

# Residual Conv-Deconv Grid Network for Semantic Segmentation

Damien Fourure<sup>1</sup>

damien.fourure@univ-st-etienne.fr

Rémi Emonet<sup>1</sup>

remi.emonet@univ-st-etienne.fr

Elisa Fromont<sup>1</sup>

elisa.fromont@univ-st-etienne.fr

Damien Muselet<sup>1</sup>

damien.muselet@univ-st-etienne.fr

Alain Tremeau<sup>1</sup>

alain.tremeau@univ-st-etienne.fr

Christian Wolf<sup>2</sup>

christian.wolf@liris.cnrs.fr

<sup>1</sup> Laboratoire Hubert Curien

Universite Jean Monnet,  
Saint-Etienne, France

<sup>2</sup> LIRIS

INSA Lyon,  
Lyon, France

## Abstract

This paper presents GridNet, a new Convolutional Neural Network (CNN) architecture for semantic image segmentation (full scene labelling). Classical neural networks are implemented as one stream from the input to the output with subsampling operators applied in the stream in order to reduce the feature maps size and to increase the receptive field for the final prediction. However, for semantic image segmentation, where the task consists in providing a semantic class to each pixel of an image, feature maps reduction is harmful because it leads to a resolution loss in the output prediction. To tackle this problem, our GridNet follows a grid pattern allowing multiple interconnected streams to work at different resolutions. We show that our network generalizes many well known networks such as conv-deconv, residual or U-Net networks. GridNet is trained from scratch and achieves competitive results on the Cityscapes dataset.

## 1 Introduction

Convolutional Neural Networks (CNN) have become tremendously popular for a huge number of applications [1, 2, 3] since the success of AlexNet [4] in 2012. AlexNet, VGG16 [5] and ResNet [6], are some of the famous architectures designed for image classification which have shown incredible results. While image classification aims at predicting a single class per image (presence or not of an object in an image) we tackle the problem of full scene labelling. Full scene labelling or semantic segmentation from RGB images aims at segmenting an image into semantically meaningful regions, i.e. to provide a class label for each pixel of an image. Based on the success of classical CNN, new networks designed especially for semantic segmentation, named fully convolutional networks have been developed. The

main advantage of these networks is that they produce 2D matrices as output, allowing the network to label an entire image directly. Because they are fully convolutional, they can be fed with images of various sizes (with some constraints).

In order to construct fully convolutional networks, two strategies have been developed: Conv-Deconv networks and dilated convolution-based networks (see Section 2 for more details). Conv-Deconv networks are composed of two parts: the first one is a classical convolutional network with subsampling operations which decrease the feature maps sizes and the second part is a deconvolutional network with upsampling operations which increase the feature maps sizes back to the original input resolution. Dilated convolution-based networks do not use subsampling operations but a "à trous" algorithm on dilated convolutions to increase the receptive field of the network.

If increasing the depth of the network has often gone hand in hand with increasing the performance on many data rich applications, it has also been observed that the deeper the network, the more difficult its training is, due to vanishing gradient problems during the back-propagation steps. Residual networks solve this problem by using identity residual connections to allow the gradient to back-propagate more easily. As a consequence, they are often faster to train than classical neural networks. The residual connections are thus now commonly used in all new architectures.

Lots of pre-trained (usually on Imagenet [1]) residual networks are available for the community. They can be fine-tuned for a new task. However, the structure of a pre-trained network cannot be changed radically which is a problem when a new architecture, such as ours, comes out.

In this paper we present GridNet, a new architecture especially designed for full scene labelling. GridNet is composed of multiple paths from the input image to the output prediction, that we call streams, working at different image resolutions. High resolution streams allow the network to give an accurate prediction in combination with low resolution streams which carry more context thanks to bigger receptive fields. The streams are interconnected with convolutional and deconvolutional units to form the columns of our grid. With these connections, information from low and high resolutions can be shared. It is especially useful for semantic segmentation because high resolution streams can give accurate predictions when low resolution ones carry more context thanks to a bigger receptive field.

In Section 2, we review the network architectures used for full scene labelling from which the GridNet takes inspiration and we show how our approach generalises existing methods. In Section 3, we present the core components of the proposed GridNet architecture. Finally, Section 4 shows results on the Cityscapes dataset.

## 2 Related Work

In traditional CNN, convolutional and non-linearity computational units are alternated with subsampling operations. The purpose of subsampling is to increase the network receptive field while decreasing the feature maps sizes. A big receptive field is necessary for the network to get bigger context for the final prediction while the feature maps size reduction is a beneficial side effect allowing to increase the number of feature maps without overloading the (GPU) memory. In the case of semantic segmentation where a full-resolution prediction is expected, the subsampling operators are detrimental as they decrease the final output resolution.

To get a prediction at the same resolution than the input image, Long, Shelhamer *et al.*

proposed recently a Fully Convolutional Networks (FCN) [18] by adding a deconvolution part after a classical convolutional neural network. The idea is that, after going down to the convolutional network, a deconvolution part, using upsampling operator and deconvolution (or fractionally-strided convolution) increases the feature maps size back to the input resolution. Noh *et al.* [19] extended this idea by using maximum unpooling upsampling operators in the deconvolution part. The deconvolution network is the symmetric of the convolution one and each maximum pooling operation in the convolution is linked to a maximum unpooling one in the deconvolution by sharing the pooling positions. Ronneberger *et al.* [20] are going even further with their U-Net by concatenating the feature maps obtained in the convolution part with feature maps of the deconvolution part to allow a better reconstruction of the segmented image. Finally, Lin *et al.* [8] used the same idea of U-Net but instead of concatenating the feature maps directly, they used a refineNet unit, containing residuals units, multi-resolutions fusions and chained residual pooling, allowing the network to learn a better semantic transformation.

All of these networks are based on the idea that subsampling is important to increase the receptive field and try to override the side effect of resolution lost with deconvolutional techniques. In our GridNet, composed of multiple streams working at different feature map sizes, we use the subsampling and upsampling operators as connectors between streams allowing the network to take decisions at any resolution. The upsampling operators are not used to correct this side effect but to allow multi-scale decisions in the network. In a recent work, Newell *et al.* [21] stacked many U-Net showing that successive steps of subsampling and upsampling are important to improve the performance of the network. This idea is improved in our GridNet with the strong connections between streams.

Yu *et al.* [22] studied another approach to deal with the side effect of subsampling. They show that, for a semantic labelling task, the pooling operations are harmful. Therefore, they remove the subsampling operators to keep the feature maps at the same input resolution. Without subsampling, the receptive field is very small so they use dilated convolution to increase it. Contrarily to classical convolutions, where the convolution mask is applied onto neighbourhood pixels, dilated convolutions have a dilatation parameter to apply the mask to more and more apart pixels. In their work Wu *et al.* [23] adapt the popular ResNet [8] pre-trained on ImageNet [9] for semantic segmentation. Residual networks [8] are very deep networks trained with residual connections allowing the gradient to propagate easily to the first layers of the network correcting the vanishing gradient problems. Wu *et al.* only keep the first layers of ResNet and change the classical convolutions into dilated ones. For memory problems, they also keep 3 subsampling operators so the final output prediction is at 1/8 of the input size, and then use linear interpolations to retrieve the input resolution. In [23], Zhao *et al.* replace the linear interpolation by a Pyramid Pooling module. The pyramid pooling module is composed of multiple pooling units of different factor, followed by convolutions and upsample operators to retrieve the original size. All the feature maps obtained with different pooling sizes are then concatenated before a final convolution operator that gives the prediction. When Zhao *et al.* add a module at the end of the network to increase the feature maps size and allow a multi-scale decision, we incorporate this multi-scale property directly into our network with the different streams.

In their work, He *et al.* [5] study the importance of residual units and give detailed results on the different strategies to use residual connections (whether batch normalisation should be used before the convolutions, whether linearity operator should be used after the additions, etc.). GridNet also benefits from these residuals units.

With their Full Resolution Residual Network (FRRN) [14], Pohlen *et al.* combine a conv-

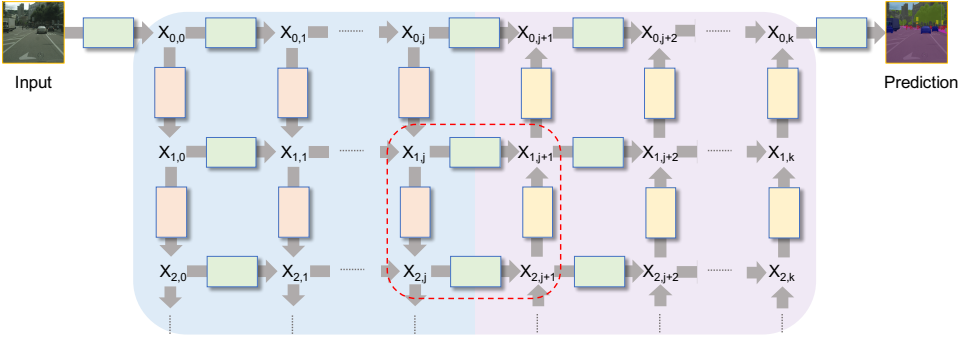


Figure 1: GridNet: each green unit is a residual bloc, which does not change the input map resolution nor the number of feature maps. Red blocks are convolutional units with resolution loss (subsampling) and the doubled number of feature maps. Yellow units are deconvolutional blocks which increase the resolution (upsampling) and divide by two the number of feature maps. A zoom on the red square part with a detailed compositions of each blocks is shown in Figure 2

deconv network with a residual one. They also use different streams but only two of them: one for the residual network linked with upsampling and subsampling operations, and one for the conv-deconv network which does not have any residual connections. GridNet subsumes FRNN and can be seen as a generalisation of this network.

Saxena *et al.* have presented a Convolutional Neural Fabrics [16] which structure looks similar to ours but which aim is different. They choose the best network architecture by training the full grid and showing which path is the most used in the grid. In our case, the full grid is used to tackle our problem.

### 3 GridNet

The computation graph of a GridNet is organised into a two-dimensional grid pattern, as shown in Figure 1. Each feature map  $X_{i,j}$  in the grid is indexed by line  $i$  and column  $j$ . Maps are connected through computation layers. Information enters the model as input to the first block of line 0 and leaves it as output from the last block of line 0. Between these two points, information can flow in several paths, either directly between these entry/exit points in a straight line or in longer paths which also involve lines with indexes  $\neq 0$ .

Information is processed in layers which connect blocks  $X_{i,j}$ . The main motivation of our model is the difference between layers connecting feature maps horizontally or vertically: We call horizontal connections “streams”. Streams are fully convolutional and keep feature map sizes constant. They are also residual, i.e. they predict differences to their input [8]. Stream blocks are green in figure 1. Vertical computing layers are also convolutional, but they change the size of the feature maps: according to the position in the grid, spatial sizes are reduced by subsampling or increased by upsampling, respectively shown as red and yellow blocks in Figure 1. Vertical connections are *NOT* residual. The main idea behind this concept is an adaptive way to compute how information flows in the computation graph. Subsampling and upsampling are important operations in resolution preserving networks, which allow to increase the size of receptive fields significantly without increasing filter sizes, which would

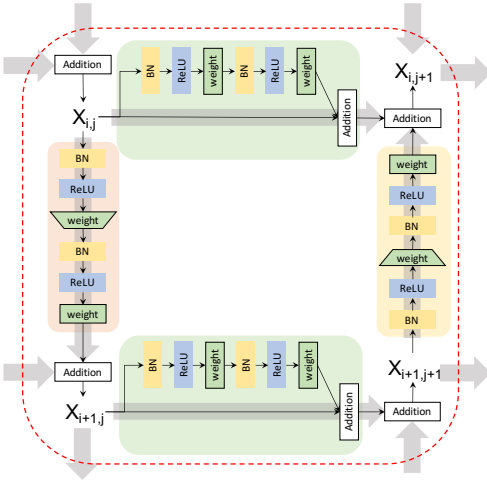


Figure 2: Detailed schema of a GridBlock. Green units are residual units keeping feature map dimensions constant between input and output. Red units are convolutional + subsampling and increase the feature dimensions. Yellow units are deconvolutional + upsampling and decrease the feature dimensions (back to the original one to allow the addition). The weight in trapezium illustrate the upsampling/subsampling operation obtained with strided convolutions. BN=Batch normalization.

require a higher number of parameters<sup>1</sup>. On the other hand, the lost resolution needs to be generated again through learned upsampling layers. In our network, information can flow on several parallel paths, some of which preserve the original resolution (horizontal only paths) and some of which pass through down+up sampling operations. In the lines of the skip-connections in U-networks [14], we conjecture that the former are better suited for details, whereas high-level semantic information will require paths involving vertical connections.

Following the widespread practise, each subsampling unit reduces feature map size by a factor 2 and multiplies the number of feature maps by 2. More formally, if the stream  $X_i$  takes as input a tensor of dimension  $(F_i \times W_i \times H_i)$  where  $F_i$  is the number of feature maps and  $W_i, H_i$  are respectively the width and height of the map, then the stream  $X_{i+1}$  is of dimension  $(F_{i+1} \times W_{i+1} \times H_{i+1}) = (2F_i \times W_i/2 \times H_i/2)$ .

Apart from border blocks, each feature map  $X_{i,j}$  in the grid is the result of two different computations: one horizontal residual computation processing data from  $X_{i,j-1}$  and one vertical computation processing data from  $X_{i-1,j}$  or  $X_{i+1,j}$  depending if the column is a subsampling or upsampling one. Several choices can be taken here, including concatenating features, summing, or learned fusion. We opted for summing, a choice which keeps model capacity low and blends well with the residual nature of the grid's streams. The details are given as follows: Let  $\Theta^{Res}(\cdot)$ ,  $\Theta^{Sub}(\cdot)$  and  $\Theta^{Up}(\cdot)$  be respectively the mapping operation for the residual unit (green block in Figure 1), subsampling unit (red block) and upsampling unit (yellow block). Each mapping takes as input a feature tensor  $X$  and parameters  $\theta$ .

If the column  $j$  is a subsampling column then:

$$X_{i,j} = X_{i,j-1} + \Theta^{Res}(X_{i,j-1}, \theta_{i,j}^{Res}) + \Theta^{Sub}(X_{i-1,j}, \theta_{i,j}^{Sub})$$

Otherwise, if the column  $j$  is an upsampling one then:

$$X_{i,j} = X_{i,j-1} + \Theta^{Res}(X_{i,j-1}, \theta_{i,j}^{Res}) + \Theta^{Up}(X_{i+1,j}, \theta_{i,j}^{Up})$$

Border blocks are simplified in a natural way. An alternative to summing is feature map concatenation, which increases capacity and expressive power of the network. Our experiments

<sup>1</sup>An alternative would be to use dilated convolutions with the *à trous* algorithm [15].

on this version showed, that it significantly increases training difficulty, especially when the network is trained from scratch.

The capacity of a GridNet is defined by three hyper parameters,  $N_S$ ,  $N_{C_S}$  and  $N_{C_u}$  respectively the number of residuals streams, the number of subsampling columns and number of upsampling columns. Inspired by the symmetric Conv-Deconv networks [18], we set  $N_{C_S}=N_{C_u}$  in our experiments, but this constraint can be lifted.

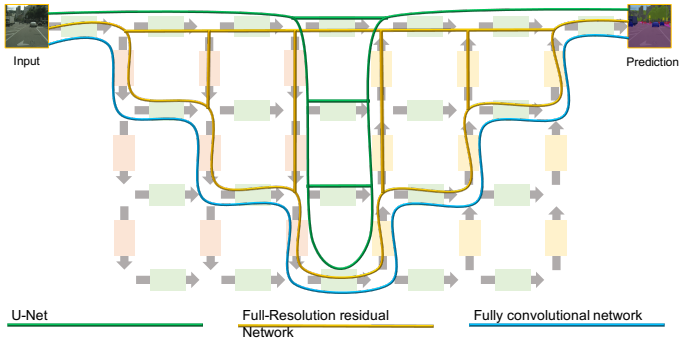


Figure 3: GridNets generalize several classical resolution preserving neural models, like Conv-Deconv networks [18] (blue connections) and U-networks [15] (green connections) and Full Resolution Residual Networks (FRRN) [14] (yellow connections).

GridNets generalize several classical resolution preserving neural models, as shown in figure 3. Standard models can be obtained by removing connections between feature maps in the grid. If we keep the connections shown in blue in figure 3, we obtain Conv-Deconv networks [18], which feature a single path without branches, first down-sampling and then up-sampling data. U-networks [15] (shown by green connections) add skip-connections between down-sampling and corresponding up-sampling stages, and Full Resolution Residual Networks (FRRN) [14] (shown as yellow connections) add a more complex structure.

### 3.1 Blockwise dropout for GridNets

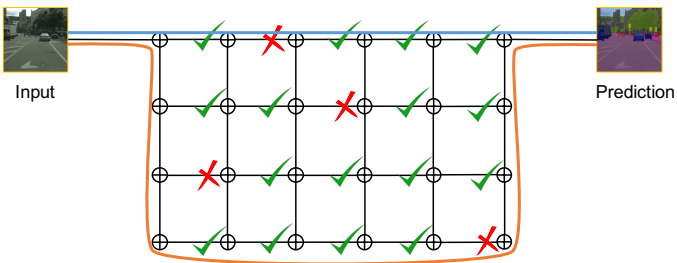


Figure 4: The blue path only using the high resolution stream is shorter than the orange path which also uses low resolution streams. To force the network to use all streams we randomly drop streams during training, indicated by red crosses.

A side effect of our 2D grid topology with input and output both situated on line 0 is that the path from the input to the output is shorter across the high resolution stream (blue

path in figure 4) than with the low resolution one (orange path in figure 4). Longer paths in deep networks may translate into well known problems of vanishing and exploding gradient. As a consequence, paths involving lower resolutions streams take a longer time to converge in our networks and generally a more difficult to train. To force the network to use all of its available streams, we employed a technique inspired by dropout, which will call total dropout. We randomly drop residual streams and set the corresponding residual mapping to zero. We force the network to use all streams during inference.

More formally, let  $r_{i,j} = \text{Bernoulli}(p)$  be a random variable taken from a *Bernoulli* distribution, which is equal to 1 with a probability  $p$  and 0 otherwise. Then the feature map computation becomes:  $X_{i,j} = X_{i,j-1} + r_{i,j}(\Theta^{\text{Res}}(X_{i,j-1}, \theta_{i,j}^{\text{Res}})) + \Theta^{\{\text{Sub}; \text{Up}\}}(X_{i+1,j}, \theta_{i,j}^{\{\text{Sub}; \text{Up}\}})$

### 3.2 Parameter count and memory footprint for GridNet

In neural networks, the memory footprint depends on both the number of activations and the number of trainable parameters. In many architectures, these two numbers are highly correlated. While it is still the case in a GridNet, the grid structure provides a finer control over these numbers. Let us consider a GridNet built following the principles from Section 3: with  $N_S$  streams,  $N_{Cs}$  subsampling columns and  $N_{Cu}$  upsampling columns, with the first stream having  $F_0$  feature maps at resolution  $W_0 \times H_0$ , and the others streams obtained by downsampling by  $2 \times 2$  and increasing feature maps by 2. From the exact computation of the number of parameters  $nb_{\text{param}}$  and the number of activation values  $nb_{\text{act}}$ , we can derive meaningful and approximations that can be interpreted.

$$nb_{\text{param}} \approx 18 \times 2^{2*(N_S-1)} F_0^2 (2.5N_{Cs} + N_{Cu} - 2)$$

Which illustrate that the number of parameters is most impacted by the number of streams  $N_S$ , followed by number of feature maps (controlled by  $F_0$ ), then only by the number of columns.

$$nb_{\text{activ}} \approx 6H_0W_0F_0(4N_{Cu} + 3N_{Cs} - 2)$$

Which shows that the size of the activations mainly depends on the first stream size (width, height and number of feature maps) and the grows linearly with the number of columns. In practice, the total memory footprint of a network at training time depends on its number of parameters and the size of the activations, but also on the optimizer and the mini-batch size. The gradient computed by the optimizer takes the same space as the parameters themselves and the optimizer may also keep statistics on the parameters and gradients (as in Adam). The mini-batch size mechanically multiplies the memory footprint as the activations of multiple inputs need to be computed and stored in parallel.

## 4 Experimental results

We evaluated the method on the Cityscapes dataset, which consists in high resolution ( $1024 \times 2048$  pixels) images taken from a car driving across 50 different cities in Germany. 2975 training images and 500 test images have been fully labelled with 30 semantic classes. However, only 19 classes are taken into account for the automatic evaluation on the Cityscapes website<sup>2</sup>, therefore we trained GridNet on these classes only. Semantic classes are also

<sup>2</sup><https://www.cityscapes-dataset.com/>



grouped into 8 semantic categories. The ground truth is not provided for the test set but an online evaluation is available on the Cityscapes website. The dataset contains also 19998 images with coarse (polygonal) annotations but, we chose not to use them for training.

The Cityscapes performance are evaluated based on the Jaccard Index, commonly known as the Pascal VOC Intersection-over-Union (IoU) metric. The IoU is given by  $\frac{TP}{TP+FP+FN}$  where  $TP$ ,  $FP$  and  $FN$  stand for the number of True Positive, False Positive and False Negative pixels. IoU is biased toward object instances that cover a large image area so an instance-level intersection-over-union metric  $iIoU$  is also used. The  $iIoU$  is computed by weighting the contribution of each pixel by the ratio of the class average instance size to the size of the respective ground truth instance. Finally, they give results accuracy for two semantic granularities (class and category) with the weighted and not weighted IoU metric leading to 4 measurements.

For our experiments we tested GridNet with 5 streams with the following feature map dimensions: [16, 32, 64, 128, 256]. GridNet is composed of 3 subsampling columns (convolutional parts) followed by 3 upsampling columns (deconvolutional part). To fit a batch of 4 images fully on a single GPU, we limit input size to tiles of  $400 \times 400$  pixels, which means that the low-resolution stream has feature maps of size  $25 \times 25$  elements and 256 dimensions.

The high resolution input images ( $1024 \times 2048$  pixels) allow us to crop patches of different resolutions. For training, we crop patches of random sizes between  $400 \times 400$  and  $1024 \times 1024$  at random locations. Then the patches are resized to  $400 \times 400$  and fed to the network. For data augmentation, we also apply random horizontal flipping. We do not apply any post-processing for the images but we added a batch normalization layer at the input of the grid. We use the classical cross-entropy loss function to train our network.

We trained our network using the Adam optimizer with a learning rate of 0.01, a learning rate decay of  $5 \times 10^{-6}$ ,  $\beta_1 = 0.9$ ,  $\beta_2 = 0.999$  and an  $\epsilon = 1 \times 10^{-8}$ . After 800 epochs the learning rate is decreased to 0.001. We stopped our experiments after 10 days leading to approximately 1900 training epochs.

## 4.1 Discussion

We conducted a study to evaluate the effects of each of our architectural components and design choices. The results are presented in Table 1. Sum† is the results given by the network presented in section 3 with total dropout operators (see section 3.1). Total dropout proved to be a key design choice, which lead to significant improvement in accuracy.

We also provide results of a fully residual version, where identity connections are added in both horizontal and vertical computing connections (whereas the proposed method is residual in horizontal streams only). Fully residuality did not prove to be an advantage. Total dropout did not solve learning difficulties and further impacted training stability negatively.

Finally, concatenation of horizontal and vertical streams, instead of summing, did also not prove to be an optimal choice. We conjecture that the high capacity of the network did not prove to be an advantage.

At this point we should indicate, that we performed architecture optimizations for different design-choices. When a design choice lead to a higher number of trainable parameters (e.g. by concatenating features), we performed experiments by removing parameters through architectural changes. The architecture described above is the one of the proposed method.



Fusion	h-residual	v-residual	Total dropout	Performance measures			
				IoU class	iloU class	IoU categ.	iloU categ.
Sum	X			57.2	35.6	83.1	68.4
Sum <sup>†</sup>	X		X	<b>65.0</b>	<b>43.2</b>	85.6	70.1
Sum	X	X		57.6	36.8	<b>86.0</b>	<b>72.6</b>
Sum	X	X	X	35.6	23.0	62.1	60.3
Concat	X			53.9	34.0	82.2	65.2

Table 1: Results of our different GridNet variants on the Cityscapes validation set: Fusion indicates how feature maps are fused between horizontal and vertical computations. The second and third columns indicate whether horizontal resp. vertical computation are residual. <sup>†</sup> proposed method.

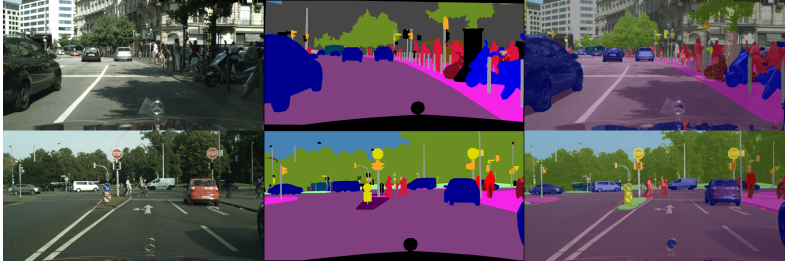


Figure 5: Semantic segmentation results obtained with GridNet. Left the input image, middle the ground truth and right our results.

## 4.2 Qualitative and Quantitative Results

Figure 5 illustrates the segmentation performance on sample images. In Table 2 we compare results by the proposed method compared to the state of the art taken from the official Cityscapes website. We restrict the comparison to methods using the same input information as us (no coarse annotation, no stereo inputs). Our network gives results comparable the state-of-the-art network, namely the FRNN network presented in Section 2.

All other results have been obtained by networks pre-trained for classification using the Imagenet dataset. Note that among the 9 other reported results of pre-trained networks, only one of them (RefineNet) give slightly better results than our networks.

Name	Without pretraining	Performance measures			
		IoU class	iloU class	IoU categ.	iloU categ.
FRNN - [10]	✓	<b>71,8</b>	<b>45,5</b>	<b>88,9</b>	<b>75,1</b>
GridNetwork - Ours	✓	69,45	44,06	87,85	71,11
RefineNet - [8]	✗	<b>73,6</b>	47,2	87,9	70,6
Lin <i>et al.</i> - [9]	✗	71,6	<b>51,7</b>	87,3	74,1
LRR - [11]	✗	69,7	48	<b>88,2</b>	<b>74,7</b>
Yu <i>et al.</i> - [12]	✗	67,1	42	86,5	71,1
DPN - [13]	✗	66,8	39,1	86	69,1
FCN - [14]	✗	65,3	41,7	85,7	70,1
Chen <i>et al.</i> - [15]	✗	63,1	34,5	81,2	58,7
Szegedy <i>et al.</i> - [16]	✗	63	38,6	85,8	69,8
Zheng <i>et al.</i> - [17]	✗	62,5	34,4	82,7	66

Table 2: Results on the Cityscapes dataset benchmark. We only report published papers which use the same data as us (no coarse annotation, no stereo inputs).

## 5 Conclusion

We have introduced a novel network architecture specifically designed for semantic segmentation. The model generalizes a wide range of existing neural models, like conv-deconv networks, U-networks and Full Resolution Residual Networks. A two-dimensional grid structure allows information to flow horizontally in a residual resolution-preserving way or vertically through down- and up-sampling layers. GridNet shows promising results even when trained from scratch without any pre-training on classification (Imagenet), which is a standard approach in the semantic segmentation literature. We believe that our network could also benefit from better weight initialization, for example by pre-training it on the ADE20K dataset.

## Acknowledgment

Authors acknowledge the support from the ANR project SoLStiCe (ANR-13-BS02-0002-01). They also want to thank Nvidia for providing two Titan X GPU.

## References

- [1] Xiang Bai, Baoguang Shi, Chengquan Zhang, Xuan Cai, and Li Qi. Text/non-text image classification in the wild with convolutional neural networks. *Pattern Recognition*, 66:437–446, 2017.
- [2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *CoRR*, abs/1606.00915, 2016. URL <http://arxiv.org/abs/1606.00915>.
- [3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- [4] Golnaz Ghiasi and Charless C. Fowlkes. Laplacian pyramid reconstruction and refinement for semantic segmentation. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III*, pages 519–534, 2016. doi: 10.1007/978-3-319-46487-9\_32. URL [http://dx.doi.org/10.1007/978-3-319-46487-9\\_32](http://dx.doi.org/10.1007/978-3-319-46487-9_32).
- [5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, pages 630–645, 2016.
- [6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *26th Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1106–1114, 2012.

- [8] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian D. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. *CoRR*, abs/1611.06612, 2016.
- [9] Guosheng Lin, Chunhua Shen, Anton van den Hengel, and Ian D. Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 3194–3203, 2016. doi: 10.1109/CVPR.2016.348. URL <http://dx.doi.org/10.1109/CVPR.2016.348>.
- [10] Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen Change Loy, and Xiaoou Tang. Semantic image segmentation via deep parsing network. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1377–1385, 2015. doi: 10.1109/ICCV.2015.162. URL <http://dx.doi.org/10.1109/ICCV.2015.162>.
- [11] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII*, pages 483–499, 2016.
- [12] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1520–1528, 2015.
- [13] Ngoc-Quan Pham, Germán Kruszewski, and Gemma Boleda. Convolutional neural network language models. In *EMNLP*, 2016.
- [14] Tobias Pohlen, Alexander Hermans, Markus Mathias, and Bastian Leibe. Full-resolution residual networks for semantic segmentation in street scenes. *CoRR*, abs/1611.08323, 2016.
- [15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*, pages 234–241, 2015.
- [16] Shreyas Saxena and Jakob Verbeek. Convolutional neural fabrics. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4053–4061, 2016.
- [17] Amir Shahroudy, Tian-Tsong Ng, Qingxiong Yang, and Gang Wang. Multimodal multipart learning for action recognition in depth videos. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(10):2123–2129, 2016.
- [18] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651, 2017.
- [19] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

- [20] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1–9, 2015. doi: 10.1109/CVPR.2015.7298594. URL <http://dx.doi.org/10.1109/CVPR.2015.7298594>.
- [21] Zifeng Wu, Chunhua Shen, and Anton van den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *CoRR*, abs/1611.10080, 2016. URL <http://arxiv.org/abs/1611.10080>.
- [22] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representations (ICLR)*, 2016.
- [23] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. *CoRR*, abs/1612.01105, 2016.
- [24] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. Conditional random fields as recurrent neural networks. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1529–1537, 2015. doi: 10.1109/ICCV.2015.179. URL <http://dx.doi.org/10.1109/ICCV.2015.179>.