Efficient Convolutional Neural Networks Recognizing Driveable Trails

Adrián Matejov
Supervisor: Mgr. Pavel Petrovič, PhD.
Consultant: Mgr. Marek Šuppa

Faculty of Mathematics, Physics and Informatics
Comenius University in Bratislava

June 18, 2020

Smelý Zajko

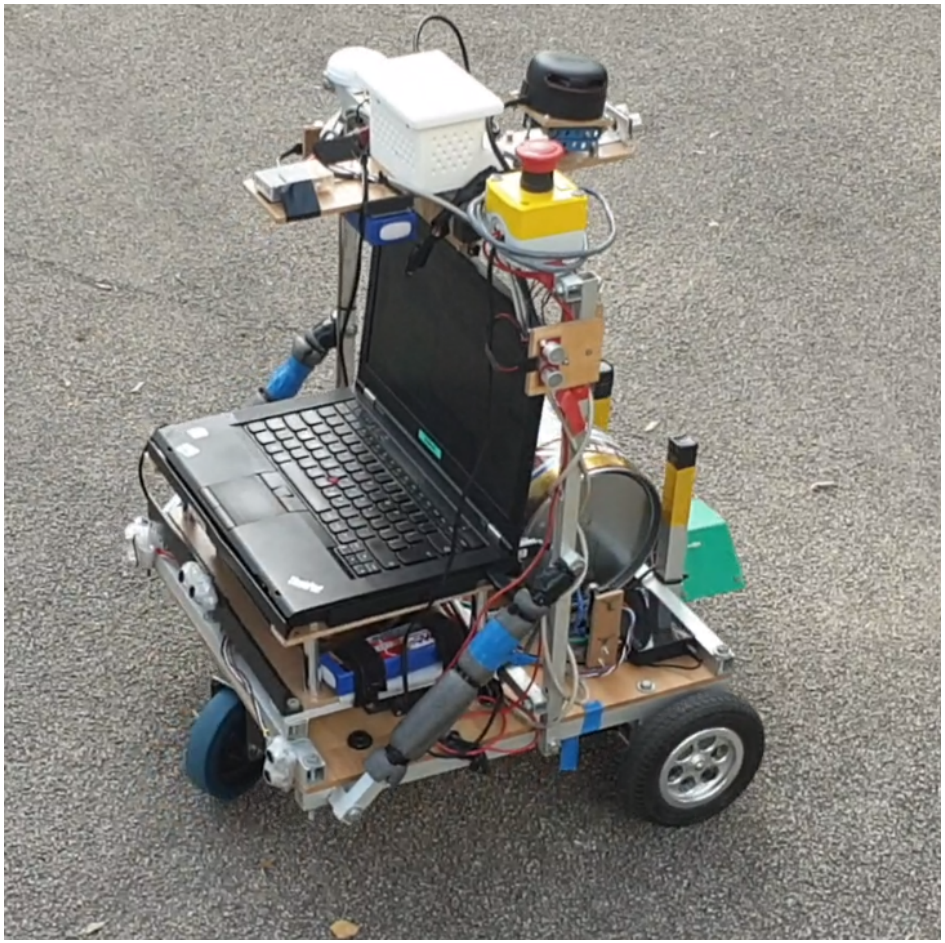


Figure: Robot Smelý Zajko at the competition in Deggendorf, heading to delivery point.

Robot's equipment

Sensors

- Hokuyo Laser - for detecting obstacles
- Rotational sensors - for adjusting robot's position in local map
- Ultrasonic sensors - for detecting obstacles behind the robot
- ZED Camera - constructs depth map and provides positional tracking
- **Android Phone** - captures images for predicting driveable path

Robot's equipment

Other

- Lenovo Thinkpad notebook - for combining all data and making decisions
- Arduino - for controlling motors
- 2x **NVIDIA Jetson TX2** - one for making predictions and the second one for working with ZED Camera

Objective

Figure: Prediction of driveable path - image resolution 640x480 on NVIDIA Jetson TX2.

Previous approaches

M. Nadhajský: RoboTour[1]

- Multi-layer perceptron
- Generates small **random** regions from input image
- Feature vector enriched by other handcrafted features

O. Jariabka, M. Šuppa and O. Rudolf: Single Camera Path Detection for Outdoor Navigation[2]

- Convolutional neural network
- Predicting only rectangles 5x5

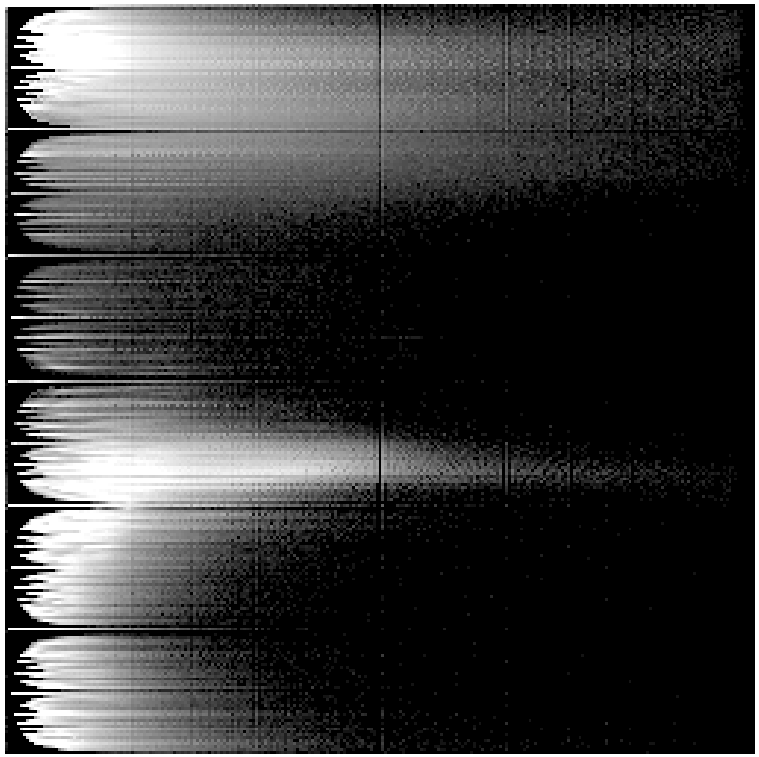
[1] M. Nadhajský. "Robotour". MA thesis. FMFI UK, 2011.

[2] O. Jariabka et al. "Single Camera Path Detection for Outdoor Navigation". *CESCG*. 2017.

Previous approaches

Statistical HSV model

- Precompute which pixels contain driveable segment (from dataset HSV images)
- Problems with lighting conditions as well
- Problems with pixels with similar color to the road



Convolutional neural network

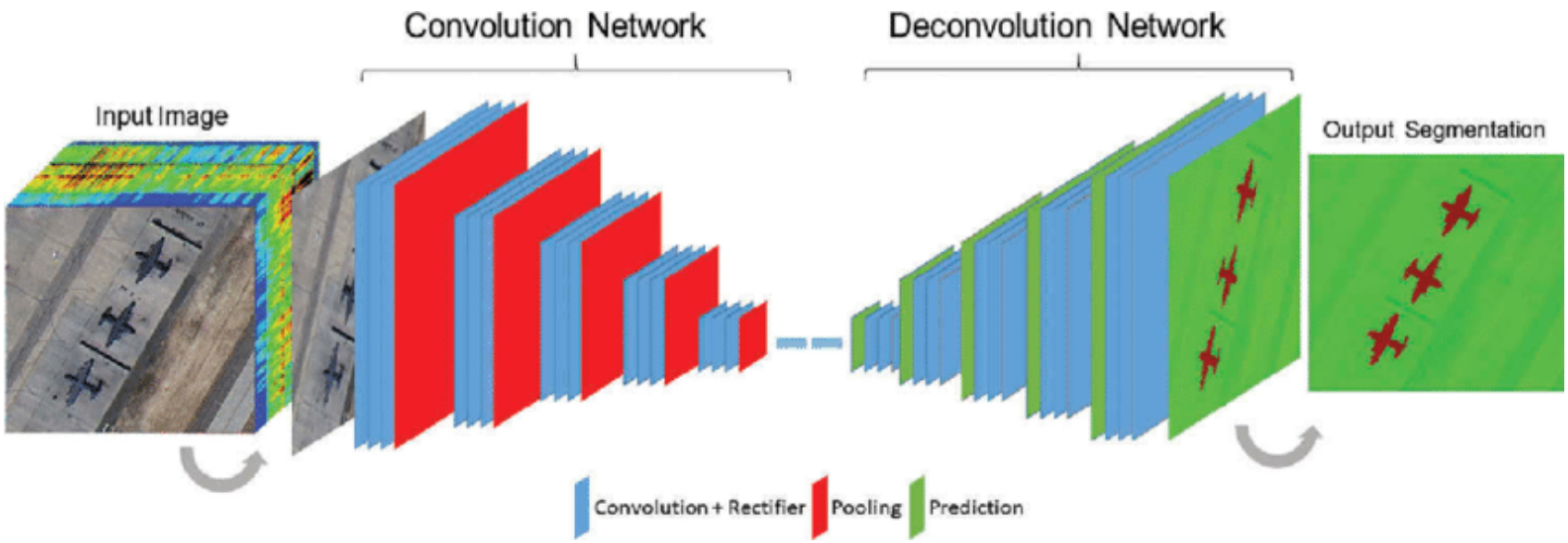


Figure: CNN

Lednice dataset

Figure: Images from the city park of Lednice (640x480).

Deggendorf dataset

Figure: Images from the city park of Deggendorf (640x480).

Metrics

- Binary Accuracy

$$\mathrm{ACC}(I) = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Intersection over Union (IoU)**

$$\mathrm{IoU}(I) = \frac{1}{|I|} \sum_{i=1}^{|I|} \frac{\text{area of overlap of i-th image}}{\text{area of union of i-th image}}$$

HSV model results

dataset	test accuracy	test IoU
Lednice	0.9037	0.8504
Deggendorf	0.8006	0.7403

Table: Results of HSV model measured on both datasets.

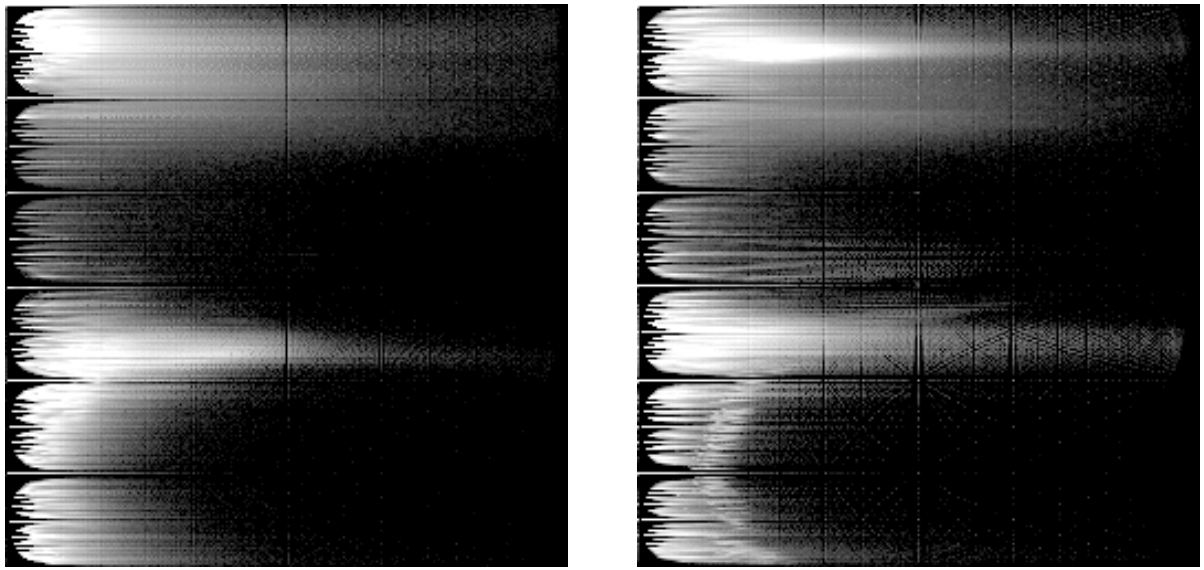


Figure: Left: Lednice, right: Deggendorf.

First attempts

Project for machine learning class

- MultiNet CNN segmentation module[3]
- Training on augmented data - slower training and almost no improvement in predictions
- **Unable to fit the model to Jetson TX2**
- Overall quite good accuracy due to pretrained encoder

[3] M. Teichmann et al. "MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving". 2016.

Models tested

- Unet[4]
- ResNet[5]
- SegNet[6]
- FCN VGG16 32S[7] (did not work)

[4] O. Ronneberger et al. "U-Net: Convolutional Networks for Biomedical Image Segmentation". 2015.

[5] K. He et al. "Deep Residual Learning for Image Recognition". 2016.

[6] V. Badrinarayanan et al. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". 2015.

[7] J. Long et al. "Fully Convolutional Networks for Semantic Segmentation". 2015.

Setup of the environment

- NVIDIA GeForce GTX 1080
- Python 3
- Keras + Tensorflow backend
- imgaug (for data augmentation - blur, transformations etc.)
- OpenCV

RoboTour 2019 - Training results

model	clr	loss	time L	time D	test IoU L	test IoU D
ResNet	rgb	dcl	81m	44m	0.9700	0.8878
SegNet	rgb	dcl	191m	122m	0.9484	0.8551
Unet	rgb	dcl	523m	499m	**0.9715**	0.9017
ResNet	rgb	bce	76m	42m	0.9711	0.8838
SegNet	rgb	bce	114m	57m	0.9643	0.8795
Unet	rgb	bce	434m	451m	0.9705	**0.9031**
ResNet	hsv	dcl	45m	61m	0.9660	**0.8868**
SegNet	hsv	dcl	144m	192m	0.9562	0.8788
ResNet	hsv	bce	54m	86m	**0.9725**	0.8861
SegNet	hsv	bce	52m	97m	0.9554	0.8748

Table: Results - Early Stopping. *L* denotes Lednice and *D* Deggendorf. *clr* - the colorspace of images, *time* - training time

Evaluating models on unseen dataset

model	clr	loss	test iou
ResNet	rgb	dcl	0.8475
SegNet	rgb	dcl	0.8502
Unet	rgb	dcl	0.8810
ResNet	rgb	bce	0.8278
SegNet	rgb	bce	0.8439
Unet	rgb	bce	**0.8812**
ResNet	hsv	dcl	0.8189
SegNet	hsv	dcl	0.8171
ResNet	hsv	bce	0.8110
SegNet	hsv	bce	0.8003

(a) Lednice models on Deggendorf dataset

model	clr	loss	test iou
ResNet	rgb	dcl	0.9289
SegNet	rgb	dcl	0.9149
Unet	rgb	dcl	0.9324
ResNet	rgb	bce	0.9271
SegNet	rgb	bce	0.9206
Unet	rgb	bce	0.9382
ResNet	hsv	dcl	**0.9300**
SegNet	hsv	dcl	0.9108
ResNet	hsv	bce	0.9242
SegNet	hsv	bce	0.9074

(b) Deggendorf models on Lednice dataset

RoboTour 2019 - dataset extension

Figure: Preview of images added to dataset at the competition because of small accuracy on images mostly covered by non-driveable segments.

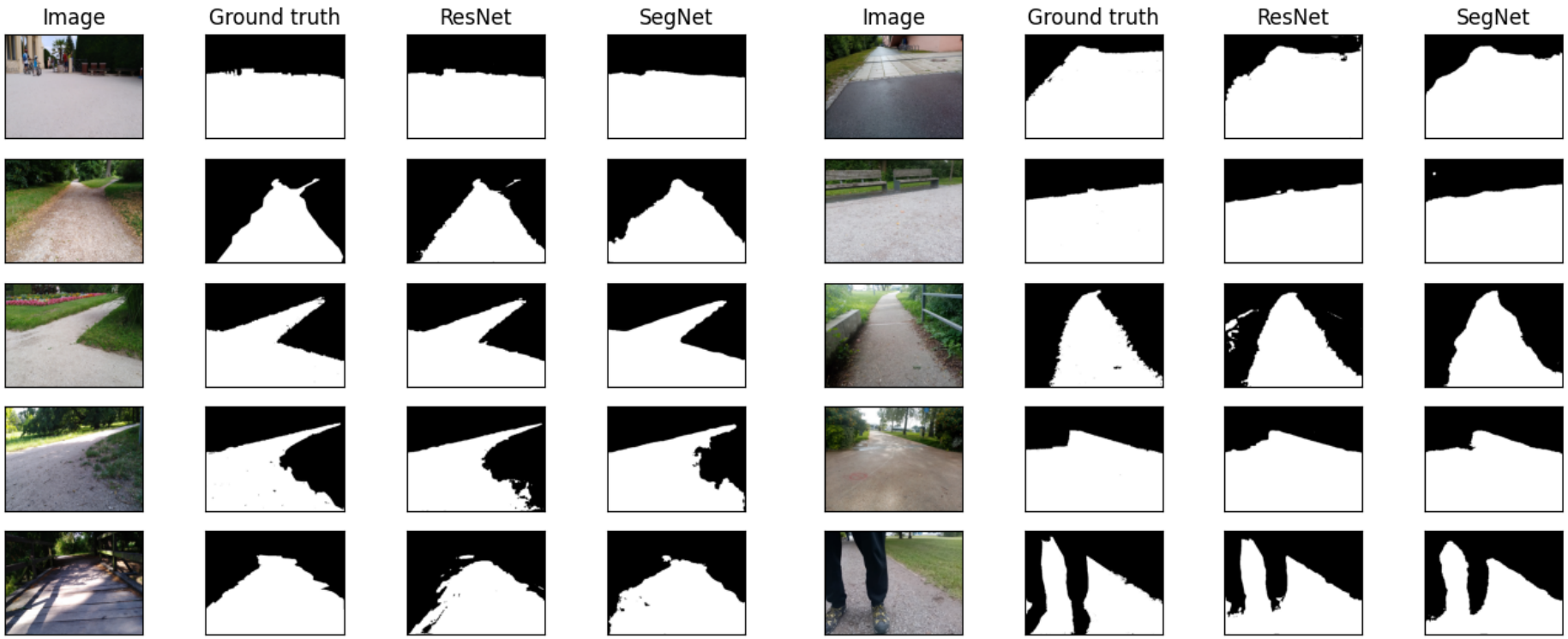
RoboTour 2019 - dataset extension - results

(a) Before

(b) After

Figure: Same model, slightly different datasets.

Predictions



(a) Lednice (b) Deggendorf

Figure: Comparison of test set predictions by ResNet and SegNet against ground truths.

Local Map

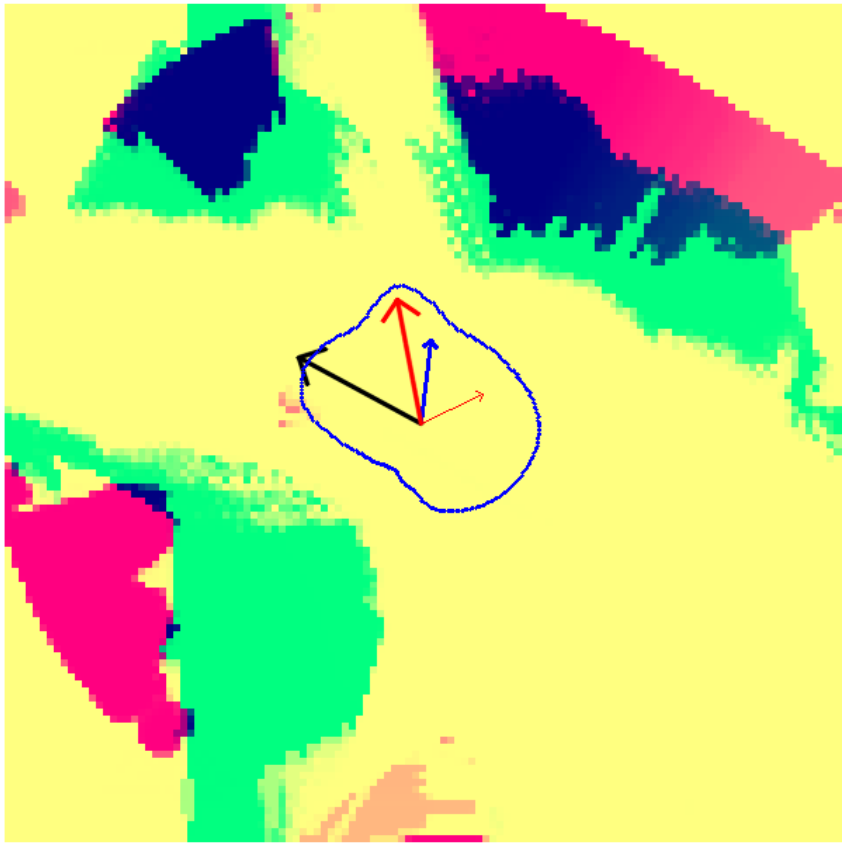


Figure: Predictions are incorporated into Local Map[8]. The robot is deciding which exit to use.

[8] M. Fikar. "Local map for a robot for the Robotour contest". MA thesis. FMFI UK, 2019.

RoboTour 2019 - Models

model	disk size	# params	**prediction time on Jetson**
ResNet	33 MB	2,753,729	0.24169 sec
SegNet	60 MB	7,818,117	0.36903 sec
Unet	356 MB	31,032,837	1.08655 sec

Table: Basic information about models

Model complexity reduction

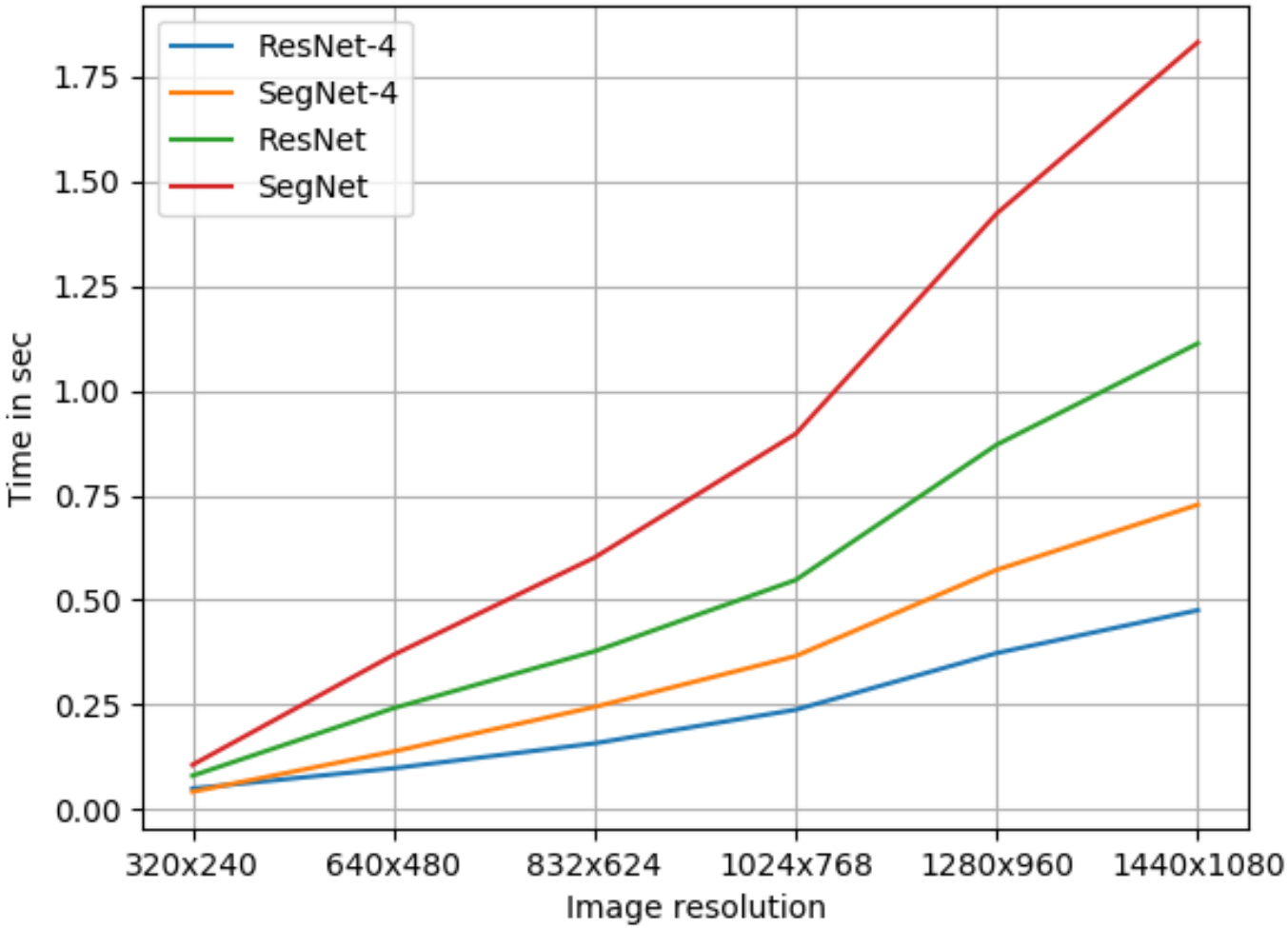
- SegNet - removal of several layers
- ResNet - removal of every second **identity block**
- reduced number of filters
- investigated usage of *dilated convolutions* (to improve accuracy)

Reduced models

model	# params	on Jetson	test acc	test IoU
ResNet-1	698,017	0.13288 sec	0.9408	0.8836
ResNet-2	179,297	0.10143 sec	0.9360	0.8796
ResNet-3	1,895,649	0.16817 sec	0.9446	**0.8919**
ResNet-4	480,817	**0.09807 sec**	0.9399	0.8850
SegNet-1	2,534,401	0.31107 sec	0.9325	0.8770
SegNet-2	1,075,009	0.22216 sec	0.9339	0.8741
SegNet-3	391,105	0.14389 sec	0.9413	**0.8833**
SegNet-4	206,145	**0.13818** sec	0.9380	0.8843

Table: Results of modified ResNet and SegNet training

Comparison of inference times



Mobile models

- ShuffleSeg[9] (ShuffleNet + SkipNet)
- ShuffleNetV2[10] (ShuffleNetV2 + DeepLabV3+)
- MobileNetV2[11] (MobileNetV2 + DeepLabV3+)
- MobileNetV3[12] (MobileNetV3 + DeepLabV3+)

[9] M. Gamal et al. "ShuffleSeg: Real-time Semantic Segmentation Network". (2018).

[10] S. Türkmen et al. "An efficient solution for semantic segmentation: ShuffleNet V2 with atrous separable convolutions". 2019.

[11] M. Sandler et al. "MobileNetV2: Inverted Residuals and Linear Bottlenecks". 2018.

[12] A. Howard et al. "Searching for MobileNetV3". 2019.

Results of mobile models

model	alpha	on Jetson	test acc	test IoU
ShuffleSeg	0.25	0.05559 sec	0.8862	0.8548
ShuffleNetV2	0.5	0.06318 sec	0.9375	0.8799
MobileNetV2	0.25	0.08698 sec	0.9396	0.8770
MobileNetV3-Large	0.25	0.07009 sec	0.9485	0.8908
MobileNetV3-Small	0.75	**0.05534** sec	0.9536	0.8977
ResNet-4	-	0.09807 sec	0.9399	0.8850

Table: Presenting only the best results. Tested on Deggendorf RGB dataset with binary crossentropy loss. alpha is the hyperparameter for adjusting the width of the network (*width multiplier*).

Active learning

- Labeling is an expensive and time-consuming process
- *Can we train the model on a smaller portion of data and reach comparable accuracy?*

Active learning

1. Train the model on a very small number of images
2. Sample another images for labeling based on some score function
3. Train the model on extended dataset
4. Repeat steps 2 and 3 until the stopping condition is met

Sampling methods - Entropy

1. Compute entropy for each pixel

$$H_i = -(p_i \log_2(p_i) + (1 - p_i) \log_2(1 - p_i))$$

where p_i is the probability that pixel belongs to driveable segment.

Sampling methods - Entropy

1. Compute entropy for each pixel

$$H_i = -(p_i \log_2(p_i) + (1 - p_i) \log_2(1 - p_i))$$

where p_i is the probability that pixel belongs to driveable segment.

2. Compute prediction entropy - aggregate over pixels

Sampling methods - Entropy

1. Compute entropy for each pixel

$$H_i = -(p_i \log_2(p_i) + (1 - p_i) \log_2(1 - p_i))$$

   where p_i is the probability that pixel belongs to driveable segment.
2. Compute prediction entropy - aggregate over pixels
3. Sample images with the **highest** prediction entropy (uncertainty)

Sampling methods - Diversity

1. Take encoded features from last encoder's layer

Sampling methods - Diversity

1. Take encoded features from last encoder's layer
2. Reduce the number of feature maps

Sampling methods - Diversity

1. Take encoded features from last encoder's layer
2. Reduce the number of feature maps
3. Run *K-means* algorithm on obtained feature vectors

Sampling methods - Diversity

1. Take encoded features from last encoder's layer
2. Reduce the number of feature maps
3. Run *K-means* algorithm on obtained feature vectors
4. Compute the entropy for each prediction

Sampling methods - Diversity

1. Take encoded features from last encoder's layer
2. Reduce the number of feature maps
3. Run *K-means* algorithm on obtained feature vectors
4. Compute the entropy for each prediction
5. Sample images with the highest entropy from **each cluster**

Active learning results

init: number of initial images, *pick*: number of sampled images, *reps*: epochs per round, *stop*: early stopping, *imgs*: number of images used

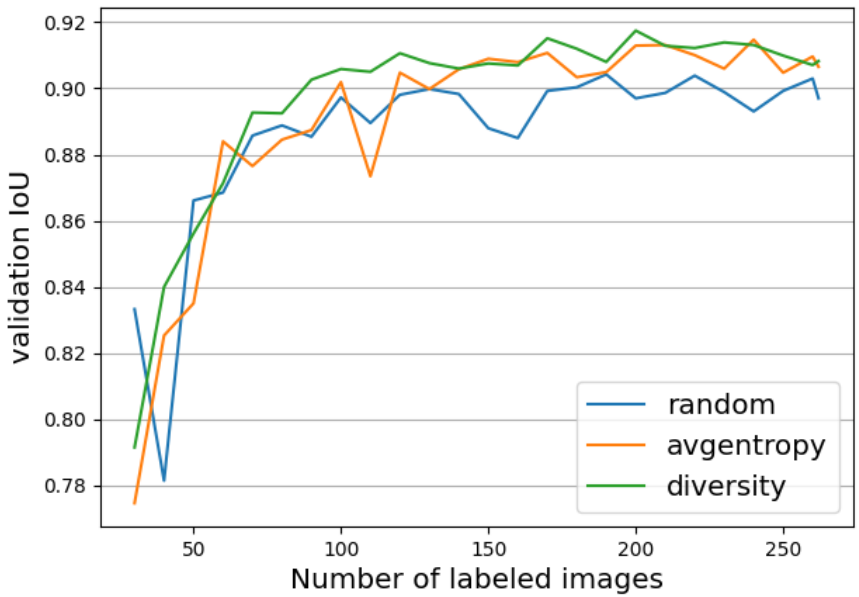
init	pick	reps	stop	imgs	test acc	test IoU
30	10	3	e-val 20	230	9320	**0.8755**
30	5	3	e-val 15	111	0.9248	0.8363
60	20	6	e-val 10	133	0.9305	0.8731

Table: Entropy sampling

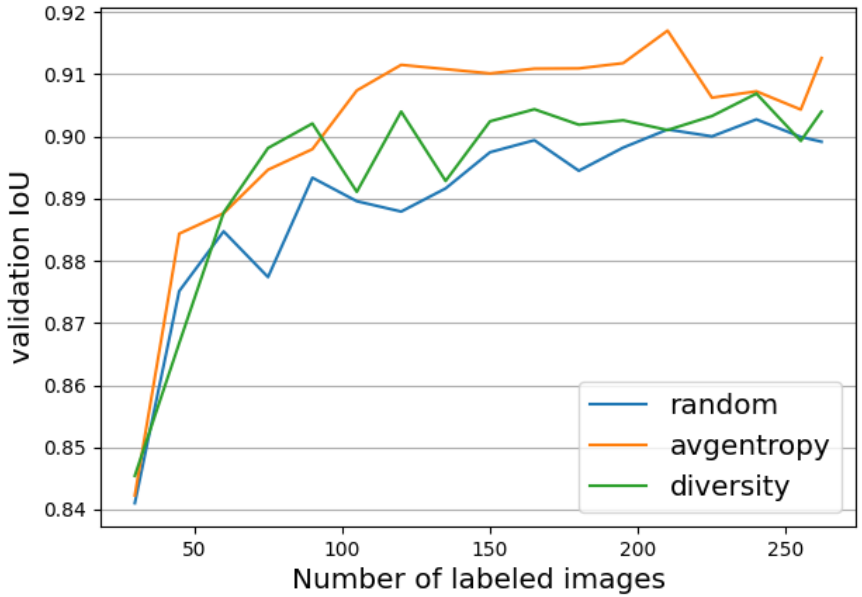
init	pick	reps	stop	imgs	test acc	test IoU
30	10	3	e-val 20	210	0.9320	0.8758
30	5	3	e-val 15	97	0.9228	0.8617
60	20	6	e-val 10	153	0.9306	**0.8770**

Table: Diversity sampling

Active learning results



(a) 10 samples and 6 epochs per round



(b) 20 samples and 6 epochs per round

Figure: Comparison of entropy, diversity and random sampling methods

Summary

1. Integrated **CNN** model to robot's vision module

Summary

1. Integrated **CNN** model to robot's vision module
2. Released **2 datasets** - Lednice (333 images) and Deggendorf (344 images)

Summary

1. Integrated **CNN** model to robot's vision module
2. Released **2 datasets** - Lednice (333 images) and Deggendorf (344 images)
3. Achieved **2nd** place at RoboTour 2019 in Deggendorf, Germany

Summary

1. Integrated **CNN** model to robot's vision module
2. Released **2 datasets** - Lednice (333 images) and Deggendorf (344 images)
3. Achieved **2nd** place at RoboTour 2019 in Deggendorf, Germany
4. Successfully reduced model complexities for **faster predictions**

Summary

1. Integrated **CNN** model to robot's vision module
2. Released **2 datasets** - Lednice (333 images) and Deggendorf (344 images)
3. Achieved **2nd** place at RoboTour 2019 in Deggendorf, Germany
4. Successfully reduced model complexities for **faster predictions**
5. Tested **Mobile** models resized with *alpha* hyper-parameter

Summary

1. Integrated **CNN** model to robot's vision module
2. Released **2 datasets** - Lednice (333 images) and Deggendorf (344 images)
3. Achieved **2nd** place at RoboTour 2019 in Deggendorf, Germany
4. Successfully reduced model complexities for **faster predictions**
5. Tested **Mobile** models resized with *alpha* hyper-parameter
6. Reduced the number of images needed for training using **Active learning** simulations

Future work

1. Try to train the models in a fully unsupervised way - first works proposed recently[13][14]
2. Incorporate depth map information from ZED mini camera into predictions

[13] M. Chen et al. "Unsupervised Object Segmentation by Redrawing". 2019.

[14] T. Nguyen et al. "DeepUSPS: Deep Robust Unsupervised Saliency Prediction via Self-supervision". 2019.

Thank you for your attention!

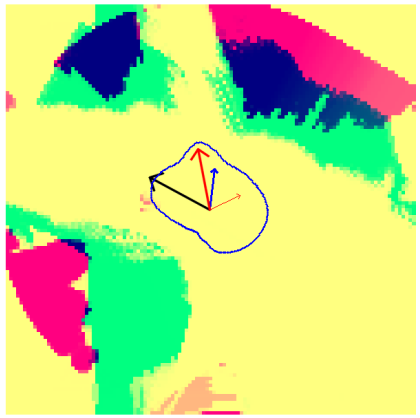
Otázky

"V aktívnom učení agregujete entropiu pomocou súčtu a priemeru. Nie sú tieto dve agregačné funkcie vzhľadom na použitý algoritmus ekvivalentné? Viete navrhnúť nejaké iné agregačné funkcie, ktoré by sa dali použiť?"

- Zaoberať sa iba entropiami nad určitým thresholdom
- Vytvoriť "superpixely" (regióny s najvyššou neistotou) na základe thresholdu a vyberať obrázky na základe veľkosti a početnosti týchto regiónov

Otázky

"Je správna klasifikácia všetkých pixelov rovnako dôležitá pre robota? Dala by sa dôležitosť pixelu zakomponovať do cieľovej funkcie, prípadne prepojiť segmentáciu priamo s tvorbou lokálnej mapy robota?"



- Lokálna mapa je "dočasná" pamäť robota
- Mapa je skonštruovaná z dát z rôznych senzorov
- 1 model, do ktorého vstupuje viacero dát a produkuje mapu/predikciu jazdy (nemáme dataset)

Otázky

"Vedeli by ste si predstaviť modifikáciu modelov tak, že budú obohatené o hĺbkovú informáciu zo zariadenia stereovidenia ZED Mini, ktoré je na robotovi inštalované?"

- Pridanie samostatného enkódera na hĺbkovú mapu a zakomponovanie tejto informácie do enkódera pre obrázok
- Dodatočná informácia pre predikciu prekážok - lepšie predpovedanie blížiacej sa prekážky (možnosť zbavenia sa laserového senzora?)