Low-Power Neural Networks for Semantic Segmentation of Satellite Images

2019 ICCV Workshop on Low-Power Computer Vision

Gaétan Bahl, Lionel Daniel, Matthieu Moretti, Florent Lafarge

Onboard semantic segmentation of satellite images

- Why?
  - Save bandwidth
  - Save processing time on the ground
  - Save storage space on the ground

Figure: OPS-SAT (ESA 2019)

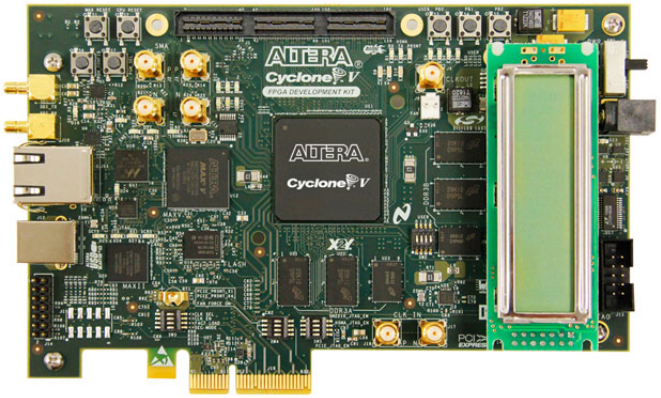


Figure: Intel/Altera Cyclone V SoC with FPGA

Onboard semantic segmentation of satellite images

- Why?
  - Save bandwidth
  - Save processing time on the ground
  - Save storage space on the ground
- Problems:
  - Power constraints
  - Memory constraints

Figure: OPS-SAT (ESA 2019)

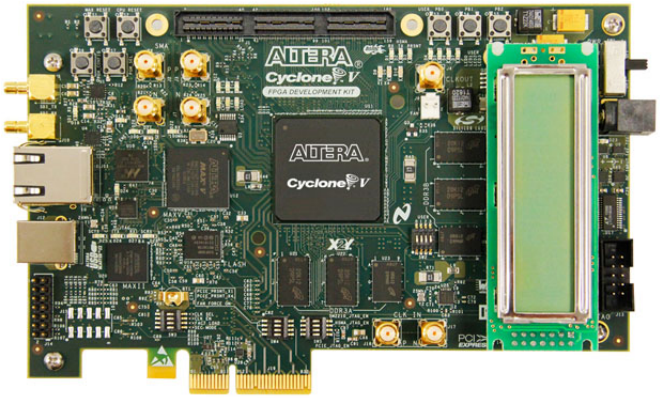


Figure: Intel/Altera Cyclone V SoC with FPGA

Onboard semantic segmentation of satellite images

- Why?
  - Save bandwidth
  - Save processing time on the ground
  - Save storage space on the ground
- Problems:
  - Power constraints
  - Memory constraints
- Targets:
  - Low-power SoCs (ARM)
  - FPGAs

Figure: OPS-SAT (ESA 2019)

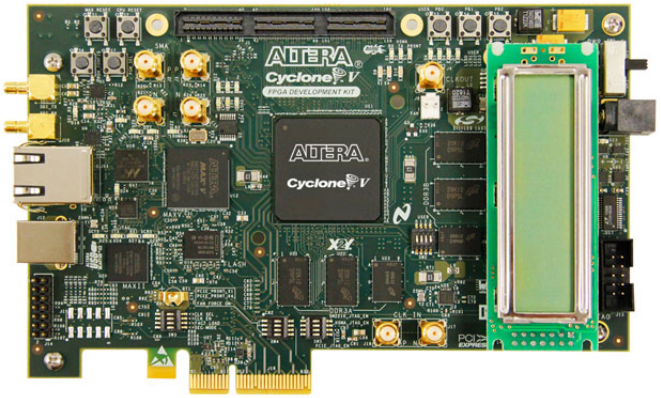


Figure: Intel/Altera Cyclone V SoC with FPGA

Onboard semantic segmentation of satellite images

- Why?
  - Save bandwidth
  - Save processing time on the ground
  - Save storage space on the ground
- Problems:
  - Power constraints
  - Memory constraints
- Targets:
  - Low-power SoCs (ARM)
  - FPGAs

  Need for compact neural network architectures

Figure: OPS-SAT (ESA 2019)

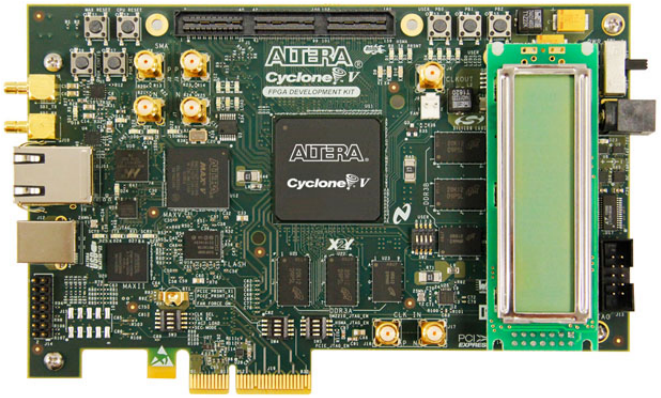
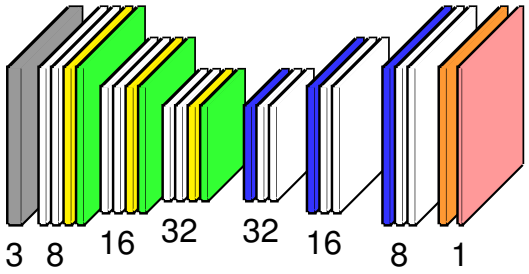


Figure: Intel/Altera Cyclone V SoC with FPGA

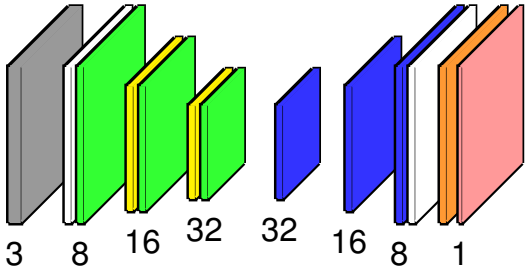
Main contributions

Two compact neural network architectures, with 2 variants:

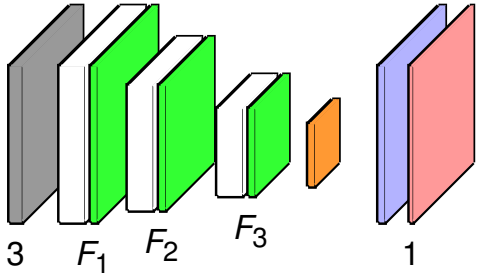
- C-UNet and C-UNet++
- C-FCN and C-FCN++



(a) C-UNet



(b) C-UNet++



(c) C-FCN and C-FCN++

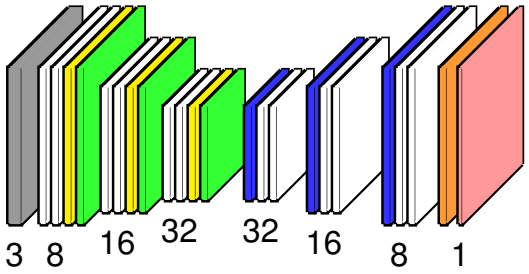
Main contributions

Two compact neural network architectures, with 2 variants:

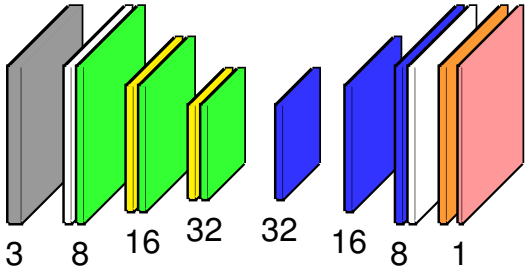
- C-UNet and C-UNet++
- C-FCN and C-FCN++

Architecture requirements:

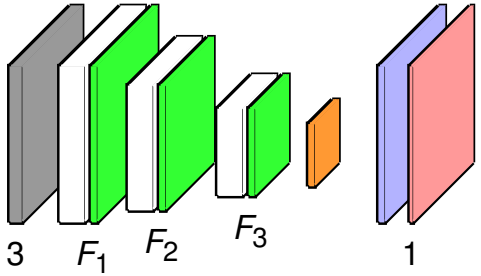
- High accuracy
- Low complexity
- High adaptability



(a) C-UNet



(b) C-UNet++



(c) C-FCN and C-FCN++

Main contributions

Two compact neural network architectures, with 2 variants:

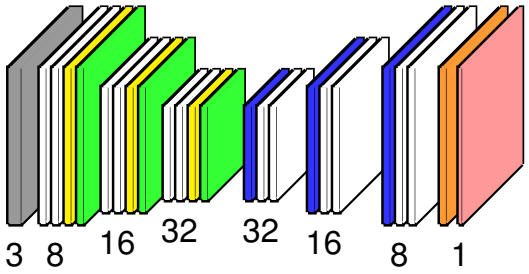
- C-UNet and C-UNet++
- C-FCN and C-FCN++

Architecture requirements:

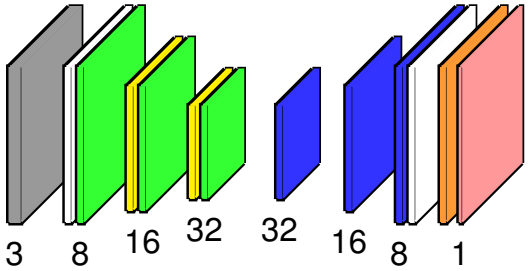
- High accuracy
- Low complexity
- High adaptability

Main Characteristics:

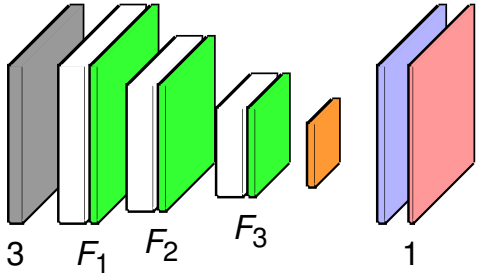
- No skip connections
- Shallow architecture
- Depth-wise separable convolutions
- Low number of filters



(a) C-UNet



(b) C-UNet++



(c) C-FCN and C-FCN++

Contents

1

Related works

Semantic segmentation Neural Networks

- Encoder-decoder architectures
- Skip connections
- Hundreds of thousands of parameters

Impossible to use in low-power systems!

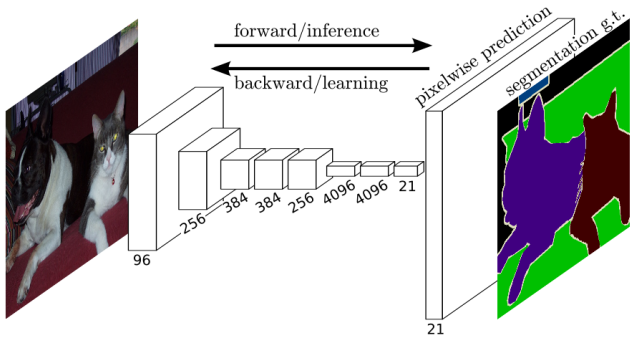


Figure: FCN architecture (Long *et al.*, 2014)

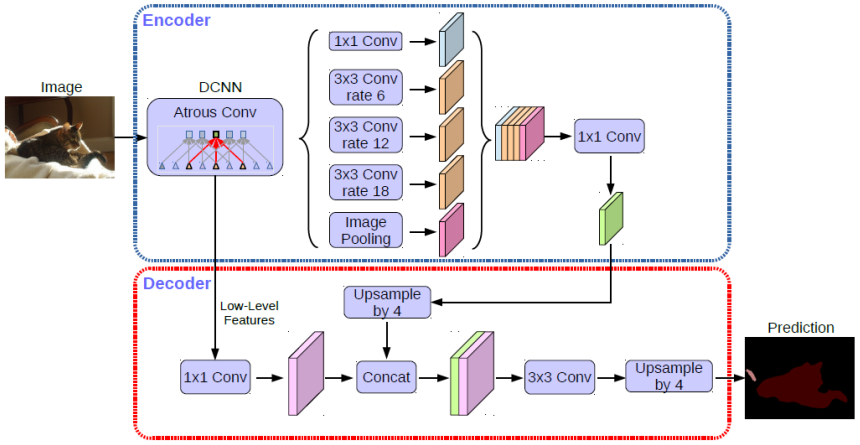


Figure: DeepLabv3+ (Chen *et al.*, 2018)

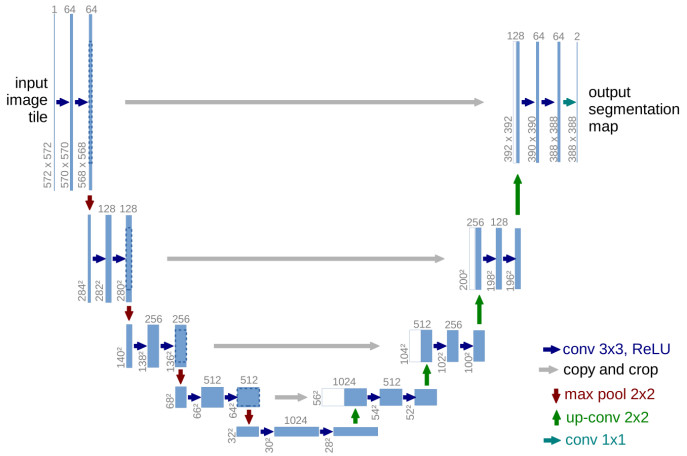


Figure: U-Net architecture (Ronneberger *et al.*, 2015)

Compact Neural Networks on FPGA

Techniques:

- Neural network pruning (Molchanov *et al.*, 2016)
- Weight quantization (Lin *et al.*, 2015)
- Compact convolution modules (Chollet, 2017)

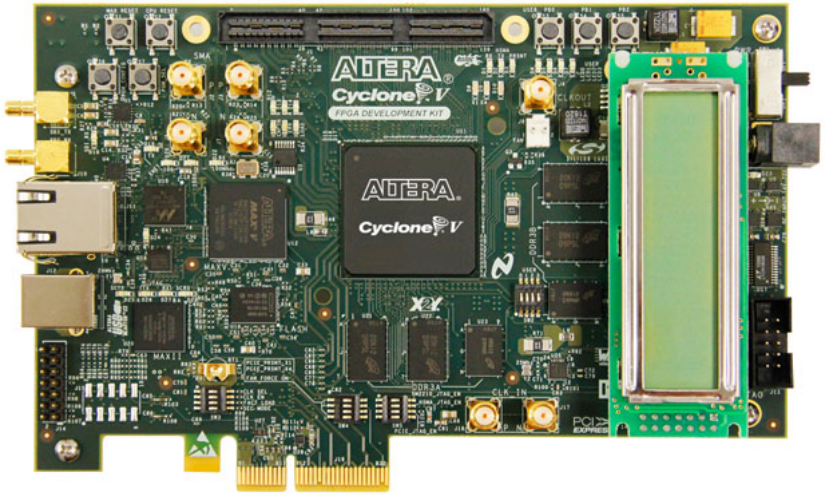


Figure: Intel/Altera Cyclone V SoC with FPGA

Compact Neural Networks on FPGA

Techniques:

- Neural network pruning (Molchanov *et al.*, 2016)
- Weight quantization (Lin *et al.*, 2015)
- Compact convolution modules (Chollet, 2017)

Not enough reduction to fit our needs.

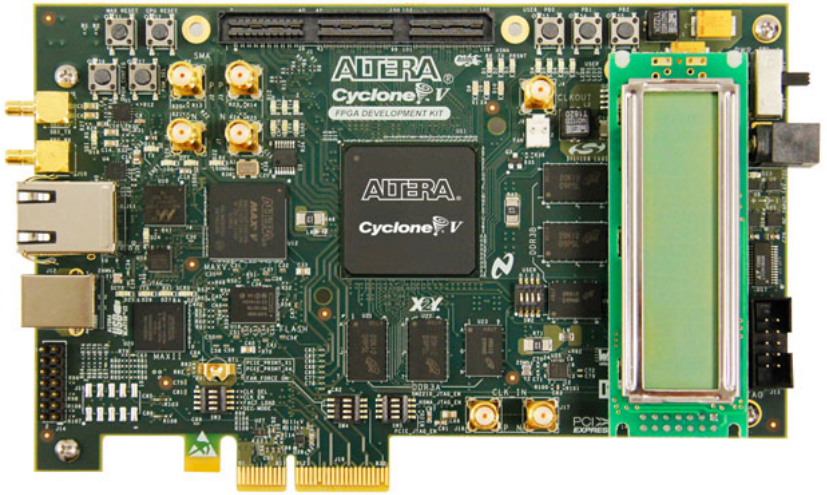


Figure: Intel/Altera Cyclone V SoC with FPGA

Semantic segmentation aboard satellites

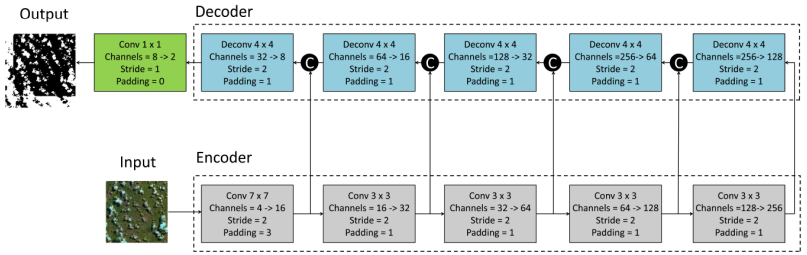


Figure: ”Strided U-Net” (Ghassemi *et al.*, 2019)

Layers	kernel size	output size	channels
Input	3×3	$a \times a$	4
conv1-1	3×3	$a \times a$	8
conv1-2	3×3	$a \times a$	16
pool1	3×3	$a/2 \times a/2$	16
dw/s2-1	3×3	$a/2 \times a/2$	32
dw/s2-2	3×3	$a/2 \times a/2$	32
pool2	3×3	$a/4 \times a/4$	32
dw/s3-1	3×3	$a/4 \times a/4$	64
dw/s3-2	3×3	$a/4 \times a/4$	64
unpool1	3×3	$a/2 \times a/2$	32
concat[unpool1, dw/s2-2]	3×3	$a/2 \times a/2$	64
conv4-1	3×3	$a/2 \times a/2$	32
conv4-2	3×3	$a/2 \times a/2$	32
unpool2	3×3	$a/2 \times a/2$	16
concat[unpool2, conv1-2]	3×3	$a \times a$	32
dw/s5-1	3×3	$a \times a$	16
dw/s5-2	3×3	$a \times a$	8
dw/s5-3	3×3	$a \times a$	2

Figure: mobUNet (Zhang *et al.*, 2018)

- Too many parameters (around 50k)
- Too much RAM

Layers	kernel size	output size	channels
Input	3×3	$a \times a$	4
dw/s1	3×3	$a \times a$	8
dw/s2	3×3	$a \times a$	16
fc3	1×1	$a \times a$	32
fc4	1×1	$a \times a$	32
deconv2	3×3	$a \times a$	16
deconv1	3×3	$a \times a$	8
output	1×1	$a \times a$	2

Figure: mobDeconvNet (Zhang *et al.*, 2018)

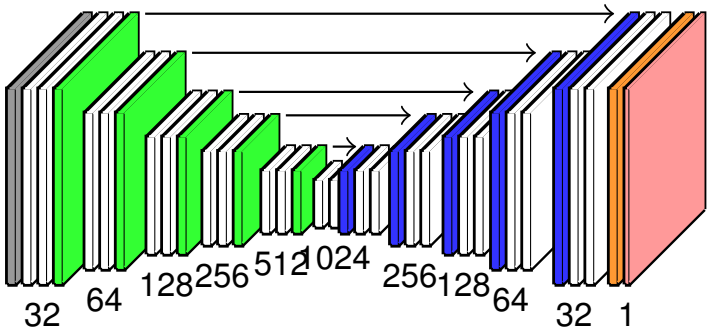
2

Contributions

C-UNet(++)

Characteristics:

- Same principle as U-Net (Ronneberger *et al.*, 2015)

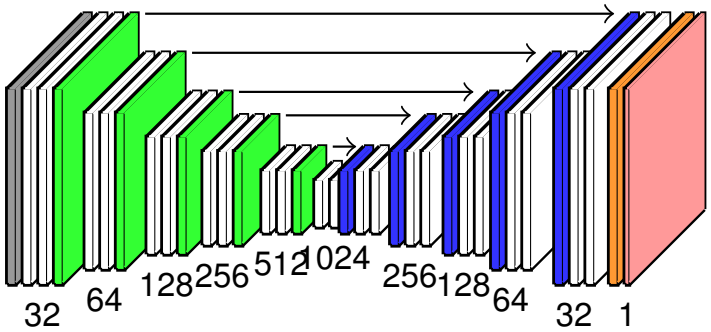


(a) U-Net

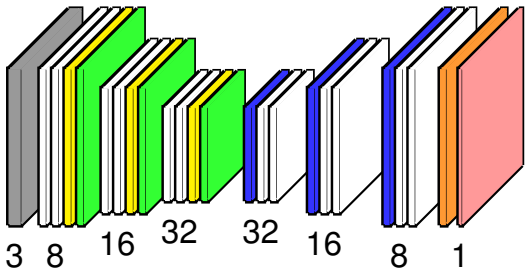
C-UNet(++)

Characteristics:

- Same principle as U-Net (Ronneberger *et al.*, 2015)
- Removed skip connections



(a) U-Net

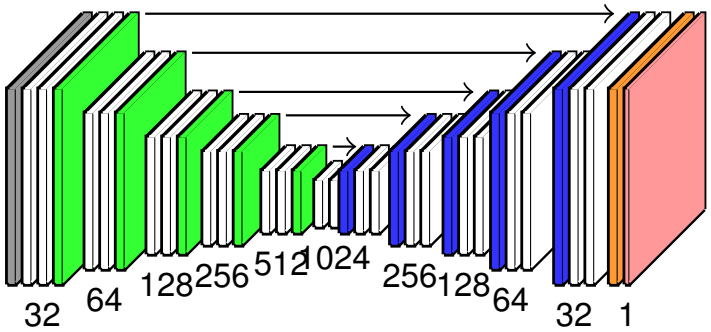


(b) C-UNet

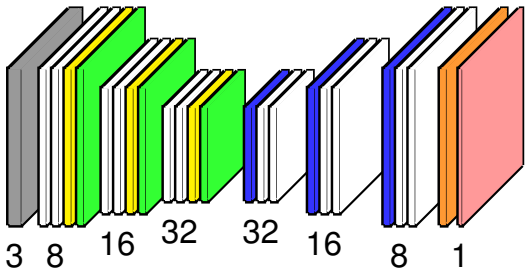
C-UNet(++)

Characteristics:

- Same principle as U-Net (Ronneberger *et al.*, 2015)
- Removed skip connections
- Use of depth-wise separable convolutions (yellow)



(a) U-Net

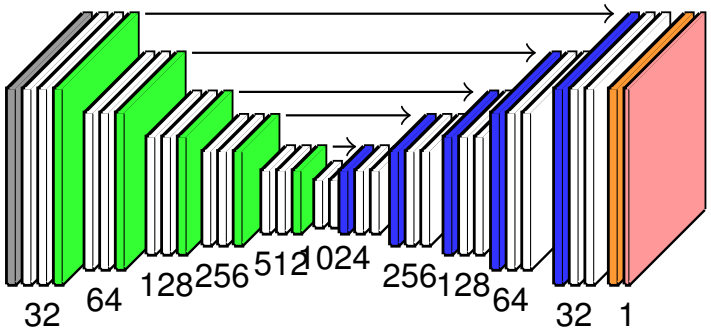


(b) C-UNet

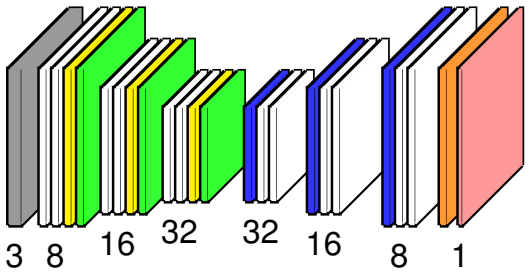
C-UNet(++)

Characteristics:

- Same principle as U-Net (Ronneberger *et al.*, 2015)
- Removed skip connections
- Use of depth-wise separable convolutions (yellow)
- Removed convolution stages



(a) U-Net

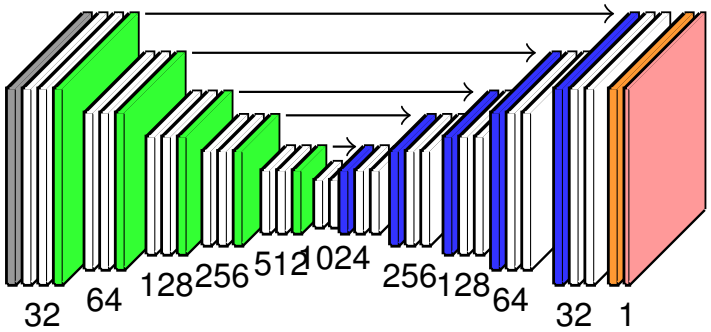


(b) C-UNet

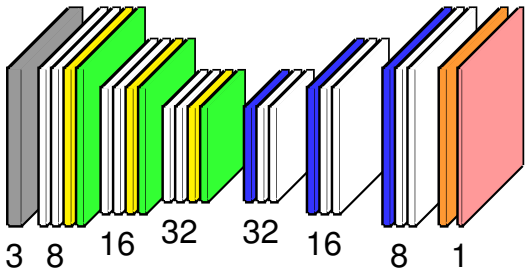
C-UNet(++)

Characteristics:

- Same principle as U-Net (Ronneberger *et al.*, 2015)
- Removed skip connections
- Use of depth-wise separable convolutions (yellow)
- Removed convolution stages
- Low number of filters



(a) U-Net



(b) C-UNet

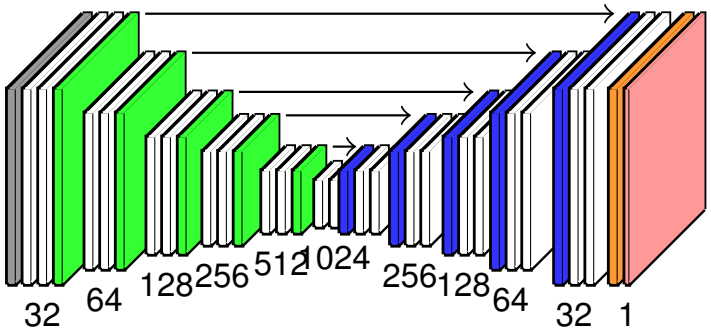
C-UNet(++)

Characteristics:

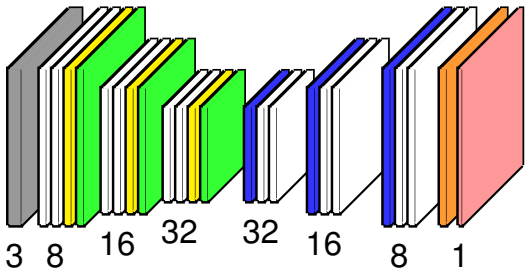
- Same principle as U-Net (Ronneberger *et al.*, 2015)
- Removed skip connections
- Use of depth-wise separable convolutions (yellow)
- Removed convolution stages
- Low number of filters

Number of parameters:

- U-Net: 30 millions
- C-UNet: Around 50k



(a) U-Net



(b) C-UNet

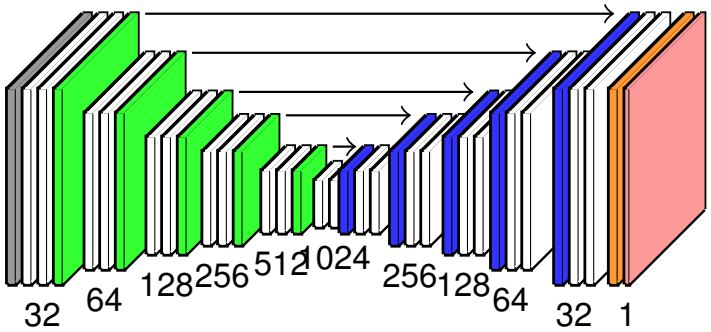
C-UNet(++)

Characteristics:

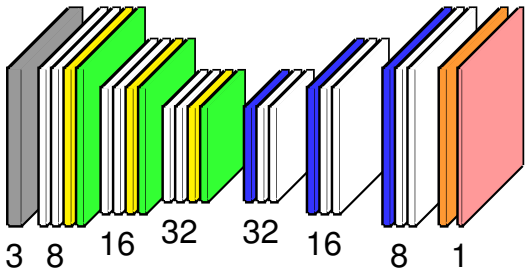
- Same principle as U-Net (Ronneberger *et al.*, 2015)
- Removed skip connections
- Use of depth-wise separable convolutions (yellow)
- Removed convolution stages
- Low number of filters

Number of parameters:

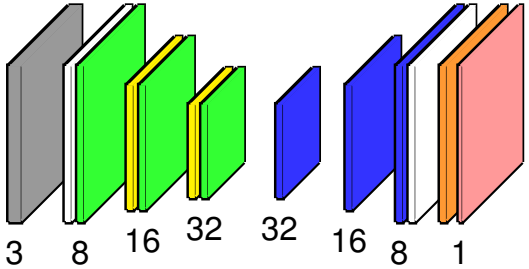
- U-Net: 30 millions
- C-UNet: Around 50k
- C-UNet++: Around 10k



(a) U-Net



(b) C-UNet



(c) C-UNet++

C-FCN(++)

Characteristics:

- Same principle as FCN (Long *et al.*, 2014)

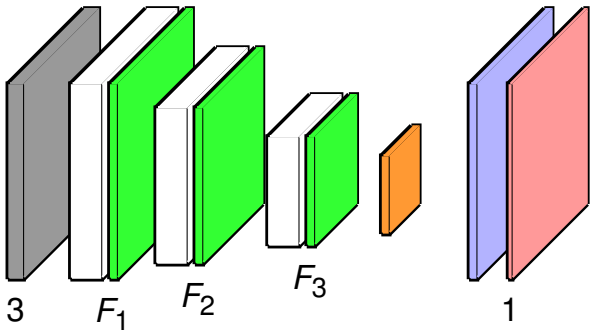


Figure: C-FCN and C-FCN++ architectures. Gray: input, white: conv3x3 ReLU, green: 2x2 max pool, orange: conv1x1 sigmoid, blue: 4x4 bilinear upscaling.

Arch.	F_1	F_2	F_3	DW	Atrous	N_{param}
C-FCN	10	20	40	Yes	No	1438
C-FCN++	5	2	2	No	1st conv.	273

Table: Characteristics of C-FCN and C-FCN++. DW = usage of depth-wise convolutions. F_x is the number of convolution filters and feature maps at stage x.

C-FCN(++)

Characteristics:

- Same principle as FCN (Long *et al.*, 2014)
- Lightweight encoder
- Depth-wise separable convolutions
- Low number of filters

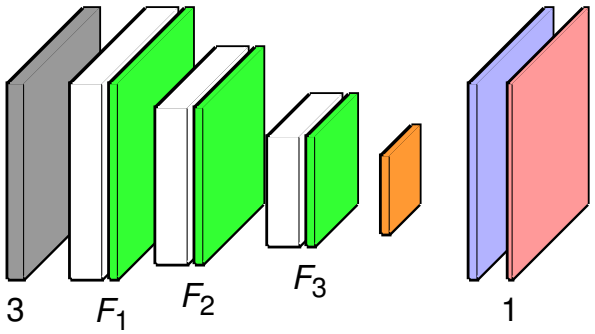


Figure: C-FCN and C-FCN++ architectures. Gray: input, white: conv3x3 ReLU, green: 2x2 max pool, orange: conv1x1 sigmoid, blue: 4x4 bilinear upscaling.

Arch.	F_1	F_2	F_3	DW	Atrous	N_{param}
C-FCN	10	20	40	Yes	No	1438
C-FCN++	5	2	2	No	1st conv.	273

Table: Characteristics of C-FCN and C-FCN++. DW = usage of depth-wise convolutions. F_x is the number of convolution filters and feature maps at stage x.

C-FCN(++)

Characteristics:

- Same principle as FCN (Long *et al.*, 2014)
- Lightweight encoder
- Depth-wise separable convolutions
- Low number of filters
- Atrous convolution (++)

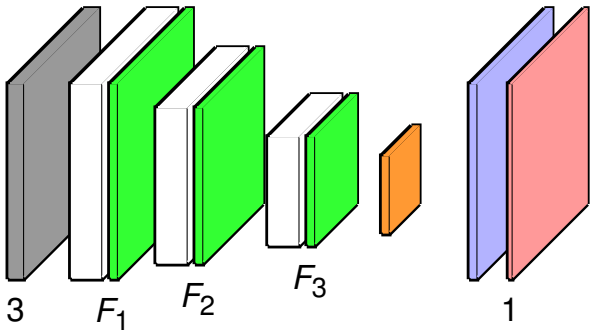


Figure: C-FCN and C-FCN++ architectures. Gray: input, white: conv3x3 ReLU, green: 2x2 max pool, orange: conv1x1 sigmoid, blue: 4x4 bilinear upscaling.

Arch.	F_1	F_2	F_3	DW	Atrous	N_{param}
C-FCN	10	20	40	Yes	No	1438
C-FCN++	5	2	2	No	1st conv.	273

Table: Characteristics of C-FCN and C-FCN++. DW = usage of depth-wise convolutions. F_x is the number of convolution filters and feature maps at stage x.

C-FCN(++)

Characteristics:

- Same principle as FCN (Long *et al.*, 2014)
- Lightweight encoder
- Depth-wise separable convolutions
- Low number of filters
- Atrous convolution (++)
- Bilinear upscaling

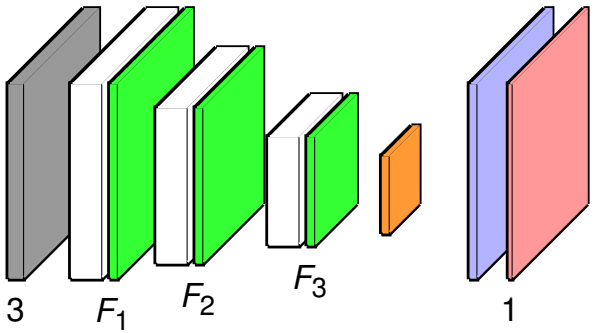


Figure: C-FCN and C-FCN++ architectures. Gray: input, white: conv3x3 ReLU, green: 2x2 max pool, orange: conv1x1 sigmoid, blue: 4x4 bilinear upscaling.

Arch.	F_1	F_2	F_3	DW	Atrous	N_{param}
C-FCN	10	20	40	Yes	No	1438
C-FCN++	5	2	2	No	1st conv.	273

Table: Characteristics of C-FCN and C-FCN++. DW = usage of depth-wise convolutions. F_x is the number of convolution filters and feature maps at stage x.

C-FCN(++)

Characteristics:

- Same principle as FCN (Long *et al.*, 2014)
- Lightweight encoder
- Depth-wise separable convolutions
- Low number of filters
- Atrous convolution (++)
- Bilinear upscaling

Number of parameters:

- C-FCN: Around 1.5k
- C-FCN++: Around 300

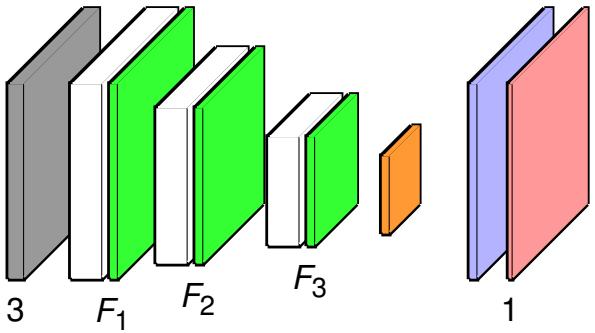


Figure: C-FCN and C-FCN++ architectures. Gray: input, white: conv3x3 ReLU, green: 2x2 max pool, orange: conv1x1 sigmoid, blue: 4x4 bilinear upscaling.

Arch.	F_1	F_2	F_3	DW	Atrous	N_{param}
C-FCN	10	20	40	Yes	No	1438
C-FCN++	5	2	2	No	1st conv.	273

Table: Characteristics of C-FCN and C-FCN++. DW = usage of depth-wise convolutions. F_x is the number of convolution filters and feature maps at stage x.

3

Experimental results

Cloud segmentation

More than 50% of Earth is covered by clouds at any time.

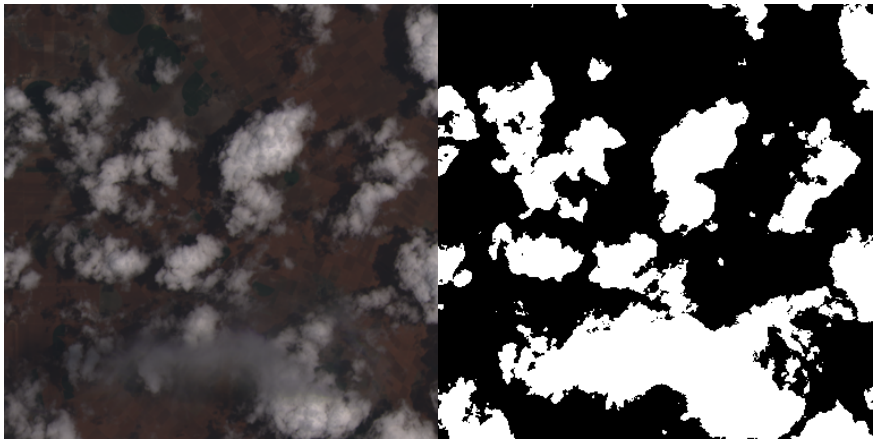


Figure: Input image and desired output

Cloud segmentation

More than 50% of Earth is covered by clouds at any time.
Onboard cloud segmentation:

- Cloud coverage estimation
- Cloud screening
- Bandwidth savings

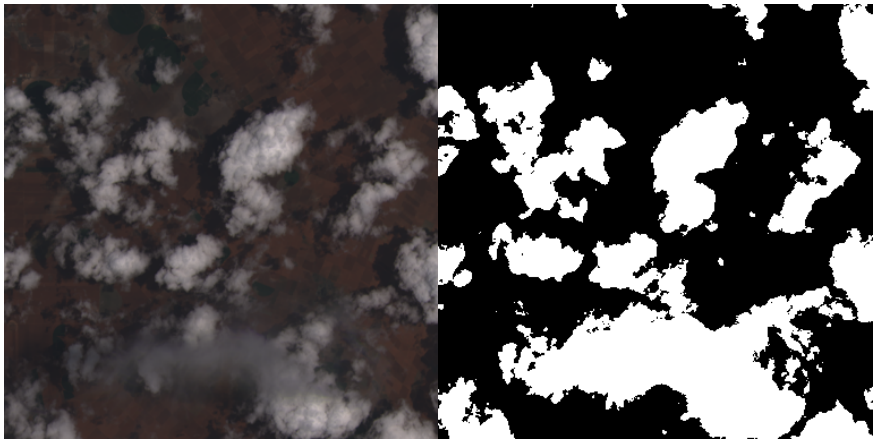


Figure: Input image and desired output

Cloud segmentation

More than 50% of Earth is covered by clouds at any time.
Onboard cloud segmentation:

- Cloud coverage estimation
- Cloud screening
- Bandwidth savings

Challenges:

- Limited number of spectral bands
- Same radiometry/texture as snow

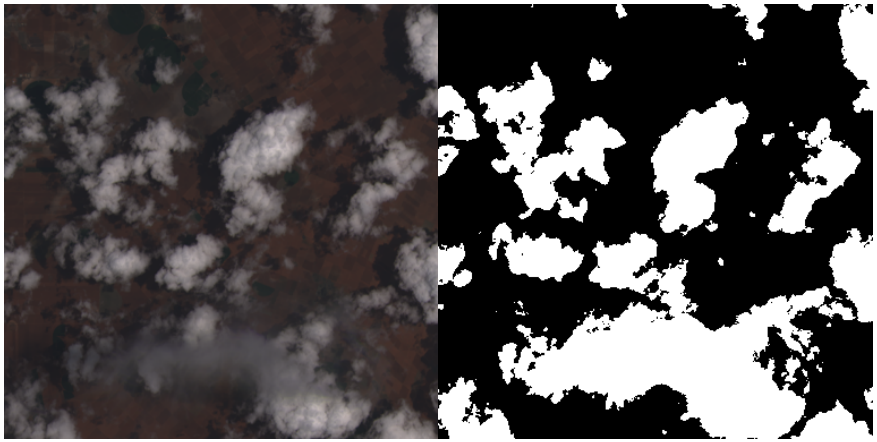


Figure: Input image and desired output

Cloud segmentation datasets

38-Cloud dataset (Mohajerani *et al.*, 2019):

- Landsat-8 images
- 30m resolution
- 4 bands
- 14 021 images of 384x384 pixels

Cloud segmentation datasets

38-Cloud dataset (Mohajerani *et al.*, 2019):

- Landsat-8 images
- 30m resolution
- 4 bands
- 14 021 images of 384x384 pixels

CloudPeru2 dataset (Morales *et al.*, 2019):

- PERUSAT-1 images
- 3m resolution
- 4 bands
- 22 400 images of 512x512 pixels

Cloud segmentation datasets

38-Cloud dataset (Mohajerani *et al.*, 2019):

- Landsat-8 images
- 30m resolution
- 4 bands
- 14 021 images of 384x384 pixels

CloudPeru2 dataset (Morales *et al.*, 2019):

- PERUSAT-1 images
- 3m resolution
- 4 bands
- 22 400 images of 512x512 pixels

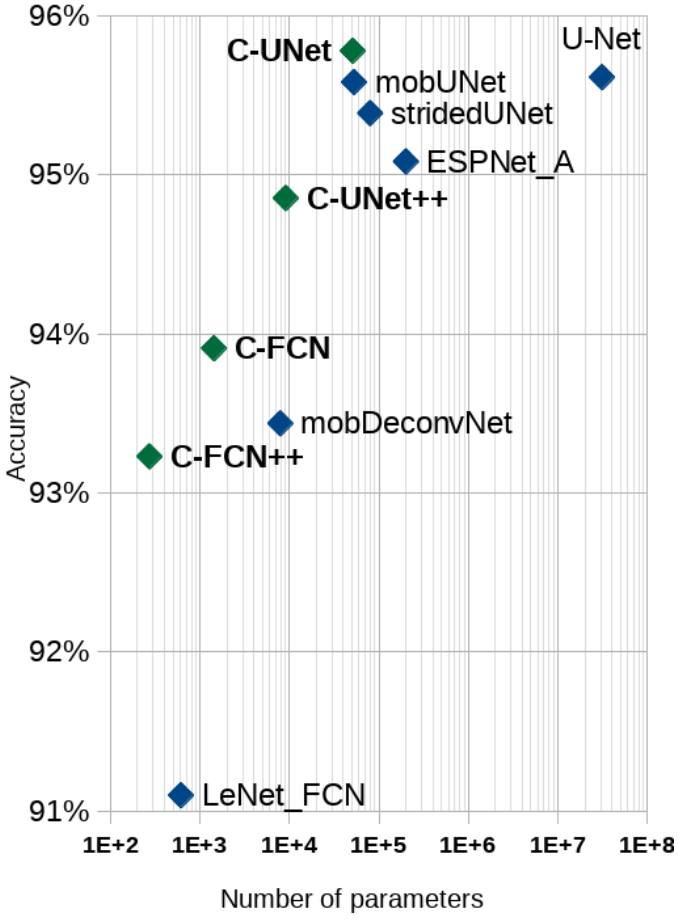
We only use 3 bands! (RGB)

Accuracy metrics

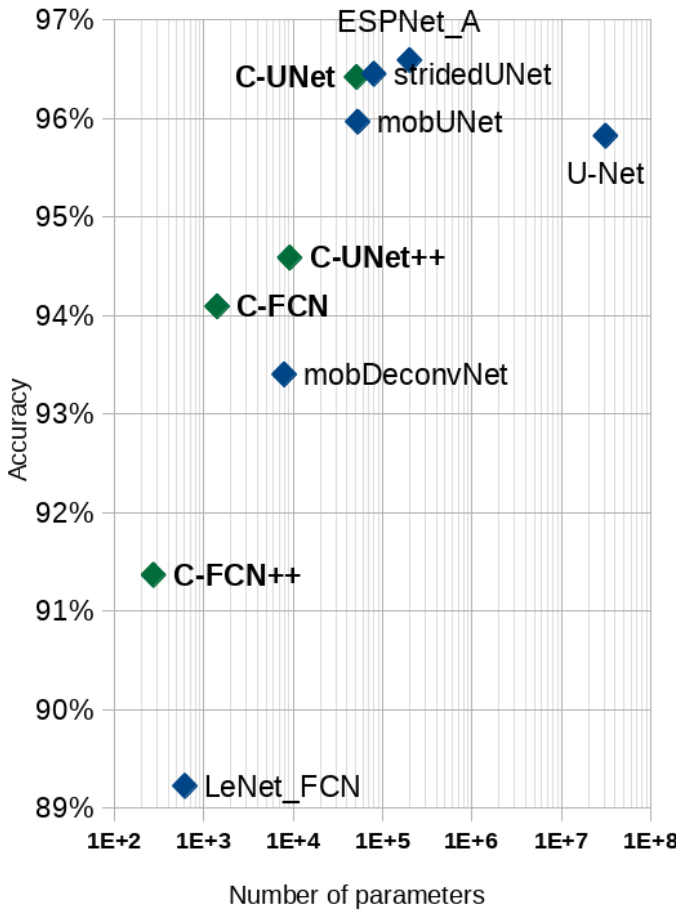
Model	Acc.	Prec.	Rec.	Spec.	Jacc.
LeNet_FCN	91.10	86.00	84.34	94.03	74.16
C-FCN++	93.23	91.67	85.45	96.62	79.30
mobDeconvNet	93.44	92.04	85.75	96.78	79.83
C-FCN	93.91	91.23	88.39	96.31	81.47
C-UNet++	94.85	94.18	88.48	97.63	83.89
ESPNet_A	95.08	93.94	89.55	97.49	84.66
StridedUNet	95.39	94.21	90.36	97.58	85.60
mobUNet	95.58	95.37	89.78	98.11	86.03
U-Net	95.61	95.69	89.56	98.25	86.08
C-UNet	**95.78**	**96.53**	89.27	**98.61**	**86.50**
Fmask	94.89	77.71	**97.22**	93.96	75.16
LeNet_FCN	89.23	93.11	83.27	94.52	78.44
C-FCN++	91.37	93.50	87.76	94.58	82.71
mobDeconvNet	93.40	94.98	90.78	95.73	86.63
C-FCN	94.09	95.15	92.15	95.82	88.01
C-UNet++	94.59	96.03	92.31	96.61	88.92
U-Net	95.82	96.29	94.78	96.75	91.44
mobUNet	95.97	**97.20**	94.14	**97.59**	91.66
C-UNet	96.42	96.54	95.83	96.94	92.65
stridedUNet	96.45	96.49	95.95	96.89	92.72
ESPNet_A	**96.59**	96.45	**96.30**	96.85	**93.01**

Table: 38-Cloud (top) and CloudPeru2 (bottom) results. Our networks in bold font.

Accuracy vs Number of parameters



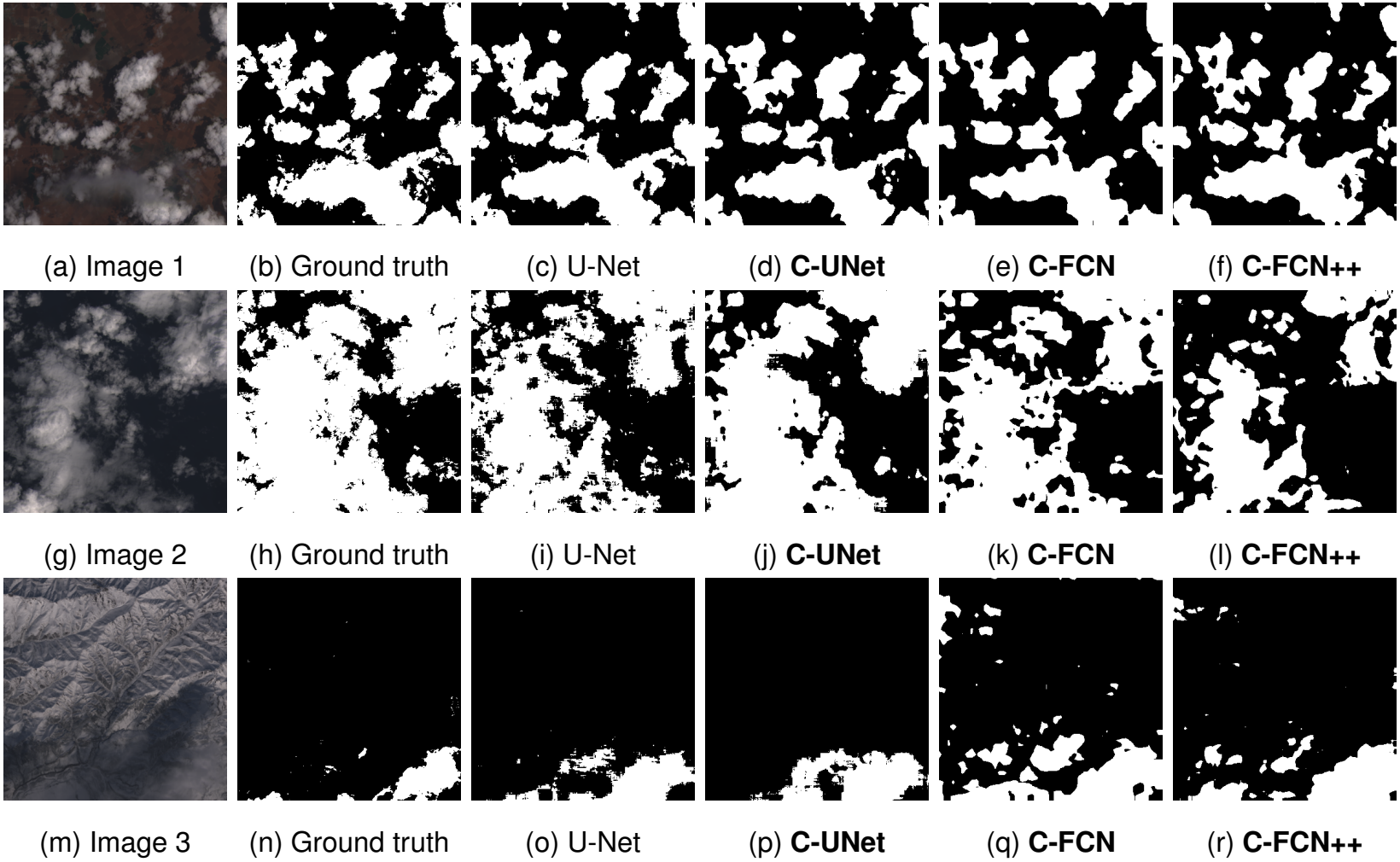
(a) 38-Cloud



(b) CloudPeru2

Figure: Test accuracy w.r.t. number of parameters on the two tested datasets. Our networks in green and bold font, others in blue. Number of parameters is on a log scale.

Qualitative results



(a) Image 1 (b) Ground truth (c) U-Net (d) **C-UNet** (e) **C-FCN** (f) **C-FCN++**

(g) Image 2 (h) Ground truth (i) U-Net (j) **C-UNet** (k) **C-FCN** (l) **C-FCN++**

(m) Image 3 (n) Ground truth (o) U-Net (p) **C-UNet** (q) **C-FCN** (r) **C-FCN++**

Forest segmentation

Applications:

- Deforestation monitoring
- Forest fire alerts

Forest segmentation

Applications:

- Deforestation monitoring
- Forest fire alerts

Slovenia 2017 dataset
(Sinergise Ltd.):

- Sentinel-2 images
- 10m resolution
- 293 images of 1000x1000

Forest segmentation

Applications:

- Deforestation monitoring
- Forest fire alerts

Slovenia 2017 dataset (Sinergise Ltd.):

- Sentinel-2 images
- 10m resolution
- 293 images of 1000x1000

Model	Acc.	Prec.	Rec.	Spec.	Jacc.
LeNetFCN	77.10	67.93	63.57	84.22	48.89
C-FCN++	77.40	69.52	61.28	85.87	48.31
C-FCN	78.87	68.95	70.38	83.33	53.44
mobDeconvNet	80.61	71.52	72.68	84.78	56.36
C-UNet++	80.62	72.92	69.61	86.41	55.31
stridedUNet	80.67	72.52	70.70	85.92	55.77
C-UNet	83.33	76.79	73.99	**88.24**	60.47
U-Net	84.04	73.37	**84.28**	83.92	**64.54**
mobUNet	**84.27**	**75.19**	81.13	85.92	63.99

Table: Slovenia 2017 forest segmentation results (10m resolution).

Impact of skip connections

C-UNet	Acc.	Prec.	Rec.	Spec.	Jac.
with skips	**95.90**	96.22	**90.01**	98.46	**86.93**
w/o skips	95.78	**96.53**	89.27	**98.61**	86.50
C-UNet++	Acc.	Prec.	Rec.	Spec.	Jac.
with skips	94.51	92.30	**89.33**	96.76	83.14
w/o skips	**94.85**	**94.18**	88.48	**97.63**	**83.89**

Table: Impact of skip connections on performance of C-UNet(++) on 38-Cloud dataset.

Impact of skip connections

C-UNet	Acc.	Prec.	Rec.	Spec.	Jac.
with skips	**95.90**	96.22	**90.01**	98.46	**86.93**
w/o skips	95.78	**96.53**	89.27	**98.61**	86.50
C-UNet++	Acc.	Prec.	Rec.	Spec.	Jac.
with skips	94.51	92.30	**89.33**	96.76	83.14
w/o skips	**94.85**	**94.18**	88.48	**97.63**	**83.89**

Table: Impact of skip connections on performance of C-UNet(++) on 38-Cloud dataset.

Architecture / Image size	384x384	2000x2000
U-Net	34.9 MB	236.5 MB
StridedUNet	4.22 MB	114.4 MB
MobUNet	1.69 MB	45.8 MB
C-UNet with skips	1.97 MB	53.4 MB
C-UNet++ with skips	1.97 MB	53.4 MB
C-UNet without skips	**0 MB**	**0 MB**
C-UNet++ without skips	**0 MB**	**0 MB**

Table: Memory footprint of skip connections for 32-bit float inference (in Megabytes), computed for two images sizes.

Number of parameters and storage

Architecture	N_{params}	FLOPs	Storage (MB)
C-FCN++	273	4 654	0.047
LeNet_FCN	614	10 455	0.049
C-FCN	1 438	24 233	0.066
mobDeconvNet	7 919	134 256	0.149
C-UNet++	9 129	154 800	0.172
C-UNet	51 113	867 712	0.735
mobUNet	52 657	893 882	0.724
StridedUNet	79 785	1 355 704	1.0
ESPNet_A	200 193	3 399 310	2.7
U-Net	31 094 497	528 578 588	357

Table: Network parameters, FLOPs (FLoating-point OPerationS) and storage size. Our architectures in bold font. FLOPs computed for a 384x384 input using Tensorflow's profiler.

Experiment on Raspberry Pi 4

Typical low-power system:

- Credit card size
- Quad-core ARM Cortex-A72 1.5GHz
- 2GB RAM
- 3W idle, 6W load

Less powerful than most phones.

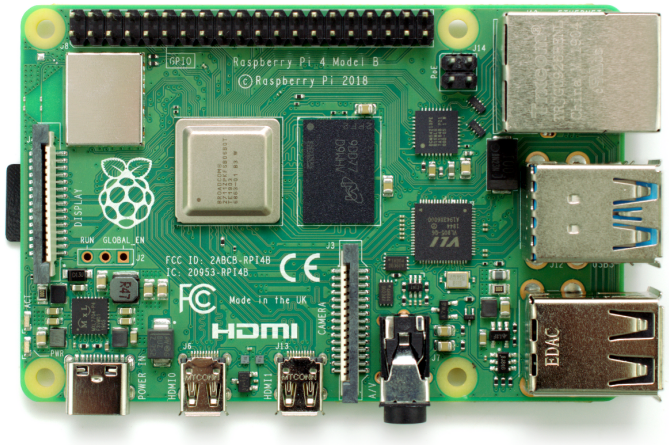


Figure: Raspberry Pi 4

Experiment on Raspberry Pi 4

Model	384x384	1024x1024	2048x2048
C-FCN++	**0.130**	0.773	**2.663**
stridedUNet	0.133	**0.752**	2.666
C-FCN	0.148	0.903	3.017
C-UNet++	0.204	1.242	4.295
C-UNet	0.404	2.867	10.355
mobUNet	0.688	5.507	OOM
mobDeconvNet	0.755	N/A	N/A
ESPNet_A	1.878	15.940	OOM
U-Net	5.035	OOM	OOM

Table: Execution time on images of different sizes in seconds on Raspberry Pi 4, averaged over 20 iterations. OOM = Out of Memory

Experiment on FPGA

Implementation:

- Altera Cyclone V 5CSXC6 @ 100MHz
- 16-bit fixed-point quantization
- Automatic HDL code generator based on VGT (Hamdan, 2018)

Figure: OPS-SAT (ESA 2019)

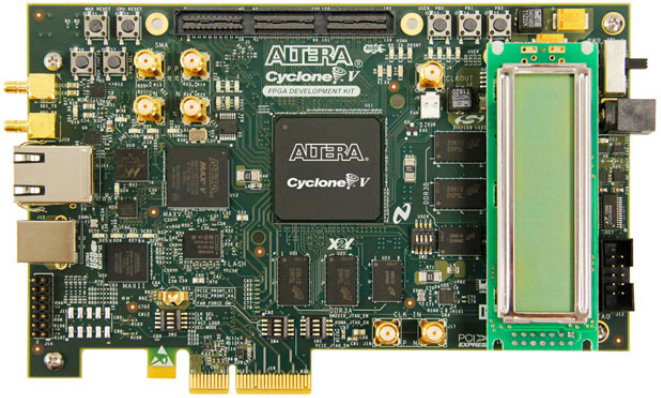


Figure: Intel/Altera Cyclone V SoC with FPGA

Experiment on FPGA

Implementation:

- Altera Cyclone V 5CSXC6 @ 100MHz
- 16-bit fixed-point quantization
- Automatic HDL code generator based on VGT (Hamdan, 2018)

Results (C-FCN++):

- Less than 150ms for a 2000x2000 image
- 64% of Adaptive Logic Modules used
- Possibility of real-time processing (7fps)

Figure: OPS-SAT (ESA 2019)

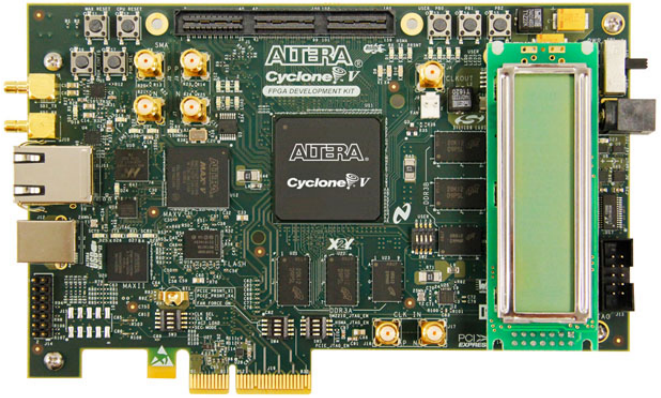


Figure: Intel/Altera Cyclone V SoC with FPGA

4

Conclusion

Conclusion

Summary:

- 4 low-power semantic segmentation architectures
- Experiments on cloud and forest segmentation
- First work on onboard satellite cloud segmentation on FPGA
- RGB is enough!

Conclusion

Summary:

- 4 low-power semantic segmentation architectures
- Experiments on cloud and forest segmentation
- First work on onboard satellite cloud segmentation on FPGA
- RGB is enough!

Future works:

- Multi-class segmentation experiments
- Building segmentation
- C-FCN and C-Unet(++) implementations on bigger FPGAs

Thank you!

This work is funded by IRT Saint-Exupéry.