USE TENSORBOARD FOR MODEL SUMMARY

* Add dropout not in encoder because it hinders learning meaningful features (and pretrained on imagenet?)
* Decoder upsamples features and combines them with skip connections from the encoder to generate final segmentation map.
* Use dropout rate of 0.2-0.5 (or 0.1?)
* Added after upsampling, after concatenation or before convolution (start with after upsampling)

“In traditional Machine Learning, most of the applied features need to be identified by a domain expert, in order to reduce the complexity of the data and make patterns more visible to learning algorithms. On the other hand a neural network has strong mapping and generalization abilities.

CNN advantages: Explain how they can capture context on different spatial levels (“network learns relatively low-dimensional yet powerful representations that greatly surpass the effectiveness of handcrafted features.”) CNNs learn hierarchical representations of images, with each layer capturing increasingly complex features. To learn these representations effectively, CNNs need a significant amount of diverse examples.

One disadvantage that comes with CNNs is requirement of big data to perform well (23, 24 in Shorten). Convolutional Networks contain a huge number of training parameters. Showing only a small amount of data leads to overfitting, adjusting to the training samples too well such that the model generalizes poorly on unseen data. Some authors discuss that data proportional to the amount of examples is required, which would means millions with regard to UNet.

“A substantial amount of data is required to capture the diversity present in the real-world domain and generalize on unseen data that can contain different scenarious and conditions.”

However, collecting large amounts of data representative for the entire Arctic is challenging. Even more challenging, data needs to be labelled. This task requires a lot of time and human expertise, as melt ponds are hardly detected in IR images.

To successfully train CNN models even with small data, different methods have been developed. Techniques that can increase the size of the training dataset are Patch Extraction (reduced image size but larger amount) and Data Augmentation (generating new samples by transforming original images). Other regularization techniques that combat overfitting in model design are Pretraining (pretrain on big dataset and finetune for problem at hand), Dropout and Batch Normalization.

Examples of meaningful results even with small data.

LITERATURE  
CNN small data overfitting

Patch extraction

Patch Extraction: Can tackle overfitting?

Dropout: Dropout tackles the problem of overfitting by disregarding random subsets of neurons during training. This increases the robustness of the model, has been shown successful in …, … and …

“Simple technique useful to avoid overfitting in which random connections are disabled during training.

Batch Normalization: Batch Normalization refers to normalizing the activations in a layer. This has been shown to remove the bias and account for more stable training. In our model, batch normalization is used in encoder and decoder before activation.

Transfer Learning: In transfer learning, a pretrained model, trained on a large dataset, is used as a starting point for the training at hand. Transfer Learning has been shown to lead to faster model convergence, saving training time and resources (reference) and especially improve learning with small datasets (reference). Pretrained models capture general pattern knowledge which helps to reduce overfitting.

In image segmentation, ImageNet can be used as classification dataset to pretrain the encoder part of the model. The pretrained model is now fine-tuned with the target dataset.

There are different opinions regarding the similarity of pretraining and target domain. In general, it is said that the more similar the two tasks, the better transfer learning will work. However, … could show equally well performing models. The assumption is that features learned by the pretrained model are generally applicable.

METHODOLOGY

encoder\_weights: imagenet, encoder\_freeze: False (🡪 encoder is fine-tuned)

DISCUSSION

In future, different datasets could be used. As there is a large amount of unlabelled data available, this could be used as a better performing feature extractor.

Resnet34: stage\_unit1\_relu1, stage3\_unit1\_relu1, stage2\_unit1\_relu1, reul0

Include\_top=False (whether to include the fully connected layer at the top of the model)

Layer (type) Output Shape Param # Connected to

==================================================================================================

data (InputLayer) [(None, 256, 256, 3 0 []

)]

bn\_data (BatchNormalization) (None, 256, 256, 3) 9 ['data[0][0]']

zero\_padding2d (ZeroPadding2D) (None, 262, 262, 3) 0 ['bn\_data[0][0]']

conv0 (Conv2D) (None, 128, 128, 64 9408 ['zero\_padding2d[0][0]']

)

bn0 (BatchNormalization) (None, 128, 128, 64 256 ['conv0[0][0]']

)

relu0 (Activation) (None, 128, 128, 64 0 ['bn0[0][0]']

)

zero\_padding2d\_1 (ZeroPadding2 (None, 130, 130, 64 0 ['relu0[0][0]']

D) )

pooling0 (MaxPooling2D) (None, 64, 64, 64) 0 ['zero\_padding2d\_1[0][0]']

stage1\_unit1\_bn1 (BatchNormali (None, 64, 64, 64) 256 ['pooling0[0][0]']

zation)

stage1\_unit1\_relu1 (Activation (None, 64, 64, 64) 0 ['stage1\_unit1\_bn1[0][0]']

)

zero\_padding2d\_2 (ZeroPadding2 (None, 66, 66, 64) 0 ['stage1\_unit1\_relu1[0][0]']

D)

stage1\_unit1\_conv1 (Conv2D) (None, 64, 64, 64) 36864 ['zero\_padding2d\_2[0][0]']

stage1\_unit1\_bn2 (BatchNormali (None, 64, 64, 64) 256 ['stage1\_unit1\_conv1[0][0]']

zation)

stage1\_unit1\_relu2 (Activation (None, 64, 64, 64) 0 ['stage1\_unit1\_bn2[0][0]']

)

zero\_padding2d\_3 (ZeroPadding2 (None, 66, 66, 64) 0 ['stage1\_unit1\_relu2[0][0]']

D)

stage1\_unit1\_conv2 (Conv2D) (None, 64, 64, 64) 36864 ['zero\_padding2d\_3[0][0]']

stage1\_unit1\_sc (Conv2D) (None, 64, 64, 64) 4096 ['stage1\_unit1\_relu1[0][0]']

add (Add) (None, 64, 64, 64) 0 ['stage1\_unit1\_conv2[0][0]',

'stage1\_unit1\_sc[0][0]']

stage1\_unit2\_bn1 (BatchNormali (None, 64, 64, 64) 256 ['add[0][0]']

zation)

stage1\_unit2\_relu1 (Activation (None, 64, 64, 64) 0 ['stage1\_unit2\_bn1[0][0]']

)

zero\_padding2d\_4 (ZeroPadding2 (None, 66, 66, 64) 0 ['stage1\_unit2\_relu1[0][0]']

D)

stage1\_unit2\_conv1 (Conv2D) (None, 64, 64, 64) 36864 ['zero\_padding2d\_4[0][0]']

stage1\_unit2\_bn2 (BatchNormali (None, 64, 64, 64) 256 ['stage1\_unit2\_conv1[0][0]']

zation)

stage1\_unit2\_relu2 (Activation (None, 64, 64, 64) 0 ['stage1\_unit2\_bn2[0][0]']

)

zero\_padding2d\_5 (ZeroPadding2 (None, 66, 66, 64) 0 ['stage1\_unit2\_relu2[0][0]']

D)

stage1\_unit2\_conv2 (Conv2D) (None, 64, 64, 64) 36864 ['zero\_padding2d\_5[0][0]']

add\_1 (Add) (None, 64, 64, 64) 0 ['stage1\_unit2\_conv2[0][0]',

'add[0][0]']

stage1\_unit3\_bn1 (BatchNormali (None, 64, 64, 64) 256 ['add\_1[0][0]']

zation)

stage1\_unit3\_relu1 (Activation (None, 64, 64, 64) 0 ['stage1\_unit3\_bn1[0][0]']

)

zero\_padding2d\_6 (ZeroPadding2 (None, 66, 66, 64) 0 ['stage1\_unit3\_relu1[0][0]']

D)

stage1\_unit3\_conv1 (Conv2D) (None, 64, 64, 64) 36864 ['zero\_padding2d\_6[0][0]']

stage1\_unit3\_bn2 (BatchNormali (None, 64, 64, 64) 256 ['stage1\_unit3\_conv1[0][0]']

zation)

stage1\_unit3\_relu2 (Activation (None, 64, 64, 64) 0 ['stage1\_unit3\_bn2[0][0]']

)

zero\_padding2d\_7 (ZeroPadding2 (None, 66, 66, 64) 0 ['stage1\_unit3\_relu2[0][0]']

D)

stage1\_unit3\_conv2 (Conv2D) (None, 64, 64, 64) 36864 ['zero\_padding2d\_7[0][0]']

add\_2 (Add) (None, 64, 64, 64) 0 ['stage1\_unit3\_conv2[0][0]',

'add\_1[0][0]']

stage2\_unit1\_bn1 (BatchNormali (None, 64, 64, 64) 256 ['add\_2[0][0]']

zation)

stage2\_unit1\_relu1 (Activation (None, 64, 64, 64) 0 ['stage2\_unit1\_bn1[0][0]']

)

zero\_padding2d\_8 (ZeroPadding2 (None, 66, 66, 64) 0 ['stage2\_unit1\_relu1[0][0]']

D)

stage2\_unit1\_conv1 (Conv2D) (None, 32, 32, 128) 73728 ['zero\_padding2d\_8[0][0]']

stage2\_unit1\_bn2 (BatchNormali (None, 32, 32, 128) 512 ['stage2\_unit1\_conv1[0][0]']

zation)

stage2\_unit1\_relu2 (Activation (None, 32, 32, 128) 0 ['stage2\_unit1\_bn2[0][0]']

)

zero\_padding2d\_9 (ZeroPadding2 (None, 34, 34, 128) 0 ['stage2\_unit1\_relu2[0][0]']

D)

stage2\_unit1\_conv2 (Conv2D) (None, 32, 32, 128) 147456 ['zero\_padding2d\_9[0][0]']

stage2\_unit1\_sc (Conv2D) (None, 32, 32, 128) 8192 ['stage2\_unit1\_relu1[0][0]']

add\_3 (Add) (None, 32, 32, 128) 0 ['stage2\_unit1\_conv2[0][0]',

'stage2\_unit1\_sc[0][0]']

stage2\_unit2\_bn1 (BatchNormali (None, 32, 32, 128) 512 ['add\_3[0][0]']

zation)

stage2\_unit2\_relu1 (Activation (None, 32, 32, 128) 0 ['stage2\_unit2\_bn1[0][0]']

)

zero\_padding2d\_10 (ZeroPadding (None, 34, 34, 128) 0 ['stage2\_unit2\_relu1[0][0]']

2D)

stage2\_unit2\_conv1 (Conv2D) (None, 32, 32, 128) 147456 ['zero\_padding2d\_10[0][0]']

stage2\_unit2\_bn2 (BatchNormali (None, 32, 32, 128) 512 ['stage2\_unit2\_conv1[0][0]']

zation)

stage2\_unit2\_relu2 (Activation (None, 32, 32, 128) 0 ['stage2\_unit2\_bn2[0][0]']

)

zero\_padding2d\_11 (ZeroPadding (None, 34, 34, 128) 0 ['stage2\_unit2\_relu2[0][0]']

2D)

stage2\_unit2\_conv2 (Conv2D) (None, 32, 32, 128) 147456 ['zero\_padding2d\_11[0][0]']

add\_4 (Add) (None, 32, 32, 128) 0 ['stage2\_unit2\_conv2[0][0]',

'add\_3[0][0]']

stage2\_unit3\_bn1 (BatchNormali (None, 32, 32, 128) 512 ['add\_4[0][0]']

zation)

stage2\_unit3\_relu1 (Activation (None, 32, 32, 128) 0 ['stage2\_unit3\_bn1[0][0]']

)

zero\_padding2d\_12 (ZeroPadding (None, 34, 34, 128) 0 ['stage2\_unit3\_relu1[0][0]']

2D)

stage2\_unit3\_conv1 (Conv2D) (None, 32, 32, 128) 147456 ['zero\_padding2d\_12[0][0]']

stage2\_unit3\_bn2 (BatchNormali (None, 32, 32, 128) 512 ['stage2\_unit3\_conv1[0][0]']

zation)

stage2\_unit3\_relu2 (Activation (None, 32, 32, 128) 0 ['stage2\_unit3\_bn2[0][0]']

)

zero\_padding2d\_13 (ZeroPadding (None, 34, 34, 128) 0 ['stage2\_unit3\_relu2[0][0]']

2D)

stage2\_unit3\_conv2 (Conv2D) (None, 32, 32, 128) 147456 ['zero\_padding2d\_13[0][0]']

add\_5 (Add) (None, 32, 32, 128) 0 ['stage2\_unit3\_conv2[0][0]',

'add\_4[0][0]']

stage2\_unit4\_bn1 (BatchNormali (None, 32, 32, 128) 512 ['add\_5[0][0]']

zation)

stage2\_unit4\_relu1 (Activation (None, 32, 32, 128) 0 ['stage2\_unit4\_bn1[0][0]']

)

zero\_padding2d\_14 (ZeroPadding (None, 34, 34, 128) 0 ['stage2\_unit4\_relu1[0][0]']

2D)

stage2\_unit4\_conv1 (Conv2D) (None, 32, 32, 128) 147456 ['zero\_padding2d\_14[0][0]']

stage2\_unit4\_bn2 (BatchNormali (None, 32, 32, 128) 512 ['stage2\_unit4\_conv1[0][0]']

zation)

stage2\_unit4\_relu2 (Activation (None, 32, 32, 128) 0 ['stage2\_unit4\_bn2[0][0]']

)

zero\_padding2d\_15 (ZeroPadding (None, 34, 34, 128) 0 ['stage2\_unit4\_relu2[0][0]']

2D)

stage2\_unit4\_conv2 (Conv2D) (None, 32, 32, 128) 147456 ['zero\_padding2d\_15[0][0]']

add\_6 (Add) (None, 32, 32, 128) 0 ['stage2\_unit4\_conv2[0][0]',

'add\_5[0][0]']

stage3\_unit1\_bn1 (BatchNormali (None, 32, 32, 128) 512 ['add\_6[0][0]']

zation)

stage3\_unit1\_relu1 (Activation (None, 32, 32, 128) 0 ['stage3\_unit1\_bn1[0][0]']

)

zero\_padding2d\_16 (ZeroPadding (None, 34, 34, 128) 0 ['stage3\_unit1\_relu1[0][0]']

2D)

stage3\_unit1\_conv1 (Conv2D) (None, 16, 16, 256) 294912 ['zero\_padding2d\_16[0][0]']

stage3\_unit1\_bn2 (BatchNormali (None, 16, 16, 256) 1024 ['stage3\_unit1\_conv1[0][0]']

zation)

stage3\_unit1\_relu2 (Activation (None, 16, 16, 256) 0 ['stage3\_unit1\_bn2[0][0]']

)

zero\_padding2d\_17 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit1\_relu2[0][0]']

2D)

stage3\_unit1\_conv2 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_17[0][0]']

stage3\_unit1\_sc (Conv2D) (None, 16, 16, 256) 32768 ['stage3\_unit1\_relu1[0][0]']

add\_7 (Add) (None, 16, 16, 256) 0 ['stage3\_unit1\_conv2[0][0]',

'stage3\_unit1\_sc[0][0]']

stage3\_unit2\_bn1 (BatchNormali (None, 16, 16, 256) 1024 ['add\_7[0][0]']

zation)

stage3\_unit2\_relu1 (Activation (None, 16, 16, 256) 0 ['stage3\_unit2\_bn1[0][0]']

)

zero\_padding2d\_18 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit2\_relu1[0][0]']

2D)

stage3\_unit2\_conv1 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_18[0][0]']

stage3\_unit2\_bn2 (BatchNormali (None, 16, 16, 256) 1024 ['stage3\_unit2\_conv1[0][0]']

zation)

stage3\_unit2\_relu2 (Activation (None, 16, 16, 256) 0 ['stage3\_unit2\_bn2[0][0]']

)

zero\_padding2d\_19 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit2\_relu2[0][0]']

2D)

stage3\_unit2\_conv2 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_19[0][0]']

add\_8 (Add) (None, 16, 16, 256) 0 ['stage3\_unit2\_conv2[0][0]',

'add\_7[0][0]']

stage3\_unit3\_bn1 (BatchNormali (None, 16, 16, 256) 1024 ['add\_8[0][0]']

zation)

stage3\_unit3\_relu1 (Activation (None, 16, 16, 256) 0 ['stage3\_unit3\_bn1[0][0]']

)

zero\_padding2d\_20 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit3\_relu1[0][0]']

2D)

stage3\_unit3\_conv1 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_20[0][0]']

stage3\_unit3\_bn2 (BatchNormali (None, 16, 16, 256) 1024 ['stage3\_unit3\_conv1[0][0]']

zation)

stage3\_unit3\_relu2 (Activation (None, 16, 16, 256) 0 ['stage3\_unit3\_bn2[0][0]']

)

zero\_padding2d\_21 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit3\_relu2[0][0]']

2D)

stage3\_unit3\_conv2 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_21[0][0]']

add\_9 (Add) (None, 16, 16, 256) 0 ['stage3\_unit3\_conv2[0][0]',

'add\_8[0][0]']

stage3\_unit4\_bn1 (BatchNormali (None, 16, 16, 256) 1024 ['add\_9[0][0]']

zation)

stage3\_unit4\_relu1 (Activation (None, 16, 16, 256) 0 ['stage3\_unit4\_bn1[0][0]']

)

zero\_padding2d\_22 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit4\_relu1[0][0]']

2D)

stage3\_unit4\_conv1 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_22[0][0]']

stage3\_unit4\_bn2 (BatchNormali (None, 16, 16, 256) 1024 ['stage3\_unit4\_conv1[0][0]']

zation)

stage3\_unit4\_relu2 (Activation (None, 16, 16, 256) 0 ['stage3\_unit4\_bn2[0][0]']

)

zero\_padding2d\_23 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit4\_relu2[0][0]']

2D)

stage3\_unit4\_conv2 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_23[0][0]']

add\_10 (Add) (None, 16, 16, 256) 0 ['stage3\_unit4\_conv2[0][0]',

'add\_9[0][0]']

stage3\_unit5\_bn1 (BatchNormali (None, 16, 16, 256) 1024 ['add\_10[0][0]']

zation)

stage3\_unit5\_relu1 (Activation (None, 16, 16, 256) 0 ['stage3\_unit5\_bn1[0][0]']

)

zero\_padding2d\_24 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit5\_relu1[0][0]']

2D)

stage3\_unit5\_conv1 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_24[0][0]']

stage3\_unit5\_bn2 (BatchNormali (None, 16, 16, 256) 1024 ['stage3\_unit5\_conv1[0][0]']

zation)

stage3\_unit5\_relu2 (Activation (None, 16, 16, 256) 0 ['stage3\_unit5\_bn2[0][0]']

)

zero\_padding2d\_25 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit5\_relu2[0][0]']

2D)

stage3\_unit5\_conv2 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_25[0][0]']

add\_11 (Add) (None, 16, 16, 256) 0 ['stage3\_unit5\_conv2[0][0]',

'add\_10[0][0]']

stage3\_unit6\_bn1 (BatchNormali (None, 16, 16, 256) 1024 ['add\_11[0][0]']

zation)

stage3\_unit6\_relu1 (Activation (None, 16, 16, 256) 0 ['stage3\_unit6\_bn1[0][0]']

)

zero\_padding2d\_26 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit6\_relu1[0][0]']

2D)

stage3\_unit6\_conv1 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_26[0][0]']

stage3\_unit6\_bn2 (BatchNormali (None, 16, 16, 256) 1024 ['stage3\_unit6\_conv1[0][0]']

zation)

stage3\_unit6\_relu2 (Activation (None, 16, 16, 256) 0 ['stage3\_unit6\_bn2[0][0]']

)

zero\_padding2d\_27 (ZeroPadding (None, 18, 18, 256) 0 ['stage3\_unit6\_relu2[0][0]']

2D)

stage3\_unit6\_conv2 (Conv2D) (None, 16, 16, 256) 589824 ['zero\_padding2d\_27[0][0]']

add\_12 (Add) (None, 16, 16, 256) 0 ['stage3\_unit6\_conv2[0][0]',

'add\_11[0][0]']

stage4\_unit1\_bn1 (BatchNormali (None, 16, 16, 256) 1024 ['add\_12[0][0]']

zation)

stage4\_unit1\_relu1 (Activation (None, 16, 16, 256) 0 ['stage4\_unit1\_bn1[0][0]']

)

zero\_padding2d\_28 (ZeroPadding (None, 18, 18, 256) 0 ['stage4\_unit1\_relu1[0][0]']

2D)

stage4\_unit1\_conv1 (Conv2D) (None, 8, 8, 512) 1179648 ['zero\_padding2d\_28[0][0]']

stage4\_unit1\_bn2 (BatchNormali (None, 8, 8, 512) 2048 ['stage4\_unit1\_conv1[0][0]']

zation)

stage4\_unit1\_relu2 (Activation (None, 8, 8, 512) 0 ['stage4\_unit1\_bn2[0][0]']

)

zero\_padding2d\_29 (ZeroPadding (None, 10, 10, 512) 0 ['stage4\_unit1\_relu2[0][0]']

2D)

stage4\_unit1\_conv2 (Conv2D) (None, 8, 8, 512) 2359296 ['zero\_padding2d\_29[0][0]']

stage4\_unit1\_sc (Conv2D) (None, 8, 8, 512) 131072 ['stage4\_unit1\_relu1[0][0]']

add\_13 (Add) (None, 8, 8, 512) 0 ['stage4\_unit1\_conv2[0][0]',

'stage4\_unit1\_sc[0][0]']

stage4\_unit2\_bn1 (BatchNormali (None, 8, 8, 512) 2048 ['add\_13[0][0]']

zation)

stage4\_unit2\_relu1 (Activation (None, 8, 8, 512) 0 ['stage4\_unit2\_bn1[0][0]']

)

zero\_padding2d\_30 (ZeroPadding (None, 10, 10, 512) 0 ['stage4\_unit2\_relu1[0][0]']

2D)

stage4\_unit2\_conv1 (Conv2D) (None, 8, 8, 512) 2359296 ['zero\_padding2d\_30[0][0]']

stage4\_unit2\_bn2 (BatchNormali (None, 8, 8, 512) 2048 ['stage4\_unit2\_conv1[0][0]']

zation)

stage4\_unit2\_relu2 (Activation (None, 8, 8, 512) 0 ['stage4\_unit2\_bn2[0][0]']

)

zero\_padding2d\_31 (ZeroPadding (None, 10, 10, 512) 0 ['stage4\_unit2\_relu2[0][0]']

2D)

stage4\_unit2\_conv2 (Conv2D) (None, 8, 8, 512) 2359296 ['zero\_padding2d\_31[0][0]']

add\_14 (Add) (None, 8, 8, 512) 0 ['stage4\_unit2\_conv2[0][0]',

'add\_13[0][0]']

stage4\_unit3\_bn1 (BatchNormali (None, 8, 8, 512) 2048 ['add\_14[0][0]']

zation)

stage4\_unit3\_relu1 (Activation (None, 8, 8, 512) 0 ['stage4\_unit3\_bn1[0][0]']

)

zero\_padding2d\_32 (ZeroPadding (None, 10, 10, 512) 0 ['stage4\_unit3\_relu1[0][0]']

2D)

stage4\_unit3\_conv1 (Conv2D) (None, 8, 8, 512) 2359296 ['zero\_padding2d\_32[0][0]']

stage4\_unit3\_bn2 (BatchNormali (None, 8, 8, 512) 2048 ['stage4\_unit3\_conv1[0][0]']

zation)

stage4\_unit3\_relu2 (Activation (None, 8, 8, 512) 0 ['stage4\_unit3\_bn2[0][0]']

)

zero\_padding2d\_33 (ZeroPadding (None, 10, 10, 512) 0 ['stage4\_unit3\_relu2[0][0]']

2D)

stage4\_unit3\_conv2 (Conv2D) (None, 8, 8, 512) 2359296 ['zero\_padding2d\_33[0][0]']

add\_15 (Add) (None, 8, 8, 512) 0 ['stage4\_unit3\_conv2[0][0]',

'add\_14[0][0]']

bn1 (BatchNormalization) (None, 8, 8, 512) 2048 ['add\_15[0][0]']

relu1 (Activation) (None, 8, 8, 512) 0 ['bn1[0][0]']

decoder\_stage0\_upsampling (UpS (None, 16, 16, 512) 0 ['relu1[0][0]']

ampling2D)

decoder\_stage0\_concat (Concate (None, 16, 16, 768) 0 ['decoder\_stage0\_upsampling[0][0]

nate) ',

'stage4\_unit1\_relu1[0][0]']

decoder\_stage0a\_conv (Conv2D) (None, 16, 16, 256) 1769472 ['decoder\_stage0\_concat[0][0]']

decoder\_stage0a\_bn (BatchNorma (None, 16, 16, 256) 1024 ['decoder\_stage0a\_conv[0][0]']

lization)

decoder\_stage0a\_relu (Activati (None, 16, 16, 256) 0 ['decoder\_stage0a\_bn[0][0]']

on)

decoder\_stage0b\_conv (Conv2D) (None, 16, 16, 256) 589824 ['decoder\_stage0a\_relu[0][0]']

decoder\_stage0b\_bn (BatchNorma (None, 16, 16, 256) 1024 ['decoder\_stage0b\_conv[0][0]']

lization)

decoder\_stage0b\_relu (Activati (None, 16, 16, 256) 0 ['decoder\_stage0b\_bn[0][0]']

on)

decoder\_stage1\_upsampling (UpS (None, 32, 32, 256) 0 ['decoder\_stage0b\_relu[0][0]']

ampling2D)

decoder\_stage1\_concat (Concate (None, 32, 32, 384) 0 ['decoder\_stage1\_upsampling[0][0]

nate) ',

'stage3\_unit1\_relu1[0][0]']

decoder\_stage1a\_conv (Conv2D) (None, 32, 32, 128) 442368 ['decoder\_stage1\_concat[0][0]']

decoder\_stage1a\_bn (BatchNorma (None, 32, 32, 128) 512 ['decoder\_stage1a\_conv[0][0]']

lization)

decoder\_stage1a\_relu (Activati (None, 32, 32, 128) 0 ['decoder\_stage1a\_bn[0][0]']

on)

decoder\_stage1b\_conv (Conv2D) (None, 32, 32, 128) 147456 ['decoder\_stage1a\_relu[0][0]']

decoder\_stage1b\_bn (BatchNorma (None, 32, 32, 128) 512 ['decoder\_stage1b\_conv[0][0]']

lization)

decoder\_stage1b\_relu (Activati (None, 32, 32, 128) 0 ['decoder\_stage1b\_bn[0][0]']

on)

decoder\_stage2\_upsampling (UpS (None, 64, 64, 128) 0 ['decoder\_stage1b\_relu[0][0]']

ampling2D)

decoder\_stage2\_concat (Concate (None, 64, 64, 192) 0 ['decoder\_stage2\_upsampling[0][0]

nate) ',

'stage2\_unit1\_relu1[0][0]']

decoder\_stage2a\_conv (Conv2D) (None, 64, 64, 64) 110592 ['decoder\_stage2\_concat[0][0]']

decoder\_stage2a\_bn (BatchNorma (None, 64, 64, 64) 256 ['decoder\_stage2a\_conv[0][0]']

lization)

decoder\_stage2a\_relu (Activati (None, 64, 64, 64) 0 ['decoder\_stage2a\_bn[0][0]']

on)

decoder\_stage2b\_conv (Conv2D) (None, 64, 64, 64) 36864 ['decoder\_stage2a\_relu[0][0]']

decoder\_stage2b\_bn (BatchNorma (None, 64, 64, 64) 256 ['decoder\_stage2b\_conv[0][0]']

lization)

decoder\_stage2b\_relu (Activati (None, 64, 64, 64) 0 ['decoder\_stage2b\_bn[0][0]']

on)

decoder\_stage3\_upsampling (UpS (None, 128, 128, 64 0 ['decoder\_stage2b\_relu[0][0]']

ampling2D) )

decoder\_stage3\_concat (Concate (None, 128, 128, 12 0 ['decoder\_stage3\_upsampling[0][0]

nate) 8) ',

'relu0[0][0]']

decoder\_stage3a\_conv (Conv2D) (None, 128, 128, 32 36864 ['decoder\_stage3\_concat[0][0]']

)

decoder\_stage3a\_bn (BatchNorma (None, 128, 128, 32 128 ['decoder\_stage3a\_conv[0][0]']

lization) )

decoder\_stage3a\_relu (Activati (None, 128, 128, 32 0 ['decoder\_stage3a\_bn[0][0]']

on) )

decoder\_stage3b\_conv (Conv2D) (None, 128, 128, 32 9216 ['decoder\_stage3a\_relu[0][0]']

)

decoder\_stage3b\_bn (BatchNorma (None, 128, 128, 32 128 ['decoder\_stage3b\_conv[0][0]']

lization) )

decoder\_stage3b\_relu (Activati (None, 128, 128, 32 0 ['decoder\_stage3b\_bn[0][0]']

on) )

decoder\_stage4\_upsampling (UpS (None, 256, 256, 32 0 ['decoder\_stage3b\_relu[0][0]']

ampling2D) )

decoder\_stage4a\_conv (Conv2D) (None, 256, 256, 16 4608 ['decoder\_stage4\_upsampling[0][0]

) ']

decoder\_stage4a\_bn (BatchNorma (None, 256, 256, 16 64 ['decoder\_stage4a\_conv[0][0]']

lization) )

decoder\_stage4a\_relu (Activati (None, 256, 256, 16 0 ['decoder\_stage4a\_bn[0][0]']

on) )

decoder\_stage4b\_conv (Conv2D) (None, 256, 256, 16 2304 ['decoder\_stage4a\_relu[0][0]']

)

decoder\_stage4b\_bn (BatchNorma (None, 256, 256, 16 64 ['decoder\_stage4b\_conv[0][0]']

lization) )

decoder\_stage4b\_relu (Activati (None, 256, 256, 16 0 ['decoder\_stage4b\_bn[0][0]']

on) )

final\_conv (Conv2D) (None, 256, 256, 3) 435 ['decoder\_stage4b\_relu[0][0]']

softmax (Activation) (None, 256, 256, 3) 0 ['final\_conv[0][0]']

==================================================================================================

Total params: 24,456,444

Trainable params: 24,439,094

Non-trainable params: 17,350

Transfer learning semantic segmentation:

What is transfer learning / pretraining

How can transfer learning / pretraining be done

Successful examples

Note on pretraining domain

Methodology: How did I stack pretrained classification network and decoder, what is Imagenet. The UNet used in this thesis is using ResNet34 as encoder and thus can be initialized with pre-trained weights from image classification Imagenet. In particular, the classification layer of resnet is cut and replaced by decoder architecture of the Unet.

“First, we train our network by initializing the encoder with pre-trained weights from ResNet34 on ImageNet. We repeat the experiment, but weights are randomly initialized using glorot uniform initialization. This allows to study if we can benefit from transfer learning. Results are presented in Figure.”

@ARTICLE{7342907,

author={Marmanis, Dimitrios and Datcu, Mihai and Esch, Thomas and Stilla, Uwe},

journal={IEEE Geoscience and Remote Sensing Letters},

title={Deep Learning Earth Observation Classification Using ImageNet Pretrained Networks},

year={2016},

volume={13},

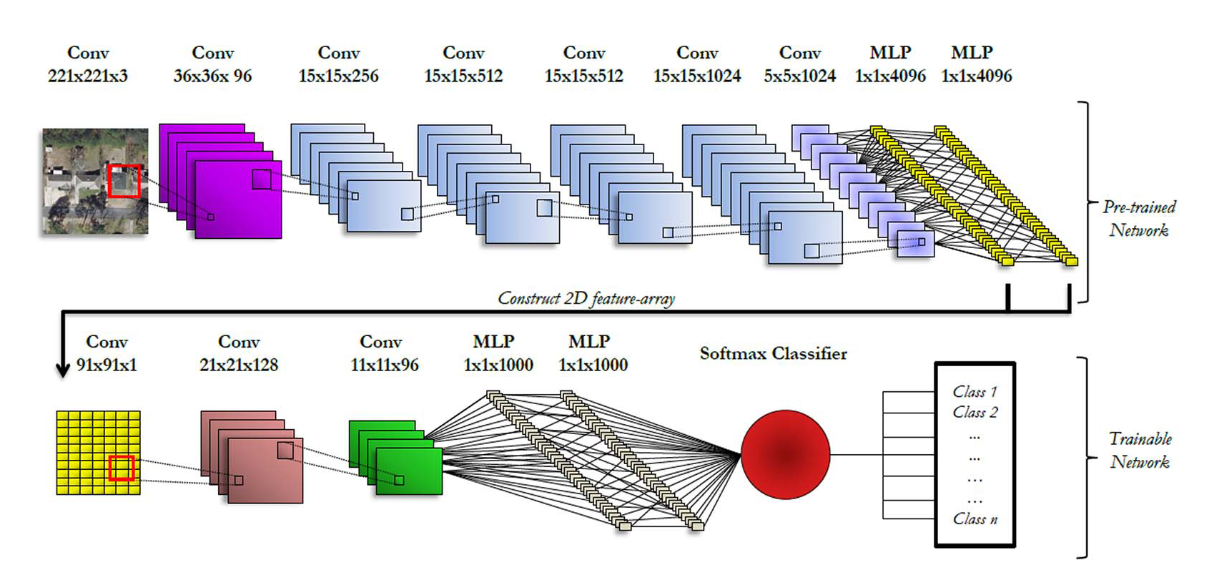
number={1},

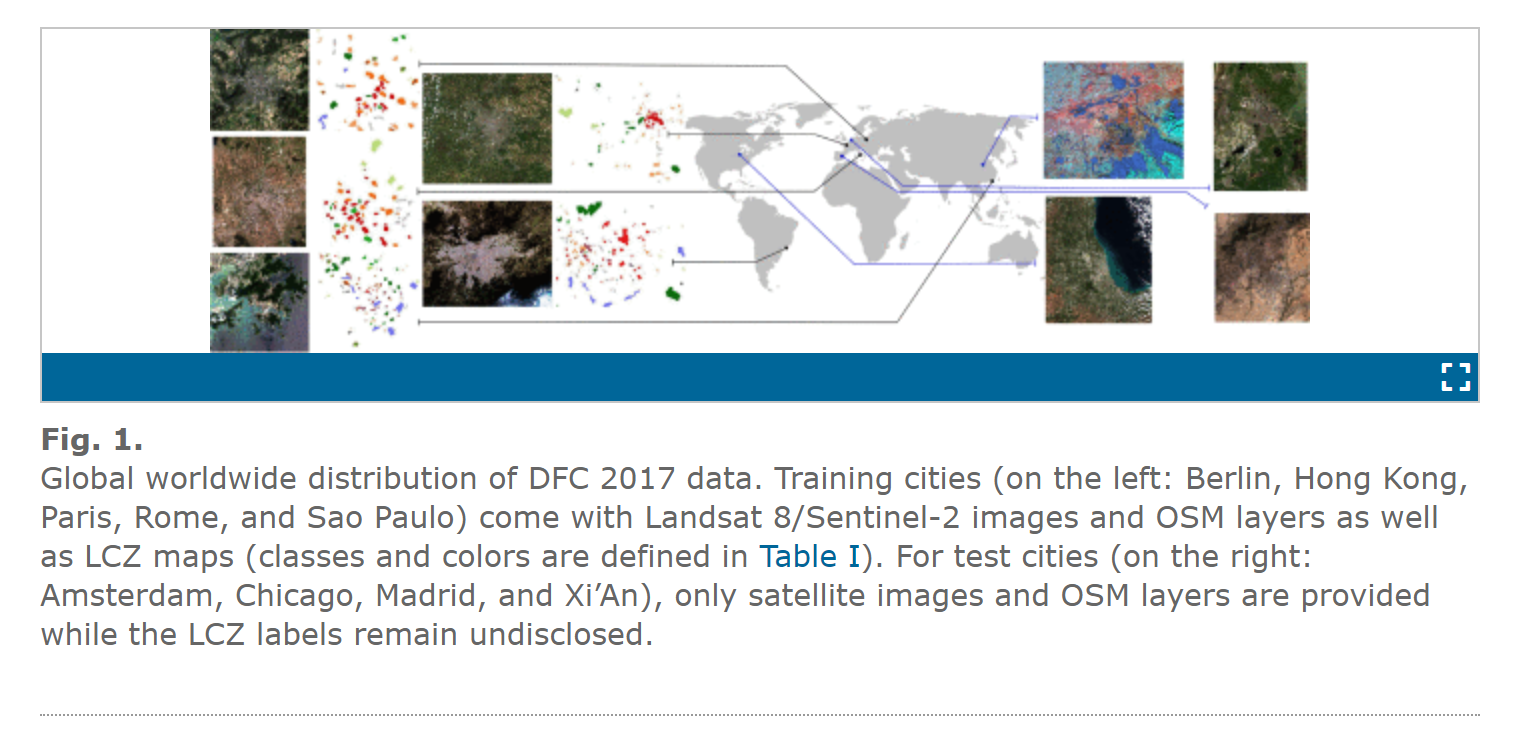
pages={105-109},

doi={10.1109/LGRS.2015.2499239}}

(Remote Sensing Imagenet Pretraining classification):

* Significantly imporved performance although entirely different domain
* “CNN classifiers from different domains can be profoundly suitable for RS classification task”
* Representations from very deep networks are generic and can facilitate transfer learning between different domains. Hypothesis why this works: Using hierarchical architectures, largely trained CNNs accumulate extended knowledge in low-level spatial descriptors and employ them to describe images in a bottom-up manner. This way, pretrained CNNs can trivially decompose images into a set of primitive elements and detect similarities on





8: “encapsulated representations contained within can work remarkably well for a large set of diverse image classification tasks and often outperform standard classification approaches.”

9