Road Segmentation from Satellite Images

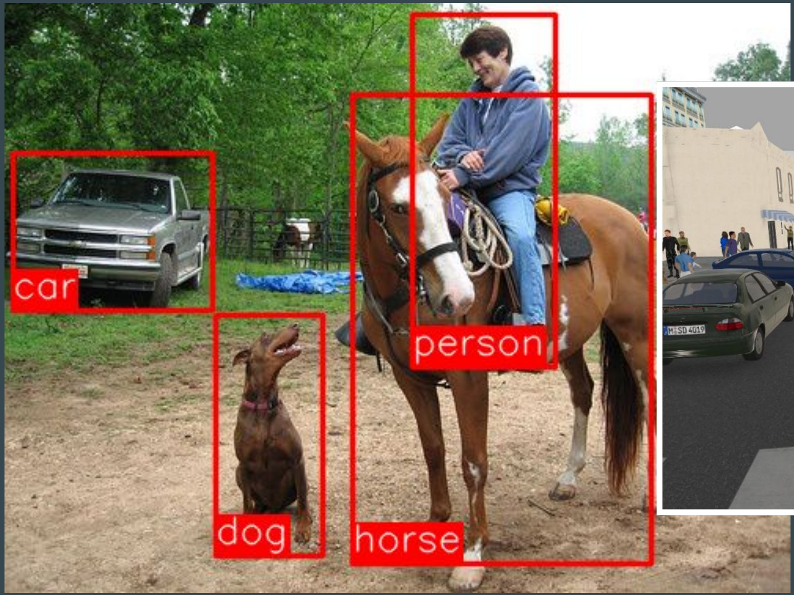
December 2018

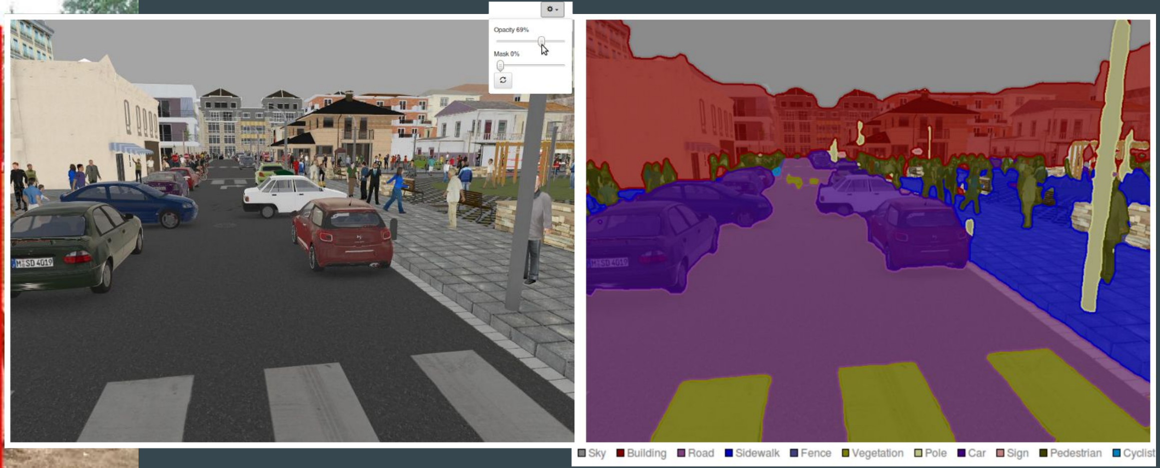
Machine Learning, EPFL

Slides: Jangwon Park

Introduction

- Deep learning has become increasingly popular in computer vision.

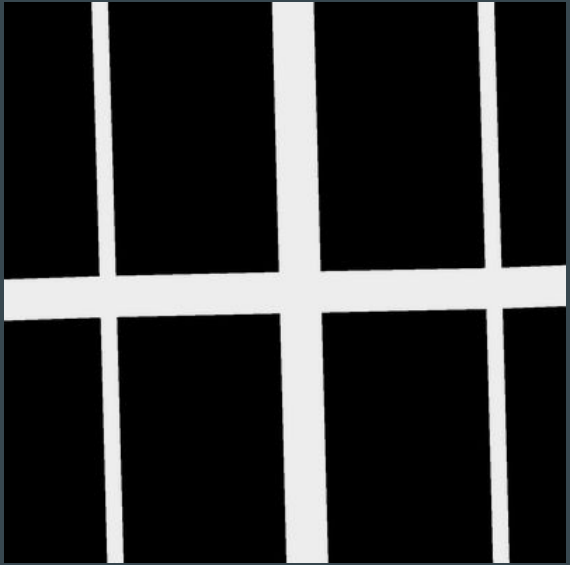




Dataset

- Satellite images from Google Maps (400 x 400 pixels)
- 100 training images, 50 test images (600 x 600 pixels)

Sample image



Groundtruth

Problem Statement

- Quick literature review reveals that CNNs are at the heart of this task.
- Classify each **16x16 pixel patches** as either {road = 1, background = 0}.
- Two goals of the project:
  - Understand the effects of numerous hyperparameters in CNNs
  - Construct a model that performs reasonably well (>= 0.7 minimum score)

$$F_1 = \left(\frac{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

CNN Evaluation Criterion

Exploratory Data Analysis

- Major challenges include:
    - Closeness in color between roads and other non-road objects *e.g.* parking lot
    - Obstacles over pixels that should be classified as roads *e.g.* trees

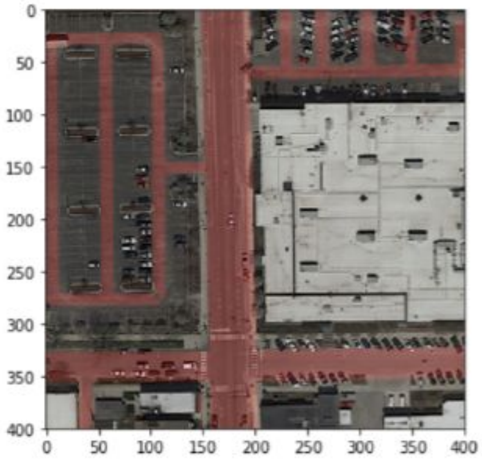


Fig. 1: Parking lot

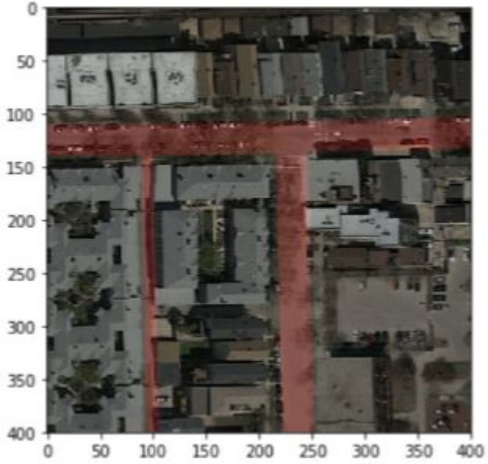


Fig. 2: Trees and buildings

Data Processing

- Image padding: extension of each 16x16 patch images by “mirror reflection”
  - Reveals previously subtle patterns so that each patch image is **easier to recognize as road**

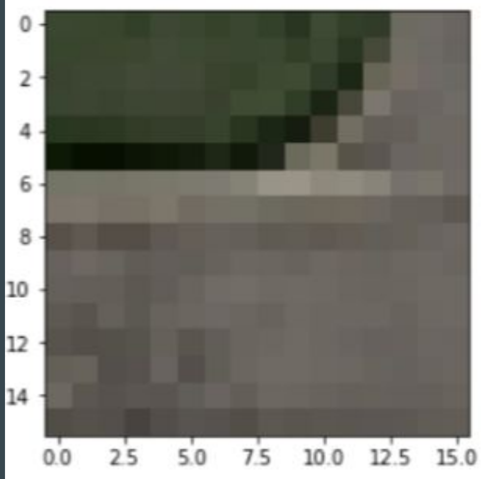


Fig. 3: Example 16x16 patch image. Hard to tell if it belongs to a road.

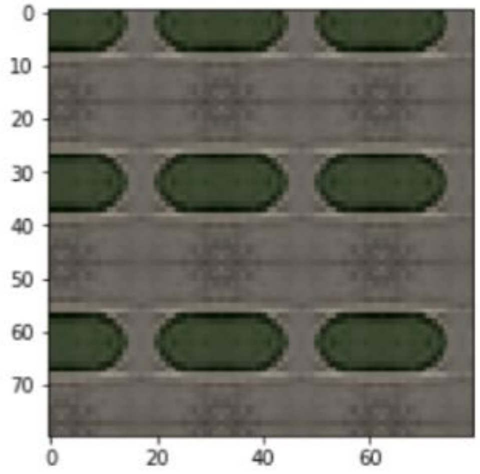


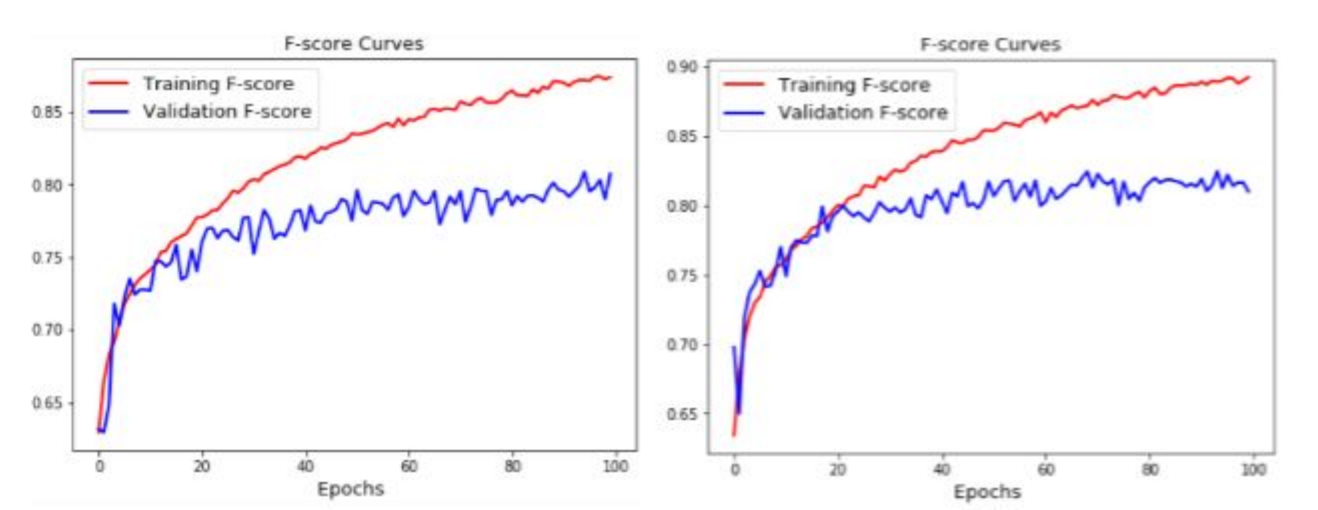
Fig. 4: Padded version of Figure 3. Easier to tell if it belongs to a road.

Optimizing Hyperparameters

- There are countless number of hyperparameters in neural networks in general
- In this section, we present only the following for brevity:
  - Choice of activation functions
  - Type of optimizer
- Please refer to the **report** in the Github repository for full details
  - https://github.com/parkjan4/RoadSegmentation

Activation Functions: ReLU vs. Leaky ReLU

- ReLU is a standard practice in most neural networks, but may suffer from the "vanishing gradient" problem.
- Leaky ReLU prevents this with a small negative slope in the negative domain.



Left: ReLU

Right: Leaky ReLU (alpha = 0.1)

Activation Functions: ReLU vs. Leaky ReLU

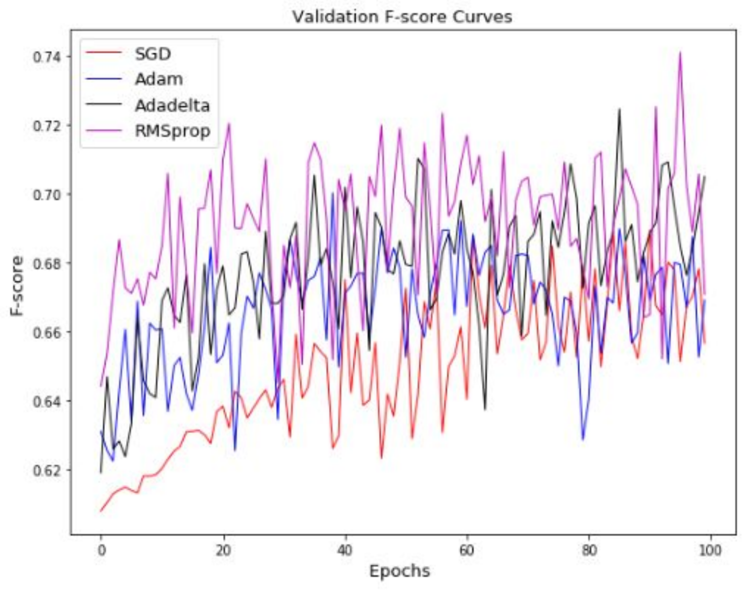
- When tested on a sample, light CNN architecture, leaky ReLU achieves both:
  - Faster convergence
  - Higher average validation F score
- Leaky ReLU also achieves lower variance in cross validation results, from which one can conjecture that it may also assist in avoiding overfitting.

Activation	Cross Validation F-scores
ReLU	0.664 ± 0.023
LReLU	0.677 ± 0.018

Comparison of 10-Fold Cross Validation Results

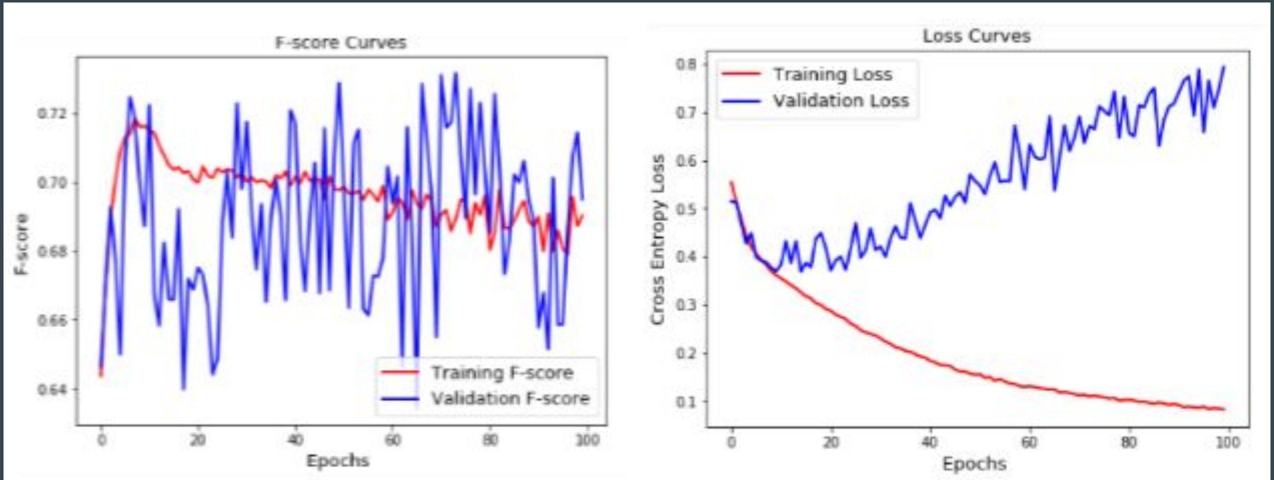
Type of Optimizers

- For different applications, optimizers may have different computational and statistical properties.
- On the same light CNN architecture as before:



Type of Optimizers

- Previous graph shows **RMSprop** and **Adadelta** show good computational properties (faster convergence).
- However, on a slightly more complex architecture, they **lack robustness** (computational-statistical property trade off)



Left: RMSprop

Right: Adadelta

Type of Optimizers

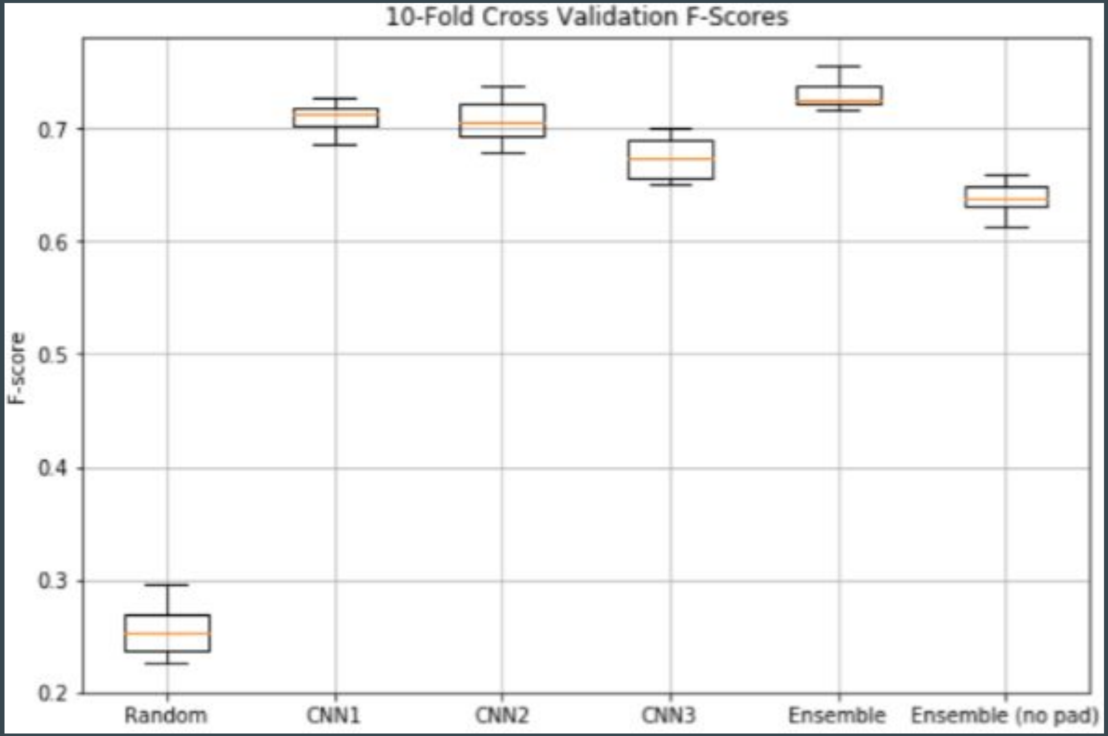
- Previous graphs show that certain optimizers can lead to overfitting on a relatively light CNN.
- Overfitting can be avoided with regularization and/or with more data, but is expected to be computationally costly.
- SGD is a good option but typically the slowest to train.
- Adam is the optimal choice for this particular application.

Model Selection

- Three architectures are proposed (refer to the report in Github for full details on hyperparameter selection)
- All three networks are comparable in terms of complexity.
- An **Ensemble Model** is also pursued by combining all three models via *majority voting rule*.

Type	Network 1	Network 2	Network 3
Input	48x48x3	64x64x3	48x48x3
Convolution filters	64 (5x5)	16 (4x4)	128 (3x3)
Max Pooling	2x2 *same*	2x2 *valid*	-
Dropout	$p = 0.25$	$p = 0.25$	-
Convolution filters	128 (3x3)	32 (4x4)	128 (3x3)
Max Pooling	2x2 *same*	2x2 *valid*	2x2 *same*
Dropout	$p = 0.25$	$p = 0.25$	-
Convolution filters	256 (3x3)	64 (4x4)	128 (3x3)
Max Pooling	2x2 *same*	2x2 *valid*	2x2 *same*
Dropout	$p = 0.25$	$p = 0.25$	-
Convolution filters	256 (3x3)	128 (4x4)	-
Max Pooling	2x2 *same*	2x2 *valid*	-
Dropout	$p = 0.25$	$p = 0.25$	-
Fully Connected	128 neurons	-	64 neurons
Dropout	$p = 0.5$	-	$p = 0.5$
Output	2 neurons	1 neuron	1 neuron
Activation	Softmax	Sigmoid	Sigmoid

Results & Discussion



- “Random” model predicts 1 with the probability equal to the proportion of “road” pixels in each image.
- “Ensemble (no pad)” is shown to verify the usefulness of the image padding technique.

Results & Discussion

- In general, CNNs are much more powerful than a “random guess” which verifies their ability to detect and learn local features in each image.
- Ensemble model outperforms individual CNNs and also achieves lower variance
- Without image padding, CNNs perform worse.

Model	Cross Validation F-score $\pm$ σ
Random	0.255 ± 0.022
CNN1	0.711 ± 0.012
CNN2	0.707 ± 0.019
CNN3	0.673 ± 0.018
Ensemble	$\mathbf{0.730 \pm 0.011}$
Ensemble (no padding)	0.640 ± 0.014

Results & Discussion

- Proposed ensemble CNN shows reasonable performance (>= 0.7 F score)

Sample Prediction 1

Sample Prediction 2

References

[1] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan,V. Vanhoucke, and A. Rabinovich, Going deeper with convolutions,2015 IEEE Conference on Computer Vision and Pattern Recognition(CVPR), 2015.

[2] H. Wu and X. Gu, Towards Dropout Training for Convolutional NeuralNetworks, Neural Networks, vol. 71, pp. 110, Nov. 2015.

[3] Karpathy, A. (2018). CS231n Convolutional Neural Networksfor Visual Recognition. [online] Cs231n.github.io. Available at:http://cs231n.github.io/neural-networks-1/ [Accessed 14 Dec. 2018]

[4] Hansen, L. and Salamon, P., ”Neural network ensembles,” IEEE Trans-actions on Pattern Analysis and Machine Intelligence, vol. 12, issue 10,pp. 2, Oct. 1990.