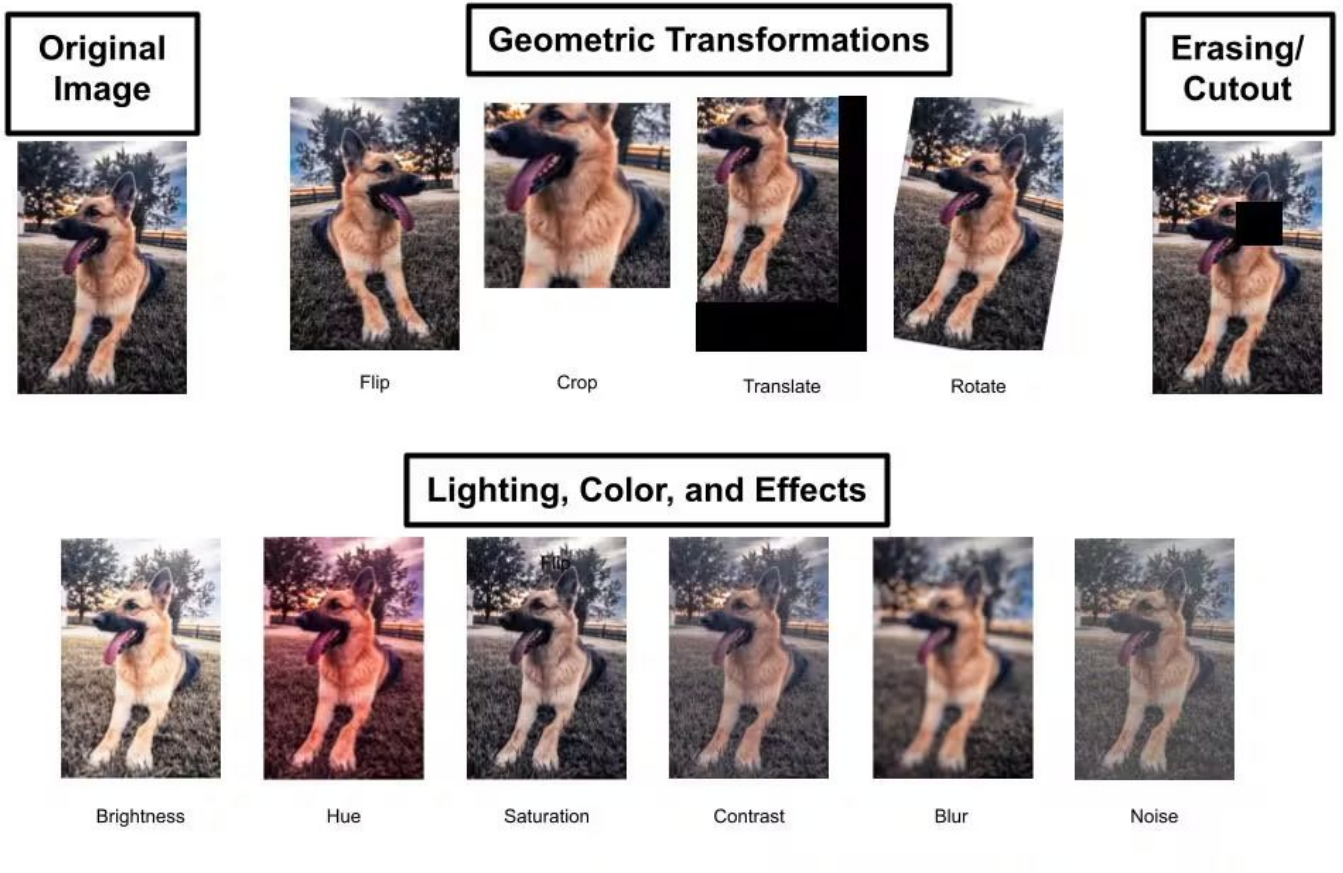
Data Augmentation is a technique to create more training data artificially.

Helps prevent overfitting by exposing models to variations of images.

Alters images in unique ways to improve model robustness.



Data Augmentation



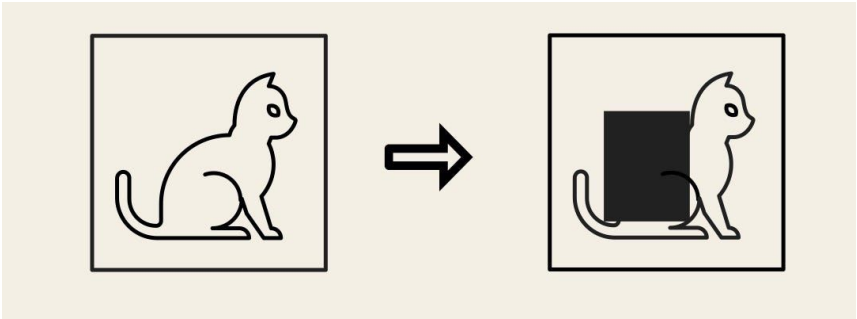
Cutout Augmentation

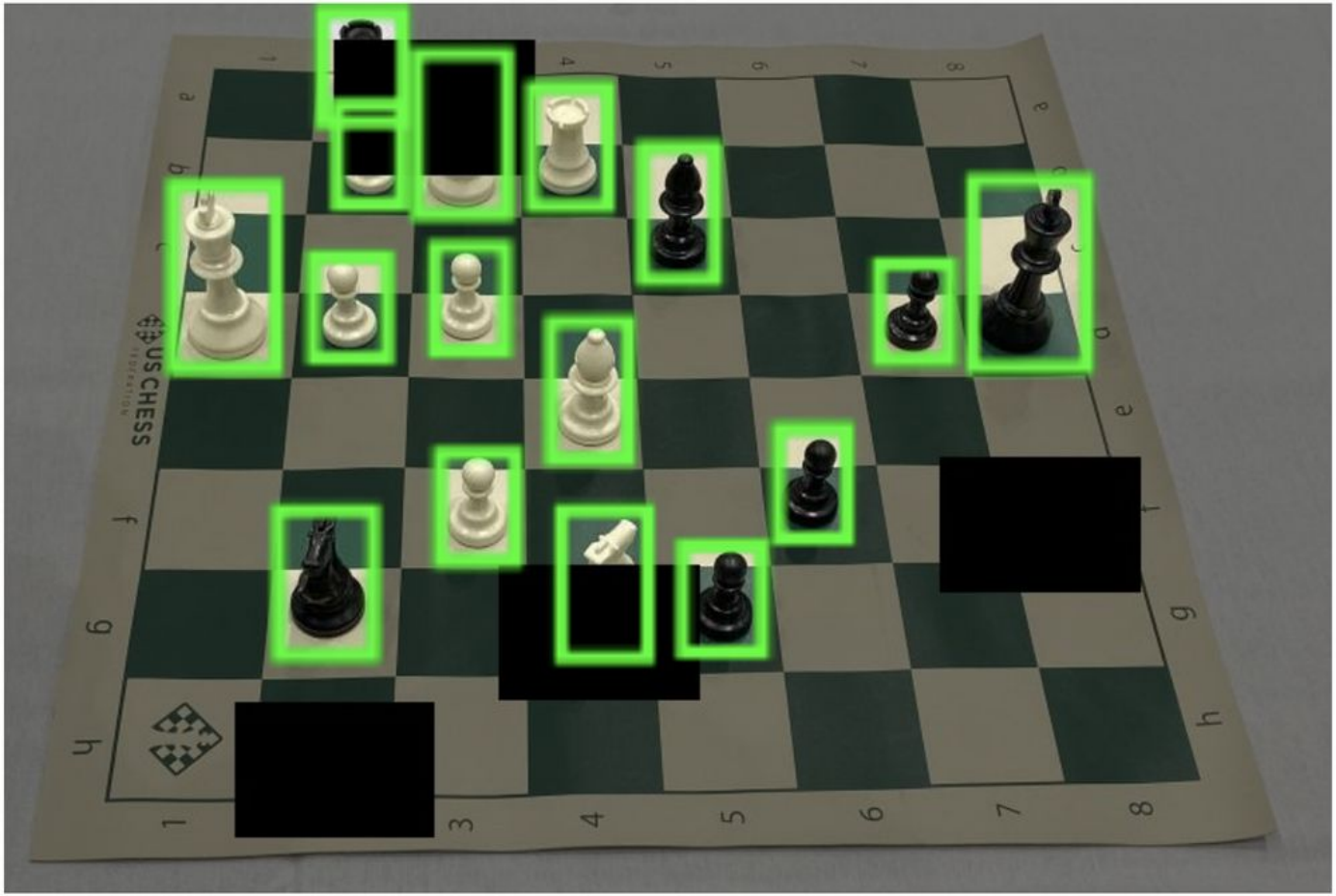
Masking Random Sections

Purpose: To encourage the model to focus on other parts of the object rather than specific details.

How It Works: A mask is randomly placed on the image, blocking out portions.

Key Benefit: Helps models generalize better by not relying on specific features.




USCHESS

Cutout

Pros:

- Helps reduce overfitting by forcing the model to rely on all features of an object.
- Encourages learning of robust, spatially distributed features.

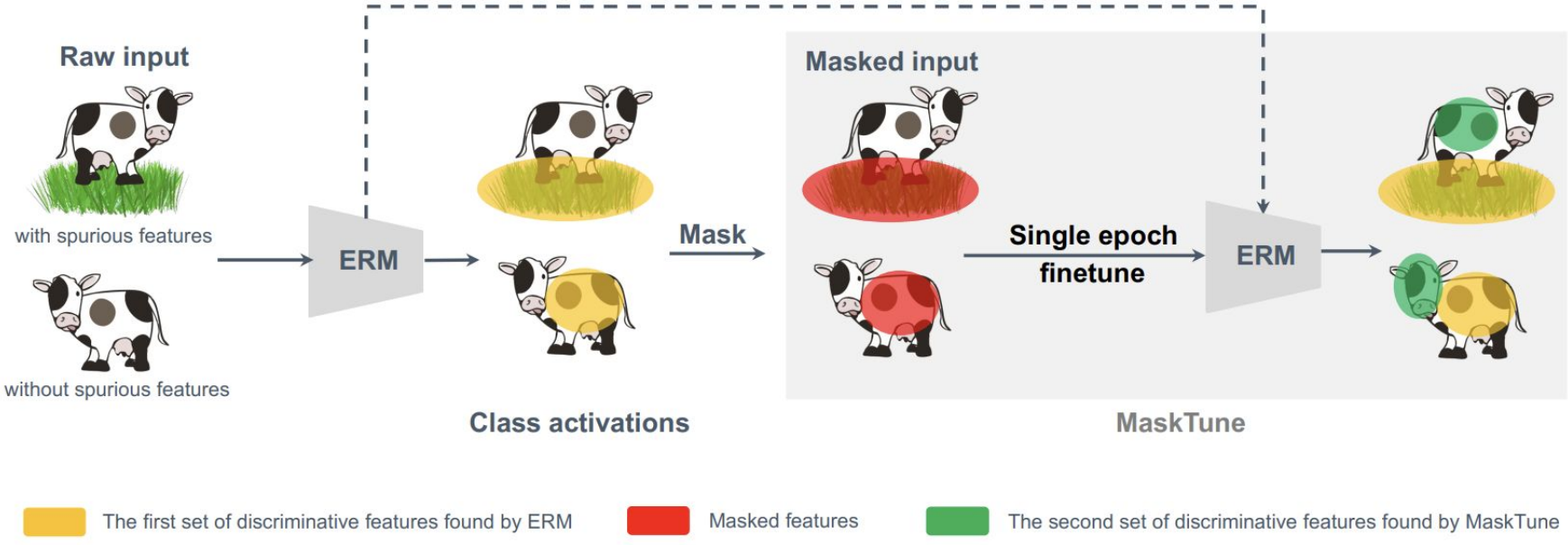
Cons:

- May lose valuable information in certain sections.
- Less effective with very small datasets or images with limited details.

MaskTune: Mitigating Spurious Correlations by Forcing to Explore

Saeid Asgari Taghanaki* Autodesk AI Lab
Aliasghar Khani* Autodesk AI Lab
Fereshte Khani* Stanford University
Ali Gholami* Autodesk AI Lab

Linh Tran Autodesk AI Lab
Ali Mahdavi-Amiri Simon Fraser University
Ghassan Hamarneh Simon Fraser University


Raw input
with spurious features
without spurious features
ERM
Mask
Masked input
Single epoch
finetune
ERM
Class activations
MaskTune
The first set of discriminative features found by ERM
Masked features
The second set of discriminative features found by MaskTune

CutMix Augmentation

Combines two images by cutting a rectangular patch from one image and pasting it onto another.

Label is also a combination, weighted by the area of the patch.

Cutmix

Pros:

- Provides richer contextual information.
- Helps with object localization, as cutouts provide occlusions.
- Improves model robustness to unseen scenarios.

Cons: (similar to Cutout)

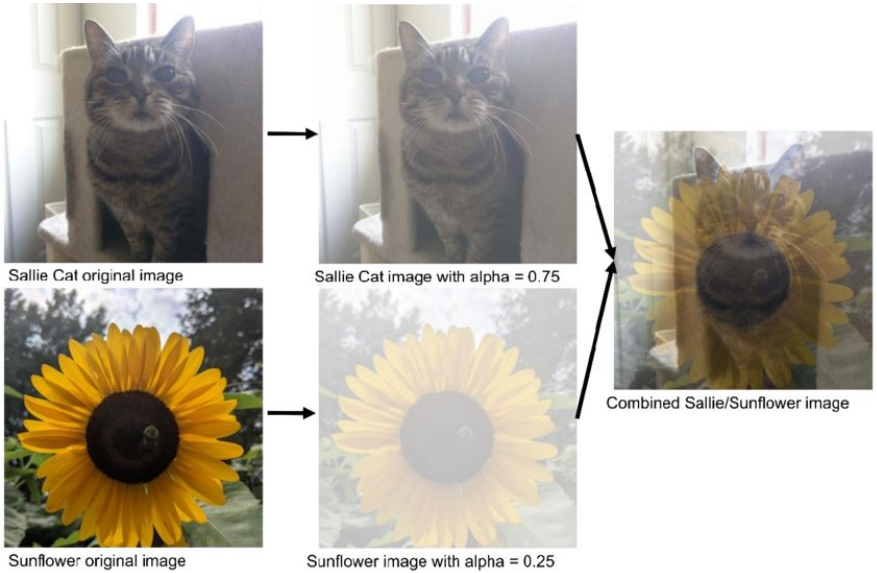
- May lose specific details of the object, especially if important regions are cut.
- Can be less effective if only a small dataset is available, as cutting and pasting reduce effective information.

MixUp Augmentation

What is MixUp?

Blends two images by mixing pixel values and labels linearly.

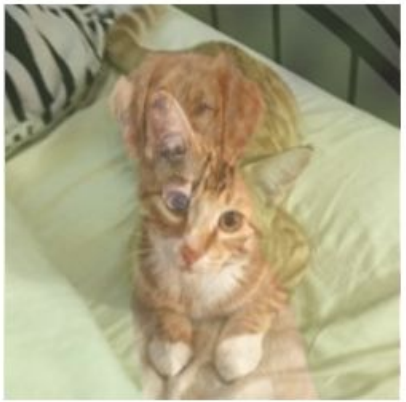
Formula: $x = \lambda * x1 + (1 - \lambda) * x2$, where λ is a mixing coefficient between 0 and 1.



$$\begin{aligned} \hat{x} &= \lambda x_i + (1-\lambda) x_j, \\ \hat{y} &= \lambda y_i + (1-\lambda) y_j, \end{aligned}$$

where $\lambda \in [0, 1]$ is a random number

Image



Label

[1.0, 0.0]
cat dog

[0.0, 1.0]
cat dog

[0.7, 0.3]
cat dog

MixUp

Pros

- Smooths decision boundaries, leading to better generalization.
- Reduces model confidence in predictions, preventing overfitting.
- Encourages learning from broader, combined feature distributions.

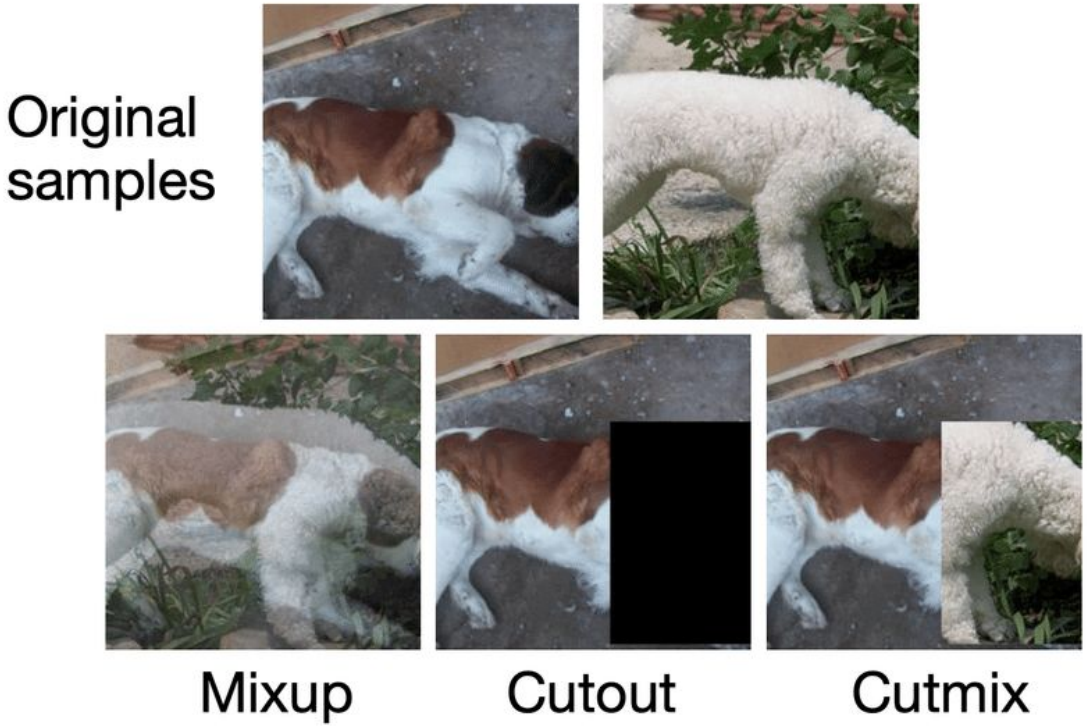
Cons

- Mixed images can look unrealistic, which might not benefit tasks requiring exact object shapes or details.
- Label ambiguity can arise if features from both images aren’t blended intuitively.

Comparison of CutMix, Cutout, and MixUp

Augmentation	Pros	Cons
CutMix	Combines context from two images	Can lose critical object details
Cutout	Forces model to learn diverse features	Can lose critical object details
MixUp	Smooth decision boundaries; good generalization	Unrealistic images; may confuse model on labels

Comparison of CutMix, Cutout, and MixUp



What is AutoAugment?

AutoAugment is a method to automatically search and find the best augmentation policies for a given dataset.

Developed by: Google Brain

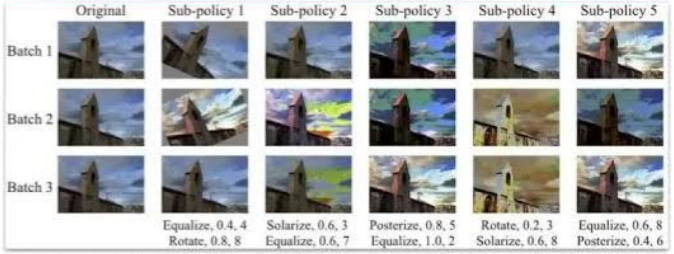
Goal: Automate the selection and combination of augmentations to maximize model performance.

AutoAugment

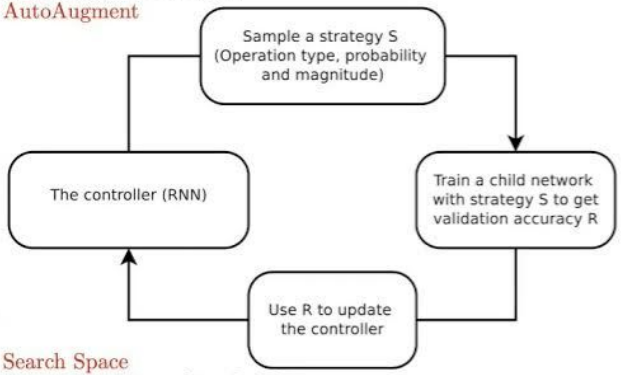
Boulder

AutoAugment: Learning Augmentation Strategies from Data

Intuitively, data augmentation is used to teach a model about invariances in the data domain.
Horizontal flipping of images during training is an effective data augmentation method on CIFAR-10, but not on MNIST, due to the different symmetries present in these datasets.



AutoAugment



PIL (Python Image Library): ShearX/Y, TranslateX/Y, Rotate, AutoContrast, Invert, Equalize, Solarize, Posterize, Contrast, Color, Brightness, Sharpness, Cutout, Sample Pairing
16 operations!
Discretize the ranges of magnitudes into 10 values (uniform spacing)
Discretize the probability of applying the operation into 11 values (uniform)
$(16 \times 10 \times 11)^2$ possibilities!
$(16 \times 10 \times 11)^{10} \approx 2.9 \times 10^{32}$ possibilities (5 sub-policies)!

Search Space
A policy consists of 5 sub-policies.
Each sub-policy consists of two image operations to be applied in sequence.
Each operation is associated with two hyper-parameters:
– the probability of applying the operation
– the magnitude of the operation
For every image in a mini-batch, choose a sub-policy uniformly at random.

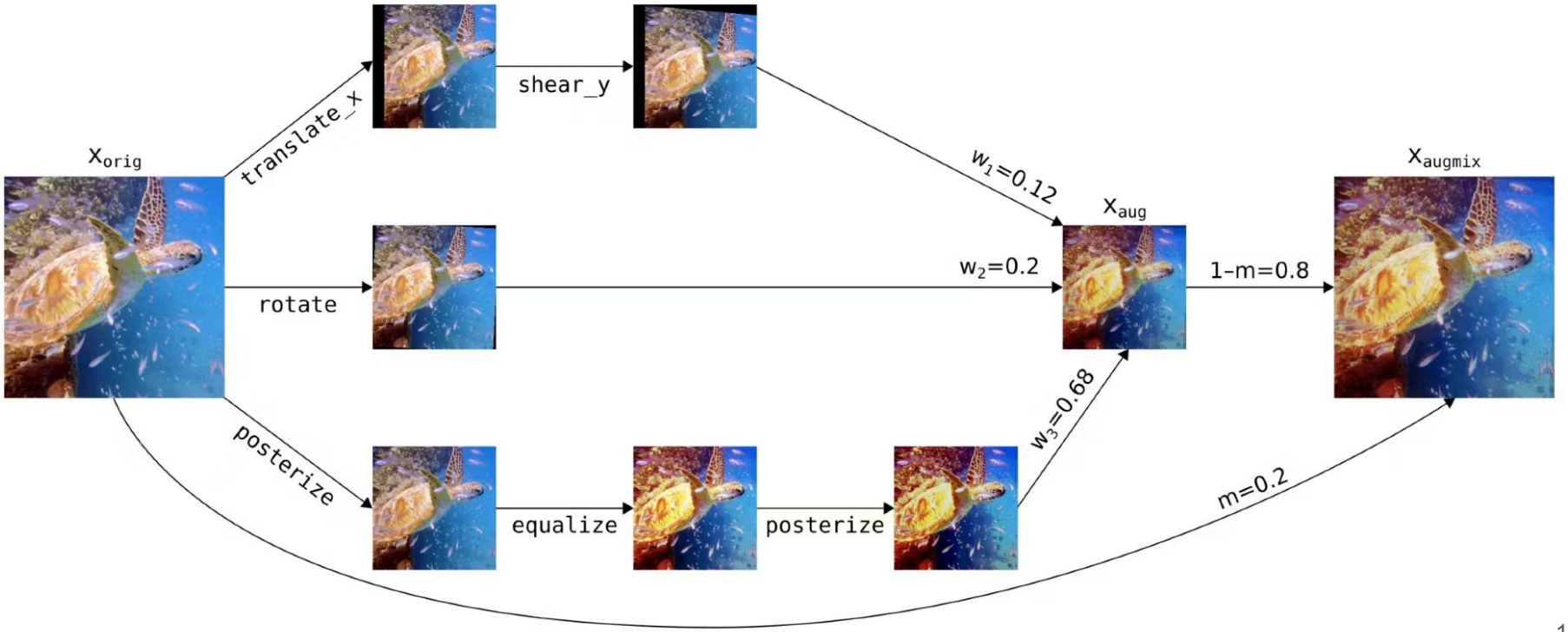
Search Algorithm
Recurrent Neural Network and Proximal Policy Optimization
30 softmax predictions in total

Cubuk, Ekin D., et al. "Autoaugment: Learning augmentation policies from data." *arXiv preprint arXiv:1805.09501* (2018).

Pros of AutoAugment

- Optimized for Specific Datasets
- Automated Process
- Improved Generalization
- High Performance

Other Augmentations (AugMix)



Test-Time Augmentation (TTA)

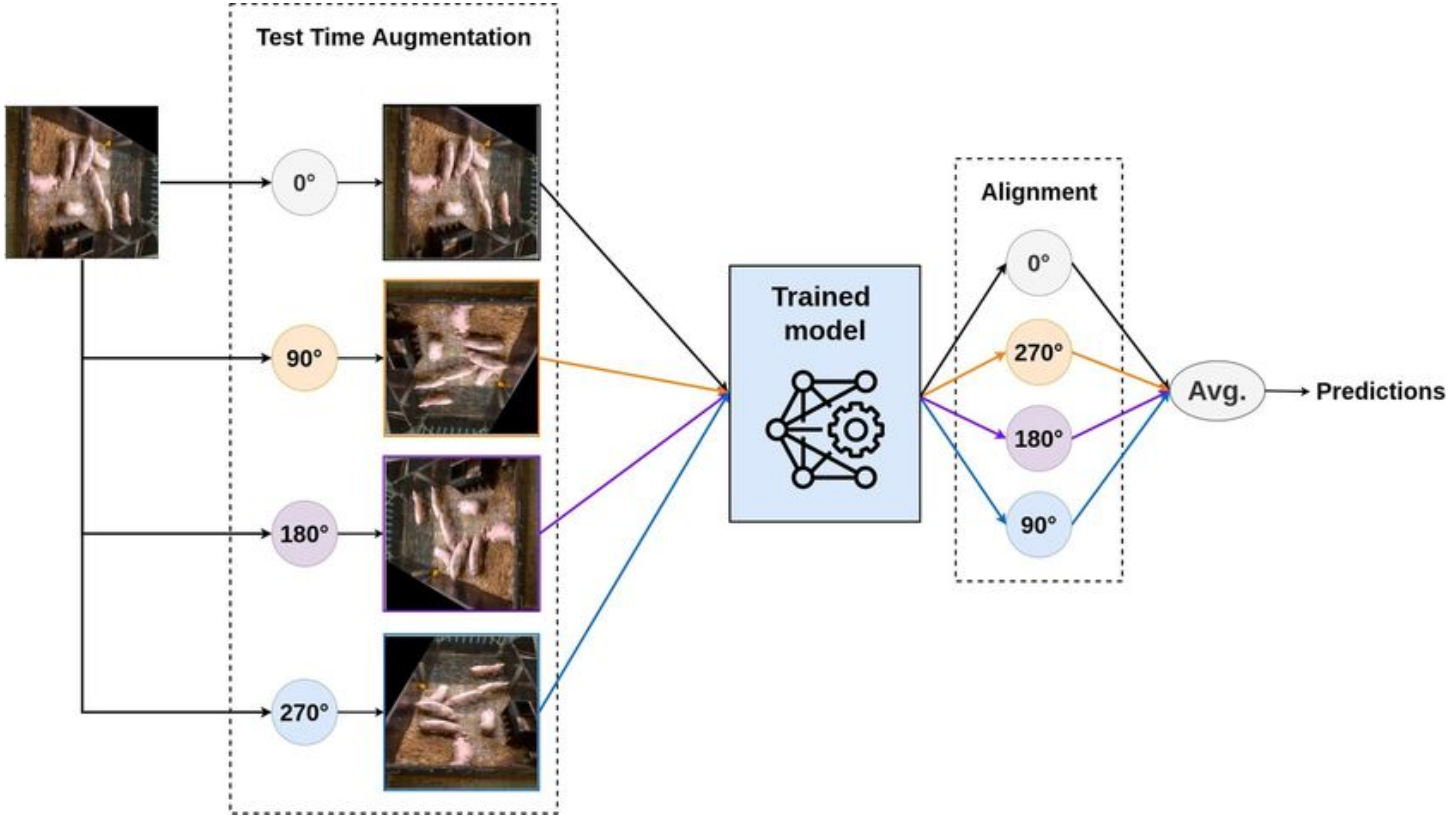
Test-Time Augmentation is a technique that applies augmentations to test data (or validation data) during inference.

Allows the model to make multiple predictions on different augmented versions of the same input.

Purpose:

Aims to improve model accuracy and robustness by averaging predictions from multiple augmented inputs.

Test-Time Augmentation (TTA)



How Test-Time Augmentation Works

Augmentation: Apply different transformations (e.g., rotations, flips, brightness adjustments) to the test image.

Prediction Ensemble: Generate predictions for each augmented version.

Result Aggregation: Average or use majority voting on the predictions to get the final output.

Test-Time Augmentation

Pros:

- Improved Prediction Robustness
- Enhanced Accuracy
- Reduced Overfitting to Specific Test Conditions

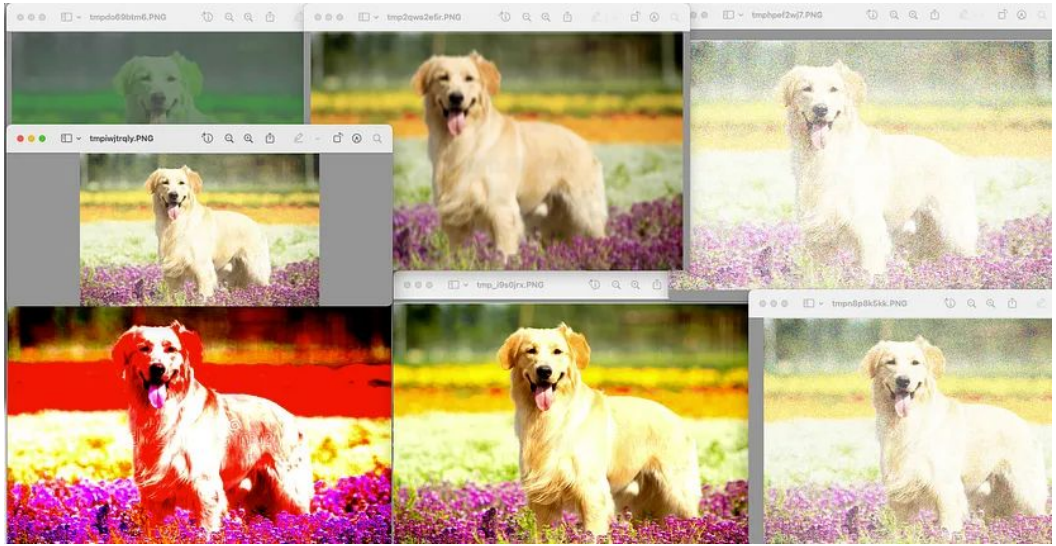
Cons:

- Limited Benefits on Simple Data

Conclusion

Each augmentation has unique strengths for various tasks.

Balance between preserving information and enhancing generalization.



Implementation Detail

Create an instance of the FoodData class by passing:

- img_path: List of image paths.
- img_dir: Directory containing images.
- size: Desired image size.
- transform: The transformation pipeline created above.

Load the image in RGB format.

Convert the image class to a numerical label using le.transform().

Return Format:

- “Gt”: The processed image tensor (ground truth).
- “Label”: The numeric label tensor for the image’s class.

Implementation Detail

Parameters:

- model: The model to be trained and validated.
- criterion: The loss function used to measure the model's prediction error (e.g., CrossEntropyLoss).
- optimizer: The optimization algorithm (e.g., SGD or Adam) that updates model parameters based on gradients.
- scheduler: A learning rate scheduler that adjusts the learning rate during training.
- num_epochs: The number of times the entire dataset is passed through the model during training.

Core Training and Validation Process:

- Tracking Performance: The total loss and number of correct predictions are accumulated for each epoch.
- Accuracy and Loss Calculation: At the end of each epoch, average loss and accuracy are computed for each phase.
- Model Saving: If the validation accuracy improves, the model weights are saved as the best model so far.
- Output: For each epoch, loss and accuracy are printed for both training and validation phases.

Implementation Detail

Select pre-trained ResNet-50 model.

Use CrossEntropyLoss as the criterion for classification.

Set up SGD optimizer with:

- Learning rate: 0.001
- Momentum: 0.9

Apply learning rate scheduler to decay the learning rate:

- Factor: 0.1
- Step size: every 3 epochs

Refs

https://encord.com/blog/data-augmentation-guide/

https://arxiv.org/abs/1708.04552

https://arxiv.org/abs/1710.09412

https://arxiv.org/abs/1805.09501

https://arxiv.org/abs/1905.04899

https://towardsdatascience.com/test-time-augmentation-tta-and-how-to-perform-it-with-keras-4ac19b67fb4d

https://github.com/DeepVoltaire/AutoAugment/