university of groningen / faculty of science and engineering

Using Intersection over Union loss to improve Binary Image Segmentation

Bachelor's Project Thesis

Floris van Beers, s2197367, f.van.beers@student.rug.nl
Supervisor: Dr M.A. Wiering

Abstract: In semantic segmentation tasks the Jaccard Index, or Intersection over Union (IoU), is often used as a measure of success. While this measure is more representative than per-pixel accuracy, state-of-the-art neural networks are still trained on accuracy by using Binary Cross Entropy Loss. In this research, an alternative is used where a neural network will be trained for a segmentation task on face detection by optimizing directly on an approximation of IoU. When using this approximation, IoU becomes differentiable and can be used as a loss function. The comparison between IoU loss and Binary Cross Entropy loss will be made by testing multiple models on multiple datasets and data splits. After testing it is found that training directly on IoU significantly increases performance for some models compared to training using the conventional Binary Cross Entropy loss.

1 Introduction

In the field of image segmentation through deep neural networks increasingly complex systems are created to improve on the semantic segmentation task. All these systems compete in complexity and state-of-the-art performance. A commonality between early works such as Shelhamer, Long, and Darrell (2017), more recent works such as SegNet (Badrinarayanan, Kendall, and Cipolla (2017)) or comparative studies (Siam, Gamal, Abdel-Razek, Yogamani, and Jägersand (2018)) is that they use Intersection over Union (IoU), also known as the Jaccard Index, as a measure of success. IoU is much more indicative of success for segmentation tasks, compared to pixel-wise accuracy, especially when the input data is significantly sparse. When labels used for training consist of 80-90% background and only a small percentage of positive labels, a naive measure such as accuracy can score up to 80-90% by labelling everything as background. Because IoU does not concern itself with true negatives, this naive solution will never occur, even with highly sparse data. While this research is focused on binary segmentation, the naive solution problem amplifies itself when applied to segmentation over multiple classes as the ratio of background to a specific class becomes worse with more classes.

Working under the assumption that the IoU as a measure of success is helpful, this research attempts to use a loss function based on IoU, proposed by Rahman and Wang (2016), to train a segmentation model directly. While research has been done extensively on which loss functions are best for which task (Janocha and Czarnecki (2017), Zhao, Gallo, Frosio, and Kautz (2017)), the default used for image segmentation is still usually Cross Entropy (Shelhamer et al. (2017), Badrinarayanan et al. (2017), Noh, Hong, and Han (2015)). Cross Entropy is a loss function that is, mathematically, much more closely related to accuracy than IoU, but which could be used as an approach to a good IoU. By defining a loss function more closely related to IoU the training process could be improved. As such the question that needs to be answered is as follows: Can a model trained on an IoU loss function perform better than a model trained on Binary Cross Entropy (BCE) loss? Since binary segmentation is considered, the comparison will be between a loss function based on IoU, proposed by Rahman and Wang (2016) and detailed in section 2, and BCE. In the research by Rahman and Wang (2016) on the same topic a model is made to train on the separate classes of several semantic seg-

mentation datasets. Significant improvements are found in Rahman and Wang (2016) by applying the new loss function on a single model. Optimization of the IoU has also been used in other work on semantic segmentation as shown in work on the Conditional Random Field by Ahmed, Tarlow, and Batra (2015) and work on probabilistic models by Nowozin (2014). In research done by Yuan, Chao, and Lo (2017) a different loss function is proposed, which is also based on IoU. This implementation strays further from IoU by applying a quadratic component. As such, in this work, the loss function proposed by Rahman and Wang (2016) is used as an implementation of IoU loss.

This research extends the work by Rahman and Wang (2016), and evaluates the performance differences between loss functions on multiple models. In addition, to focus solely on the difference in performance based on the loss functions, the models will use the same structure and parameters. A base model has been created that functions when changing the loss function, keeping all other parameters constant. To further emphasize the difference in performance based purely on the use of a new loss function, the dataset has been chosen to accommodate this. A sufficiently dense dataset has been chosen to avoid skewing the results in favor of IoU loss, which theoretically performs better with a sparser dataset. In the research by Rahman and Wang (2016) the data is much sparser due to the use of separate classes in multiclass segmentation datasets, such as PASCAL VOC2011 (Everingham, Van Gool, Williams, Winn, and Zisserman (2011)).

In section 2 the models, dataset, and the loss functions will be elaborated upon. Some mathematical adaptation of IoU will be used in order to make the calculation differentiable. In section 3 the experimental setup will be detailed, both in terms of hyperparameters and dataset splits. In section 4 the outcome of the experiments will be presented numerically. These results will be put into context in section 5 and evaluated for statistical relevance. Furthermore, this section shows some example output. Final conclusions and suggestions for future work will be conveyed in section 6.

2 Methods

In order to provide a consistent experimental environment for the loss function comparison, some established models and datasets are used.

2.1 Models

In this research, two different encoders are extended into fully convolutional networks (FCNs) for their use as a semantic segmentation model. These encoders are the convolutional parts of VGG-16 (Simonyan and Zisserman (2014)) and ResNet-50 (He, Zhang, Ren, and Sun (2015)).

We have trained our own custom neural network systems using pretrained weights from the earlier mentioned neural network systems. These were previously trained on the ImageNet dataset (Deng, Dong, Socher, Li, Li, and Fei-Fei (2009)). The training of the custom systems employs face images from two separate face datasets used as input to the proposed methods. All experiments were carried out with the aid of Keras deep learning framework (Chollet et al. (2015)), because it contains rich libraries for Computer Vision tasks

2.1.1 VGG-16/BFCN-32s

The VGG-16 network (Simonyan and Zisserman (2014)) is a well-established deep neural network used for classification. It is a convolutional network with 16 convolutional layers after which there are several fully connected layers. Finally, a softmax layer determines the classification outcome. It has been trained extensively on the ImageNet dataset (Deng et al. (2009)). Previously this work has been extended to a fully convolutional network by Shelhamer et al. (2017). For this research the same extension from VGG-16 to FCN-32s was used as described by Shelhamer et al. (2017). A notable difference is the number of output classes. Where the original FCN-32s is modelled to segment 21 classes, i.e. the 20 classes of the VOC-2011 dataset (Everingham et al. (2011)) and background, the adaptation used here considers only 2 classes. These classes are face and background.

The main steps in adapting VGG-16 to the segmentation task remain the same as described by Shelhamer et al. (2017). The fully connected layers of VGG-16, which are 4096 neurons each, are

replaced with fully connected convolutional layers. The first of these convolutional layers has 4096 feature maps, a kernel size of 7×7 and a stride of 1. This means that they are essentially fully connected feature maps. The second fully connected layer is replaced by a similar layer, but with a kernel size of 1×1. These layers use a rectified linear unit activation. Finally, the fully connected layer that is the size of the label space is replaced with a fully convolutional layer with the same purpose. This layer has 21 feature maps, one for each class, in the original creation of FCN-32s (Shelhamer et al. (2017)) and has 1 feature map in our adaptation, named BFCN-32s (Binary FCN-32s). This final layer goes from feature space to label space and uses a linear activation to achieve the reduction of feature maps. To make the model into a fully convolutional model, the resulting 7×7 feature maps have been upsampled by a trainable deconvolutional layer. This uses a stride of 32 to regain the original image size and counteracts the size decreases performed by the max-pooling layers done in each convolutional block of the decoder. Since the number of classes is reduced to 2, a softmax layer as used in Shelhamer et al. (2017) is no longer necessary. Instead, the upsampling layer uses a sigmoid activation so the output pixels are between 0 and 1. These steps create a model where the input image is the same size as the output image, allowing the model to be trained pixel-wise end-to-end.

2.1.2 ResNet/FCResNet

Another well-performing classification network is ResNet-50 (He et al. (2015)). This model uses residual learning in a 50-layer DNN to classify images. It has been trained on the ImageNet dataset (Deng et al. (2009)) as well. The same steps as described in section 2.1.1 were taken to make a fully convolutional version from ResNet, named FCResNet in this research. Because ResNet does not have fully-connected layers, but only a softmax layer the size of the label space, a single convolutional layer was used to replace this and perform the same conversion from feature space to label space as mentioned in section 2.1.1. This layer reduces the feature maps from 2048 to 1 by using linear activations. Finally, a similar upsampling layer was added to obtain an output image the same size as the input image, resulting in a pixel-wise end-to-end trainable version of Resnet. This upsampling layer uses a sigmoid activation, as explained in section 2.1.1

2.2 Datasets

To explore the differences in performance from these models two pixel-wise labelled datasets have been used. These are the Labelled Faces in the Wild: Part Labels dataset (LFW) (Kae, Sohn, Lee, and Learned-Miller (2013)) and the HELEN dataset (Le, Brandt, Lin, Bourdev, and Huang (2012)). Both sets have been chosen for their relatively dense appearance of positive labels. The reason for this is that the data needs to be evenly distributed instead of sparse, which might skew the results in favor of an IoU approach, as explained in section 1.

2.2.1 LFW

Labelled Faces in the Wild: Part Labels (Kae et al. (2013)) is a dataset containing 2927 images of faces, an example of which is shown in figure 2.1. These images are labelled in two ways, but for this research, only the pixel-wise labelling is considered, as seen in figure 2.2. This labelling is in three classes, namely face, hair and background. Since the task being considered is regarding binary segmentation, some preprocessing had to be performed. For different experimental settings, we consider the hair to be either part of the face, as shown in figure 2.3 or not, as shown in figure 2.4. This results in a binary distinction where pixels are either part of the face or background. The data was split by using 292 images (10%) for testing. The amount of training and validation images are dependent on the experimental setting as detailed in section 3.

2.2.2 HELEN

The HELEN dataset (Le et al. (2012)) is also a dataset of faces, containing 2330 images. It is labelled in multiple classes for separate parts of the face, such as mouth and hair. As with LFW this has to be preprocessed in such a way that it can be used for binary image segmentation. As such we construct new labels out of the provided labels that either label all parts of the face as face and the rest as background, or label the hair as background as well. This results in the same data structure as

Figure 2.1: Example Input Image LFW

Figure 2.2: Example 3-class Label LFW

with the LFW dataset. With this dataset 233 images (10%) are used for testing. The amount of images for training and validation are dependent on the experimental setup.

2.3 Loss Functions

In assessing the effectiveness of an IoU loss function, a baseline has to be established. Binary Cross Entropy (BCE) loss, or log loss, is such a baseline in that it is used by default in a wide range of recent classification and segmentation works, namely in Shelhamer et al. (2017), Badrinarayanan et al. (2017), Siam et al. (2018), Noh et al. (2015). The comparative study by Janocha and Czarnecki (2017) also concludes that BCE loss is used too frequently as a loss function without considering alternatives.
The formula for binary cross entropy loss can be seen in equation 2.1. In this equation T refers to the true label image, T_x refers to a single element of that label, P refers to the prediction of the output image and P_x to a single element of that prediction.

$$L_{BCE} = \sum_{x} -(T_x \log(P_x) + (1 - T_x) \log(1 - P_x)) \tag{2.1}$$

In this equation, it can be seen that BCE, while incorporating an element of probability, smoothed out by the log component, still awards both true positives and true negatives, while penalizing false positives and false negatives. Referring back to the problem described in section 1, this can lead to simplistic solutions to segmentation when the data is significantly sparse, by labelling all output as background.
The loss function we use is one that directly incorporates the value for Intersection over Union. This loss function is proposed by Rahman and Wang (2016), and this previous work describes the mathematical aspects, which will be elaborated upon here as well. The original equation for IoU can be given as:

$$IoU = \frac{|T \cap P|}{|T \cup P|} \tag{2.2}$$

As before, in equation 2.2, T stands for the true label image, P for the prediction of the output image and the symbols are taken from set theory. This IoU is then taken as the average over the entire set to be considered producing an IoU value between 0 and 1. These set symbols are, however, not differentiable. To apply these set symbols in their true form the numbers in T and P need to be absolute 1's and 0's. However, while the label T contains these values, the output P contains values between 1 and 0 due to the sigmoid activation in the final upsampling layer of the network. To solve this an approximation of IoU can be made using probabilities. This gives the equation for this approximation IoU':

$$IoU' = \frac{|T * P|}{|T + P - (T * P)|} = \frac{I}{U} \tag{2.3}$$

Figure 2.3: Example LFW label with hair

Figure 2.4: Example LFW label without hair

Here T and P remain the same, but $T * P$ is the element-wise multiplication of T and P. In the numerator, this gives an approximation of Intersection by giving the probability of P_x when T_x is 1 and giving 0 otherwise. As such, the intersection is highest when P_x is 1 wherever T_x is 1, exactly as is to be expected. The denominator is an addition of T and P with a deduction of the Intersection, just as in a regular calculation for the union, to mitigate the effect of counting the intersection area twice.
Now that the equation for IoU has been reduced from using set operations to IoU', which uses arithmetic operations, it is also differentiable. As a loss function, the error needs to approach 0 when results become better. To achieve this the loss function is defined in terms of IoU' as such:

$$L_{IoU} = 1 - IoU' \tag{2.4}$$

This loss L_{IoU} is applied to each element in a batch and added producing a value between 0 and batchsize, when IoU for each sample approaches 1 or 0, respectively. This is a loss function that can, again, be minimized. To achieve this it needs to be differentiated, which is done in the following way:

$$\frac{\partial F}{\partial P_x} = \frac{-U * \frac{\partial I}{\partial P_x} + I * \frac{\delta U}{\delta P_x}}{U^2} \tag{2.5}$$

$$\frac{-U * T_x + I * (1 - T_x)}{U^2} \tag{2.6}$$

These derivatives and the backpropagation that follows are computed by the Keras framework during training when the equation used to calculate IoU is replaced with the differentiable approximation.

3 Experiments

Training the models using the datasets and loss functions described in section 2 is done according to certain design choices. These choices are mostly hyperparameters of the models, but also pertain to choices in the dataset usage. Table 3.1 shows the different splits of the datasets such that 8 distinct experimental setups are created. Applying these 8 data splits to each combination of BFCN32-s and FCResNet, either with BCE or IoU loss results in 32 distinct setups, the results of which are presented in section 4.
To ensure no other factors would influence the outcome of these experiments, any other hyperparameters have been kept constant. The relevant hyperparameters can be seen in table 3.2. While most parameters are found through a parameter sweep based on the results from previous work such as Shelhamer et al. (2017), the patience parameter is less self-explanatory. It determines the number of epochs without improvements after which the training stops. This is our point of convergence. As such this value is more important for total learning time than the number of epochs, the maximum of which was never reached.
Using these data splits, models and parameters, each of the models was trained until convergence as

Dataset	# Training	# validation	Hair?
LFW	2342	292	yes
LFW	1000	100	yes
LFW	2342	292	no
LFW	1000	100	no
HELEN	1864	233	yes
HELEN	1000	100	yes
HELEN	1864	233	no
HELEN	1000	100	no

Table 3.1: Different data uses: Number of images used for training and validation is given by # training and # validation respectively. The Hair? column determines whether hair is included in the face (yes) or in the background (no).

Parameter	Value
Epochs	1000
Batch-size	100
Patience	20
Learning Rate	0.0001
Optimizer	RMSprop

Table 3.2: Experimental parameters

determined by the patience parameter. After training the model was tested on previously unseen data of the dataset it was trained on. For both datasets this was the last 10%, i.e. 292 images for LFW and 233 images for HELEN, as mentioned in section 2.2.

4 Results

The results of the experiments described in section 3 can be seen in tables 4.1, 4.2, 4.3 and 4.4. These tables are each structured similarly. Each table displays the results of a different combination of model and dataset and presents each of the 4 data splits within that dataset. Results are evaluated on three metrics: original binary accuracy, Intersection over Union and epochs before convergence. For each metric, the results from the models trained on IoU loss are in the F_{IoU} column and the results from binary cross entropy loss are in the F_{BCE} column. Better performances are marked in bold.
What is clear from these tables is that while accuracy and convergence favor both IoU and BCE seemingly at random, the IoU-score is distinctly higher for the new IoU loss function in most experimental settings. The exact nature and significance of this improvement will be detailed in section 5.

5 Discussion

The experiments performed and the results reported in section 4 show the patterns for the three metrics used to compare the two loss functions. Each of these patterns will be discussed briefly. While binary accuracy has been established as being less relevant as a measure of success for a segmentation task, it is noteworthy nonetheless to show that it does not decrease with the use of the IoU loss function.

5.1 Aberrations in the results

Before the metrics can be discussed, it is necessary to assess some individual results at face value. It can be seen that some combinations of model and dataset produce unexpected results, such as "big, no hair, F_{IoU}", "small, no hair, F_{IoU}" and "small, no hair, F_{BCE}" in table 4.3. It is most likely that these test settings terminate too quickly because hyperparameters that have been tuned to BFCN-32s do not allow FCResNet to perform optimally. However, in the interest of keeping test settings comparable, it was decided these hyper-parameters were to be unchanged.
Another possible reason for these aberrations in the results is that some combinations of data with the FCResNet model do not allow FCResNet to be trained optimally, due to the model being much deeper than BFCN-32s. FCResnet would therefore not be able to learn all the patterns in the data as a result of underfitting.

5.2 Binary Accuracy

As can be seen from tables 4.1, 4.2, 4.3 and 4.4, IoU-loss and BCE-loss score higher on accuracy in 43.75% and 50% of the cases, respectively. However, whether IoU-loss or BCE-loss results in a higher binary accuracy, the results are within 2% in every comparison. A paired t-test on these values remains inconclusive, with a p-value of 0.49, implying no significant difference between the mean performance

Name	Accuracy		IoU-Score		Convergence	
	F_{IoU}	F_{BCE}	F_{IoU}	F_{BCE}	F_{IoU}	F_{BCE}
big, hair	0.974	**0.978**	**0.921**	0.906	**91**	146
small, hair	0.962	**0.963**	**0.887**	0.857	116	**98**
big, no hair	**0.977**	0.976	**0.896**	0.852	89	**83**
small, no hair	**0.970**	0.967	**0.872**	0.837	155	**117**

Table 4.1: Results from BFCN-32s trained and tested on LFW: The name describes whether that setting used all of the training and validation images (big) or only 1000 training images and 100 validation images (small) and whether hair is included as part of the face (hair) or not (no hair). Accuracy refers to the pixel-wise accuracy score for that setting. Time needed for convergence is measured in epochs. Columns marked F_{IoU} show results for IoU loss. Columns marked F_{BCE} show results for binary cross entropy loss.

Name	Accuracy		IoU-Score		Convergence	
	F_{IoU}	F_{BCE}	F_{IoU}	F_{BCE}	F_{IoU}	F_{BCE}
big, hair	0.961	0.961	**0.877**	0.838	**125**	130
small, hair	**0.943**	0.940	**0.820**	0.779	**118**	131
big, no hair	0.983	**0.984**	**0.905**	0.877	**83**	125
small, no hair	**0.978**	0.977	**0.871**	0.837	**93**	129

Table 4.2: Results from BFCN-32s trained and tested on HELEN: see table 4.1 for clarification.

of these loss functions based on binary accuracy.

5.3 Intersection over Union

Contrary to binary accuracy, the results for Intersection over Union show consistent and significant improvements, albeit not in all experiments. While some exceptions to this are the instances where we can see that training has not been done optimally, as mentioned in section 5.1, there is also a clear distinction in performance between using the BFCN-32s model and the FCResNet model.
When including all test settings, a paired t-test shows that IoU-loss scores significantly better with a p-value of 0.042. While this is significant, it is meagre. When excluding the settings where aberrations occur, this value changes to 0.025, making the distinction more significant.
Because performance is mainly determined by the choice of a model, it is interesting to view the results for each model separately. When performing a similar t-test on only the results of BFCN-32s, the resulting p-value is 1.8×10^{-5}. This indicates that for the BFCN-32s model, IoU loss improves performance very significantly. However, for FCResNet the p-value is 0.41, which is represented by the fact that either loss function performs better in half the test settings.

5.4 Convergence

While the focus of this research was on a comparison of performance of the two loss functions, an improvement to training time would also be relevant. From the results, however, it can be seen that these training times vary wildly. A paired t-test shows that the average training time for IoU is lower with a p-value of 0.043, which is again meagre, but significant. When excluding the aberrations, however, this value becomes 0.064. Therefore no claim can be made as to a statistically relevant improvement in training times.

5.5 Example output images

In figures 5.1 and 5.2 an example output is shown. These images are from the experimental setting using the BFCN-32s model and the LFW dataset. They correspond to the example input and label shown in figure 2.1 and figure 2.4, respectively. While neither segmentation is perfect, the output for IoU in figure 5.1 is clearly showing the contours of the hairless face better than the output for

Name	Accuracy		IoU-Score		Convergence	
	F_{IoU}	F_{BCE}	F_{IoU}	F_{BCE}	F_{IoU}	F_{BCE}
big, hair	**0.939**	0.934	**0.783**	0.716	146	**92**
small, hair	**0.932**	0.922	**0.754**	0.686	**104**	134
big, no hair	0.939	**0.942**	0.632	**0.672**	**33**	110
small, no hair	**0.933**	0.921	**0.613**	0.453	44	**29**

Table 4.3: Results from FCResnet trained and tested on LFW: see table 4.1 for clarification.

Name	Accuracy		IoU-Score		Convergence	
	F_{IoU}	F_{BCE}	F_{IoU}	F_{BCE}	F_{IoU}	F_{BCE}
big, hair	0.913	**0.914**	**0.701**	0.686	**88**	111
small, hair	0.908	**0.916**	0.686	**0.716**	**99**	242
big, no hair	0.960	**0.961**	0.672	**0.727**	**61**	198
small, no hair	0.957	**0.959**	0.648	**0.654**	**111**	175

Table 4.4: Results from FCResnet trained and tested on HELEN: see table 4.1 for clarification.

BCE in figure 5.2. Unfortunately, for both images, some upsampling artifacts remain. This is most likely due to the coarse deconvolution layer, which upsamples by a factor of 32. These upsampling artifacts are much more pronounced upon initialization and are never fully removed during training.

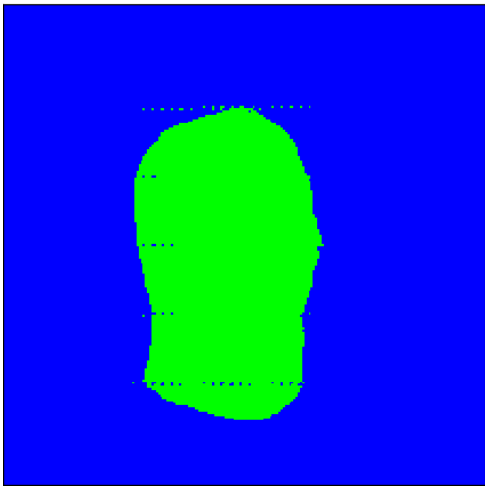

Figure 5.1: Example Output: BFCN-32s with IoU loss on LFW without hair

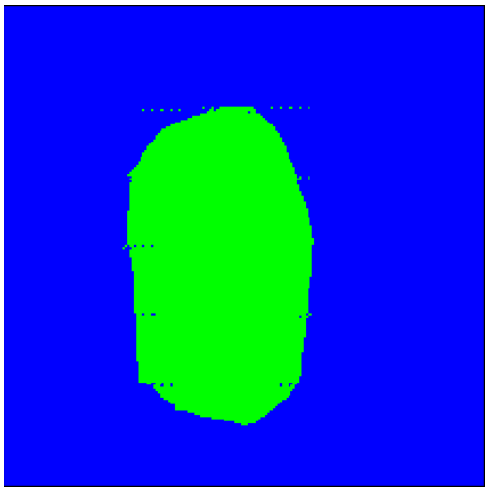

Figure 5.2: Example Output: BFCN-32s with BCE loss on LFW without hair

5.6 Dataset Size

In section 3 multiple dataset splits were proposed. The distinction between large and small was made to evaluate the effect of the size of the dataset on the effectiveness of either loss function. With the results presented in section 4 it can be seen that whenever a smaller dataset is used, performance suffers slightly, as is to be expected. However, whether IoU or BCE performs better does not change based on the size of the dataset.

6 Conclusion

Taking into account both the statistical analysis of the results and the output images presented, it is clear that training a segmentation model directly on Intersection over Union can lead to significant improvements. While these improvements were not the case for every experimental setting, and thus it cannot be claimed that training on IoU is objectively better, it is definitely an option worth considering when attempting to improve a model for state-of-the-art performance. This improvement on segmentation is in accordance with the results from Rahman and Wang (2016). This research shows that these improvements are independent of the sparsity of data, yet do depend on the choice of model.

What is also shown by this research is that in an ever-continuing attempt to improve state-of-the-art Deep Neural Nets, the area of loss functions has not been fully explored. This is in agreement with the conclusions from Janocha and Czarnecki (2017), which state that while cross entropy has been an unquestionable favourite, adopting one of the various other losses can be equally, if not more, effective. Mentioned in the work by Janocha and Czarnecki (2017) are squared hinge loss for pure accuracy focused research and expectation losses, as explained in the paper, for very noisy datasets. These conclusions, together with the conclusion from this and other research on the effectiveness of IoU loss show the same thing. More and more research is being done towards architectures, creating deeper or more convoluted networks, while an improvement can already be made by choosing a different loss function.

While this research shows that in some situations IoU loss improves performance, it does not do so in every circumstance. As such we propose that future research focuses on whether these circumstances can benefit from IoU loss with different hyper-parameters. In this research, it has been attempted to train a fully convolutional model based on ResNet (He et al. (2015)) with hyper-parameters that where optimized for a model based on FCN-32s (Shelhamer et al. (2017)). Further research can attempt to test each network with hyper-parameters of its own on multiple datasets to explore whether the lack of improvements for FCResNet could be due to the use of the new loss function or due to the initial model being flawed. Aside from this, the search for models to which IoU loss can be applied can be widened.

In section 1 an explanation is given why the loss function by Rahman and Wang (2016) is preferred to the loss function proposed by Yuan et al. (2017). In future research, it is also interesting to compare both these loss functions based on IoU directly.

Finally, the claim has been made that the benefit from training on IoU directly will only magnify when a model is presented with sparse data. This has not been evaluated in this research and can be done by expanding the models presented here to perform a segmentation class on multiple classes. This would significantly reduce the amount of positive samples in a dataset and thus be a way to explore the hypothesis that an IoU loss function outperforms binary cross entropy on sparser data. In the work by Rahman and Wang (2016) sparse data is already used. However, here the sparsity of the data is taken as is and not isolated to determine its effect on the performance of the IoU loss function. As such future work could focus specifically on certain datasets, comparing performance on sparse and dense data.

References

F. Ahmed, D. Tarlow, and D. Batra. Optimizing expected intersection-over-union with candidate-constrained CRFs. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1850–1858, Dec 2015. doi: 10.1109/ICCV.2015.215.

V Badrinarayanan, A Kendall, and R Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

François Chollet et al. Keras. `https://keras.io`, 2015.

J. Deng, W. Dong, R. Socher, L. J. Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009. doi: 10.1109/CVPR.2009.5206848.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results. http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html, 2011.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL `http://arxiv.org/abs/1512.03385`.

Katarzyna Janocha and Wojciech Marian Czarnecki. On loss functions for deep neural networks in classification. *CoRR*, abs/1702.05659, 2017. URL `http://arxiv.org/abs/1702.05659`.

Andrew Kae, Kihyuk Sohn, Honglak Lee, and Erik Learned-Miller. Augmenting crfs with boltzmann machine shape priors for image labeling. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

Vuong Le, Jonathan Brandt, Zhe Lin, Lubomir Bourdev, and Thomas S. Huang. Interactive facial feature localization. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part III*, ECCV'12, pages 679–692, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-33711-6.

H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. *Proceedings of the IEEE International Conference on Computer Vision*, 2015 International Conference on Computer Vision, ICCV 2015:1520–1528, 2015.

S. Nowozin. Optimal decisions from probabilistic models: The intersection-over-union case. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 548–555, June 2014. doi: 10.1109/CVPR.2014.77.

Md. Atiqur Rahman and Yang Wang. Optimizing intersection-over-union in deep neural networks for image segmentation. In *ISVC*, 2016.

Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4), 2017.

Mennatullah Siam, Mostafa Gamal, Moemen Abdel-Razek, Senthil Yogamani, and Martin Jägersand. Rtseg: Real-time semantic segmentation comparative study. *CoRR*, abs/1803.02758, 2018. URL `http://arxiv.org/abs/1803.02758`.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. URL `http://arxiv.org/abs/1409.1556`.

Y. Yuan, M. Chao, and Y. C. Lo. Automatic skin lesion segmentation using deep fully convolutional networks with jaccard distance. *IEEE Transactions on Medical Imaging*, 36(9): 1876–1886, Sept 2017. ISSN 0278-0062. doi: 10.1109/TMI.2017.2695227.

H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for image restoration with neural networks. *IEEE TRANSACTIONS ON COMPUTATIONAL IMAGING*, 3(1):47–57, 2017.