Image Classification

Barbaros Cetiner, PhD, PE
NHERI SimCenter

August 21, 2024

This material is based upon work supported by the National Science Foundation under Grant No. (1612843 & 2131111).

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Outline

- Traditional Machine Learning Methods for Image Classification
- Deep Learning Methods
- Building Blocks of CNNs
- Attention Networks and Transformers
- Training a Deep Learning Model

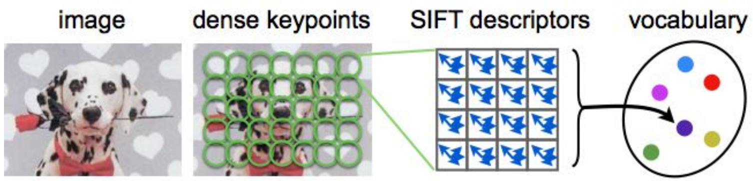
Traditional ML Methods for Image Classification

Traditional Methods

Traditional image classification methods consist of two steps

1. Feature extraction (encoding)
2. Establishing a connection between features and image labels (decoding)

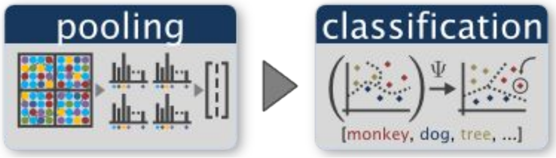
Encoding



Typical Algorithms:

- Scale invariant feature transform (SIFT)
- Histogram of oriented gradients (HOG)
- Binary robust invariant scalable keypoints (BRISK)

Decoding



Typical Algorithms:

- Support vector machines (SVM)
- Decision tress
- Neural networks

Deep Learning Methods

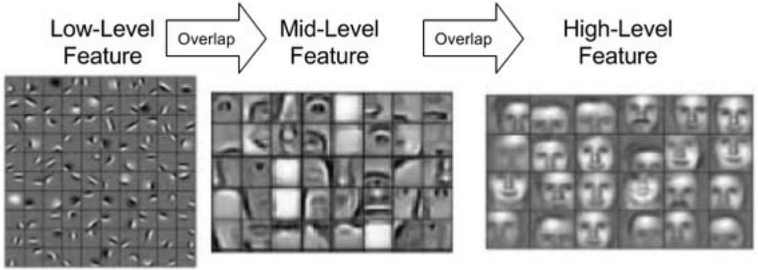
Traditional vs Deep Learning Methods

Traditional Approaches

- Identify the features to solve a problem
- Develop methods to extract these features

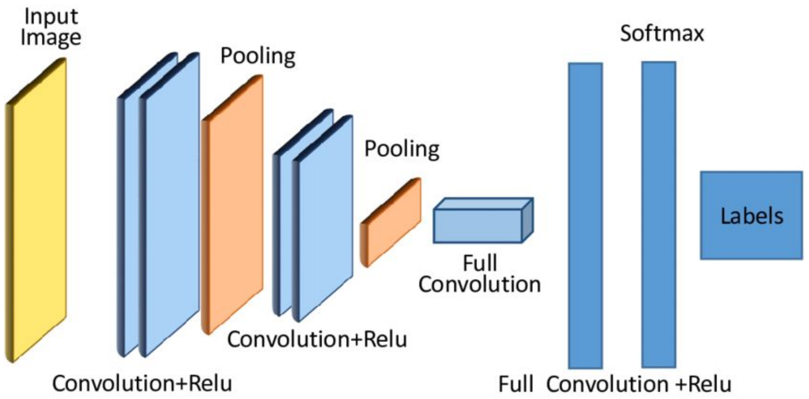
Deep Learning Methods

- Identifies the features it needs to solve a problem using the presented data



Advantages of Deep Learning Methods

- End-to-end methods that require less feature engineering
- Consistently more accurate than traditional methods

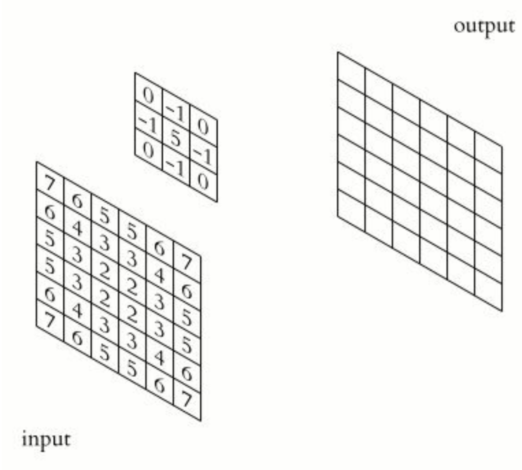


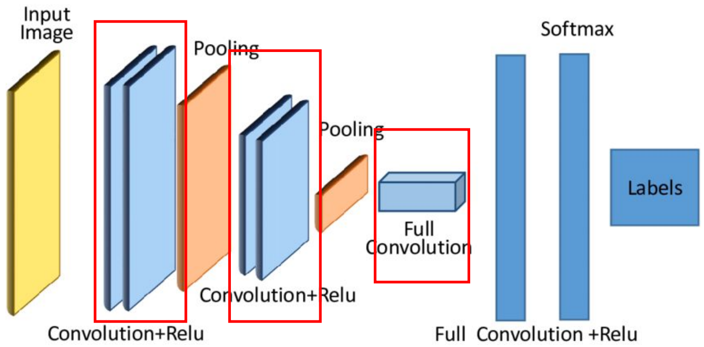
Convolutional Neural Networks (CNN) are one of the most popular deep learning methods for image recognition.

Building Blocks of CNNs

Convolutional Layers

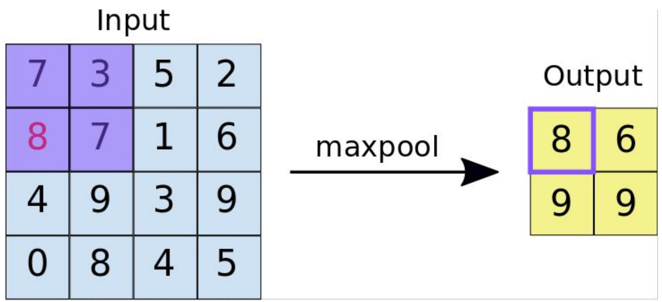
- Set of filters (or kernels), parameters of which are to be learned throughout the training
- Each filter convolves with the image and creates a feature (activation) map
- The kernel on the right has a **stride** of 1

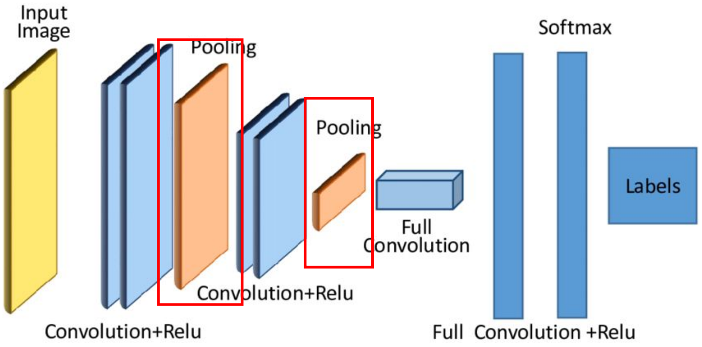




Pooling Layers

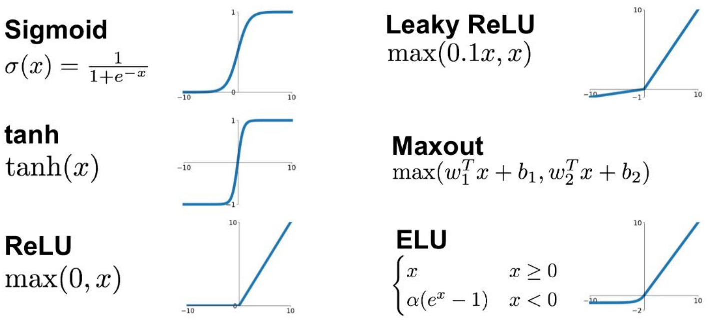
- Reduces the dimensionality of the computations
- Similar to convolutional layer, runs a filter across the entire input
- Unlike convolutional layer applies an aggregation function to the values
- What you see on the right is **Max pooling**

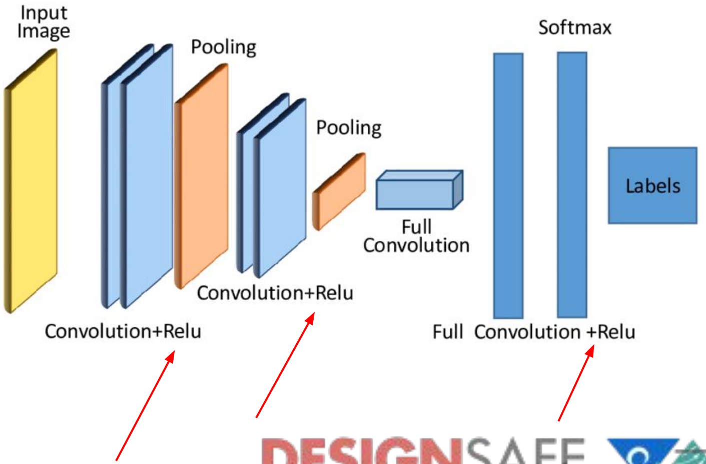




Activation Functions

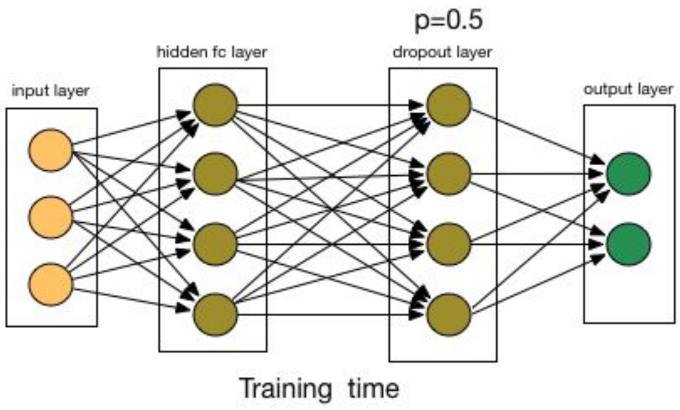
- **Note:** the composition of two linear functions is a linear function
- Activation functions enable nonlinear modeling

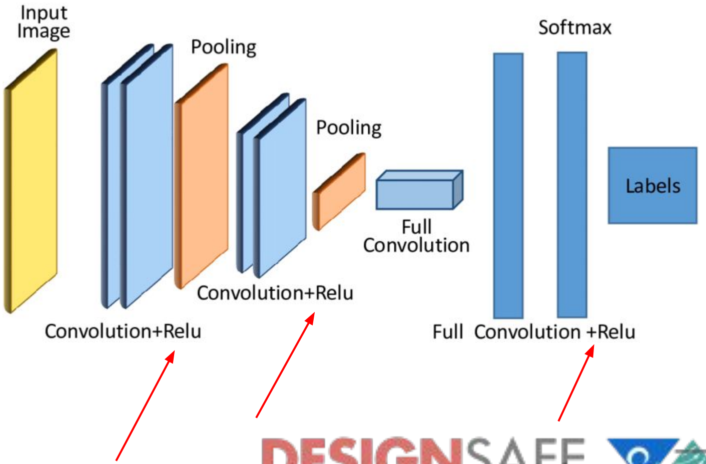




Dropout Layers

- Ensures that the model does not only rely on just the dominant features (avoids overfitting)
- Works by randomly killing some activations at each epoch
- **p=0.5** kills half of the connections



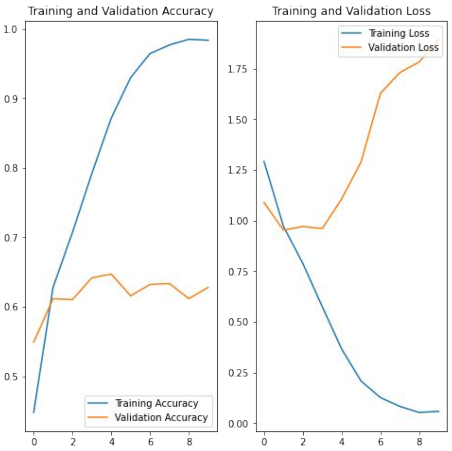


Quick Detour: What is Overfitting?

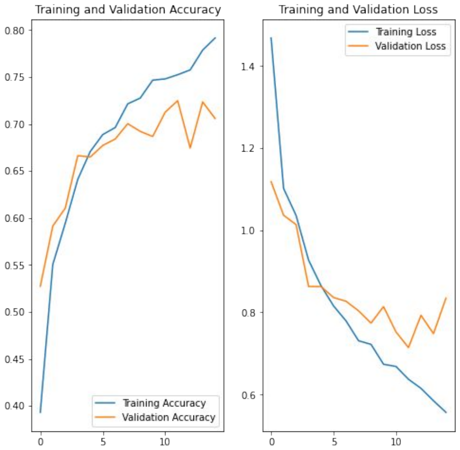
Overfitting: memorizing, instead of learning

Model remembers patterns, noise and random fluctuations, hence fail to perform well on unseen data (i.e., generalize)

Tell-tale sign: Divergence between training & validation accuracies (or losses)



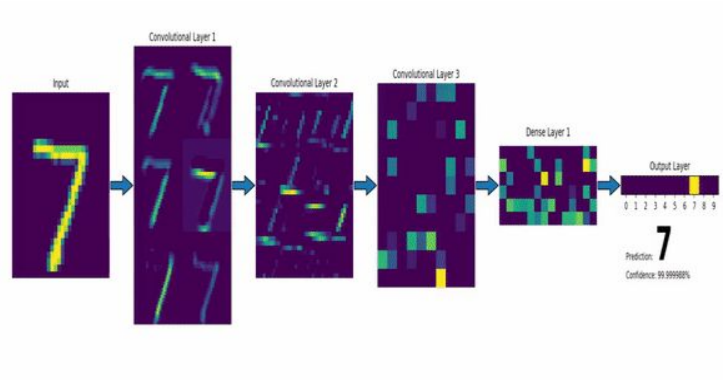
Overfitting ❌

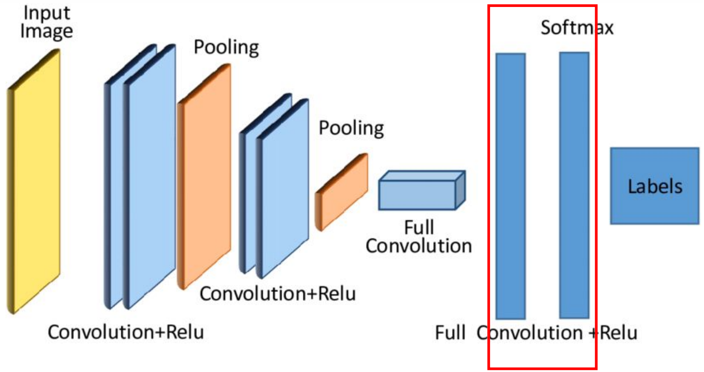


Not overfitting ✓

Fully Connected (FC) Layers

- Also known as the Classifier Head or Dense Layer
- This layer performs classification based on the features extracted through the previous layers
- FC layers usually use SoftMax activation function to classify inputs

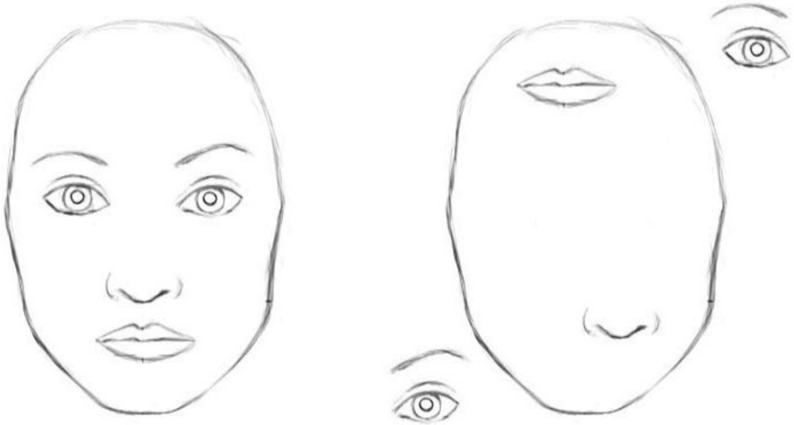




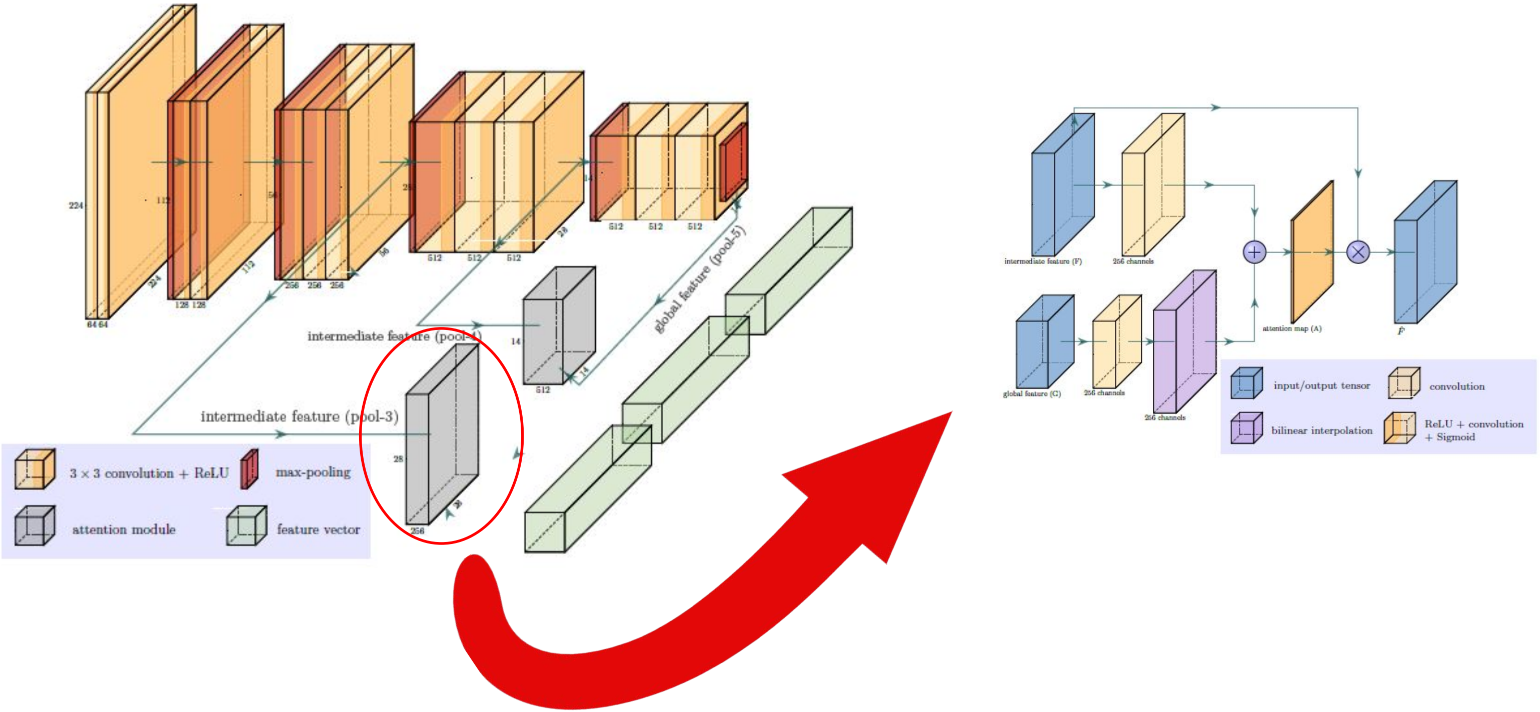
Attention Networks and Transformers

Why Use Attention-Based Models?

- CNNs do not capture the relative positions of different features
- The two figures below are the same to a CNN
- CNNs start at a single pixel and zoom out, attention-based models slowly bring the whole fuzzy image into focus

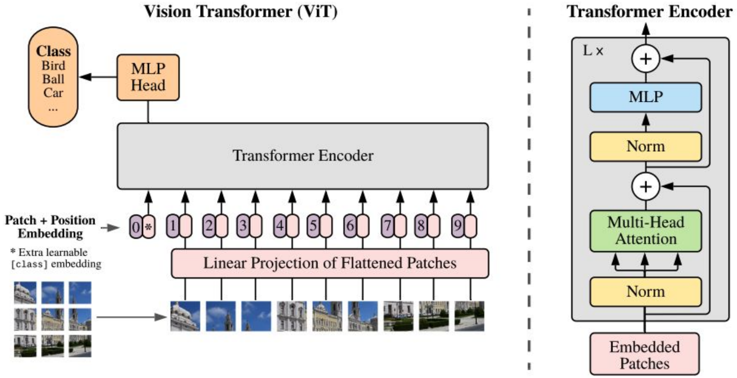


Attention Networks

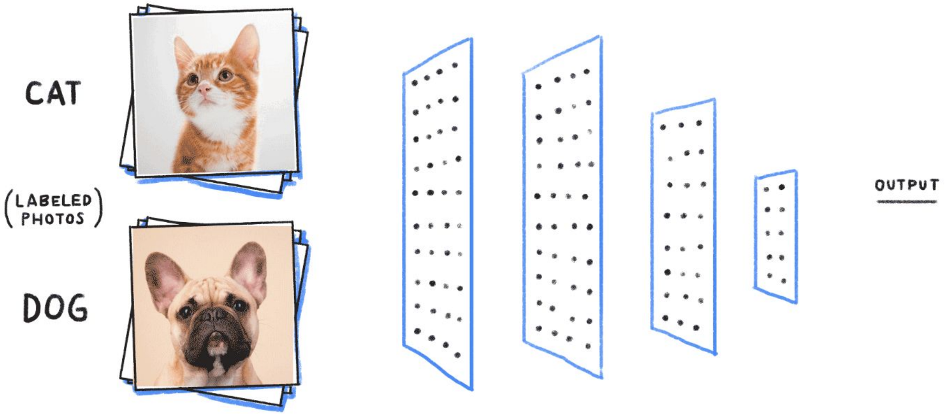


Transformers

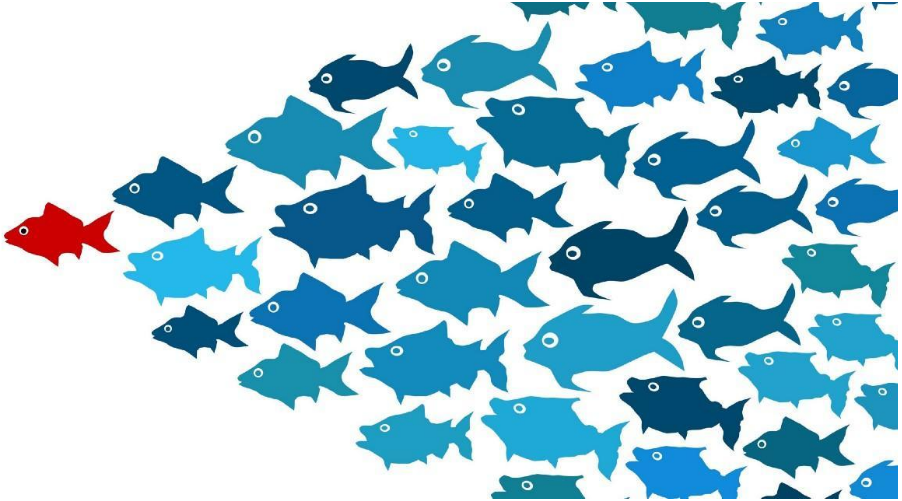
- Split an image into patches:
    - Image patches are treated the same way as tokens (words) in an NLP application.
- Flatten the patches
- Produce lower-dimensional linear embeddings from the flattened patches
- Add positional embeddings
- Feed the sequence as an input to a standard transformer encoder



Training a Deep Learning Model for Image Recognition

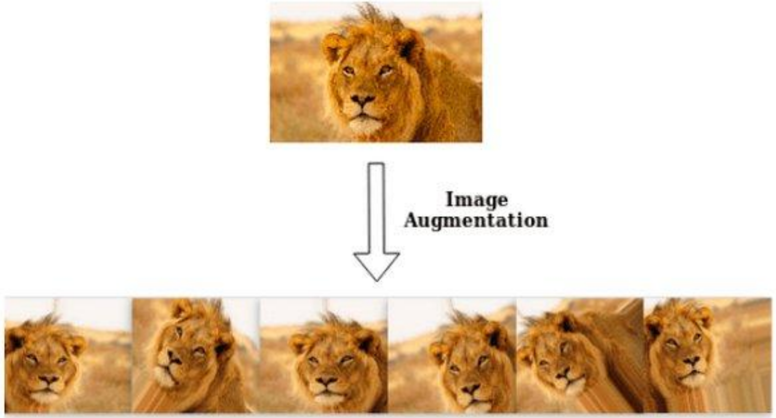


Dataset Preparation: Class Imbalance



- **Always have a balanced training dataset** (i.e., equal number of samples for each class)
- Else, the training model will spend most of its time on the majority classes (and possibly have a bias towards these classes)

Dataset Preparation: Image Augmentation



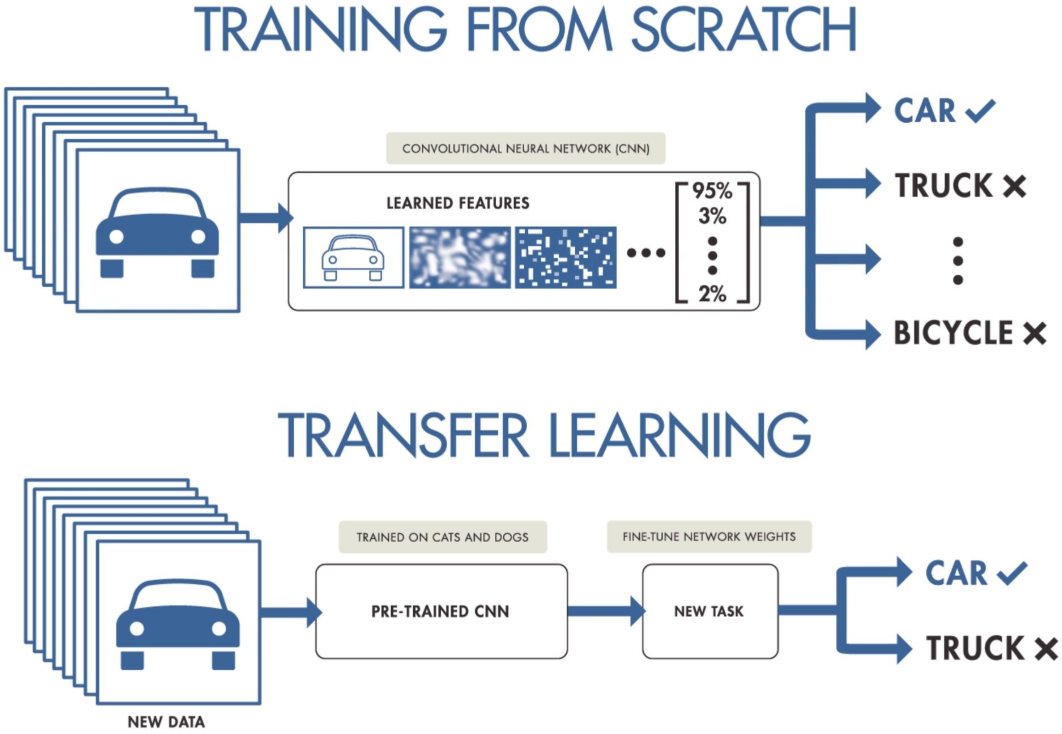
- Improves model performance by forming new and different examples to train datasets (expands the training dataset)
- Improves robustness of the model

Dataset Preparation: Image Augmentation

Popular techniques:

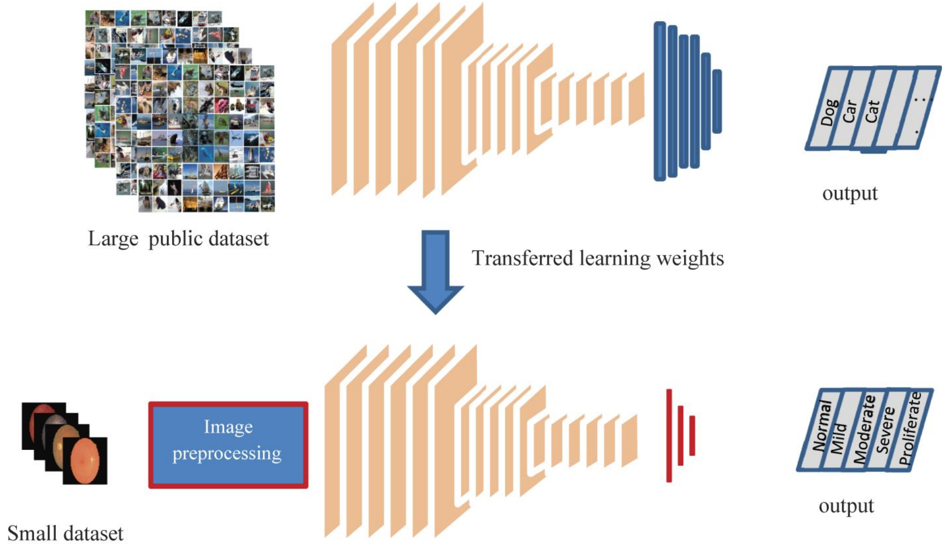
- Rotation,
- Gaussian noise,
- Crop,
- Hue and saturation adjustment,
- Elastic transform,
- Coarse dropout

Model Training: Transfer Learning



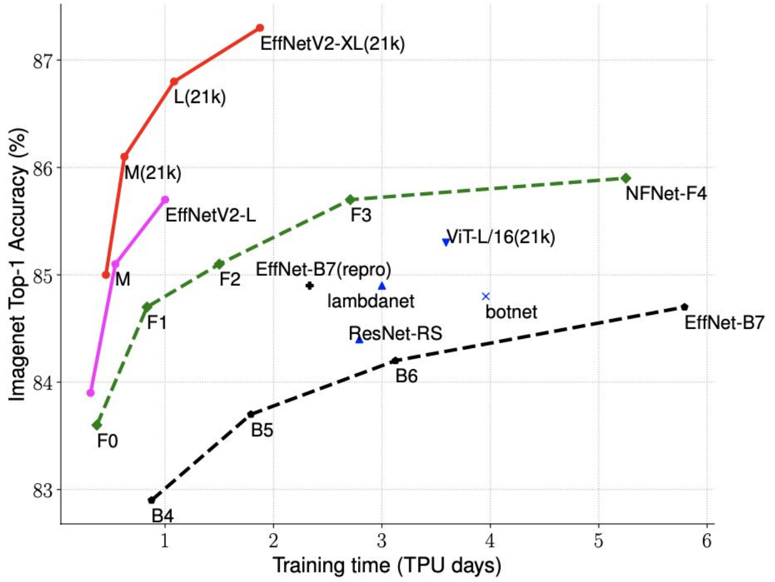
Model Training: Transfer Learning

- CNNs detect low level features in the first layer, forms in the middle layer, and task-specific features in the latter layers
- Transfer learning benefits from the lower features detected for a large dataset.
- Benefits:
  - Reduced training time,
  - Improved neural network performance in the absence of large training dataset



Model Training: The Right Architecture

For civil engineering applications (small datasets) using an architecture with a good accuracy level on a standard dataset is the best!



Model Training: Right Programming Approach

- Pytorch and Tensorflow are the most popular Python libraries for Deep Learning
- SimCenter's BRAILS provides easy-to-use end-to-end pipelines for training deep learning models
  - Little to no knowledge of deep learning is required to use BRAILS
  - Makes the difficult training decisions for the user

TensorFlow

PyTorch

BRAILS

Questions?