

Lab CudaVision
Learning Vision Systems on Graphics Cards (MA-INF 4308)

CudaLab Project

03.02.2023

PROF. SVEN BEHNKE, ANGEL VILLAR-CORRALES

Contact: villar@ais.uni-bonn.de

Video Semantic Segmentation

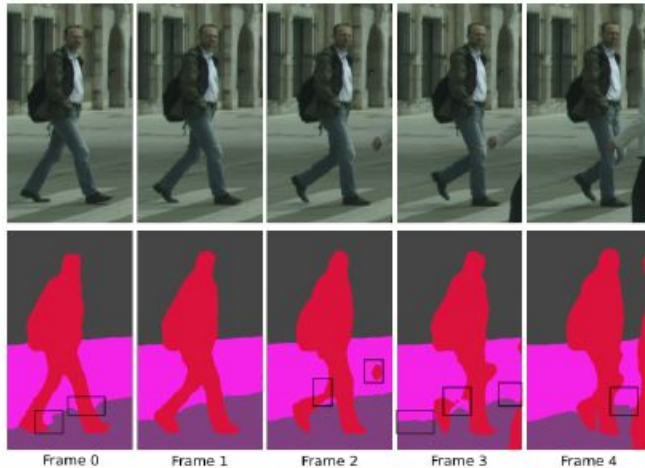
Video Semantic Segmentation

- **Semantic Segmentation:** Predicting a semantic category for every pixel in an image
- **Video Semantic Segmentation:** Predicting a semantic category for every pixel in every frame of a video sequence:
 1. Apply model frame-by-frame
 2. Exploit temporal dependencies
- **Applications:**
 - Autonomous driving
 - Robotics
 - Agriculture
 - ...



Challenges

- Processing frame by frame leads to errors (flickering, ghosting, ...)
- Difficult handling of occlusions
- Temporal consistency can correct most issues



Proposed Approach

Inspiration

Recurrent U-Net for Resource-Constrained Segmentation

Wei Wang^{*} Kaicheng Yu^{*} Joachim Hübner^{*} Pascal Fua^{*}
Mathieu Salzmann
CVLab, EPFL, 1015 Lausanne
{wang.1, yu, huebner, fua}@epfl.ch

Abstract

State-of-the-art segmentation methods rely on very deep networks that are not always easy to train without very large training datasets and tend to be intractably slow to run on standard GPUs. In this paper, we introduce a novel recurrent U-Net architecture that preserves the compactness of the original U-Net [10], while substantially increasing its performance to the point where it outperforms the state of the art on several benchmarks. We will demonstrate its effectiveness for several tasks, including hand segmentation, retina vessel segmentation, and road segmentation. We also introduce a large-scale dataset for hand segmentation.

1. Introduction

While recent semantic segmentation methods achieve impressive results [6, 17, 18, 25], they require very deep networks and their architectures tend to focus on high-resolution and large-scale datasets and to rely on pre-trained backbones. For instance, state-of-the-art models, such as DeepLab [5, 3], PSPNet [16] and RefNetNet [11], use a ResNet101 [25] as their backbone. This results in high GPU memory usage and inference time, and makes them less than ideal for operation in power-limited environments where real-time performance is nevertheless required, such as when segmenting bands using the onboard resources of an Augmented Reality headset. This has been addressed by architectures such as the ICNet [4] at the cost of a substantial performance drop. Perhaps even more importantly, training very deep networks usually requires either massive amounts of training data or image statistics close to that of ImageNet [10], which may not be appropriate in fields such as biomedical image segmentation where the most compact U-Net architecture remains prevalent [13].

In this paper, we argue that these state-of-the-art methods do not naturally generalize to resource-constrained situations and introduce a novel recurrent U-Net architecture that preserves the compactness of the original U-Net [10], while substantially increasing its performance to the point

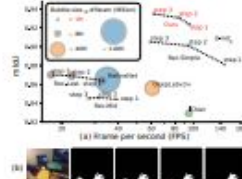


Figure 1: Speed vs. accuracy. Each circle represents the performance of a model in terms of frame-per-second and mIoU accuracy on our Keyboard Hand Dataset using a Titan X(Pascal)/GPU. The radius of each circle denotes the model's number of parameters. For our recurrent approach, we plot these numbers after 1, 2, and 3 iterations, and for these corresponding segmentations in the bottom row. The performance of our approach is plotted in red and the other approaches are defined in Section 3.2. ICNet [17] is slightly faster than RefNetNet at the cost of a significant accuracy drop, whereas RefNetNet [11] and DeepLab [5] are both slower and less accurate on our dataset, presumably because there are not enough training samples to learn their early parameters, when it comes to the current state of the art on 5-band segmentation datasets, one of which is showcased in Fig. 1, and a retina vessel segmentation one. With only 0.3 million parameters, our model is much smaller than the ResNet101-based DeepLabV3+ [1] and RefNetNet [11], with 40 and 118 million weights, respectively. This helps explain why we can outperform state-of-the-art networks on specialized tasks. The pre-trained ImageNet features are not necessarily the best and training sets are not quite as large as CityScapes [15]. As a result, the large networks tend to overfit and do not perform as well as compact models trained from scratch.

The standard U-Net takes the image as input, processes it, and directly returns an output. By contrast, our recurrent

Semantic Segmentation of Video Sequences with Convolutional LSTMs

Andreas Pfaff¹, Katrin Scholz¹, and Klaus Dittmann²

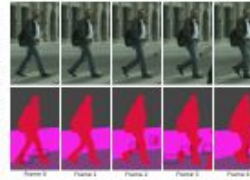


Fig. 1: Segmentation map of a video sequence guided by the ICNet. The black boxes show typical errors such as poorly segmented objects, flickering edges, and flickering (ghost) objects.

1. INTRODUCTION

A challenge of autonomous driving is to understand the environment as good as possible. Hence, multiple sensors are used in self-driving cars, such as the classical RGB cameras. In order to reduce the flood of information of the camera, the images are segmented. Image segmentation denotes the task to assign each image pixel a predefined class, such as ego, pedestrian, or road. State-of-the-art approaches, such as PSPNet [26] or DeepLab [4], are based on convolutional neural networks (CNNs) and achieve very good results on several datasets. However, these approaches are not applicable in the case of autonomous driving, since the inference time for one image amounts to about one second and more, for instance the PSPNet [26] takes about 1.2 seconds and the DeepLab v3+ [4] about 5 seconds on a Nvidia Titan X. In contrast, the performance of current real-time capable approaches, such as SegNet [12], ICNet [15], and ICNet [25], is much worse, and more errors occur. Typical segmentation errors are blurred and flickering object objects, poorly segmented objects, and flickering (ghost) objects. Many of these errors often only occur in a single frame of a video sequence, and are classified correctly in the next frame, as shown in Fig. 1, where a short video sequence was segmented by the ICNet. For instance, the pedestrian was classified correctly in the first to third and in the last frame of the video, while parts of the leg were

not detected in the fourth frame. Furthermore, the border between road and sidewalk is flickering during the video, and a ghost object occurs at time step two. The described errors can be avoided by additionally considering image information of the previous frames instead of processing each image independently. One possibility to take account of the last frames is to use recurrent neural networks. They are able to store information of the past time steps and to reuse them in the current time step. Frequently used recurrent networks are Long-Short-Term Memory networks (LSTM) [16], which can be easily trained and integrated in comparison to other recurrent neural networks. An extension of LSTMs are convolutional LSTMs (convLSTM) [19], which are more suitable for image processing tasks. Convolutional LSTM layers can be added at different positions in the network. For instance, they can be integrated directly in front of the softmax layer, which corresponds to a temporal filtering of the results. Another possible location is between the encoder and decoder in the case of a encoder-decoder network architecture, and is motivated by the fact that the encoder extracts global features of the image. These global features should not change very much between two neighbouring frames so that the usage of the previous global image features may improve the segmentation. In this work, several positions of convLSTM layers in different, real-time capable, state-of-the-art semantic segmentation approaches are compared in terms of accuracy, inference time, and inference time on the CityScapes dataset. It is also investigated if the LSTM based semantic segmentation approaches outperform the pure

^{*}Andreas Pfaff, Katrin Scholz, and Klaus Dittmann are with the Institute of Measurement, Control, and Robotics, TUWAT University, 35001 Ulm, Germany. {andreas.pfaff, katrin.scholz, klaus.dittmann}@tuw.at

This work has been submitted to the IEEE for possible publication. Copyright may be transferred without notice, after which this version may no longer be accessible.

Convolutional Gated Recurrent Networks for Video Segmentation

Mennatallah Siam^{*} Sepehr Valipour^{*} Martin Jagersand Nilanjana Ray
University of Alberta
{mennatallah, valipour, m.j, ray}@ualberta.ca

Abstract

Semantic segmentation has recently witnessed major progress, where fully convolutional neural networks have shown to perform well. However, most of the previous work focused on improving single image segmentation. To our knowledge, no prior work has made use of temporal video information in a recurrent network. In this paper, we introduce a novel approach to implicitly utilize temporal data in videos for online semantic segmentation. The method relies on a fully convolutional network that is embedded into a gated recurrent architecture. This design receives a sequence of consecutive video frames and outputs the segmentation of the last frame. Convolutional gated recurrent networks are used for the recurrent part to preserve spatial connectivity in the image. Our proposed method can be applied in both online and batch segmentation. This architecture is tested for both binary and semantic video segmentation tasks. Experiments are conducted on the recent benchmarks in SegTrack V2, Davis, CityScapes, and Synthia. Using recurrent fully convolutional networks improved the baseline network performance in all of our experiments. Namely, 5% and 3% improvement of F-measure in SegTrack2 and Davis respectively, 5.7% improvement in mean IoU in Synthia and 3.5% improvement in categorical mean IoU in CityScapes. The performance of the RGCN network depends on its baseline fully convolutional network. Thus RGCN architecture can be seen as a method to improve its baseline segmentation network by exploiting spatiotemporal information in videos.

1. Introduction

Semantic segmentation, which provides pixel-wise labels, has witnessed a tremendous progress recently. As shown in [14][16][20][25], it outputs dense predictions and partitions the image to semantically meaningful parts. It has numerous applications including autonomous driving[28][21][7], augmented reality[15] and robotics[23]

^{*}Authors contributed equally

[26]. The work in [14] presented a fully convolutional network and provides a method for end-to-end training of semantic segmentation. It yields a coarse heat-map followed by in-network upsampling to get dense predictions. Following the work of fully convolutional networks, many attempts were made to improve single image semantic segmentation. In [16] a full deconvolution network is presented with stacked deconvolution layers. The work in [25] provided a method to incorporate contextual information using recurrent neural networks. However, one missing element is that the real-world is not a set of still images. In real-time camera or recorded video, much information is preserved from temporal cues. For example, the difference between a walking or standing person is highly recognizable in still images but it is obvious in a video.

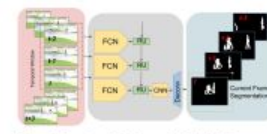
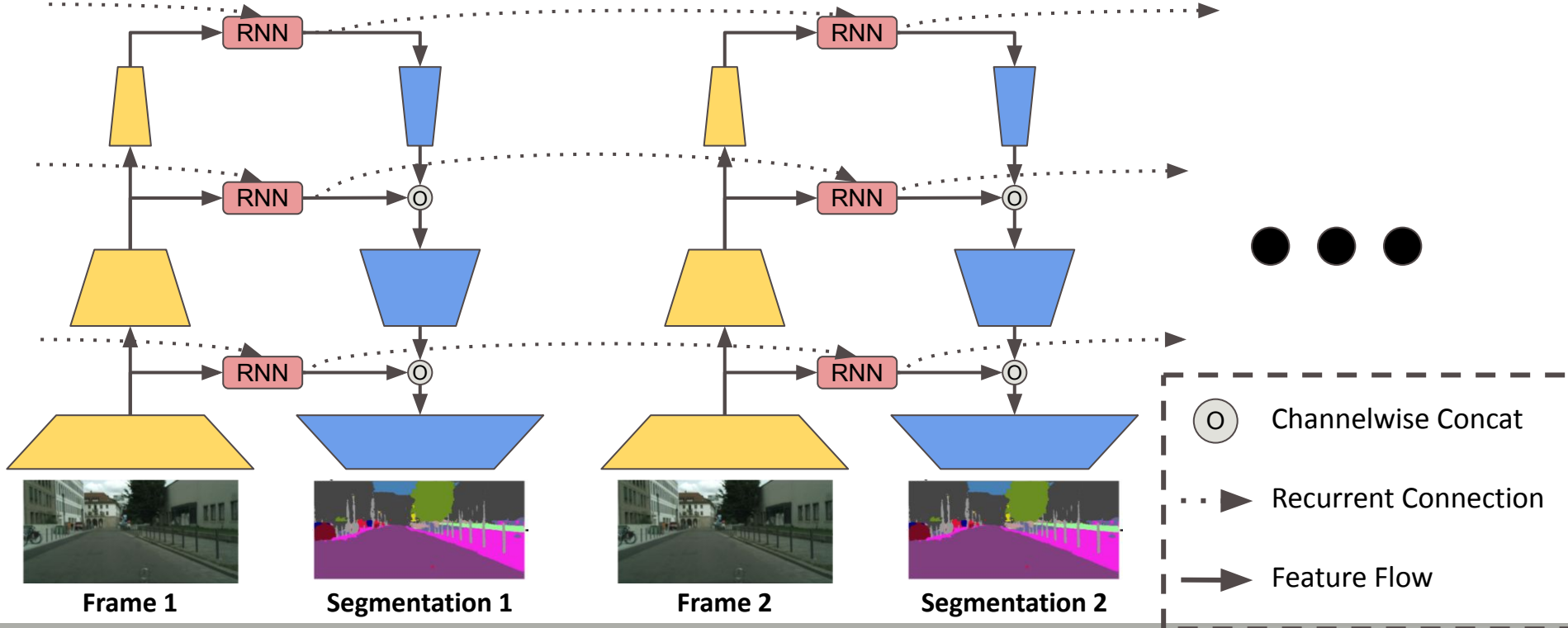


Figure 1: Overview of the Proposed Method of Recurrent FCN. The recurrent part is unfolded for better visualization

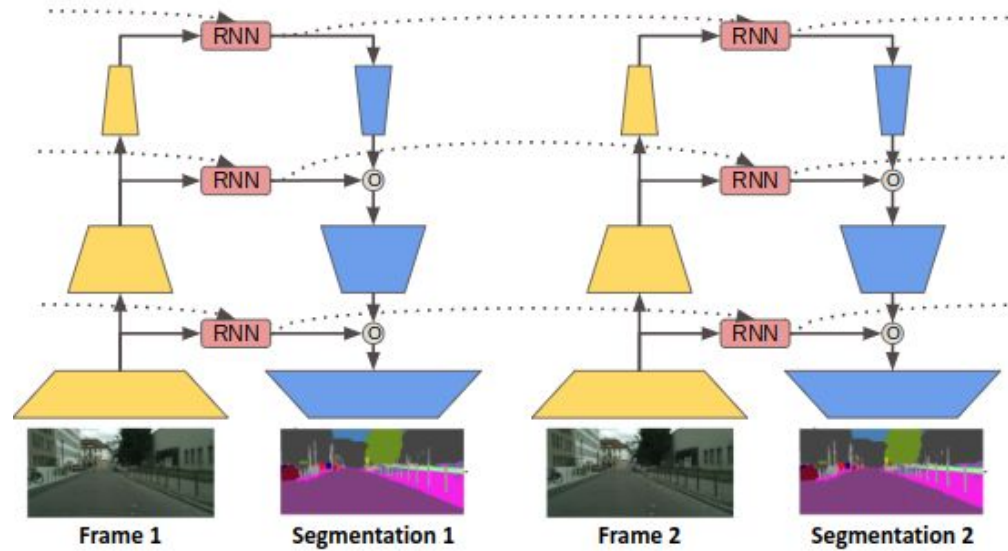
Video segmentation has been extensively investigated using classical approaches. The work in [18] reviews the literature in binary video segmentation. It mainly focuses on semi-supervised approaches[1][19][20] that propagate the labels in one or more annotated frames to the entire video. In [17] a method that uses a combination of Recurrent Neural Networks (RNN) and CNN for RGB-D video segmentation is presented. However, their proposed architecture is difficult to train because of the vanishing gradient. It does not utilize pre-trained networks and it cannot process large images as number of their parameters is quadratic with respect to the input size.

Proposed Model



Proposed Model

- Recurrent UNet-like model
- Convolutional encoder/decoder
 - Convolutional blocks
 - Three residual paths
 - Recurrent modules



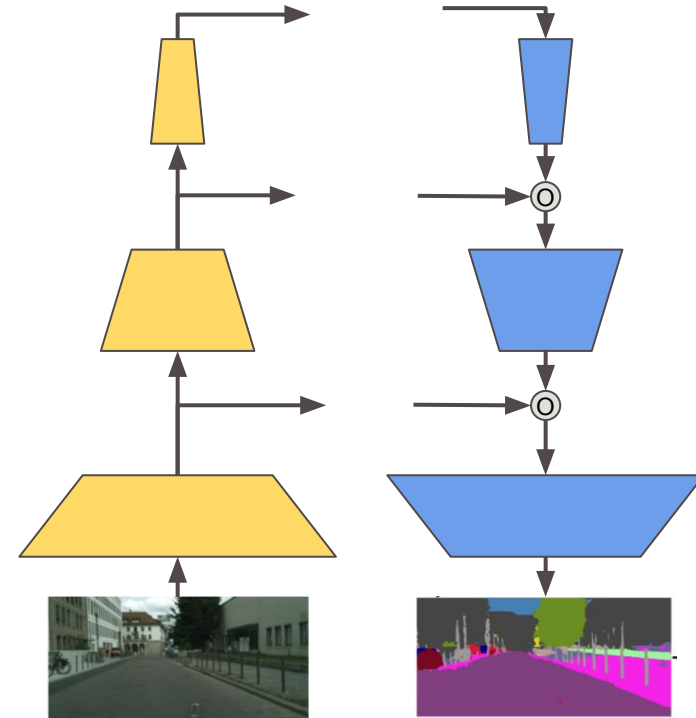
Encoder and Decoder

Encoder

- Extracts features of different level and spatial resolution from the input frames
- Convolutional module
 - ResNet-like
 - VGG-like

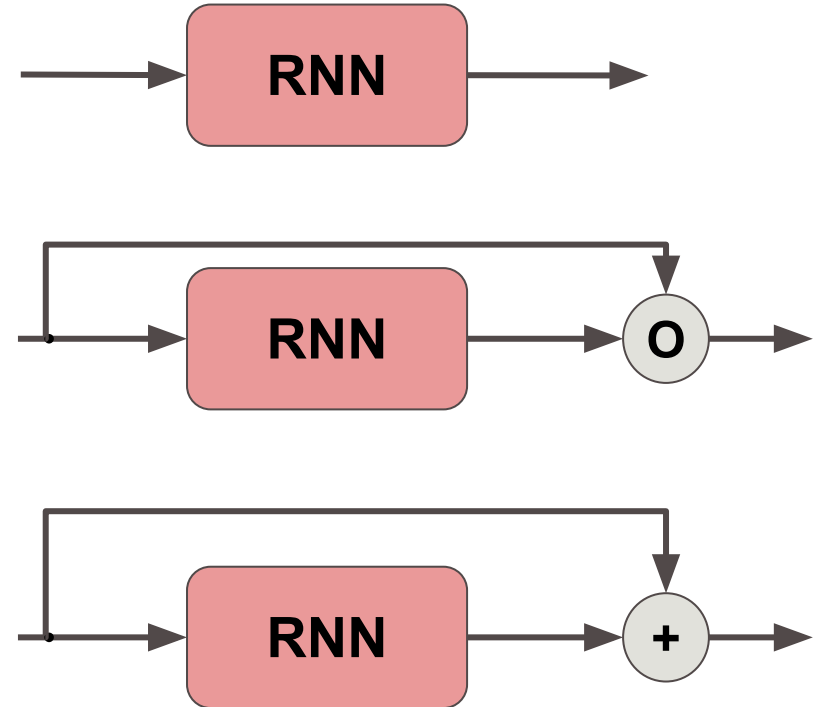
Decoder

- Decodes feature maps into segmentation masks
- Combines features of different level via concatenation in a U-Net manner
- Usually the mirrored version of encoder



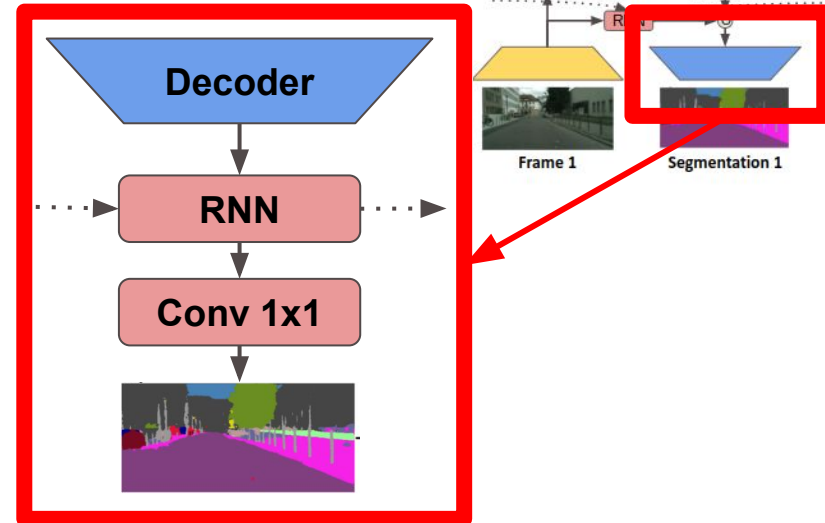
Recurrent Lateral Connections

- Provide features of different level and spatial resolution to the decoder
- Model temporal dependencies via recurrent cells
- Different possible cells:
 - ConvLSTM
 - ConvGRU
- Design choices:
 - Standard (top)
 - Dense-connection (middle)
 - Residual-connection (bottom)



Final RNN?

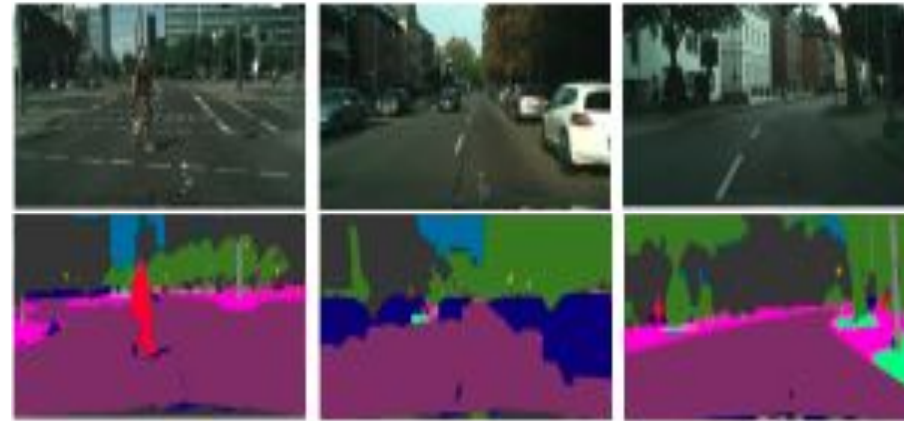
- Model temporal dependencies directly at the output
 - Can be seen as filtering
- Two needed components:
- Recurrent cell:** temporal modelling
- Conv. Layer:** aligns the number of channels and produces desired output
- This recurrent module is sometimes used, however it is not mandatory



Datasets

Cityscapes Dataset

- Dataset for autonomous driving related tasks
 - Object detection
 - Semantic segmentation
 - Depth estimation
- 5000 images of size (1024x2048)
 - 2975 training imgs
 - 500 validation imgs
 - 1525 test imgs (no ground truth)
- 30 classes, but we will only use 19
- Sequences of 30 frames, but only 20th is labelled



 Available in `home/nfs/inf6/data/datasets/cityscapes/leftImg8bit_sequence/`

Dataset is Small

Data Augmentation

- Standard to augment Cityscapes
 - Taking image crops
 - Resizing
 - Mirroring
 - ...
- Temporal augmentations
 - Skip frames
 - Interpolation
 - ...
- Try to get the most out of the few images that you have

Pretraining on MS-COCO

- Pretrain the segmentation-only model on a larger dataset, e.g., MS-COCO
- MS-COCO
 - Over 45K annotated images
 - 91 semantic categories
- This is a large dataset, and training here will take a long time (perhaps days)



Available in `/home/nfs/inf6/data/datasets/coco`

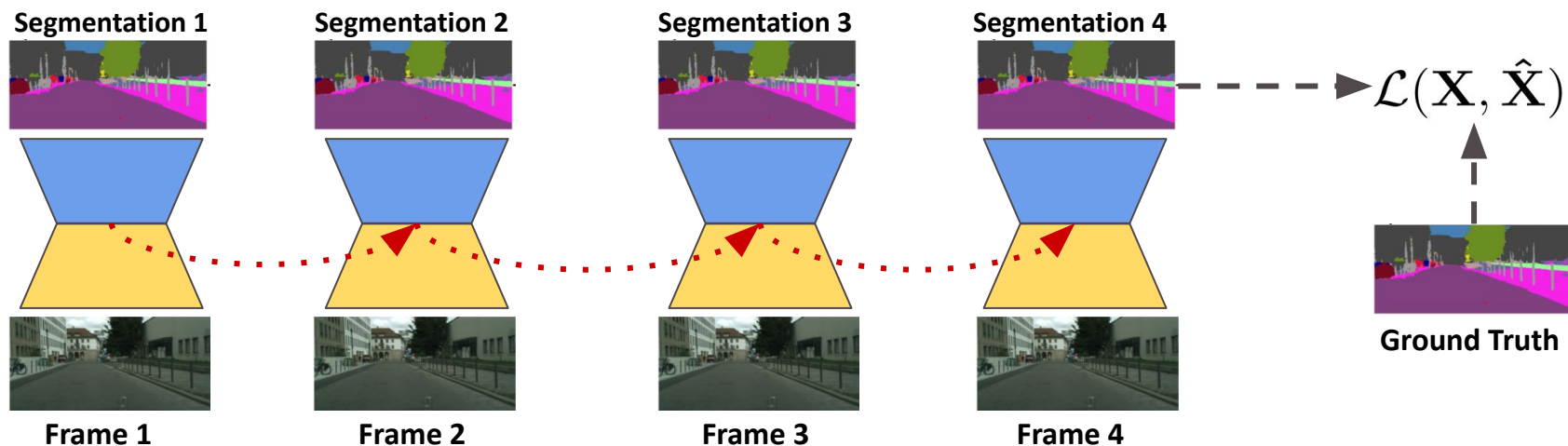
Training & Evaluation

Train/Eval on Cityscapes

- Training:
 - Image crops of size: (3, 512, 1024)
 - Sequences of 5 frames
- Evaluation:
 - Original image size: $\approx (3, 1024, 2048)$
 - Sequences of 5 frames and 12 frames
- *CrossEntropy* Loss function for training
- mAcc and mIoU quantitative evaluation metrics
- Make GIFs for qualitative evaluation

Training Protocol

- Only the 20th frame in every sequence of 30 is labelled
- We cannot enforce full supervision
- Compute the loss only on the annotated frame
- Temporal Regularization?



Project Goals and Deliverables

Passing Requirements

1. Implement the required model, pipelines and utils
2. Train your models to achieve best possible results on Cityscapes
 - You must implement and train the described model
 - Make changes and train further model variants to achieve better results
3. Beat the naive framewise baseline
 - **Baseline:** applying the image segmentation model frame-by-frame
4. Create overview notebook
5. Write project report

Deliverables

- Complete codebase
 - Clean and structured
 - Not just a notebook!
- Trained model checkpoint and (tensorboard, WandB, ...) logs
- Overview notebook (.ipynb & .html) showing main functionalities:
 - Load data and display some samples
 - Load pretrained model and display the structure or some stats
 - Display some qualitative results (e.g. results on 5 sequences)
 - Show the quantitative evaluation
- Project report

Grading

- Results and Experiments **55%-60%**:
 - Performing several experiments and obtaining good results
 - **Additional experiments**: ablation study, changes in the model, ...
 - This grade partly depends on how your results compare to the class
- Codebase & Overview Notebook **20-25%**:
 - Implement all functionalities
 - Modularity and structure
- Report **20%-25%**

Project Report

- Document your work in the project report
- Try to be brief, but readable and informative
- Include figures and tables
- Use *BibTex* for the references
- I expect 8-12 pages, but highly depends on number and size of imgs/tables
- Use the following template
 - <https://www.overleaf.com/read/tmnvhrsdmjrp>

Additional Experiment Ideas

- Try your own ideas!
- Training and pretraining:
 - Supervised or self-supervised pretraining
 - Use an encoder pretrained on ImageNet and perform transfer learning
- Tweak the model
 - Use a nice backbone (e.g. ResNet or ConvNext)
 - Adapt a different architecture (e.g. DeepLab v3+)
 - Investigate the type and positioning of the recurrent modules
- Investigate different training strategies:
 - Use different loss functions
 - Regularization to enforce temporal consistency
 - Use adversarial supervision
 - Advanced data augmentation (e.g. mix-up) and regularization (e.g. label smoothing)

Important Dates

- **03.02:** Starting date
- **10.03-21.03:** Revision session (flexible dates)
- **20.03:** Draft submission due
- **31.03:** Final submission:

Questions?



References

1. Cordts, Marius, et al. "The cityscapes dataset for semantic urban scene understanding." IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2016.
2. Ronneberger, Olaf, et al. "U-net: Convolutional networks for biomedical image segmentation." International Conference on Medical image computing and computer-assisted intervention (MICCAI). 2015.
3. Siam, Mennatullah, et al. "Convolutional gated recurrent networks for video segmentation." IEEE International Conference on Image Processing (ICIP). 2017.
4. Wang, Wei, et al. "Recurrent U-Net for resource-constrained segmentation." IEEE/CVF International Conference on Computer Vision (ICCV). 2019.
5. Pfeuffer, Andreas, Karina Schulz, and Klaus Dietmayer. "Semantic segmentation of video sequences with convolutional lstms." IEEE Intelligent Vehicles Symposium. 2019.
6. Lin, Tsung-Yi, et al. "Microsoft coco: Common objects in context." European Conference on Computer Vision. (ECCV), 2014.

