Deep Learning

Computer Vision Applications

Edgar F. Roman-Rangel.
edgar.roman@itam.mx

Digital Systems Department.
Instituto Tecnológico Autónomo de México, ITAM.

February 26th, 2021.

Outline

Transfer learning

Remember: Any DL model is but a collection of matrices (weights). i.e., a network with three layers,

$$\hat{y} = \sigma\left(\mathbf{w}^{(3)T}\sigma\left(\mathbf{w}^{(2)T}\sigma\left(\mathbf{w}^{(1)T}\mathbf{x}\right)\right)\right).$$

Once the network is trained, we can store the weights $\Omega = \{\mathbf{w}^{(i)}\}$, and use them later in a test set, as we already know. Furthermore, we might reuse them for a completely different data base (similar problem): **transfer learning**.

We might need to modify a few layers, e.g., output layer.

Transfer learning pipeline

Straightforward

Simply apply the trained model to a new data base, just as it is.

Fine tuning

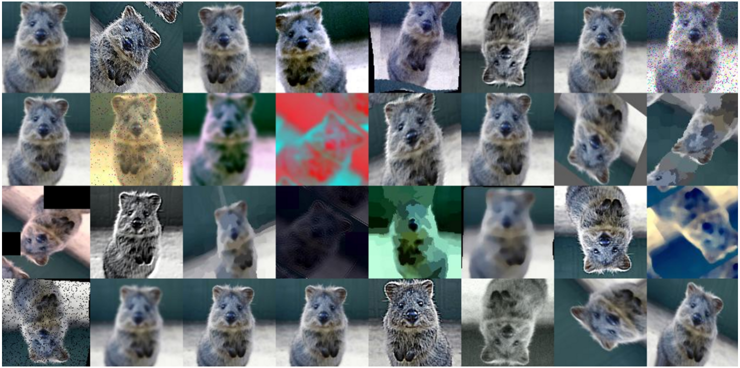
Retrain the model on the new data, starting from the previously learned set of parameters (instead of random initialization).

Warm up

1. Freeze the set of previously learned parameters.
2. Train any newly added parameter.
3. Unfreeze initial weights, and fine tune the whole model.

Data augmentation

Increase variability of input data, so the model learns to be robust against potential variations, e.g.,



Outline

Improving CNN performance

Object detection and localization

Pure convolution

Style transfer

Definitions

Assume there is, at least, one object of interest in the image.

Detection

- Binary answer.
- Telling whether or not the object is found in the image.

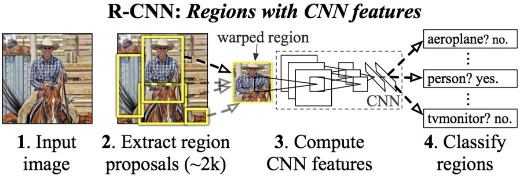
Localization

- Vector form answer.
- Bounding box framing the object of interest, $\mathbf{y} = [x, y, w, h, c]$.

Optionally, include the class label.

R-CNN, I

Girshick et al. “Rich feature hierarchies for accurate object detection and semantic segmentation”. CVPR, 2014.



- Uses a pretrained CNN as feature extractor.
- Defines candidate regions within the image.
- Produces a vector of 4096-D for each region.
- Feeds them into a classifier.

R-CNN, II

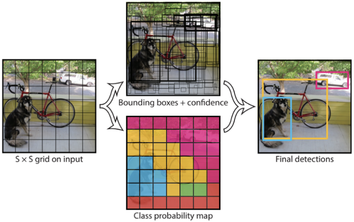
R-CNN also produces a correction off-set value, which compensates for objects cuts produced by the process that generates candidate regions.

Extension:

- Fast R-CNN: Girshick. "Fast R-CNN". arXiv:1504.08083. 2015.
- Faster R-CNN: Ren et al. "Faster R-CNN: towards real-time object detection with region proposal networks". NIPS. 2015.
- Mask R-CNN: He et al. "Mask R-CNN". ICCV. 2017.

YOLO, I

Redmon et al. “You Only Look Once:Unified, Real-Time Object Detection”. CVPR. 2016.

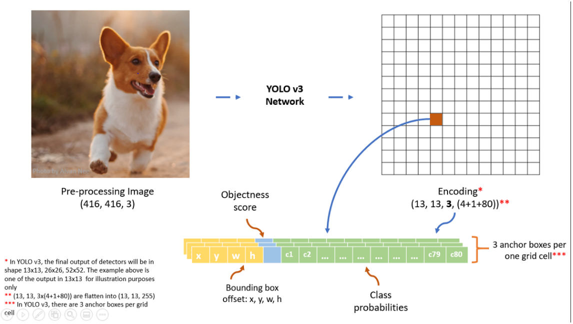


- Single CNN.
- Predicts both bounding boxes and the class probabilities.
- It is trained with the whole information integrated.
- Real time, i.e., 45 fps.

YOLO, II

Extension:

- v2 to v5, with incremental improvements.
- v3 is the latest one proposed by the original authors.



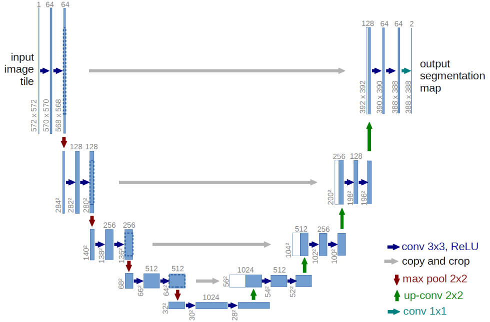
Outline

Segmentation, I

Classify each pixel as belonging, or not, to a specified entity.

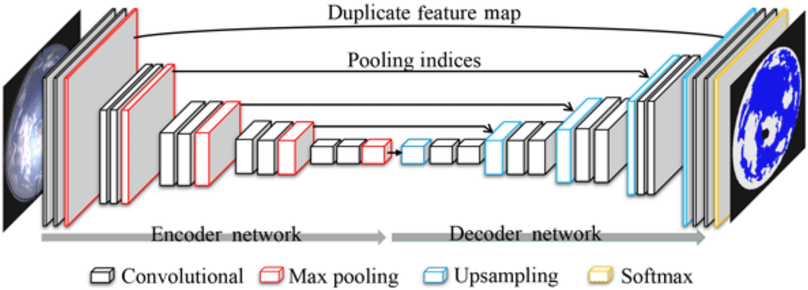
- **Object segmentation**: classify pixels by object identity.
- **Semantic segmentation**: classify pixels by class identity.

Ronneberger et al. "U-Net: Convolutional Networks for Biomedical Image Segmentation". MICCAI. 2015.



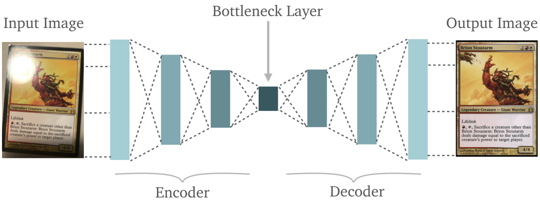
Segmentation, II

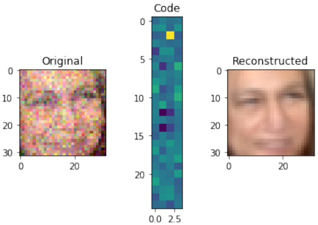
U-Net input and output.

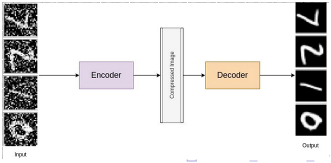


Denoisig and compression

Fully convolutional NN are also used for image denoising, inpainting and compression.



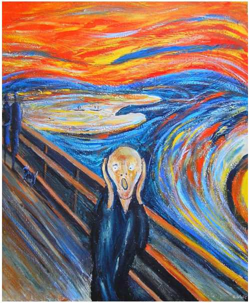




Outline

Transferring styles

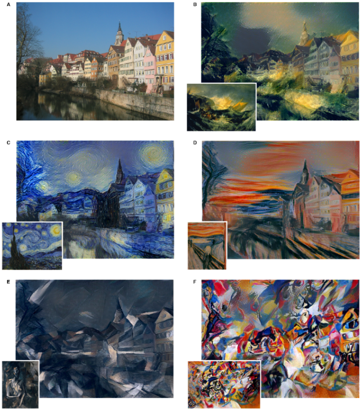
- Merge the content of one image with the style of another one.
- Apply the style of an image onto the content of another one.


Style transfer

Gatys et al., exploited the knowledge about of filter outputs.

- Gatys et al. 2014. A Neural Algorithm of Artistic Style.
- Gatys et al. 2016. Image Style Transfer Using CNNs. CVPR.



Pipeline

1. Load a pre-trained model.
2. Select some hidden layers for content and style outputs.
3. Pass contents image and obtain contents reference (output).
4. Pass style image and obtain style reference (output).
5. Pass a random image and get its outputs.
6. Compute loss against references.
7. Back-propagate error all the way to the input random image.
8. Update random image.
9. Repeat from step 5 until convergence.

Q&A

Thank you!

edgar.roman@itam.mx