أكاديمية كاوست
KAUST ACADEMY

Day 2: Computer Vision Summary

Note: Do Not depend entirely on this and study from the official slides.

Contributed by: Hassan Mohammed Nasr

Dataset Class & DataLoaders

Component	Dataset Class	DataLoaders
• Role	Stores and process samples(images) and their labels,	Delivers data from the dataset class to the model
• Key Function	Retrieve single item from the dataset	Groups samples into batches and deliver them to model

Notes:

- For Custom Dataset class, we need two main functions:
  - __len__(): returns dataset size.
  - __getitem__(): fetches single data item —» returns single data item with its label
- DataLoader is used to load data in mini batches, SHUFFLES, and utilize multiprocessing

Data Augmentation

- Data Augmentation is a technique to artificially expand the dataset size and add variety to improve the model and prevent overfitting.
  - Goal: Teach the model invariance (e.g. cat is still cat if rotated, or scaled or cropped…)
  - Types: Geometric transformations such as flipping, rotation, cropping, and translation…. Or other transformations related to color, brightness, noise, contrast….
    - Pick the type that solves model's weaknesses (e.g if u noticed that the model always predicts the images that are cropped or the main object is not fully inside the image, then we could use cropping, randomly cropping some images in the training)
- Test Time Augmentation is a technique of averaging multiple augmented versions of the same image

Note: Data Augmentation Is applied only on training not testing or validation

Transfer Learning

- Transfer Learning Using good trained models and adapt it to solve a new specific problem

1 – Fine – Tuning

Concept: Adjusting the weights of the pre trained model to fit my new data

- Strategy based on Data Size and similarity
  - Data is small & Similar(similar to the data the model trained) : FREEZE the feature extractor, and train only the final classifier layer
  - Data is small & Different : freeze earlier layers, but train later layers
  - Data is Large : Train (fine tune) the whole network
- Advantages
  - Saves time and computational resources
  - Improves performance, especially with limited data

2 – Ensembling

Train multiple different models on the same data and average their predictions (no single model is perfect, different models make different errors, thus combining the predictions can reduce error and improve accuracy)

Recall boosting and bagging in decision trees

- Hard Voting $\longrightarrow$ Majority of votes
- Soft Voting $\longrightarrow$ Average the predicted probabilities, take the highest

Regularization Techniques

- Regularization Techniques : Methods to stabilize the training and prevent it from memorizing noise (overfitting)

1 - Dropout

- Concept : Randomly "deactivates" (put zero) a percentage of neurons during each training step
- Effect : the network learns not to rely too heavily on specific connections, act as efficient ensemble

Note: Applied during training only, but the problem is using all neurons in inference will be inconsistent

Solution: Scale the output by 'p' to keep the same expected activation as in training

- model.train(): Enables dropout
- model.eval(): disable dropout and apply scales

2 - Batch Normalization

- Problem : As weights change during training, the distribution of inputs to the next layer keeps shifting. The next layer has to constantly re adjust to these changes, which slows the training. Moreover forcing every layer to output zero mean and one unit variance is restrictive

Solution: Let the network learn if it wants normalization

- Step 1, Normalize: apply normalization (get zero mean and 1 standard deviation)
- Step 2, Give the network control: let the network scale how much mean it needs and shift the variance as it wants $y = \gamma \hat{x} + \beta$

- Advantages
  - Gives the network the power to choose
  - it learns exactly when and how much to normalize for optimal performance

- model.train(): Use batch statistics
- model.eval(): use running statistics (fixed values from training)

Full Training Workflow

- Step 1: Initial setup
  - Start with pretrained model and fine tune it
  - Don't use data augmentation and Regularization
  - Get a baseline score
- Step 2: Improvement Process
  - Error Analysis: Analyze where the model fails
  - Regularization: Add dropout / batch normalization if overfitting
  - Augmentation use data augmentation if model struggles with variations
  - Tune: Tune your hyper parameters
  - Ensemble

  Note: always keep track of scores, improvements, and what made the improvements at every step
- Step 3: Finalization
  - Save the optimized model for later use