FEATURES OF DEEP LEARNING MODELS IN CONTEXT OF SEMANTIC SEGMENTATION TASK

Pęczek Paweł
Faculty of Computer Science, Electronics and Telecommunications
AGH University of Science and Technology
Cracow
peczek@student.agh.edu.pl

Droński Michał
Faculty of Computer Science, Electronics and Telecommunications
AGH University of Science and Technology
Cracow
dronski@student.agh.edu.pl

ABSTRACT

Deep Learning models have shown their expression power in a wide range of applications including Computer Vision, Natural Language Processing and graph processing. One of the topics that bring attention of researchers is image semantic segmentation which can be seen as a way towards visual scene comprehension. Capability to retrieve context of visual scenes is a key factor contributing success of various practical applications ranging from medical imaging to vehicle autonomy. There are, however, several difficulties that must be overcome to fully solve semantic segmentation challenge in general case, such as modelling diversity of objects scale and their semantic dependencies. In this work we aim to evaluate possibility of training stable semantic segmentation model on relatively small dataset finding a trafe-off between accuracy and computational efficiency.

Keywords Computer Vision · Semantic Segmentation · Deep Learning · Convolutional Neural Networks

1 Introduction

Empowered by modern GPU units, Deep Learning (DL) models are able to achieve outstanding performance on wide range of tasks focused on extensive and automatic information retrieval from data in various form. Since AlexNet [10] was released, the continuous growth of attention on Convolutional Neural Networks (CNNs) development can be observed. CNNs are successfully used to solve many classical Computer Vision (CV) tasks such as image classification. Over years CNNs have proved their tremendous expression power in modelling CV problems ranging from classification to much more sophisticated ones, like semantic segmentation. Despite emergence of other approaches towards modelling visual scenes (e.g. [27]), conovolutional bias seems still guarantee the best performance in complex CV challenges. In essence, CNN can be seen as an parameterized approximation f_θ^* of a function $f : \mathcal{I} \to \mathcal{T}$ that maps input $i \in \mathcal{I}$ into target $t \in \mathcal{T}$. Domain and range of function f differ w. r. t. specific task. In case of semantic segmentation $\mathcal{I}_{semseg} = \mathbb{R}^{H \times W \times C}$ and $\mathcal{T}_{semseg} = \mathbb{R}^{H \times W \times S}$ (H, W, C, S are images height, width, color chanels and number of semantic classes respectively). Element $t \in \mathcal{T}_{semseg}$ is a pixel-wise map of probability distribution over semantic classes describing which class a pixel belongs to. Parameters θ are determined while training procedure that is supposed to optimize objective function J formulated in a way specific for a target task.

Research effort on improving current state-of-the-art semantic segmentation methods is important due to possible applications of systems based on models or algorithms that solve the task. Such systems can be utilized to improve autonomous vehicles by ensuring more comprehensive surroundings understanding. What is more, in the field of medical imaging there is a chance to develop systems to assess wide range of scans without human supervision. This

application may have tangible effect on the quality of treatment and effectiveness of medical screening. Nevertheless, the general and robust method for solving semantic segmentation challenge is not yet known. The task complexity lays in wide range and high variety of visual scenes that may be segmentation input. Such scenes may contain very different objects both among different possible images and within a particular example. Objects scale variety is also potential challenge. Additionally, a truly robust semantic segmentation solution should also grasp the semantic dependency between objects inside image. For instance high building in urban scene should be recognized as *skyscraper*, whereas large building in industry context probably as *factory*. Unfortunately, CNNs (even particularly deep) cannot solve all mentioned issues without improvements in their architectures or training methods.

DO ZMIANY!!!
In this work we evaluated several features of CNNs that may be important in context of improving models solving semantic segmentation task. In particular, we tried to evaluate the nature of visual features learned by CNN while training w. r. t. semantic segmentation objective and compare them with ones reported to be learned by CNN trained w. r. t. classification objective. The goal of this evaluation is to verify the correctness of an approach to reuse models pre-trained on image classification as features encoders in semantic segmentation task. Apart from that, we evaluated the possibility of achieving non-supervised pre-training for semantic segmentation with use of convolutional Auto Encoder (AE), as well as we started discussion about the correctness of choice optimization method for semantic segmentation by comparing a model training convergence between one of the most popular optimizer (SGD with Momentum [30]) with a novel one called AdamW [29]. We have chosen CityScapes [7] data set as a baseline for our experiments due to the fact that it contains complex real scenes. To narrowed resources requirements of this work we decided to choose ICNet [22] as a baseline architecture.

The structure of this work is the following. Section 2 contain detailed description of state-of-the-art methods for semantic segmentation. In Section 3 we present the research methodology used to obtain results which are outlined in Section 5. Conclusions, general observations and proposals for further work are presented in Section 5.

2 Related work

Presented DL semantic segmentation models usually follow one of two major design approaches. One is to focus on best possible accuracy regardless of computational cost (see Section 2.1) and the other is to optimize models w. r. t. memory footprint and inference speed to enable on-line applications (see Section 2.2).

2.1 General methods

One of the first successful approaches towards semantic segmentation was proposed by Shelhamer et al. [9]. It uses standard CNN (like LeNet [8], or AlexNet [10]) as an encoder of image features, but instead of last layers that usually serves as dense classifier, it utilizes convolutional layers, turning the output from fixed-size and non-spatial matrix into classification map. This subtle change allows the FCN to take inputs of any size in contrast to classical CNNs. Shelhamer et al. in [9] proposed also the natural way of upsampling features map in a form of backwards strided convolutions so that the model output could be the same size as input. FCN established state-of-the art 62.2% mIoU accuracy on PASCAL VOC 2012 [11].

A natural weakness of previously discussed models is not taking advantage of global context information when solving semantic segmentation task. Zhao et al. proposed PSPNet [12] that employs pyramid pooling module in order to provide a way to make pixel predictions with regard to scene context. For datasets with a wide range of labels that make the segmentation task significantly complex, global contextual prior is extremally helpful for making accurate predictions. Examples of classes that may be misleading for neural network are described in details in [12]. Worth noting is the fact that in PSPNet [12] a CNN output forwarded as an input of pyramid pooling module. Its output after required upsampling is concatenated with original feature map. This kind of modules composition occurs once, after the final convolutional layer of backbone model. A natural extension of this approach should be a desire to inject similar context-aware blocks directly into the encoder network. Regardless of that, the PSPNet [12] achieved state-of-the art performance in 2017 on CityScapes [7] (80.2% mIoU) and PASCAL VOC 2012 [11] (85.4% mIoU). Worth noting also its accuracy (44.94% mIoU) on significantly hard ADE20K [13] dataset.

Another neural network model based shaped in a fully-convolutional fashion was proposed by Ronneberger et al. [14]. Its architecture is composed of two major parts, named by the authors contracting and expansive paths. The contracting path follows the typical CNN architecture and acts as image features encoder. The second path, on the contrary, serves as decoder and at each step, it consists of upsampling operation followed by convolution. Feature

channels number is doubled at each step of contracting path and halved on expansive path. Probability distributions over each pixel of the one before last activation map is produced by convolution with kernel size 1×1. The whole architecture is U-shaped and additionally employing skip connections between corresponding layers at both paths. Due to the significant difference in the field of applications that U-Net [14] was designed to, in the original paper there are no evaluation results on datasets like CityScapes [7] or ADE20K [13]. The value of evaluation metrics mainly discussed in this work is reported only for medical datasets PhC-U373 [15] (92.03% mIoU) and DIC-HeLa [16] (77.56% mIoU).

Slightly different approach was proposed by Guosheng et al. in [17]. Their main assumption was to provide the model multiple paths over which contextual information from different regions of network may be used by the following layers to achieve improvement in segmentation accuracy. In order to reach desired property, the authors proposed the RefineNet architecture. The backbone ResNet [1] is divided into 4 blocks with regards to the feature map sizes and serves as an input for RefineNet blocks. Their internal architecture provides a way to fuse feature maps in different resolutions coming from either backbone ResNet or another RefineNet block. RefineNet block as such provides a lot of flexibility in terms of network composition. The architecture described in [17] is therefore only one example of NN possible to be formulated as a part of architectures family that employs RefineNet blocks. Worth noting is the fact that Guosheng et al. in [17] put a lot of effort on ensuring the best possible gradient flow during backpropagation through the RefineNet blocks. Results reported on datasets relevant to this work: Cityscapes [7] – 73.6% mIoU, PASCAL VOC [11] – 83.4% mIoU, ADE20K [13] – 40.7% mIoU.

In-depth analysis of ResNet [1] architecture was done by Wu et al. in [18]. Based on research conducted, among others, by Szegedy et al. [19] and Veit et al. [20], Wu has shown interesting properties of residual blocks used as basic structure unit of ResNet [1]. One of the main field of interest in [18] was to investigate the way that residual networks learn and discover their properties dependent on chosen architecture design. The natural question that arises is about the appropriate trade-off between model width and depth. In the light of findings included in [18] the answer is following. Wider, but quite shallow residual network is able to outperform the deep one, in particular ones that are commonly in use - like ResNet-152 [1] or Inception-ResNet [19]. Such an architecture could be used both for image classification and (after some changes) semantic segmentation task. The accuracy achieved by the best model proposed in [18] is following. On validation set of CityScapes [7] – 77.86% mIoU. In case of Pascal VOC [11] – 80.84% mIoU on validation set. On validation set of ADE20K [13] – 43.73% mIoU.

Wang et al. in [21] proposed another approach towards architecture of semantic segmentation model. With DUC model they redefined the view on decoder part of a network. The output from standard image feature extractor (like ResNet [1]) is prepended by sophisticated functional blocks that are designed to alleviate the detrimental effect of commonly used operations on the model accuracy. Dilated convolutions are widely used as a simple form of enlarging the receptive field of convolutional kernels. Such an approach may enable the possibility to capture characteristics of larger objects without extending the depth of neural network. For tasks like semantic segmentation, this feature may significantly improve the model accuracy. There are, however, downsides of this kind of operations. One of them is called in [21] "gridding". The phenomenon concerns the basic mechanism that lays behind dilated convolutions. Stacking dilated convolution layers with the fixed dilation rate, it can be observed that information used to compute one particular value of the output feature map is limited only to small fraction of theoretical receptive field. What is more, the alignment of bottom feature maps' regions that influence on it are arranged in a checkerboard fashion. This itself may lead to a mismatch in terms of keeping the local information, as well as to lost of relevant information. As a theoretical solution to this problem Wang et al. proposed a block called *Hybrid Dilated Convolution (HDC)*. It consists of a stack of convolutional layers with different dilation rates chosen carefully such that the convolutional kernels fully cover the region of interest. Authors of [21] have also raised a problem of feature map upsampling in order to obtain predictions in FCN fashion. As in many concurrent approaches [12, 19, 22] in they discuss the inability of simple bilinear upsampling to adopt to improve the quality of feature maps being processed. However, instead of standard deconvolution operation, Wang et al. proposed *Dense Upsampling Convolution (DUC)*. DUC applies only convolutional operations to get dense prediction map. In order to achieve desired result DUC block outputs the feature map of shape $(H/d) \times (W/d) \times (d^2 + L)$ where H and W are height and width of output prediction respectively, L is the number of classes to predict and d is chosen so that the output map consists of d^2 subparts. The output of DUC is reshaped later on to achieve desired size of prediction map. Obtained results clearly show the advantage of DUC both over bilinear upsampling and deconvolutions in terms of ability to restore information from low-resolution feature maps. Results reported on particular datasets are following: CityScapes [7] – 80.1% mIoU, Pascal VOC 2012 [11] – 83.1% mIoU.

2.2 Real-Time methods

Zhao et al. in [22] proposed architecture called ICNet based on the PSPNet [12] that reaches 30fps on Titan X GPU in high resolution (images from CityScape [7] sized 1024×2048). All of this was achieved preserving trade-off between

prediction accuracy and inference speed. Considering available time budget the model architecture relies on a cascade of processing pipelines, each operating on input with different resolution and effective feature fusion in *Cascade Fusion Units (CFU)* both with *Cascade Label Guidance (CLG)*. CFU is a sophisticated method for fusion the feature map that comes from different pipelines in such a way that allows to improve coarse, low-resolution prediction and build better, upsampled one. Feature map coming from a lower-resolution stream is upsampled by bilinear interpolation. To improve the quality of upsampling, the dilated convolution is applied directly at the top of it. The result is fused with the second feature map projected by 1×1 convolution operation. Both feature maps are added up after batch normalization layer being put on top each of them. CLG, on the other hand, is a form of loss function formulation in such manner that encourages intermediate low-resolution feature map to be as close as correspondingly resized ground-truth as possible. Results reported in [22] are following: CityScapes [7] – 70.6% mIoU, CamVid [23] – 67.1% mIoU.

U-shape network architecture used as design pattern when creating various semantic segmentation models (like [22]) has few drawbacks. The most important ones are following: a) requirements to operate on high-resolution input feature maps at the initial stages of feed-forward pass and b) weak capacity to recover information from high-resolution branches due to use of low number of convolutional kernels (tight layers). Yu et al. in [24] proposed a different approach based on two major part: Spatial Path (SP) and Context Path (CP). SP handles spatial information preservation whereas CP (Xception-based [25] architecture) serve as context features extractor. Yu et al. proposed also a new fusion method to improve model performance. In order to ensure large effective receptive field, CP is based on standard, lightweight model (commonly used to solve classification task) followed by Global Average Pooling layer (GAP). What is more, due to refining features of particular CP stages, the authors proposed to use attention-based modules for guiding valuable features learning. Attention-based approach is also used in Feature Fusion Module which merges the results from SP and CP. Thanks to its design, BiSeNet achieves 74.8% mIoU on CityScapes [7] (validation set) and 68.7% mIoU on CamVid [23]. Results reported above in terms of CityScapes [7] evaluation are correct under assumption of full-size (1024x2048) input and real-time regime (NVIDIA Titan XP - 65.5 fps). Without inference time constraints results are better and reaches 80.3% mIoU on CityScapes [7] (validation set).

Transfer learning is a common practice in wide range of Machine Learning application. Oršić et al. in [26] proposed an architecture utilizing ImageNet pre-trained features to achieve better performance on target task. Pretrained model serves as features encoder, whereas upsampling decoder is jointly trained from scratch on semantic segmentation task. At the end of encoder (in order to increase receptive field under real-time inference regime), simplified version of Spatial Pyramid Pooling block (inspired by one proposed in [12]) is attached. Decoder is composed of blocks with two inputs - one for previous encoder block output and second for corresponding encoder output (transformed by 1x1 conv). Thanks to ImageNet pretraining, the model achieves 75.5% mIoU on CityScapes [7] (validation set) and 39.9 fps inference speed (GTX 1080Ti) on full-size (1024x2048) input. What is more interesting - the same architecture without pretraining achieves 70.4% mIoU on CityScapes (validation set), which is less than [22]. When training without real-time constraint, 76.5% mIoU on the same dataset was achieved.

3 Method

3.1 Training strategy

DO OPISANIA SPOSÓB PODZIAŁU NA FOLDY, włącznie z konkretnymi wartościami kątów - trzeba wspomnieć, że dla klas, dla których nie dysponowaliśmy konkretnymi anotacjami ustaliliśmy podział zgrubnie. Trzeba też wspomnieć że wiemy, że zaburza to train-test data distribution, ale zrobiliśmy tak dla sprawdzenia jak "roboust" są modele, jak dobrze generalizują na nieznane przypadki i w ogólności dla zbadania jak wrażliwe są modele segmentacji trenowane na małej ilości danych na takie delikatne shifty perspektywy. Każdy model był trenowany 32 epoki. Opis klas z data setu. Każdy model był trenowany na tym samym podziale danych w danej turze. Opis datasetu - ile przykładów, jakie klasy, SZTUCZNE TŁA - fajnie by było wrzucić kilka obrazków. ICNET i UNET dostosowane do potrzeb tego datasetu - podać liczbę parametrów.

3.2 Evaluation strategy

In binary segmentation setup, Dice coefficient serves as segmentation efficiency metrics in a very natural manner. Assuming objects of interests presence on segmented scene, Dice can be used as evaluation metrics without any adjustments. There are, however, several difficulties that must be overcome when it comes to multi-class generalization of Dice coefficient in context of semantic segmentation model evaluation. One of such issue can be lack of presence of a particular class representative on a particular image under evaluation. Model that predicts perfectly fine result yield *NaN* value of Dice score. The issue can be easily solved by setting Dice score as 1.0 in such cases. On the other hand,

if a model output contains negligibly small amount of positive values (taking into account the same setup), the Dice score is 0.0 which seems not to be a fair outcome. Allowing small *noise level* ϵ representing the fraction of output to be FP for a particular class if this class does not occur in *ground-truth* can efficiently solve this issue. Yet another issue that may occur is the following. Assuming that a particular image does only contain one class instance(s) and the model is evaluated w. r. t. N segmentation classes, in case that a model performs badly on this image and produces only FN values, then the Dice score value for this class is 0.0. The level on how meaningful this value is in the overall score (among all classes) depends on the evaluation strategy. If the chosen approach is to average the score among all classes, then with growth of N, the influence of value 0.0 decreases (to the level of being almost impossible to notice). To alleviate such cases, final score averaging should be done only w. r. t. classes actually present in the picture and those one that were predicted by a model (possibly with small noise-level acceptance strategy described above).

In the light of potential issues discussion conducted above we would like to outline the evaluation strategy applied on results of our experiments. First of all, we decided to train two models - ICNet [22] and U-Net [14] with and without naive and unconstrained Dice loss coefficient. The loss term always present is cross-entropy. Putting explicit and unconstrained Dice coefficient was intended to validate the differences in training outcomes. Final scores presented in Section 4 are calculated both per-class and w. r. t. all classes in data set (with additional *background* class). We provided two set of results - one averaged on *pixel-wise* level and *example-wise* one. *Example-wise* averaging means using binary weights $w_b^{i,c}$ for all examples i and classes c when calculating Dice score. Value 1 was assigned to an example i in context of a particular class c if a class was present either in *ground-truth* or *prediction* (in case of prediction - above noise level ϵ). *Pixel-wise* averaging means using as weights $w_p^{i,c}$ values calculated as $TP + FP + FN$ (the only ones that Dice takies into account) for a particular example i and class c (noise tolerance was also applied on prediction). Noise tolerance was 24 pixels per image (as $\epsilon = 0.0015$ and images size was 128×128). Background class was treated as equally important as other ones.

As a result, Dice score for a particular class c was calculated as:

- $D_c^{pw} = \frac{\sum_i w_p^{i,c} d_i}{\sum_i w_p^{i,c}}$ in case of *pixel-wise* averaging w. r. t. a particular class c,

- $D_c^{ew} = \frac{\sum_i w_b^{i,c} d_i}{\sum_i w_b^{i,c}}$ in case of *example-wise* averaging w. r. t. a particular class c,

- $D^{pw} = \frac{\sum_c \sum_i w_p^{i,c} d_i}{\sum_c \sum_i w_p^{i,c}}$ in case of *pixel-wise* averaging w. r. t. all classes,

- $D^{ew} = \frac{\sum_c \sum_i w_b^{i,c} d_i}{\sum_c \sum_i w_b^{i,c}}$ in case of *pixel-wise* averaging w. r. t. all classes.

where d_i is Dice score constrained w. r. t. noise tolerance ϵ.

For clarity of results presentation we utilized the following abbreviations for class names: *background - bg*, *box - bx*, *clamp - cl*, *drill - dr*, *duck - du*, *adapter - ad* and *bottle - bo*.

4 Results

Experiments results determined in regime described in Section 3.2 are presented in Tables $1 - 4$. In contrast to majority model performance reports, highlighted values are the worst among detected ones. Majority of worst results (regarding Tables $1 - 4$ columns) in a number of 88% are connected with *camera rotation-based* splits of data set. Such phenomenon was supposed to occur due to mismatch in train-test data distribution, but its scale can be seen as a measurement of how weak the models are when it comes to data generalization. The *camera rotation-based* splits were chosen in such a way to test models on unseen angles that are only slightly different from ones that were seen by model while training. Interestingly, the lowest score values (in terms of absolute magnitude) can be observed in case of rot_1. This particular split was created in a way that training set contains extreme angles values and test set - the ones from the middle of available range, so the results are rather counter-intuitive. There are cases where absolute difference in score magnitude between *random-* and *rotation-based* splits are above 0.35.

In this work we do not aim to compare ICNet [22] and U-Net [14] architectures head-to-head for several different reasons. First of all, according to results reported in Tables $1 - 4$, VISAPP [31] data set seems not to be sufficiently complex to draw meaningful conclusions regarding particular model features (such conclusions could be drawn rather from meta-analysis than a single experiment). Second of all, those two models differ by large in terms of parameters number. Our version of ICNet [22] has less than 1M parameters, whereas our version of U-Net [14] - more than 20M.
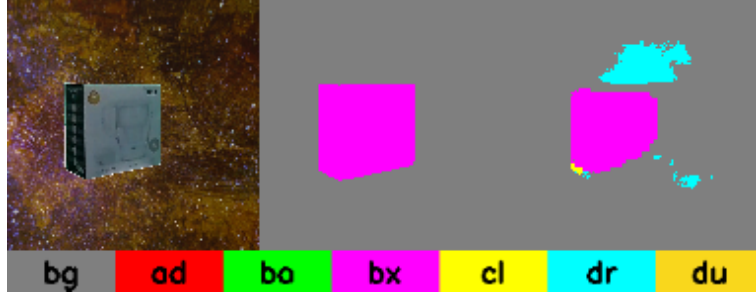
Figure 1: Example of bad prediction from ICNet [22] model. In relatively easy case, model predicted three different classes to be present. Picture alignment is the following: from left to right - original example, ground truth and prediction.

This factor contributes significantly not only in computational cost of model inference, but also in model stability. Despite the fact that particularly weak predictions can be observed in case of both models, model with less number of parameters tends to produce less stable output (part of strange model predictions can be influenced by augmentation, but augmentation was needed due to small data set size).

Worth mentioning is the fact that particular classes can be easier to learn that other ones. An example of such class can be *duck*. Objects that belong to this class have characteristic color and shape that is compact and relatively easily to describe. There are, however, classes that are quite challenging for both models, like *box*, *clamp*. The reason those classes are problematic is quite different. In case of *box* class, objects are relatively simple in terms of shape (the shape is even less sophisticated than in terms of *duck*). The reason why models cannot predict segmentation map for *box* instances is probably the objects color in correspondence to backgrounds. The assumption is originated in a fact that even for *human expert* it is harder to visually separate object from background if the object is similar in terms of color to the background. Regardless of the true reason for this phenomenon, its presence indicates how data set complexity influence results (similar conclusions can be drawn from evaluating results achieved by *state-of-the-art* models on CityScapes [7] in contrast to ones reported foe ADE20K [13] - see Section 2). On the other hand, *clamp* class contain objects which are the most complex in terms of shape among all classes in VISAPP [31] data set, what seems to explain the magnitude of difficulties in training phase for a model to learn how to produce stable predictions in case of this particular class. Additionally, in some cases (like U-Net [14] trained only with CE loss - see Table 3) *clamp* class tends to be predicted instead of proper classes that should be assigned by a model. Cases like that are source of low D_c^{ew} scores. Issues discussed in this paragraph are illustrated on Figure 2.

Split name	D_{bg}^{pw}	D_{bx}^{pw}	D_{cl}^{pw}	D_{dr}^{pw}	D_{du}^{pw}	D_{ad}^{pw}	D_{bo}^{pw}	D^{pw}	D_{bg}^{ew}	D_{bx}^{ew}	D_{cl}^{ew}	D_{dr}^{ew}	D_{du}^{ew}	D_{ad}^{ew}	D_{bo}^{ew}	D^{ew}
$random_0$	0.99	0.84	0.86	0.87	0.96	0.82	0.88	0.97	0.99	0.49	0.54	0.79	0.77	0.81	0.88	0.83
$random_1$	0.99	0.8	**0.73**	0.83	0.96	0.92	0.9	0.97	0.99	0.59	0.48	0.49	0.8	0.71	0.69	0.76
$random_2$	0.99	0.83	0.82	0.83	0.97	0.92	0.85	0.97	0.99	0.4	0.48	0.56	0.97	0.72	0.85	0.77
$random_3$	0.99	0.84	0.86	0.88	0.97	0.77	0.87	0.97	0.99	0.4	0.53	0.69	0.97	0.76	0.83	0.81
$random_4$	0.99	0.79	0.78	0.86	0.97	0.91	0.84	0.97	0.99	0.65	**0.36**	0.83	0.88	0.78	0.4	0.74
$random_5$	0.99	0.86	0.82	0.81	0.95	0.88	0.88	0.97	0.99	0.48	0.52	0.62	**0.54**	0.85	0.57	0.76
$random_6$	0.99	0.85	0.77	0.79	0.95	0.9	0.87	0.97	0.99	0.38	0.37	0.65	0.96	0.87	0.8	0.77
$random_7$	0.99	0.7	0.75	0.83	0.96	0.93	0.9	0.97	0.99	0.57	0.37	0.56	0.88	0.93	0.68	0.77
rot_0	**0.98**	0.46	0.74	0.74	**0.89**	0.84	0.85	**0.95**	**0.98**	0.39	0.37	0.59	0.62	0.84	0.44	**0.69**
rot_1	**0.98**	**0.42**	0.78	0.72	0.96	0.82	0.79	0.96	**0.98**	**0.13**	0.6	0.51	0.88	**0.53**	0.62	0.72
rot_2	**0.98**	0.9	0.82	**0.67**	0.94	**0.59**	**0.72**	**0.95**	**0.98**	0.9	0.65	**0.43**	0.58	0.58	**0.36**	**0.69**

Table 1: Results of adjusted ICNet [22] model (CE loss) evaluation w. r. t. evaluation strategy described in Section 3.2. Table rows are indexed by *split name*, where $random_i$ refers to randomly generated split of data set (with randomly adjusted backgrounds) and rot_i refers to data set split prepared w. r. t. angle that a particular image was taken regarding to object. Values marked as **bold** in particular columns highlight the weakest score in both for *pixel-wise* and *example-wise* score averaging.

Figure 2: Predictions from U-Net [14] trained only with CE loss term (see results in Table 3). For each example (from top to bottom), original image, ground truth and model prediction (from left to right) are presented. First case shows relatively good model prediction on not challenging class (beak of a duck is not predicted correctly, probably due to similarity to background region) even if background contain other objects. Second case shows totally mismatched prediction for a problematic class. It also illustrates the situation when other classes are predicted incorrectly which lowers D^{ew}_c scores significantly. Third case illustrates the model issues with predicting proper segmentation map for some areas of *clamp* instance, where object shape is complex and background have similar color to foreground.

Split name	D^{pw}_{bg}	D^{pw}_{bx}	D^{pw}_{cl}	D^{pw}_{dr}	D^{pw}_{du}	D^{pw}_{ad}	D^{pw}_{bo}	D^{pw}	D^{ew}_{bg}	D^{ew}_{bx}	D^{ew}_{cl}	D^{ew}_{dr}	D^{ew}_{du}	D^{ew}_{ad}	D^{ew}_{bo}	D^{ew}
$random_0$	0.99	0.87	0.75	0.82	0.95	0.9	0.87	0.97	0.99	0.8	0.32	0.74	0.74	0.82	0.59	0.76
$random_1$	0.99	0.76	0.8	0.84	0.97	0.88	0.78	0.97	0.99	0.41	0.46	0.69	0.97	0.71	0.4	0.74
$random_2$	0.99	0.89	0.85	0.85	0.98	0.89	0.9	0.98	0.99	0.64	0.48	0.7	0.88	0.86	0.76	0.83
$random_3$	0.99	0.85	0.85	0.86	0.97	0.8	0.89	0.98	0.99	0.84	0.48	0.66	0.77	0.64	0.7	0.79
$random_4$	0.99	0.9	0.88	0.87	0.97	0.89	0.92	0.98	0.99	0.66	0.62	0.53	0.97	0.89	0.84	0.83
$random_5$	0.99	0.91	0.9	0.83	0.97	0.81	0.8	0.98	0.99	0.63	0.79	0.58	0.97	0.8	**0.39**	0.78
$random_6$	0.99	0.87	0.78	0.81	0.97	0.9	0.84	0.97	0.99	0.54	0.45	0.59	0.97	0.82	0.44	0.76
$random_7$	0.99	0.67	0.68	0.8	0.96	0.94	0.87	0.97	0.99	0.38	0.39	0.63	0.92	0.88	0.42	0.74
rot_0	**0.98**	0.43	0.75	0.74	0.97	**0.78**	0.83	**0.95**	**0.98**	0.4	0.39	0.52	0.95	0.78	0.44	0.71
rot_1	**0.98**	**0.2**	0.72	0.65	0.95	0.87	**0.77**	**0.95**	**0.98**	**0.09**	0.46	**0.4**	0.77	**0.58**	0.71	0.7
rot_2	**0.98**	0.8	**0.66**	**0.63**	**0.82**	0.87	0.85	0.96	**0.98**	0.74	**0.27**	0.44	**0.5**	0.79	0.5	**0.69**

Table 2: Results of adjusted ICNet [22] model (CE + Dice loss) evaluation w. r. t. evaluation strategy described in Section 3.2. Table rows are indexed by *split name*, where $random_i$ refers to randomly generated split of data set (with randomly adjusted backgrounds) and rot_i refers to data set split prepared w. r. t. angle that a particular image was taken regarding to object. Values marked as **bold** in particular columns highlight the weakest score in both for *pixel-wise* and *example-wise* score averaging.

Split name	D_{bg}^{pw}	D_{bx}^{pw}	D_{cl}^{pw}	D_{dr}^{pw}	D_{du}^{pw}	D_{ad}^{pw}	D_{bo}^{pw}	D^{pw}	D_{bg}^{ew}	D_{bx}^{ew}	D_{cl}^{ew}	D_{dr}^{ew}	D_{du}^{ew}	D_{ad}^{ew}	D_{bo}^{ew}	D^{ew}
$random_0$	0.99	0.59	0.65	0.68	0.95	0.89	0.82	0.96	0.99	0.5	0.36	0.36	0.91	0.75	0.44	0.67
$random_1$	0.99	0.68	0.74	0.71	0.95	0.8	0.79	0.96	0.99	0.5	0.38	0.4	0.54	0.54	0.64	0.66
$random_2$	0.98	0.48	0.68	0.66	0.96	0.88	0.74	0.95	0.98	0.13	0.32	0.34	0.96	0.59	0.31	0.54
$random_3$	0.98	0.6	0.65	0.72	0.92	0.82	0.75	0.96	0.98	0.23	0.29	0.5	0.44	0.48	0.38	0.58
$random_4$	0.99	0.79	0.78	0.77	0.95	0.91	0.8	0.97	0.99	0.33	0.32	0.51	0.76	0.67	0.32	0.63
$random_5$	0.99	0.72	0.74	0.72	0.95	0.93	0.84	0.97	0.99	0.47	0.31	0.42	0.95	0.78	0.45	0.68
$random_6$	0.99	0.55	0.6	0.68	0.95	0.9	0.82	0.96	0.99	0.54	0.29	0.53	0.82	0.66	0.55	0.69
$random_7$	0.98	0.55	0.7	0.63	0.87	0.84	0.71	0.95	0.98	0.32	0.34	0.38	0.45	0.51	0.3	0.58
rot_0	**0.97**	0.35	0.56	**0.5**	0.93	0.88	0.75	**0.93**	**0.97**	0.33	0.31	**0.22**	0.83	0.73	0.47	0.62
rot_1	0.98	**0.12**	0.66	0.63	0.93	**0.68**	**0.68**	0.94	0.98	**0.07**	0.49	0.28	0.82	**0.38**	0.55	0.63
rot_2	0.98	0.52	**0.49**	0.64	**0.74**	0.81	0.7	0.94	0.98	0.18	**0.18**	0.33	**0.38**	0.52	**0.27**	**0.49**

Table 3: Results of adjusted U-Net [14] model (CE loss) evaluation w. r. t. evaluation strategy described in Section 3.2. Table rows are indexed by *split name*, where $random_i$ refers to randomly generated split of data set (with randomly adjusted backgrounds) and rot_i refers to data set split prepared w. r. t. angle that a particular image was taken regarding to object. Values marked as **bold** in particular columns highlight the weakest score in both for *pixel-wise* and *example-wise* score averaging.

Split name	D_{bg}^{pw}	D_{bx}^{pw}	D_{cl}^{pw}	D_{dr}^{pw}	D_{du}^{pw}	D_{ad}^{pw}	D_{bo}^{pw}	D^{pw}	D_{bg}^{ew}	D_{bx}^{ew}	D_{cl}^{ew}	D_{dr}^{ew}	D_{du}^{ew}	D_{ad}^{ew}	D_{bo}^{ew}	D^{ew}
$random_0$	0.99	0.72	0.64	0.76	0.96	0.85	0.87	0.97	0.99	0.48	0.38	0.42	0.96	0.85	0.61	0.73
$random_1$	0.99	0.79	**0.58**	0.68	0.96	0.88	0.88	0.97	0.99	0.32	0.37	0.47	0.71	0.76	0.67	0.71
$random_2$	0.99	0.82	0.71	0.79	0.95	0.86	0.89	0.97	0.99	0.35	0.37	0.46	0.61	0.73	0.57	0.68
$random_3$	0.99	0.7	0.71	0.85	0.96	0.86	0.82	0.97	0.99	0.17	0.24	0.65	0.92	0.51	0.46	0.58
$random_4$	0.99	0.75	0.59	0.62	0.93	0.9	0.86	0.96	0.99	0.55	0.38	0.41	0.44	0.66	0.48	0.66
$random_5$	0.99	0.69	0.64	0.8	0.95	**0.73**	0.72	0.96	0.99	0.23	**0.28**	0.58	0.9	0.69	**0.32**	0.62
$random_6$	0.99	0.49	0.59	0.77	0.95	0.81	0.88	0.96	0.99	0.2	0.31	0.48	0.82	**0.46**	0.74	0.65
$random_7$	0.99	0.46	0.65	0.75	0.96	0.89	0.8	0.96	0.99	0.15	**0.28**	0.5	0.76	0.64	0.38	**0.61**
rot_0	**0.98**	0.66	0.71	0.69	0.95	0.78	0.82	**0.95**	**0.98**	0.47	**0.37**	0.38	0.67	0.65	0.41	0.65
rot_1	**0.98**	**0.32**	0.72	0.77	0.94	0.86	**0.69**	**0.95**	**0.98**	**0.08**	0.46	0.37	0.81	0.62	0.63	0.66
rot_2	0.99	0.77	0.78	**0.61**	**0.91**	0.79	0.83	0.96	0.99	0.29	0.54	0.52	**0.53**	0.58	0.38	0.64

Table 4: Results of adjusted U-Net [14] model (CE + Dice loss) evaluation w. r. t. evaluation strategy described in Section 3.2. Table rows are indexed by *split name*, where $random_i$ refers to randomly generated split of data set (with randomly adjusted backgrounds) and rot_i refers to data set split prepared w. r. t. angle that a particular image was taken regarding to object. Values marked as **bold** in particular columns highlight the weakest score in both for *pixel-wise* and *example-wise* score averaging.

5 Conclusions

On the basis of results presented in Section 4 the following conclusions can be drawn:

- There is possibility to train a semantic segmentation model with low number of parameters that achieves results that can be seen as better than random noise. At the same time, the quality of such models must be evaluated w. r. t. proper performance metrics and wide range of data sets.

- Semantic Segmentation is challenging and not yet solved Computer Vision task. The complexity lays in variety of potential visual scenes and lack of large and open Semantic Segmentation data sets (due to effort that must be put in order to collect annotations at good quality).

- Small data sets may not be a proper evaluation benchmarks for Deep Learning models trained to solve Semantic Segmentation task.

- The choice of proper evaluation metrics is crucial to validate Semantic Segmentation model quality, in particular, in multi-class segmentation task.

- Lack of original backgrounds in VISAPP [31] data set influence results significantly. Training a model with artificially created backgrounds are not a perfect set up when the goal is to deeply investigate model quality, as backgrounds choice induces artificial bias in data.

References

[1] He K., Zhang X., Ren S., Sun J.: Deep Residual Learning for Image Recognition. *arXiv:1512.03385*, 2015

[2] Szegedy C., Liu W., Jia Y., Sermanet P., Reed S., Anguelov D., Erhan D., Vanhoucke V., Rabinovich A.: Going Deeper with Convolutions. *arXiv:1409.4842*, 2014

[3] Chen T., Xu B., Zhang C., Guestrin C.: Training Deep Nets with Sublinear Memory Cost. *arXiv:1604.06174*, 2016

[4] Li Y., Kaiser Ł., Bengio S., Si S.: Area Attention. *arXiv:1810.10126*, 2019

[5] Chen Y., Kalantidis Y., Li J., Yan S., Feng J.: A2-Nets: Double Attention Networks. *arXiv:1810.11579*, 2018

[6] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A., Kaiser Ł., Polosukhin I.: Attention Is All You Need. *arXiv:1706.03762*, 2017

[7] Cordts M., Omran M., Ramos S., Rehfeld T., Enzweiler M., Benenson R., Franke U., Roth S., Schiele B.: The Cityscapes Dataset for Semantic Urban Scene Understanding. in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016

[8] LeCun Y., Boser B., Denker J., Henderson D., Howard R., Hubbard W., Jackel L.: Gradient-Based Learning Applied to Document Recognition. in *Proc. of the IEEE, number 11*, 2010

[9] Shelhamer E., Long J., Darrell T.: Fully Convolutional Networks for semantic segmentation. *arXiv:1605.06211*, 2016

[10] Krizhevsky A., Sutskever I., Hinton G.: Imagenet classification with deep convolutional neural networks. in *NIPS*, 2012

[11] Everingham M., Gool L., Williams C., Winn J., Zisserman A.: The Pascal Visual Object Classes (VOC) Challenge in *International Journal of Computer Vision, vol. 88*, 2010

[12] Zhao H., Shi J., Qi X., Wang X., Jia J.: Pyramid Scene Parsing Network *arXiv:1612.01105*, 2016

[13] Zhou B., Zhao H., Puig X., Fidler B., Barriuso A., Torralba A.: Scene Parsing through ADE20K Dataset. in *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[14] Ronneberger O., Fischer P., Brox T.: U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv:1505.04597*, 2015

[15] Data set provided by Dr. Sanjay Kumar. Department of Bioengineering University of California at Berkeley. *PhC-U373*, Berkeley CA (USA)

[16] Data set provided by Dr. Gert van Cappellen Erasmus Medical Center. *DIC-HeLa*, Rotterdam (Netherlands)

[17] Lin G., Milan A., Shen C., Reid I.: RefineNet: Multi-Path Refinement Networks for High-Resolution semantic segmentation. *arXiv:1611.06612*, 2016

[18] Wu Z., Shen C., van den Hengel A.: Wider or Deeper: Revisiting the ResNet Model for Visual Recognition. *arXiv:1611.10080*, 2016

[19] Szegedy C., Ioffe S., Vanhoucke V., Alemi A.: Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *arXiv:1602.07261*, 2016

[20] Veit A., Wilber M., Belongie S.: Residual Networks Behave Like Ensembles of Relatively Shallow Networks. *arXiv:1605.06431*, 2016

[21] Wang P., Chen P., Yuan Y., Liu D., Huang Z., Hou X., Cottrell G.: Understanding Convolution for semantic segmentation. *arXiv:1702.08502*, 2017

[22] Zhao H., Qi X., Shen X., Shi J., Jia J.: ICNet for Real-Time semantic segmentation on High-Resolution Images. *arXiv:1704.08545*, 2017

[23] Brostow G., Fauqueur J., Cipoll R.: Semantic object classes in video: A high-definition ground truth database. in *Pattern Recognition Letters*, 2009

[24] Yu C., Wang J., Peng C., Gao C., Yu G., Sang N.: BiSeNet: Bilateral Segmentation Network for Real-time semantic segmentation *arXiv:1808.00897*, 2018

[25] Chollet F.: Xception: Deep Learning with Depthwise Separable Convolutions *arXiv:1610.02357*, 2016

[26] Oršić M., Krešo I., Bevandić P., Šegvić S.: In Defense of Pre-trained ImageNet Architectures for Real-time semantic segmentation of Road-driving Images *arXiv:1903.08469*, 2019

[27] Kosiorek A., Sabour S., Teh Y., Hinton G.: Stacked Capsule Autoencoders *arXiv:1906.06818*, 2019

[28] Schroff F., Kalenichenko D., Philbin J.: FaceNet: A Unified Embedding for Face Recognition and Clustering *arXiv:1503.03832*, 2015

[29] Loshchilov I., Hutter F.: Decoupled Weight Decay Regularization *arXiv:1711.05101*, 2017

[30] Qian N.: On the momentum term in gradient descent learning algorithms. in *Neural networks: the official journal of the International Neural Network Society*, 12(1):145–151, 1999

[31] VISAPP data set, `http://home.agh.edu.pl/˜majcher/src/visapp/`