GeoX Young Academy:
Machine Learning in Remote Sensing
Best practice and recent developments
-
Part 4: Modern Approaches to ML

Ronny Hänsch & Andreas Ley

Dept. Computer Vision and Remote Sensing

TU Berlin - Germany

Challenges in classification today?

- Increasing amount & openness of data, e.g.:
  - Pléiades: entire earth every day ($<$ 1 m resolution)
  - USGS public domain aerial images

  $\Rightarrow$ Scalability: temporal/space complexity
- Intra-class variability:

Chicago

Vienna

Austin

- Interest in semantic classes (e.g., *building*, *road*, *lane*)

  $\Rightarrow$ Need for high-level contextual reasoning (shape, patterns,...)

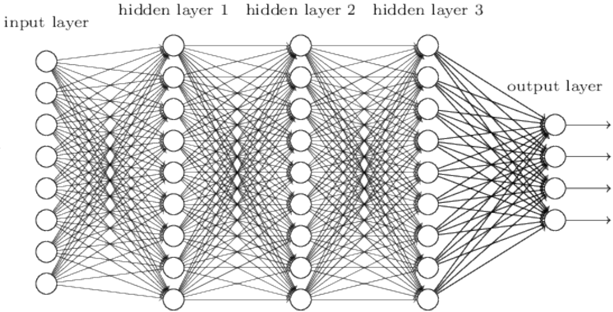
  $\Rightarrow$ Generalization to different locations

Outline

Recap: Artificial neural networks

Multilayer perceptron (MLP)

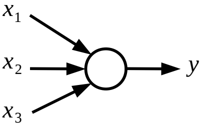
Features
$\rightarrow$



Classif. scores
(e.g., "dog: 0.9")

Fully connected neuron layers

Neuron



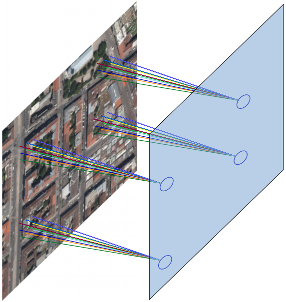
- $y = \sigma(\sum a_i x_i + b)$, σ nonlinear
- Parameters (a_i, b of all neurons) define the function
- Trained from samples by stoch. gradient descent

Recap: Convolutional neural networks (CNNs)

- Input: the image itself
- $\{$*Convolutional* layers $+$ *pooling* layers$\}^*$ $+$ MLP

Convolutional layer

Learned convolution filters $\rightarrow$ feature maps



Special case of fully connected layer:

- Only local spatial connections
- Location invariance

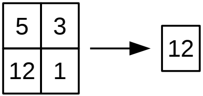
$\Rightarrow$ Makes sense in image domain (or text, time series,...)

Recap: Convolutional neural networks (CNNs)

Pooling layers

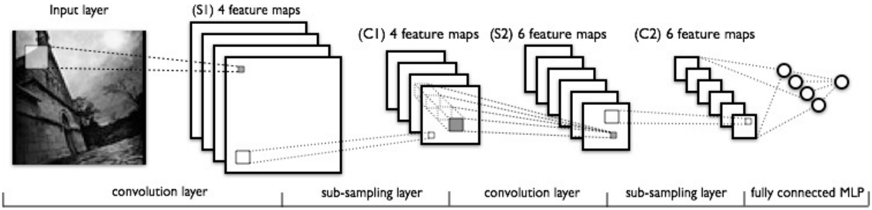
Subsample feature maps

- Increase *receptive field* ☺
- Downgrade resolution
  - Robustness to spatial variation ☺
  - Not good for *pixelwise* labeling ☹



Max pooling

Overall categorization CNN

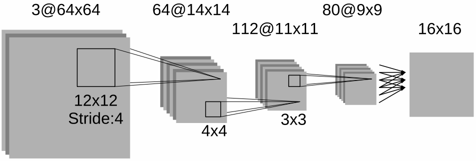


Source: deeplearning.net

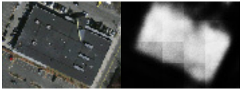
Remote sensing: *dense* labeling with CNNs?

Pioneering works:

1. Predict and entire patch centered in input patch (Mnih, 2013)



   - Allows to learn "in-patch location" priors
     → Patch border artifacts



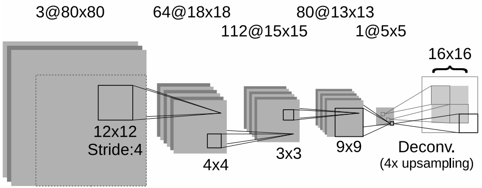
2. Predict the central pixel in the patch and shift one by one
   (e.g., Paisitkriangkrai et al., CVPR Earthvision 2015)
   - Too many redundant computations

State of the art: fully convolutional network (FCN)

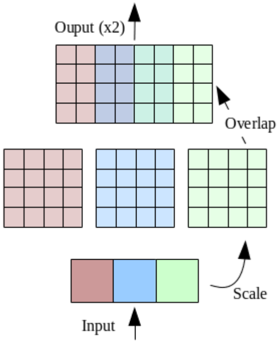
Fully convolutional networks (FCNs)

[Long et al., CVPR 2015]

- Convolutions & subsampling
- "Deconvolutional" layer to upsample



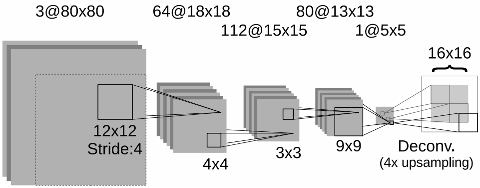
Proposed FCN for remote sensing



Deconv. layer

E. Maggiori, Y. Tarabalka, G. Charpiat, P. Alliez. "Fully convolutional neural networks for remote sensing image classification", IGARSS 2016.

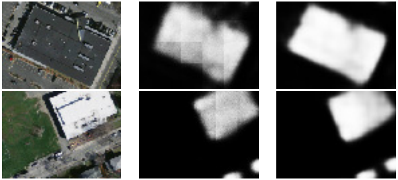
State of the art: fully convolutional network (FCN)



- Output size varies with input size (with fixed number of parameters)
- Location invariant (same logic used to compute every output)
- Avoid redundant computations
- *Especially* relevant in remote sensing (arbitrary tiling, azimuth)

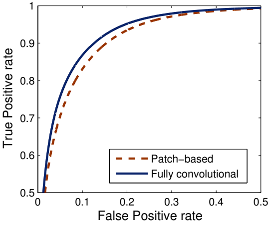
FCN: experiment

- Patch artifacts removed by construction
- More accurate
- 10x faster



Input Patch-based FCN

Massachusetts dataset (Mnih, 2015)



Once again...

Imposing sensible restrictions

- improves the learning process,
- reduces execution times.

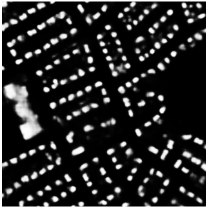
FCN: experiment

Massachusetts dataset

[Dataset: Mnih, 2013]

Color input

Reference



FCN

SVM

- Classification of 22.5 km^2 (1 m resolution): 8.5 seconds

Dealing with imperfect training data

Frequent misregistration/omission in large-scale data sources:

Pléiades image + OpenStreetMap (OSM) over Loire department

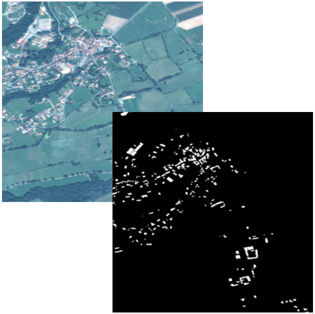
Possible strategy

Two-step training process:

1. Pretrain on large amounts of imperfect data
   → Learn dataset generalities
2. Fine-tune on a small piece of manually labeled reference

Imperfect training data: experiment

1. Pretrain on 22.5 km^2 Pléiades + OpenStreetMap data
2. Fine-tune on a manually labeled tile (2.5km^2, 3000×3000 px.)



Fine-tuning tile

Close-up

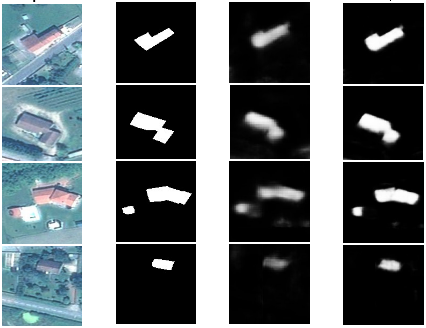
E. Maggiori, Y. Tarabalka, G. Charpiat, P. Alliez. "Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification", TGRS 2017.

Imperfect training data: experiment

Test on a different manually labeled tile

Test tile

Results



Method	Accuracy	AUC*	IoU
FCN	99.13%	0.98154	47%
FCN + FT	99.57%	0.99836	72%

*AUC: area under the ROC curve

Concluding remarks

- Fully convolutional networks for remote sensing classification
  - FCNs have now become the standard dense labeling architecture
  - Other FCN comparisons (Kampffmeyer et al., 2016; Sherrah, 2016)
- Combining OSM + manual data sources to improve predictions
  - Growing interest in crowd-sourced data
    - Correcting OSM roads (Mattyus et al., 2016)
    - Combining diverse data sources (Kaiser, 2016)
    - OSM as an additional input (Audebert et al., 2017)

Concluding remarks

Recognition/localization trade-off

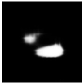
Subsampling:

- increases the receptive field (improving recognition)
- reduces resolution (hampering localization)

⇒ “Blobby” objects

Input

Ref.

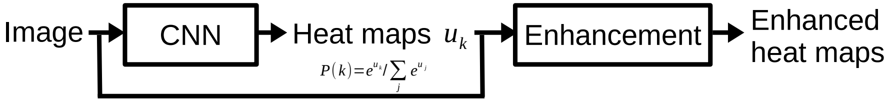


CNN

Solutions

1. Post-process the CNN’s output (e.g., CRF)
2. Use innovative (e.g., multiscale) architectures

Enhancing CNNs' outputs



Recent approaches

- CNN + Fully connected CRF (Chen et al., ICML 2015)
- CNN + Fully connected CRF as RNN (Zheng et al., CVPR 2015)
- CNN + Domain transform (Chen et al., CVPR 2016)

In remote sensing:

- CNN + CRF (Paisitkriangkrai et al., CVPR Worshops 2015)
- CNN + Fully connected CRF (Marmanis et al., ISPRS 2015; Sherrah 2016,...)

Goal

Learn iterative enhancement process

Partial differential equations (PDEs)

- **One strategy:** progressively enhance the score maps by using partial differential equations
- Given heat maps u_k, image I:
  - Heat flow *(Smooths out u_k)*

$$\frac{\partial u_k(x)}{\partial t} = \mathrm{div}(\nabla u_k(x))$$

- **Divergence** represents the volume density of the outward flux of a vector field from an infinitesimal volume around a given point

Partial differential equations (PDEs)

Given heat maps u_k, image I:

- Heat flow
  (Smooths out u_k)
  $$\frac{\partial u_k(x)}{\partial t} = \mathrm{div}(\nabla u_k(x))$$
- Perona-Malik
  Edge-stopping function $g(\nabla I, x)$
  $$\frac{\partial u_k(x)}{\partial t} = \mathrm{div}(g(\nabla I, x)\nabla u_k(x))$$
- Anisotropic diffusion
  Diffusion tensor $D(I, x)$
  $$\frac{\partial u_k(x)}{\partial t} = \mathrm{div}(D(\nabla I, x)\nabla u_k(x))$$
- Geodesic active contours
  Edge-stopping function $g(\nabla I, x)$
  $$\frac{\partial u_k(x)}{\partial t} = |\nabla u_k(x)|\mathrm{div}\left(g(\nabla I, x)\frac{\nabla u_k(x)}{|\nabla u_k(x)|}\right)$$
- ...

Partial differential equations (PDEs)

- Different PDE approaches can be devised to enhance classification maps
- Several choices must be made to select the appropriate PDE and tailor it to the considered problem
  - For example, edge-stopping function $g(\nabla I, x)$ must be chosen

Partial differential equations (PDEs)

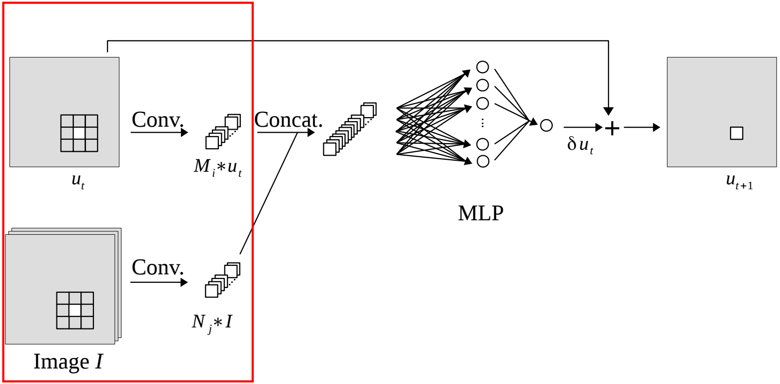
- Different PDE approaches can be devised to enhance classification maps
- Several choices must be made to select the appropriate PDE and tailor it to the considered problem
  - For example, edge-stopping function $g(\nabla I, x)$ must be chosen
- Can we let a machine learning approach discover by itself a useful iterative process?

Partial differential equations (PDEs)

- Different PDE approaches can be devised to enhance classification maps
- Several choices must be made to select the appropriate PDE and tailor it to the considered problem
    - For example, edge-stopping function $g(\nabla I, x)$ must be chosen
- Can we let a machine learning approach discover by itself a useful iterative process?
- PDEs are usually discretized in space by using finite differences
    - Derivatives as discrete convolution filters

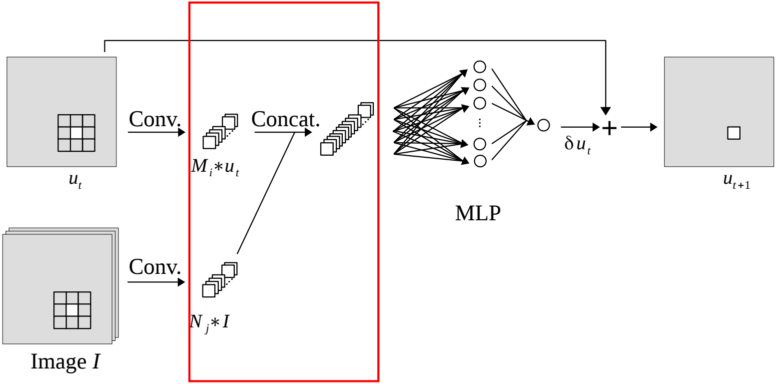
A generic enhancement process

- Differential operations ($\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial^2}{\partial x \partial y}, \frac{\partial^2}{\partial x^2}, \ldots$) applied on u_k and image I
- Implemented as convolutions: $M_i * u_k$, $N_j * I$
  $\{M_1, M_2, \ldots\}$, $\{N_1, N_2, \ldots\}$ conv. kernels (e.g., Sobel filters)



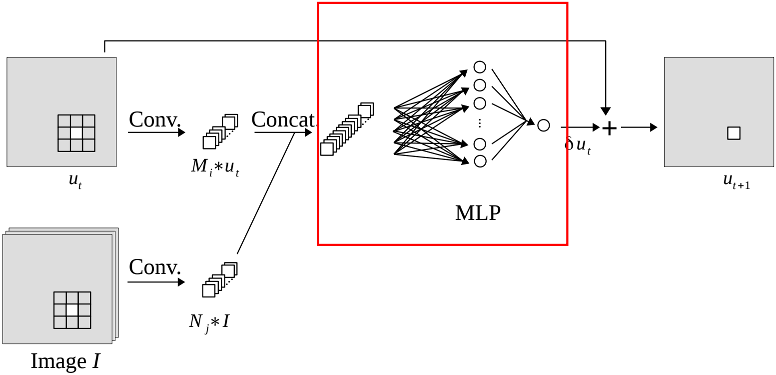
A generic enhancement process

- $\Phi(u_k, I) = \{M_i * u_k, \;\; N_j * I \; ; \;\; \forall i, j\}$, set of responses



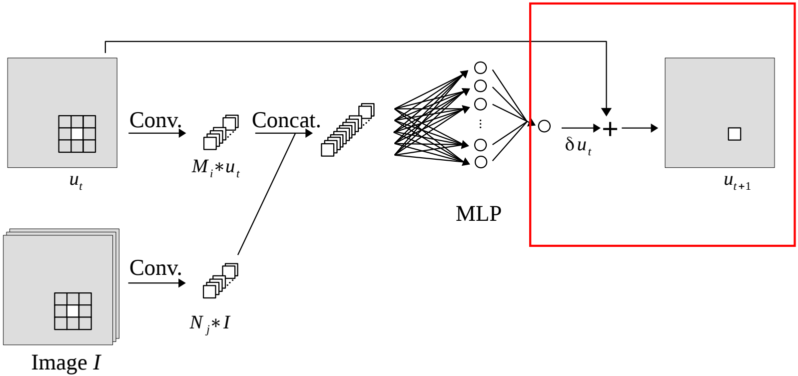
A generic enhancement process

- Overall update on u_k at x: $\delta u_k(x) = f_k\left(\Phi(u_k, I)(x)\right)$
- Class-specific f_k, implemented as multilayer perceptron
- M_i and N_j convey spatial reasoning (e.g., gradients), f_k their combination (e.g., products)



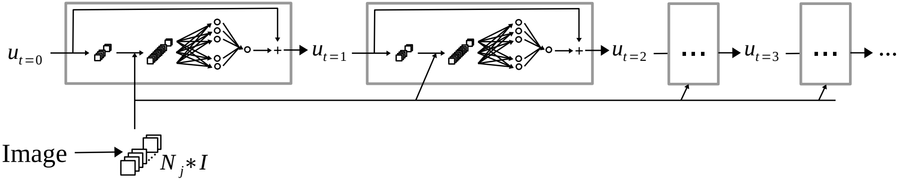
A generic enhancement process

- Discretized in time:
  $u_{k,t+1}(x) = u_{k,t}(x) + \delta u_{k,t}(x)$, overall update δ



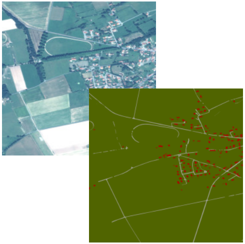
Iterative processes as recurrent neural networks (RNNs)

- "Unroll" iterations
- Enforce weight sharing along iterations
- Train by backpropagation as usual ("through time")
- Every iteration is meant to progressively refine the classification maps



Experiments

- FCN trained on Pléiades + OSM data
- Manually labeled tiles
  for RNN training/testing
- Unroll 5 iterations
- 32 M_i and 32 N_j
- MLP: 1 hidden layer, 32 neurons

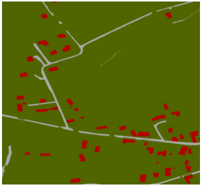


Building, Road, Background

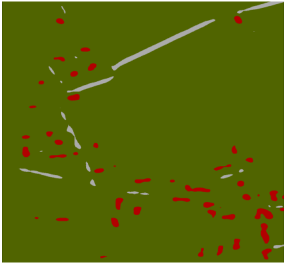
E. Maggiori, G. Charpiat, Y. Tarabalka, P. Alliez. "Recurrent Neural Networks to Correct Satellite Image Classification Maps". TGRS 2017.

Experiments

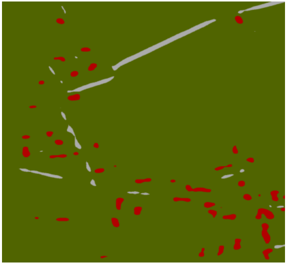
Color input



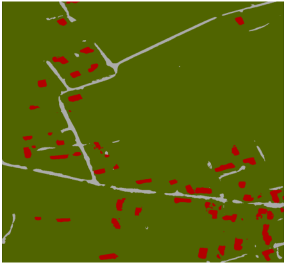
Reference



Coarse CNN

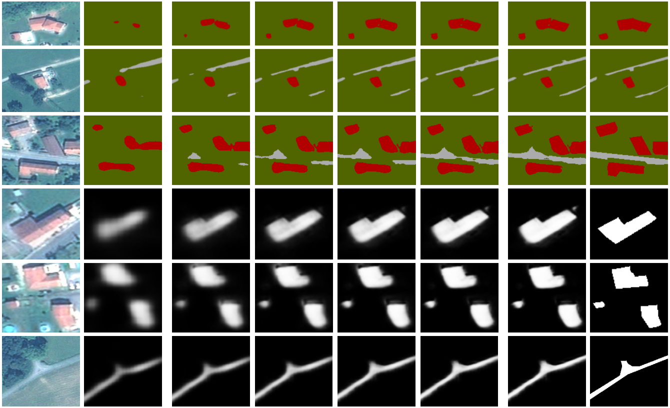


→ RNN enhancement →

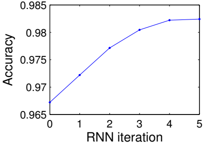


RNN output

Experiments

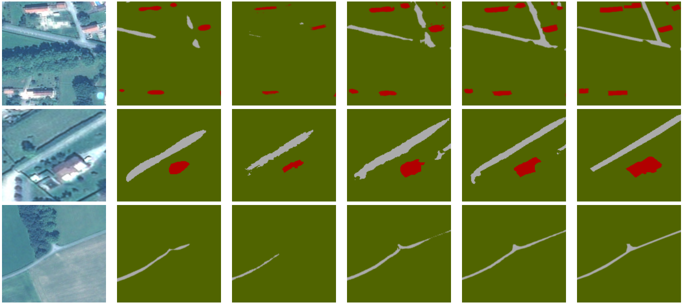


Color (RNN input) | CNN map | — Intermediate RNN iterations — | RNN output | Reference



Experiments

Comparison

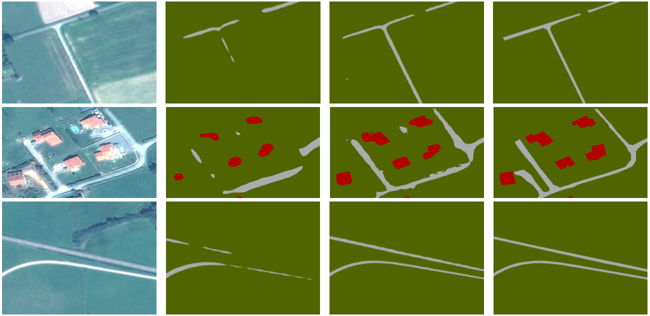


Color image | Coarse CNN | CNN+CRF | Class-agnostic CNN+RNN | CNN+RNN | Reference

Method	Overall accuracy	Mean IoU	Class-specific IoU Build.	Road	Backg.
CNN	96.72	48.32	38.92	9.34	96.69
CNN+CRF	96.96	44.15	29.05	6.62	96.78
Class-agn. CNN+RNN	97.78	65.30	59.12	39.03	97.74
CNN+RNN	**98.24**	**72.90**	**69.16**	**51.32**	**98.20**

Experiments

More examples



Color image | Coarse CNN | RNN output | Reference

Concluding remarks

- A small set of accurately labeled data can be used to enhance classification maps
- We can *learn* the specifics of an iterative enhancement process
- Removing the recurrence constraint significantly deteriorates results

Yielding high-resolution outputs

Very recent works

Four families of architectures:

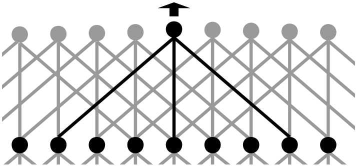
- *Dilation* (Chen et al., 2015; Dubrovina et al., 2016,...)
- *Unpooling/deconv.* (Noh et al., 2015; Volpi and Tuia, 2016,...)
- *Skip networks* (Long et al., 2015; Badrinarayanan et al., 2015,...)
- *MLP network* (Maggiori et al., 2017 ⇒ attend talk of E. Maggiori (July 28, 13:40, ballroomB))

Ultimate goal: CNN architecture that addresses recognition/localization trade-off

Analysis of SoA: E. Maggiori, Y. Tarabalka, G. Charpiat, P. Alliez. "High-Resolution Semantic Labeling with Convolutional Neural Networks", arXiv, Nov. 2016.

Dilation networks

- Based on the shift-and-stitch approach:
  - Conduct predictions at different offsets to produce low-resolution outputs
  - Interleave these outputs to compose the final high-resolution result
- Such an interleaving can be implemented as convolutions on non-contiguous locations

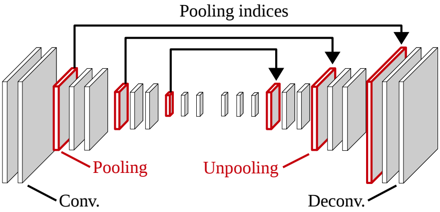


$\Rightarrow$ Larger context without introducing more parameters

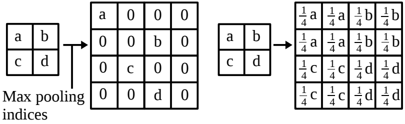
- Not robust to spatial deformation
  (e.g., detect road located *exactly* 5px away)

Unpooling/deconvolution networks

- The CNN is “mirrored” to learn the deconvolution:



- Max (left) and average (right) unpooling

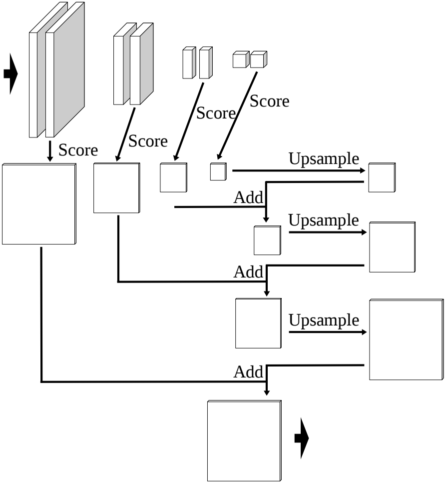


- The depth of deconv. networks is significantly larger ($\sim$ twice FCN)

Skip networks

1. Extract intermediate features
2. Classify
3. Upsample/add (pairwise)

- Addresses trade-off
- Inflexible/arbitrary at combining resolutions



MLP network

Premise

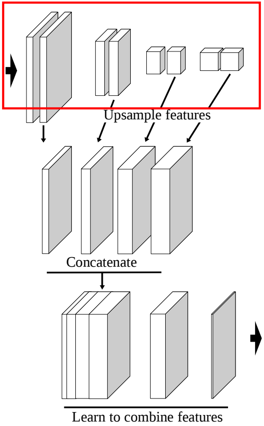
- CNNs do not need to “see” everywhere at the same resolution
- E.g., to classify central pixel:

Full resolution context

Full resolution only near center

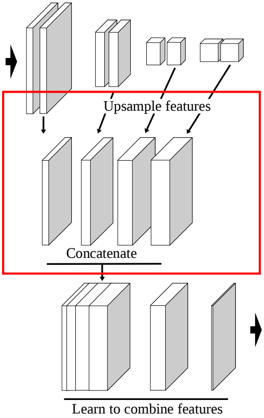
⇒ Combine resolutions to address trade-off, in a flexible way

MLP network



Base FCN

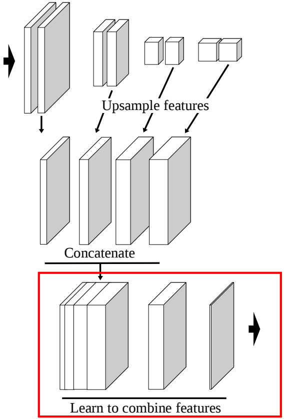
MLP network



- Extract intermediate features
- Upsample to the highest res.
- Concatenate

⇒ Pool of features (e.g., edge detectors, object detectors)

MLP network



- Multi-layer perceptron (MLP) learns how to combine those features
  $\Rightarrow$ Output classif. map
- Pixel by pixel (series of 1×1 convolutional layers)
  $\Rightarrow$ 128 hidden neurons, nonlinear activation
- Addresses trade-off in a flexible way

Experiments

Datasets

ISPRS 2D semantic labeling contest:

Vaihingen (9 cm)

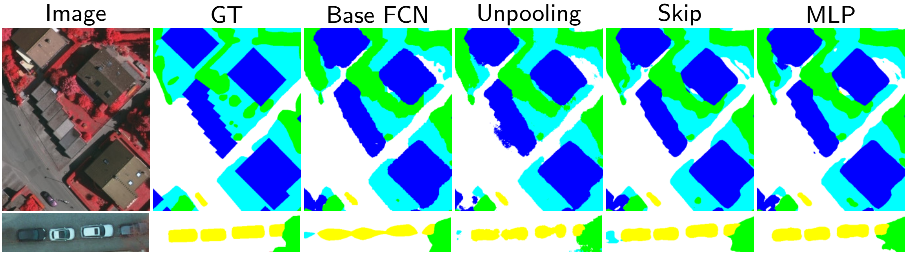
Potsdam (5 cm)

- Color infra-red + Elevation model

Results: Base FCN vs derived architectures

Vaihingen	Imp. surf.	Building	Low veg.	Tree	Car	Mean F1	Acc.
Base FCN	91.46	94.88	79.19	87.89	72.25	85.14	88.61
Unpooling	91.17	95.16	79.06	87.78	69.49	84.54	88.55
Skip	91.66	95.02	79.13	88.11	77.96	86.38	88.80
MLP	**91.69**	**95.24**	**79.44**	**88.12**	**78.42**	**86.58**	**88.92**

Potsdam	Imp. surf.	Building	Low veg.	Tree	Car	Clutter	Mean F1	Acc.
Base FCN	88.33	93.97	84.11	80.30	86.13	75.35	84.70	86.20
Unpooling	87.00	92.86	82.93	78.04	84.85	72.47	83.03	84.67
Skip	89.27	94.21	84.73	**81.23**	93.47	75.18	86.35	86.89
MLP	**89.31**	**94.37**	**84.83**	81.10	**93.56**	**76.54**	**86.62**	**87.02**



Classes: Impervious surface (white), Building (blue), Low veget. (cyan), Tree (green), Car (yellow), Clutter (red).

Results: Comparison with other methods

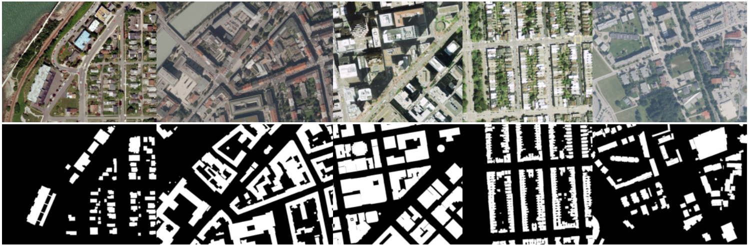
Vaihingen	Imp. surf.	Build.	Low veg.	Tree	Car	F1	Acc.
CNN+RF	88.58	94.23	76.58	86.29	67.58	82.65	86.52
CNN+RF+CRF	89.10	94.30	77.36	86.25	71.91	83.78	86.89
Deconvolution						83.58	87.83
Dilation	90.19	94.49	77.69	87.24	76.77	85.28	87.70
Dilation + CRF	90.41	94.73	78.25	87.25	75.57	85.24	87.90
MLP	**91.69**	**95.24**	**79.44**	**88.12**	**78.42**	**86.58**	**88.92**

Submission of the MLP-network results to ISPRS server

- Overall accuracy: 89.5%
- Second place (out of 29) at the time of submission
- Significantly simpler and faster than other methods

New deep approaches are coming!

⇒ https://project.inria.fr/aerialimagelabeling/:

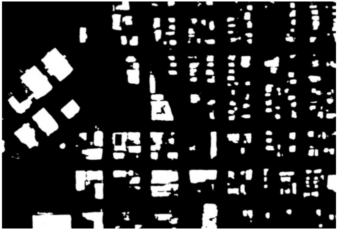


Leaderboard

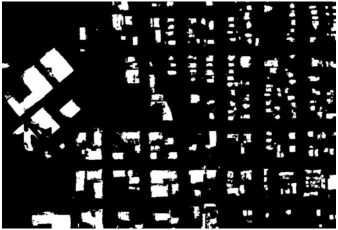
Method	Date	Bellingham		Bloomington		Innsbruck		San Francisco		East Tyrol		Overall	
		IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.	IoU	Acc.
Inria1	3-Jan-17	52.91	95.14	46.08	94.95	58.12	95.16	57.84	86.05	59.03	96.40	55.82	93.54
Inria2	3-Jan-17	56.11	95.37	50.40	95.27	61.03	95.37	61.38	87.00	62.51	96.61	59.31	93.93
TeraDeep	5-May-17	58.08	95.88	53.38	95.61	59.47	95.26	64.34	88.71	62.00	96.57	60.95	94.41
RMIT	16-July-17	57.30	95.97	51.78	95.60	60.70	95.69	66.71	89.23	59.73	96.59	61.73	94.62

New deep approaches are coming! - some results

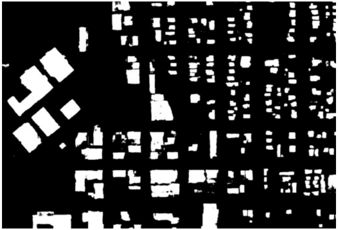
Input



Inria



TeraDeep



RMIT

Concluding remarks

- Modern CNN architertures address well recognition/localization trade-off
- Good generalisation potential
- How to implement?
  - Some codes: `https://github.com/emaggiori/CaffeRemoteSensing`
    - Extending Caffe framework for pixelwise labeling of aerial remote sensing imagery.

Concluding remarks

Key to CNNs' success

Imposing *sensible* restrictions to neuronal connections reduces optimization search space w.l.o.g:

- Better minima → better accuracy
- Computational efficiency

⇒ Win-win

A recurrent pattern: simpler is better

- FCNs → More accurate and 10x faster
- RNNs → Removing recurrence significantly degrades results
- MLP net → More accurate than more complicated models

Concluding remarks

The "no free lunch" principle in machine learning (Wolper, 1996)

There is no such thing as a universally better classifier. A classifier is better under certain assumptions.

- CNNs exploit the properties of images particularly well
- Shifting efforts from feature engineering to network engineering
- Good *payoff* of the efforts,
  e.g., learning better features than handmade ones,
  convolutions $\rightarrow$ GPUs, borrowing pretrained network
- Still many remaining challenges to solve:
  $\rightarrow$ Rounded corners, unstructured outputs, etc.
  ...
  $\rightarrow$ Classifying the Earth