Semi-supervised learning through adversary networks for baseline detection

Romain Karpinski, Abdel Belaïd

Semi-supervised learning through adversary networks for baseline detection

Romain Karpinski
Université de Lorraine - LORIA
Campus scientifique, 54500 Vandoeuvre, France
romain.karpinski@yahoo.fr

Abdel Belaïd
Université de Lorraine - LORIA
Campus scientifique, 54500 Vandoeuvre, France
abdel.belaid@loria.fr

***Abstract*—The aim of this paper is to propose a new strategy adapted to the semantic segmentation of document images in order to extract baselines. Inspired by the work of Grüning [7], we used a convolutional model with residual layers enriched by an attention mechanism, called ARU-Net, a post-processing for the agglomeration of predictions and a data augmentation to enrich the database. Then, to consolidate the ARU-Net and help explicitly model dependencies between feature maps, we added a module of "Squeeze and Excitation" as proposed by Hu et al. [9]. Finally, to exploit the amount of unrated data available, we used a semi-supervised learning, based on ARU-Net, through the use of adversary networks. This approach has shown some interesting predictive qualities, compared to Grüning's work, with easier processing and less task-specific error correction. The resulting performance improvement is a success.**

***Keywords*-Semi-supervised learning; Semantic segmentation; ARU-Net; Adversary networks**

I. Introduction

There are two tasks: detection and extraction of baselines. Baseline detection can be thought of as a semantic segmentation task where every pixel over which baselines pass should be classified as such. The extraction is done from the prediction of the detection and provides the coordinates of the lines. The U-net architecture, as proposed by [12], is designed to perform semantic segmentation on biomedical image documents. This architecture consists of a contraction path (encoder) that captures the context of the image, and an expansion path (decoder) that, from the encoded feature maps, returns to the original image size.

At least two systems of the literature [5], [7] use this architecture for the extraction of baselines but remain either very greedy in place memory or too consuming data or computing time due to the multiplication of layers. In this work, we propose an improvement of this architecture by the addition of a Squeeze and Excitation module (SE) and a semi-supervised automatic learning through the use of adversary networks. We used the same dataset, cBAD, as [5], [7] to be able to compare our results.

The paper is organized as follows: In section II, a study of the state-of-the-art baseline detection methods is performed. In section III, we detail the different modules of the system. In Section IV, the experiments are commented. Finally, section V concludes the work and gives some research ideas for the future.

II. Related work

We focus here on the study of three methods in particular, all of which are recent and two of the three participated in the cBAD competition [3].

The first is that of Renton et al. [11] detecting X-Heights instead of baselines. To make the labeling of the pixels, they use an architecture inspired from VGG16. This architecture has a series of convolutions followed by a max pooling, with a number of filters increasing after each max pooling, except for the penultimate max pooling. The VGG16 architecture is modified to make it fully convolutional. To do this, the authors remove the max pooling operations and keep the first six layers of convolutions. To compensate for this, they use dilated convolutions to obtain a broader context. The results obtained on the basis of cBAD for the sub-base "simple" are: 75% F-Measure, 66% Precision and 86% Reminder on baselines. This method is interesting because the size of the image does not change. However, this architecture is greedy in memory.

The second method proposed by Fink et al. [5] consists of several steps. At first, a U-net is applied to image patches and predicts a probability for each feature on each patch. They add to U-net a fully connected layer before performing the classification. Once the characteristics of each patch have been extracted, they use a second U-net to perform baseline detection. The idea behind this strategy is to provide the context network to perform labeling and avoid edge effects. Instead of using the cross entropy as a loss function, they use a modified version of the Dice coefficient. This allows them to improve the accuracy of baseline extraction from 76.59% to 81.41%. The extraction is performed from the predictions of the second network using the least squares method. The extracted baselines then go through a post-processing step which aims at removing the erroneous lines and joining the lines being cut into pieces. A weakness of this method is that it uses several networks that make it slower compared to a method with a single network.

The third method described by Grüning et al. [7] is a two-step method: at first, one uses an ARU-net (a U-net having residual connections in its blocks and an attention mechanism) to detect the pixels belonging to the baselines. Then, a post-processing is operated to extract the baselines from the predictions. The ARU-net works by using the input image with n extra scales of the input image. These n different scales denoted S_n, correspond to the successive sub-samples of the input image by a factor f via the average pooling operation. The input of the post-processing used corresponds to the prediction of the network performing the detection of the baselines and the separation of lines. This mechanism has a cost in terms of time because it is necessary to obtain the predictions of the RU-net for each scale. The presented algorithm includes a heavy post image processing. The prediction is binarized before selecting points of interest. One uses curvature and distance conditions to group the points. The treatment is effective but heavy. It corrects errors made by the system but is task specific.

III. Proposed Approach

The proposed method is based on a well-reinforced U-Net to which we have added different modules. It uses semi-supervised learning with adversary networks. We will describe in the following all these developments.

A. *Squeeze and excitation module*

The Squeeze and Excitation (SE) module, as defined by Hu et al. [9], is added to the residual connections to help explicitly model the dependencies between the feature maps. At first, the branch of the module SE reduces the size of the characteristic maps to a value by calculating the average of the values for each map; This operation is called Global Average Pooling. Then, two fully connected layers are applied to the maps to model their relationships. In order to obtain a weight, each output of the last layer is activated by a Sigmoid function which gives a value in the interval [0; 1]. This value is then used to weight the values of the second inactivated branch. We verify in experiments the effectiveness of this module and show that we can improve the ARU-net architecture by adding it to the residual connections.

B. *Attention network*

The attention for each scale is calculated with a simple FCN (Fully Convolutional Network) containing only 4 convolutional layers and a small number of filters whose last layer is the classification layer which gives a single map per scale. This unique map corresponds to the A_i maps and it will be used to weight the feature maps. Although this network is too small to perform baseline detection directly; It has a sufficient number of filters and layers to detect the different attentions to be worn for each scale. This attention mechanism will specialize the attention scales to identify a type of feature such as background, text, text contours or noise, depending on the ease of detection that is a function of the scale. Indeed, it is easy to roughly detect the location of lines when the image is small, but it is difficult to accurately predict the location of the baselines. On the other hand, with a high scale, it is difficult to capture the entire context of the baselines because the pixels can be remote and out of the local scope of successive convolutions.

C. *Post-processing*

The method used to extract baselines from predictions consists of 6 steps:

1) The prediction of baselines by the network is binarized in order to simplify the extraction and to filter the low probabilities.
2) Next, the algorithm Mean Shift [2] is used to group the points close to each other.
3) The Density-Based algorithm (DBScan from [4]) is used to group the previously formed groups into larger ones that will represent the final lines.
4) The end group points are plotted on a blank image before extracting the related components.
5) The orientation of each of the related components is calculated to detect whether they are horizontal or vertical for the next step.
6) The baselines are calculated by performing a polynomial regression from the related components.

We have carried out different experiments by varying the parameters of the architectures and we will report those that have produced interesting results compared to the state of the art (see Table I). The optimizer used is RMSprop from [8] and the learning rate is 10^{-3}.

Experiment	Action	Precision (%)	Recall (%)	F-Measure(%)	Cross Entropy - Validation
A	RU-net without attention mechanism	88,92	88,99	88,96	0,0416
B	RU-net with attention mechanism (ARU-net)	90,16	89,61	89,89	0,0421
C	Crossed summed entropy replaced by average crossed entropy	88,42	90,52	89,46	0,0379
D	ARU-net V2: Attention distributed according to the characteristics extracted	89,91	90,37	90,14	0,0395
E	Impact of image resizing during learning.	90,21	95,69	92,87	0,0337
F	E with ARU-net V2	89,99	**95,99**	92,89	0,0340
G	E + image rotations	89,69	95,16	92,35	0,0357
H	G with ARU-net V2	79,84	95,24	86,86	0,0356
I	H by decreasing the learning rate	92,17	94,94	93,54	0,0351
J	Polynomial decreasing vs. exponential decreasing	89,24	95,23	92,14	0,0336
K	Curriculum learning	**92,47**	94,53	93,49	0,0243
L	Data augmentation	91,89	95,62	**93,62**	**0,1904**
[7]	ARU-net	96.27	96.15	96.36	-

Table I
Summary of results for each experiment on the test dataset.

The experiment L corresponds to the best results obtained and to the best convergence. In this experiment, instead of making a progressive learning which simultaneously increases the difficulty of each transformation, we decide to follow on these transformations once the current configuration is well acquired.

D. Semi-supervised learning

The principle, depicted in Figure 1, is to train the network first on annotated documents and then use the result of this training on non-annotated documents in order to obtain predictions. Then, from these predictions, the goal is to determine which pixels can be trusted. Once the correct areas or pixels are identified, we can use them to drive the network training.

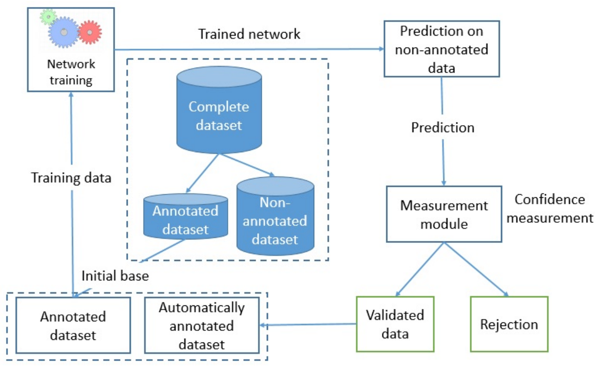


Figure 1. Functioning schema of a semi-automatic learning algorithm.

The pixel selection can be defined by a set of rules and constraints induced by the task to be performed. For example, in the case of baseline detection, one can rely on the form of predictions to filter out areas of the image that are not clearly predicted as lines. However, the difficulty in this case would be to identify the missing baselines and those that are in excess. Another way to make this selection is to learn to predict the correct areas, of ambiguous zones, that need to be set aside. In order to achieve this semi-automatic learning, we choose to use adversarial networks as defined by [10].

1) Adversarial neural networks: The adversarial neural networks, more specifically the networks of generative adversarial neurons, propose a new method which makes it possible to estimate a generative model by means of an adversarial training. This method uses two neural networks: a generator G and a discriminator D. The generator G must capture the distribution of the truth data while the discriminator D must estimate the probability that a data originates from the truth or it is generated by the generator. D must therefore minimize the probability of making a mistake. G's goal is to maximize the probability of deceiving D into believing that the data generated comes from the truth. This training can be seen as a min-max game with two players: G and D.

Let p_{data} be the truth distribution of the data, p_g, the generator distribution and y, the data coming from either the truth or generated by the generator. In order to train the generator, we need a random noise $p_z(z)$ that allows the generator to match this noise with the truth distribution, in such a way that $G(z) \in p_{data}$.

Let D be a discriminator that takes an input y and gives as output a value $D(y)$ representing the probability that y comes from the generator in relation to the truth. D is used to differentiate between p_{data} and p_g.

In order to allow the generator to capture the difference determined by the discriminator, G is trained by minimizing $log(1 - D(G(z))$. We can then formulate the problem by determining that the training corresponds to a min-max set defined by the function $V(G, D)$ as shown by the Equation 1. The first part of the equation (before the sign $+$) aims to minimize the probability of not correctly classifying a data as a given data belonging to the distribution of the truth. The second part (after the sign $+$) is true for the min part of the min-max problem.

$$min_G \ max_D \ V(G,D) = \mathbb{E}_{y\sim p_{data}(y)}[log(D(y)]+ \mathbb{E}_{z\sim p_z(z)}[1 - log(D(G(z))] \quad (1)$$

Figure 2 shows a schematic example of the training of an adversarial network. More concretely, we use two loss functions $\mathcal{L}_d$ and $\mathcal{L}_{adv}$ in order to train the two networks. The first, $\mathcal{L}_d$, is used to oblige the discriminator to differentiate between $y\sim p_{data}$ and $G(z)\sim p_z$. The second, $\mathcal{L}_{adv}$, is used to drive the generator through the discriminator. Gradients are calculated from the discriminator. However, the weights of the latter are not changed. Then, by back-propagation, the gradients of the discriminator are transmitted to the generator until the input. It is by this mechanism that the generator absorbs the difference between its distribution p_g and that of data p_{data} which has been captured by the discriminator.

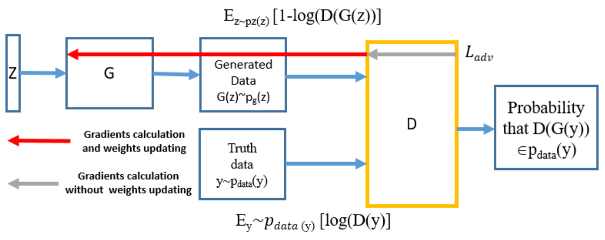

$E_{z\sim pz(z)}$ [1-log(D(G(z))]
L_{adv}
Z
G
Generated Data G(z)~p_g(z)
D
Probability that D(G(y)) ∈p_{data}(y)
Gradients calculation and weights updating
Gradients calculation without weights updating
Truth data y~p_{data}(y)
$E_y\sim p_{data\,(y)}$ [log(D(y)]

Figure 2. Functioning schema of adversarial neural network.

2) Baseline detection: In the configuration of [10], the truth distribution p_{data} is given by classes of baseline pixels in images. Also, random noise is no longer a noise nor random because we do not want to generate images from a given space. Actually, the space of p_z is already defined in our case and it corresponds to the images. The generator must, from an image, generate the associated prediction. The discriminator always has the same role and must differentiate between a baseline prediction from the generator and a truth image from the p_{data} distribution.

Instead of training the adversarial networks directly, the authors combine the classical learning that drives the generator by measuring the error associated with its prediction in relation to a truth, with the adversarial learning. In addition, the discriminator (an FCN) no longer gives a single value for an entry, but provides a probability of belonging to p_{data} for each pixel of the prediction. The authors determine that this modification is essential to improve the results. The loss function of the generator is therefore defined by the Equation 2 with $\mathcal{L}_{ec} = H(Y, G(z))$, summed cross entropy, $\mathcal{L}_{adv}$, the cross entropy of the adversarial training defined by the Equation 3, and λ_{adv}, the weight given to the adversarial training.

$$\mathcal{L}_g = \mathcal{L}_{ec} + \lambda_{adv}\mathcal{L}_{adv} \tag{2}$$

$$\mathcal{L}_{adv} = H(Y, D(G(z))) \tag{3}$$

In addition, they consider that the probability given by the discriminator, can be used as a confidence measure to detect reliable areas. In this case, the correct areas are those that have the same distribution as p_{data}. This allows us to set up a semi-supervised learning on non-annotated images. A non-annotated image $z = S_n$ is labeled by the generator $G(z)$, then the discriminator determines the areas of the image that correspond to the truth distribution $D(G(z))$. The output of the discriminator is thresholded by a value T_d to obtain a binary confidence mask $I(D(G(z)) > T_d$, with $I(\cdot)$ the indicator function. The generator's prediction is also thresholded by T_g giving $I(G(z)) > T_g = \hat{Y}$ to obtain the class of each pixel.

$$\mathcal{L}_{semi} = -\sum_{i=1}^{m}\sum_{j=1}^{|C|} I(D(G(z)) > T) \cdot \hat{Y} \times log(G(z)) \tag{4}$$

$$\mathcal{L}_g = \mathcal{L}_{ec} + \lambda_{adv}\mathcal{L}_{adv} + \lambda_{semi}\mathcal{L}_{semi} \tag{5}$$

As we saw earlier, we replace the summed cross entropy by the mean cross entropy. We will now study through experiments how the network can be trained with this new architecture and what are the performances according to several scenarios.

IV. Experiments

We have experimentally studied how the network can be driven with this new architecture and what are the performances according to several scenarios. This is related in the following paragraphs and the results are reported in the table II.

We started by studying the impact of the number of elements in the training set on the performance of the system. We choose to train the network conventionally with $\lambda_{adv} = 0$ and $\lambda_{semi} = 0$ using 25% (L2) and 50% (L1) of the training data. As one might expect, performance decreases as the amount of data in the training set also decreases. However, it can be noted that even with half of the data, the results are close to the L0 experiment (100% of training data). This means that the network used has a good ability to generalize with little data.

A. *Adversarial training*

The next experiment was to use an adversarial training by following the Equation 2. We start by training a network with 25% of the data and using a weight of $\lambda_{adv} = 0.01$, as defined in [10]. The results obtained are not at all satisfactory because the network does not converge. Indeed, the adversarial learning completely blocks all the training because the discriminator can easily make the difference between the truth data and the predictions.

Since the discriminator is good, the generator must be as good as it is to make the task of the discriminator more difficult. However, it is easier to tell the difference between the predictions and the truth than it is to label the historical images. As a result, the generator fails to follow the rate imposed by the discriminator. Even having a weight of $\lambda_{adv} = 0.01$ which makes the adversarial loss function small compared to the cross entropy loss function. The simplest solution found by the generator to fool the discriminator is to predict no baseline. Since the predicted image is homogeneous, it is difficult for the discriminator to differentiate from images of truth. So we get a solution that is not satisfactory because the generator is blocked in this configuration and can hardly get out.

Balancing the two networks is a complex task that is very time consuming. We tried to slow down the pace of the discriminator but without success. At each attempt, the discriminator finally takes over the generator which causes a collapse of the system. To overcome this problem, there is a technique known as label diffusion that makes it possible to disseminate class values for images of truth. This technique consists in adding a noise in the probabilities to be predicted. That is, instead of having to predict 0 or 1, the network must predict $0{+}b$ or $1{-}b$ with b a fairly low value and $b < \frac{1}{|C|}$. For example, with $b = 0.2$, the network must predict for a two-class classification, the vector $[0.8; 0.2]$ instead of having to predict $[1.0; 0.0]$. If we draw a different value b for each pixel, we end up with a distribution p_{data} closer to that naturally given by a network. In this case, the task of the discriminator becomes much more complex because it can no longer be based on remarkable values.

In the case of baseline detection images, we apply a Gaussian noise on the baseline pixels, making them more like a prediction. The results obtained with this method are visible in the Table II by the L4 experiment, compared to the systematic failure of the L3 experiment. The authors of [10] explain that in their case, this diffusion is not necessary and that the network converges correctly. In the case of baselines, the peculiarity of the form of truth necessitates the use of this technique.

The next experiment, L5, aims to determine whether the concatenation of the source image with its prediction or truth helps the discriminator to determine the correct regions and to differentiate between bad or correct labeling. Indeed, the generator could predict a labeling that strongly resembles the truth without being fair. For example, on an image containing no line of text, predicting regular lines would be considered fair by the discriminator because it is an image that can be present in the truth. The only condition for determining the error is to have the information at the origin of the labeling. The discriminator must therefore predict $P(y|z)$ for each pixel instead of $P(y)$ denoted as "DX" in the tables. It can be seen that this addition improves system performance with a gain of +0.8% F-Measure.

The experiment L6 uses a weight $\lambda_{adv} = 0.001$ instead of $\lambda_{adv} = 0.01$ to see if the adversarial training is degrading or improving performance. We can see that the results obtained with $\lambda_{adv} = 0.01$ are the best of the two configurations with a difference of 1.89% F-Measure. The adversarial training allows, compared to the traditional training, to improve the results of 2% at the level of the F-Measure which is a consequent improvement, considering the small number of data.

We conduct further the experiments by training the adversarial network with 100% of the data (experiment L9). The results obtained are the best, with all experiments combined with 94.66% of F-Measure (+0.91% F-Measure compared to the second). This result confirms again the interest of using this training method.

Figure 3(c) shows the prediction obtained with the L9 experiment. It can be noted that the prediction obtained with the L9 experiment is much more precise and clear than that of the L experiment. Figure 3(d) shows the result of the prediction given by [7]. We find the same prediction characteristics as with our implementation, that is to say, the prediction is diffuse in some locations. With the adversary training, we do not find diffuse regions as in Figure 3(b) and Figure 3(d). This limits the false detection of baselines and makes them much easier to extract and allows the use of a simple extraction method. We can therefore conclude that with the use of a more efficient system than ours combined with the adversarial training, an increase of the results of the state of the art is expected.

B. Semi-supervised adversarial training

We have conducted an experiment on the semi-supervised part of the learning phase of the system (experiment L10 in Table II). The network learning is defined by the Equation 5 with $T_d = 0.8$ and $T_p = 0.2$. In addition, we set $\lambda_{semi} = 0$ for the first 20 epochs. As the authors [10] explain, this avoids using noisy predictions of the discriminator and gives the two networks time to give correct results. The results of the experiment are not very conclusive. Indeed, the performances are degraded (-0.8 %) of F-Measure. However, it can be noticed that the accuracy obtained is greater with an improvement of 0.98% at the expense of a recall loss of 2.49%. This can be explained by the fact that the network learns from predictions where baselines are completely missing. Increasing the accuracy indicates that the learning is reliable when the lines are detected and the discriminator has validated them. However, when the lines are missing, if the discriminator does not classify their pixels as false, the network will learn to ignore them which will lower the recall.

Experiment	Quantity	DX	Diffusion	λ_{adv}	λ_{semi}	Precision (%)	Recall (%)	F-Measure (%)
L/L0	-	-	-	-	-	89.06	95.7	92.27
L1	0.25	No	No	0	0	81.13	94.75	87.41
L3	0.25	No	No	0.01	0	-	-	-
L4	0.25	No	Yes	0.01	0	86.28	95.01	90.43
L5	0.25	Yes	Yes	0.01	0	86.44	95.66	**90.82**
L6	0.25	Yes	Yes	0.001	0	81.64	96.22	88.33
L10	0.25	Yes	Yes	0.01	0.1	87.42	93.17	90.02
L2	0.5	No	No	0	0	87.93	95.98	91.78
L7	0.5	Yes	Yes	0.01	0	91.42	96.19	**93.75**
L8	0.5	Yes	Yes	0.001	0	88.52	96.71	92.43
L9	1.0	Yes	Yes	0.00	0.1	92.87	96.52	**94.66**

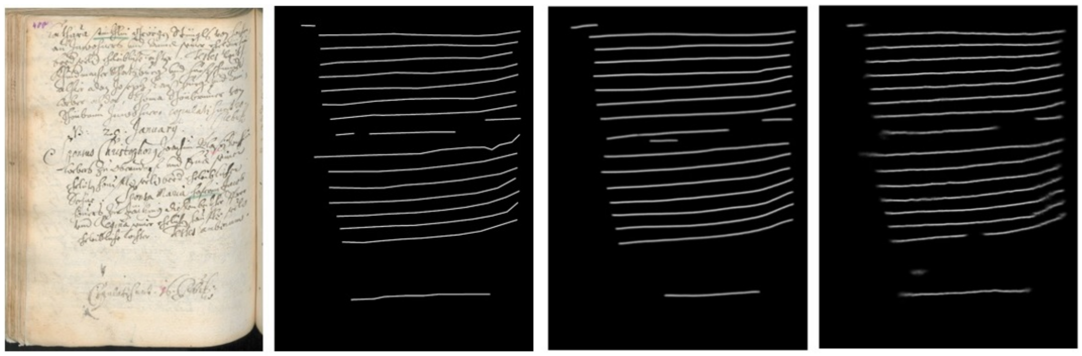
Table II
EXPERIMENTS RESULTS WITH A SIMPLE BASELINE EXTRACTION METHOD.

The experiments carried out made it possible to discern several points. First, the SE module improves network performance, which confirms that the ARU-net architecture must be used in combination with this module for better results. Second, cross entropy is not always related to network performance and may not be the best loss function for this task. Indeed, the calculation of the entropy does not take into account the form of the predictions. Nothing prevents the network from predicting a scatter plot for example, while predictions must look like lines. If we could integrate the shape of the elements to predict, then, the performances would be improved.

The networks of opposing neurons make it possible to correct this point by considerably improving the form of the prediction. As the architecture is stabilized, the pursuit of work oriented towards semi-supervised learning is now possible to make use of non-annotated data. The use of a simple post-processing allows, the case of an industrial use, to reduce the time required to treat an image which can result in a saving of time and money for a company.

The method was inspired by [7] work but we get good results with a simpler post-processing and better predictions. The addition of the SE module has made a strong contribution to making the network even better, although this has not been explicitly demonstrated for lack of time.

We have no better results than [7] in the quantitative sense but more qualitative. The quality of the predictions obtained is characterized by the shape of the predicted lines which makes it possible to apply a simpler post-processing. More



(a) Complex image from cBAD (b) Truth image of (a) (c) Prediction by L9 experiment (d) Prediction by Grüning et al. (2018)

Figure 3. Comparison of the truth image of line detection with respect to predictions on the image (a)

efforts should have been put in place to develop a post-processing to obtain better quantitative results. The problem with the baseline extraction case is that it does not evaluate the performance of the detection of these lines.

V. Conclusion

A semi-supervised approach, through the use of adversary networks, has been studied and adapted to the detection of baselines in historical documents. The use of the diffusion of the labels was essential in order to stabilize the architecture. Moreover, we have seen that the concatenation of the source image and its prediction or truth given to the discriminator increases the performances. It also allows the discriminator to differentiate between good and bad labeling based on the original image. In conclusion, the consequent improvement in performance through the use of limited data is a success. On the second point, more experiments are required to obtain a stable learning mechanism on unannotated data. The work done lays the groundwork for further work and describes a method for implementing semi-supervised learning through adversarial networks. We have no better results than [7] in the quantitative but more qualitative sense. The quality of the predictions obtained is characterized by the shape of the predicted lines which makes it possible to apply a simpler post-processing. More efforts should have been put in place to develop a post-processing to obtain better quantitative results. The problem with the baseline extraction case is that it does not evaluate the performance of the detection of these lines.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[2] Comaniciu, D., Meer, P., Mean shift: A robust approach toward feature space analysis. TPAMI 24, pp. 603–619, 2002.

[3] Diem, M., Kleber, F., Fiel, S., Grüning, T., Gatos, B., Scriptnet: Icdar 2017 competition on baseline detection in archival documents (cBAD), 2017.

[4] Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al., A density-based algorithm for discovering clusters in large spatial databases with noise., in: Kdd, pp. 226–231, 1996.

[5] Fink, M., Layer, T., Mackenbrock, G., Sprinzl, M., Baseline detection in historical documents using convolutional u-nets, DAS, pp. 37–42, 2018.

[6] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., Generative adversarial nets, in: Advances in neural information processing systems, pp. 2672–2680, 2014.

[7] Grüning, T., Leifert, G., Strauß, T., Labahn, R. A Two-Stage Method for Text Line Detection in Historical Documents URL: http://arxiv.org/ abs/1802.03345, 2018.

[8] Hinton, G. and Srivastava, N. and Swersky, K., Rmsprop: Divide the gradient by a running average of its recent magnitude, Neural networks for machine learning, Coursera lecture 6e, 2012.

[9] Hu, J., Shen, L., Sun, G., Squeeze-and-excitation networks. arXiv preprint arXiv:1709.01507 7, 2017.

[10] Hung, W.C., Tsai, Y.H., Liou, Y.T., Lin, Y.Y., Yang, M.H., Adversarial learning for semi-supervised semantic segmentation. arXiv preprint arXiv:1802.07934, 2018.

[11] Renton, G., Chatelain, C., Adam, S., Kermorvant, C., Paquet, T., 2017. Handwritten text line segmentation using fully convolutional network, ICDAR, pp. 5–9, 2017.

[12] Ronneberger, O., Fischer, P., Brox, T., U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical image computing and computer-assisted intervention, Springer. pp. 234–241, 2015.