Project Machine Learning
— Milestone 3 —

Hazem Lahoual, Shashi Durbha, Wai Tang Victor Chan

March 29, 2020

1 Outline

Our report in this milestone is organized as follows: we start in section 2 by an overview of two failures cases of CycleGAN model, namely image transfiguration and shape deformation. After that, section 3 describes the attention-guided GAN (AGGAN), which is proposed to overcome the stated failures above. Qualitative and quantitative evaluations are conducted in section 4 to compare the performance of AGGAN to CycleGAN. Section 5 verifies if there is any correlation between Inception Distance (IS) and Fréchet Inception Distance (FID), two evaluation metrics used to evaluate the performance of GANs. In section 6, we describe our experiments and results in an attempt to the application of CycleGAN in the use of cell-tracking in biomedical images. Finally, we state the conclusions in section 7.

2 Failure Cases

This section gives a brief overview of two failure cases of CycleGAN, namely image transfiguration and shape deformation.

2.1 Image transfiguration

Image transfiguration means the change of appearance without change of shape. For this task, CycleGAN performs well and the generators learn how to change textures and colors between domains. But, when translating the input image, the changes do not only affect the desired object but also other parts of the image that should be kept unchanged. Figure 1 shows two examples of failure cases.

The first example in figure 1b shows that even though the CycleGAN manages to translate successfully the horse in figure 1a to zebra, the background is affected by the change (it turns grey).

The second example in figure 1d demonstrates that the generator fails to understand the context of the image and to separate semantically the different objects in the input. What we mean by this is that the generator can not separate between the horse in the background, to be translated to zebra, and the boy in the foreground. This leads to the changes during the translation will affect both objects (the skin of the boy in the foreground changed to zebra texture).

(a) real (b) generated (c) real (d) generated

Figure 1: Examples of failure of CycleGAN in the case of image transfiguration.

(a) real (b) generated (c) real (d) generated

Figure 2: Examples of failure of CycleGAN in the case of shape deformation.

2.2 Shape Deformation

CycleGAN fails in the task where there is a geometric change in the input shape. We see in figure 2 two examples of failure in the case of shape deformation. In both cases (cat to dog and dog to cat), the output is different from the target domain. In fact, the generated images are unrealistic images of cat and dog, and far from being considered as real by a human. This can be explained by the fact that CycleGAN fails to extract high-level features (what is in the image), and limit itself to translate low-level features (change of textures, etc.).

3 Attention-Guided GAN

In this section, we present first, the attention-guided GAN as an improvement to CycleGAN. We describe after that the process of attention-guided cycle, followed by the losses used. Finally, we formulate the problem as a minimization problem.

3.1 Idea

Using attention mechanism is proposed in the literature (Zhang et al. (2018)) as a way to avoid the artifacts and failures in the generative adversarial networks: We have a way to focus on a certain part of the input and ignore the other parts. As an example of use in the case of domain translation, e.g. horse to zebras, we focus on the pixels that represents the horses and keep the background unchanged when translating to zebras.

Tang et al. (2019) propose the Attention-Guided Generative Adversarial Networks (AGGAN) to overcome the limits of CycleGAN, by introducing attention networks to the generators. This added networks will produce attention masks, with the goal to learn to detect the key parts of the input image. These parts will be the only ones to be affected by translation while everything else is kept unchanged. As a result, this will help to limit the undesired changes and artifacts.

3.2 Process

We want to train two mappings between two domains X and Y. For this, we train two generators with a built-in attention mechanism: $G : x \in X \rightarrow G(x) \in Y$ and $F : y \in Y \rightarrow F(y) \in X$. Unlike the CycleGAN, the generators are composed of three sub-generators:

- Parameter-sharing encoder G_E: extracts both low-level and high-level features of the input image.
- Content Mask generator G_C: generates content masks, which will be combined to generate the output image.
- Attention mask generator G_A: generates attention masks, which determine a pixel-wise intensity maps that define how much each pixel of the content masks generated by G_C contributes to the output image.

To explain the process of attention-guided cycle, we take the case of translation from domain X to domain Y, the output of the attention mask generator G_A is $n-1$ foreground attention masks $\{A_y^f\}_{f=1}^{n-1}$ and one background attention mask A_y^b (here n is

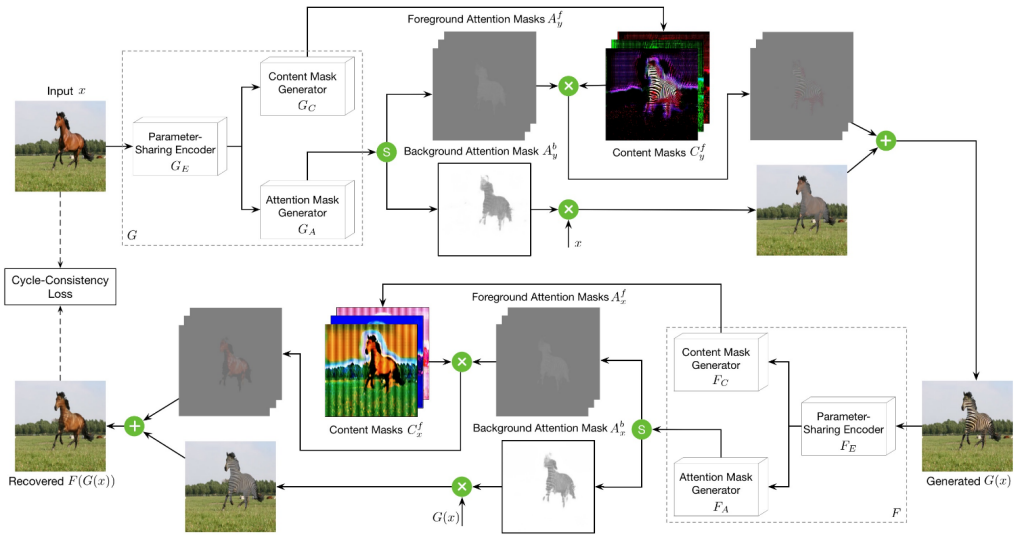


Figure 3: Attention-guided cycle (source: Tang et al. (2019))

a hyperparameter to choose before the training). This allows detecting the foreground (the parts of the image to translate between domains) and to preserve the background. Furthermore, the content mask generator G_C generates $n-1$ content masks $\{C_y^f\}_{f=1}^{n-1}$. So, these content masks combined with the input image will be used as intermediate masks to generate the output image.

After generation of masks, the attention masks are normalized by a channel-wise softmax activation function and then multiplied by the corresponding content masks to obtain the output images:

$$G(x) = \sum_{f=1}^{n-1}(C_y^f * A_y^f) + x * A_y^b$$

In the first summand $\sum_{f=1}^{n-1}(C_y^f * A_y^f)$, we have $n-1$ element-wise multiplication between the content and foreground attention masks, which help to focus on the parts to be translated. In the second summand $x * A_y^b$, we have an element-wise multiplication between input image and background mask, which allows to preserve the background. Finally, the intermediate results are merged to obtain the output image.

We have the same steps for the translation from domain Y to X. The output is defined by the following equation.

$$F(y) = \sum_{f=1}^{n-1}(C_x^f * A_x^f) + y * A_x^b$$

Figure 3 summarizes the steps of attention-guided cycle.

3.3 Losses

Here, we discuss briefly the losses used in AGGAN. We refer the reader to section 2 of milestone (MS) 2 report to have more details about the different losses used in CycleGAN.

3.3.1 Cycle-consistency Loss

Similar to CycleGAN, we want that an input image $x \in X$ and the reconstructed image $F(G(x)) : x \to G(x) \in Y \to F(G(x)) \in X$ to be as similar as possible, where:

$$F(G(x)) = \sum_{f=1}^{n-1}(C_x^f * A_x^f) + G(x) * A_x^b$$

We want the same for an image $y \in Y$ and the reconstructed image $G(F(y))$, where:

$$G(F(y)) = \sum_{f=1}^{n-1}(C_y^f * A_y^f) + F(y) * A_y^b$$

The cycle-consistency loss in the case of AGGAN can be expressed by:

$$\mathcal{L}_{cycle}(G, F) = \mathbb{E}_{x \sim p_{data}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{data}(y)}[\|G(F(y)) - y\|_1]$$

3.3.2 Adversarial Loss

The two adversarial losses are same as the ones used in CycleGAN:

$$\mathcal{L}_{GAN}(G, D_Y) = \mathbb{E}_{y \sim p_{data}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{data}(x)}[\log (1 - D_Y(G(x))] \quad (1)$$

$$\mathcal{L}_{GAN}(F, D_X) = \mathbb{E}_{x \sim p_{data}(x)}[\log D_X(x)] + \mathbb{E}_{y \sim p_{data}(y)}[\log (1 - D_X(F(y))] \quad (2)$$

3.3.3 Identity Loss

Also here, the identity loss is the same as the one used in CycleGAN:

$$\mathcal{L}_{identity} = \mathbb{E}_{x \sim p_{data}(x)}[\|F(x) - x\|_1] + \mathbb{E}_{y \sim p_{data}(y)}[\|G(y) - y\|_1]$$

3.4 Problem Formulation

Having discussed the losses, we can formulate the attention-guided GAN process as the following optimization problem:

$$\min_{G_X, G_Y} \max_{D_{XA}, D_{YA}} \mathcal{L}_{GAN}(G, D_Y) + \mathcal{L}_{GAN}(F, D_X) + \lambda_{cycle}\mathcal{L}_{cycle} + \lambda_{id}\mathcal{L}_{identity}$$

where λ_{cycle} and λ_{id} control the importance of each type of loss.

4 AGGAN Evaluation

We evaluate in this section the performance of AGGAN model in the cases where CycleGAN fails, namely image transfiguration and shape deformation. A qualitative evaluation is conducted first, followed by a quantitative evaluation.

4.1 Experiments Setup

We want to compare the performance of AGGAN to CycleGAN. For the latter model, we train a 9-blocks resnet as a generator, with log loss and patchGAN discriminator (see section 3.3 of MS 2 report for a full description of the architecture). We use the same architecture for the AGGAN model. This time, the output of the generator consists of n attention masks and $n-1$ content masks. We choose $n = 10$ as the number of attention masks.

For the image transfiguration case, we use the horse2zebra dataset. For the shape deformation case, we use cats and dogs dataset (Elson et al. (2007)). The training set is large (more than 40000 images of cats and 8000 images of dogs), so we limit ourselves to 1500 images for the training phase.

4.2 Qualitative Evaluation

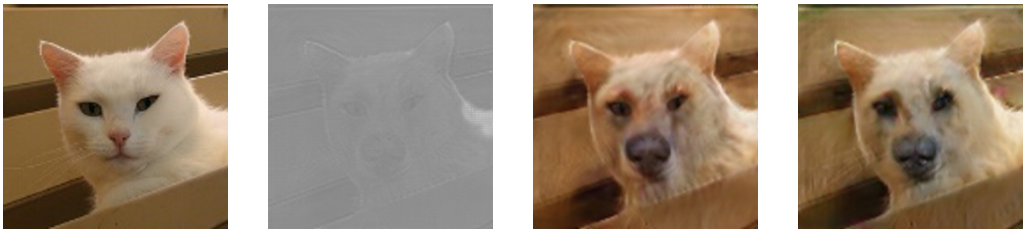
We want to evaluate the models qualitatively, which means to evaluate the quality of the output images.

- **Image transfiguration:** figure 4 shows an example of horse to zebra translation. We can see clearly that AGGAN gives better output than CycleGAN. In fact, the background is kept unchanged, and only the pixels that represent the horse are translated between domains. CycleGAN fails to detect to limits of the input domain to translate, that is why the background is also affected by changes: it turns grey, the same color used for zebra generation.

  Figure 4b represents the background mask learned by AGGAN. Here, the bright pixels are considered as a part of the background, while the dark pixels are not (here value 1 means white color and value 0 means dark color, as the masks are normalized). AGGAN manages successfully to separate the foreground from the background: the dark pixels take almost the same shape as horses in the input image.

(a) real horse (b) background mask (c) AGGAN (d) CycleGAN

Figure 4: Output images generated by AGGAN and CycleGAN models in the case of horse to zebra translation.



(a) real cat (b) foreground mask (c) AGGAN (d) CycleGAN

Figure 5: Output images generated by AGGAN and CycleGAN models in the case of cat to dog translation.

- **Shape deformation:** We have in figure 5 an example of cat to dog translation. Here, both AGGAN and CycleGAN fail to give good results, with the former having slightly better performance than the latter. In fact, the output dog in the case of AGGAN has a better shape of nose and eyes compared to one given by CycleGAN, but it is still far from being considered as a dog.

  Figure 5b shows one the foreground masks learned by AGGAN. Here, we decide to show a foreground mask and not the background mask, because the shape deformation will affect the background, so there is no point to focus on it. Instead, the foreground mask determines the quality of the generators. The learned foreground mask fails to detect the high-level features of the input image, and this explains the low quality of the output.

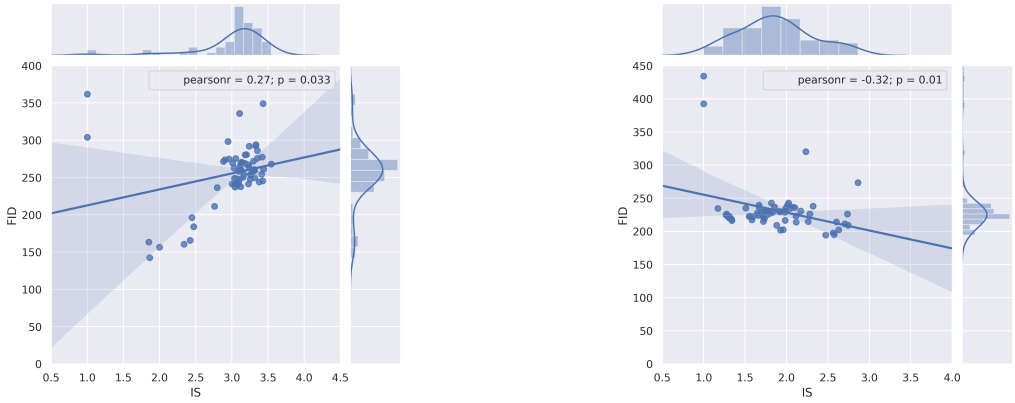
4.3 Quantitative Evaluation

We want to verify our results above quantitatively. So, we apply the Fréchet Inception Distance (FID) to the generated outputs (see section 4 of MS2 report for more details about the evaluation metrics).

- **Image transfiguration:** table 1 contains the FID scores applied to the generated test images of AGGAN and CycleGAN for horse to zebra (H2Z) and zebra to horse (Z2H) experiments. AGGAN outperforms CycleGAN in both cases, but the difference between the FID scores is not large: 71.76 versus 77.20 in the case of H2Z for example. This can be explained by the use of a small number of content masks: even though we manage to separate the foreground and the background, and we keep the latter unchanged, we don't have enough information about how to translate the foreground. This can be solved by increasing the content and foreground masks, which leads to better extraction of high-level features from the input images.

model	α ($\times 10^{-3}$)	β_1	β_2	λ_{cycle}	λ_{id}	FID H2Z	FID Z2H
CycleGAN	0.2	0.5	0.999	10	0.5	77.20	135.72
AGGAN	0.2	0.5	0.999	10	0.5	**71.76**	**124.84**

Table 1: FID scores of AGGAN and CycleGAN models using the horse2zebra dataset.



(a) horse to zebra translation

(b) zebra to horse translation

Figure 6: Marginal distributions of Inception score (IS) and Fréchet Inception Distance (FID) score. The Pearson correlation coefficient and p-value are also showed

- **Shape deformation:** this time, we apply the FID scores applied to the generated test images of AGGAN and CycleGAN for cat to dog (C2D) and dog to cat (D2C) experiments. Table 2 summarizes the results. Again here, AGGAN has slightly better performance than CycleGAN, but both of them fail to have any good scores: in the case of C2D, both models have FID scores larger than 240, which means they fail to generate realistic images (this can be seen clearly from the qualitative evaluation). It is also the same for D2C, where the two models have large scores (more than 150). That means, the output images can not be considered as cat images by humans.

model	α ($\times 10^{-3}$)	β_1	β_2	λ_{cycle}	λ_{id}	FID C2D	FID D2C
CycleGAN	0.2	0.5	0.999	10	0.5	249.63	164.80
AGGAN	0.2	0.5	0.999	10	0.5	**246.42**	**159.70**

Table 2: FID scores of AGGAN and CycleGAN models using the cats and dogs dataset.

5 Correlation between Metrics

We discussed in MS2 different evaluation metrics for GANs, and we showed the correlation between the Fréchet Inception Distance (FID) and the Kernel Inception Distance(KID). We want to check in this section if there is any inverse correlation between the Inception Score (IS) and the FID.

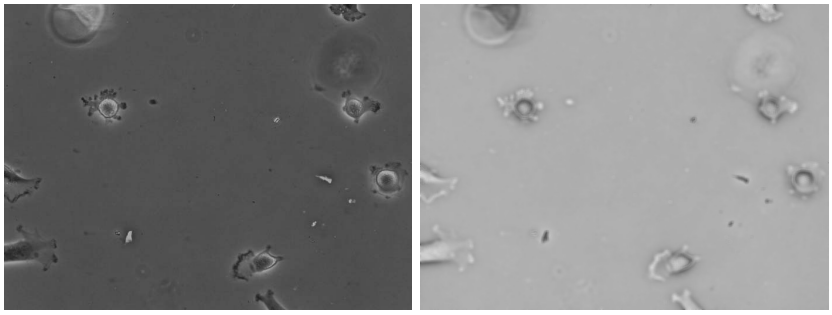
Figure 6 shows the joint plots (marginal distribution) of IS and FID obtained from all models trained in MS2 (70 models), along with the Pearson correlation coefficient (measures the linear correlation between two datasets with values between 1, which indicates correlation and -1, which indicates inverse correlation) and the p-value (insignificant in our case as we have small dataset: 70 samples).

For the case of zebras generation (figure 6a), surprisingly we have a positive correlation coefficient, which was not expected. This can be explained that our first generator (from horse to zebra) is not generating images similar to zebras, but adding black and white noises to input images, so it gives good IS score, but when using FID score, we have bad scores, because we are using both real and fake images for the computations of the FID score.

For the case of horses generation (figure 6b), we have a small inverse correlation. This can be explained by the fact that IS and FID have different ranges, that means a large change of FID only results in a small change of IS. However, the results are not good enough to assume the existence of a correlation between IS and FID as for KID and FID.

(a) original image (b) cell tracker (c) ground truth

Figure 7: The use of cell-tracking application on NIKON.



(a) original image (b) artificially modified

Figure 8: Artificially generated image imitating the effects of the use of a different microscope with different lens.

6 Experiment: Transfer Learning

We would like to explore the idea of the application of image style transfer as an image pre-processing step in a pipeline for an image processing utility of a purpose other than image stylization.

Suppose we have previously trained a model S_A that can process images of style A to produce an associated desired images from them or to retrieve specific information from them, we would like to obtain a similar model S_B that can work on images of style B and produce the same results. Instead of training this model of such particular functionality, we would like to explore the possibility of training the stylizer $G_{B\to A}$ that can transfer an image of style B to an image of style A. Doing so maybe beneficial, if the image application S_A is very difficult to train from scratch, and applying the stylizer $G_{B\to A}$ is time-efficient enough to be added in the application pipeline.

$$S_B = S_A \circ G_{B\to A}$$

6.1 Description of Experiments and Datasets

Ronneberger et al. (2015a) proposed an auto-encoder that can efficiently perform the task of cell tracking in microscopic images. The application released on the website of Ronneberger et al. (2015b) is trained to identify cell-occupied pixels of images of Glioblastoma-astrocytoma U373 cells on a polyacrylamide substrate. An example is shown in Figure 7, where the cell-tracking script identifies all the pixels corresponding to cells.

We will consider microscopic image sets provided for the cell tracking challenge from the IEEE International Symposium on Biomedical Imaging (ISBI). They are Glioblastoma-astrocytoma U373 cells on a polyacrylamide substrate (NIKON), HeLa cells on a flat glass (ZEISS), HeLa cells stably expressing H2b-GFP (OLYMPUS) and GFP-GOWT1 mouse stem cells (LEICA). The brand of the microscope for each dataset is written in bracket and will be used to denote the dataset.

Unfortunately all these datasets correspond to cell type different from the NIKON, upon which the original cell-tracking script is based. Therefore, we also artificially created a new dataset (ARTIFICIAL), which is obtained by applying colour inversion and Gaussian blur to the original images of NIKON. An example can be seen in Figure 8.

(a) original image (b) cell tracker (c) ground truth

Figure 9: The use of cell-tracking application on ZEISS.

In Figure 9, the result obtained by feeding a microscopic slide from ZEISS is illustrated. The majority of the non-cell-pixels were wrongly classified as cell-pixels. It is a good example to show the limitation of the original script when we extend it for use of microscopic images of other styles without image preprocessing.

Our goal is to apply domain-transfer to images from each of ZEISS, OLYMPUS, LEICA and ARTIFICIAL so that the output images resemble those from NIKON. These resultant images will be fed to the cell-tracking application and the results will indicate the effectiveness of the domain-transfer works.

We will compare the accuracy of the cell-tracking script with the F-score, which is the harmonic mean of precision (percentage of correctly identified pixels amongst all pixels classified as cell-pixels) and recall (percentage of correctly identified pixels amongst all the actual cell-pixels). This measure is more robust against datasets where cell-pixels are sparse.

6.2 Model Training Condition

From the experience of the hyperparameters tuning in Milestone 2, we choose the CycleGAN model to use a 9-block resnet, with the same learning rate ($\alpha = 0.0002$, $\beta_1 = 0.5$, $\beta_2 = 0.999$) and $\lambda = 10$, but the model will not use any identity loss.

6.3 Summary of Results

Example of cell-tracking results are illustrated in Figure 10. The average accuracy of cell identification is given in Table 3.

Datasets	NIKON	ZEISS	OLYMPUS	LEICA	ARTIFICIAL
Before Style-Transfer	0.974	0.650	0.000	0.127	0.392
After Style-transfer	NA	0.247	0.157	0.041	0.342

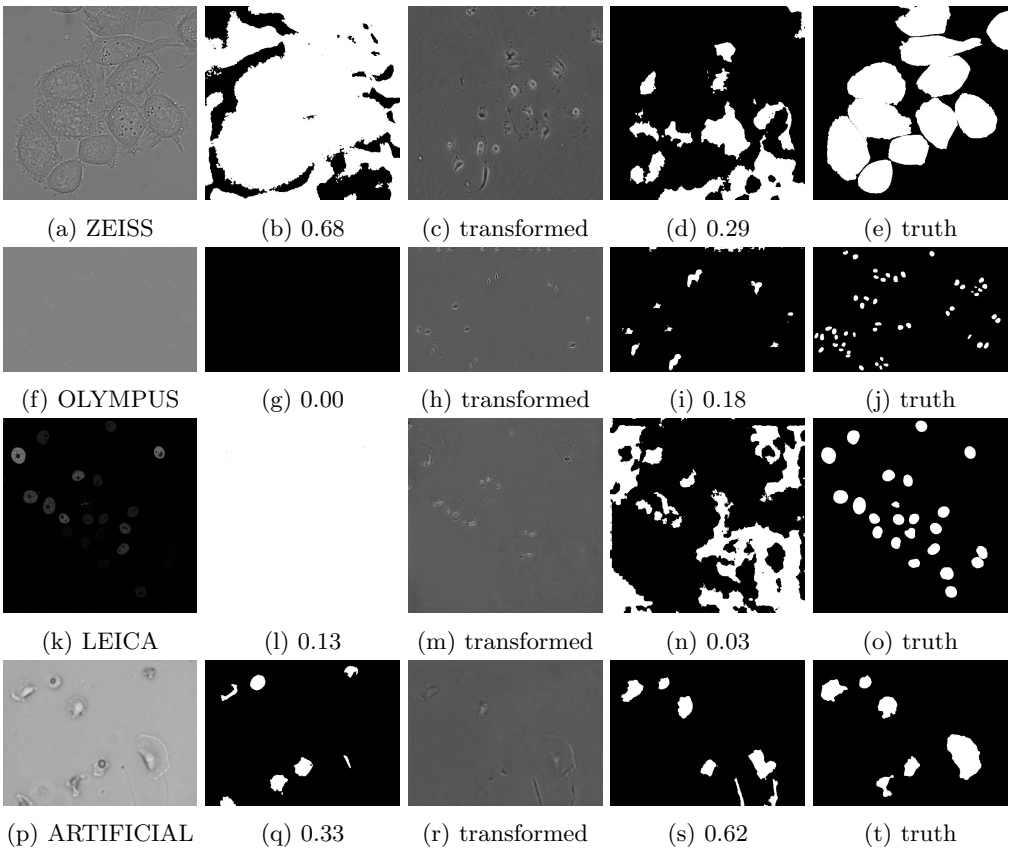
Table 3: F-scores of cell images from different data-sets

6.4 Discussion: ZEISS

We can observe that the results given by the domain-transfer give very poor representation of the cells from the original microscopic images. Due to the difference in size (measured pixels) between the cells, the bigger cells are disintegrated into smaller pieces in the transferred images. This definitely accounts for the poor performance of the overall cell-tracking from this data-set.

6.5 Discussion: OLYMPUS

Not only are the original images with rather low overall contrast (the background and the cells have very similar hue), the majority of the images are quite noisy and the cell cluster patterns are distinctively different from the cells from NIKON. The results of



(a) ZEISS (b) 0.68 (c) transformed (d) 0.29 (e) truth

(f) OLYMPUS (g) 0.00 (h) transformed (i) 0.18 (j) truth

(k) LEICA (l) 0.13 (m) transformed (n) 0.03 (o) truth

(p) ARTIFICIAL (q) 0.33 (r) transformed (s) 0.62 (t) truth

Figure 10: Examples of microscopic images style transfer. The figures below the second and fourth column are the F-scores of the classification accuracy of the result from the cell-tracking application.

the domain-transfer are visually quite unsatisfactory, and they are also reflected by the objective evaluation from the output of the cell-tracking application.

6.6 Discussion: LEICA

The individual cells from the original images have a wide range of grayscale values, which may create some confusion to the domain-transfer program, making it unable to semantically identify the cells. We can observe from exemplary outputs that dimmer cells were treated merely as part of the background in the style-transferred images, which contributes to a lower recall score for the classification.

6.7 Discussion: ARTIFICIAL

The style-transferred images create an emphasis on the edge of the cells, instead of the whole cell, misleading the classification program into identifying only the cell-membranes. The examples we collected also suggest that our cycleGAN model does not perform well against inverted images. However, grayscale value inversion is quite common in biomedical applications so such results indicated a substantial point of weakness of cycleGAN for real applications.

7 Conclusion

While we have repeatedly displayed the power of cycleGAN for the use of domain transfer, we have identified several limitations of such a model in this milestone. These can be taken into consideration when one tries to further extend the area of application and robustness of the results from cycleGAN, as did attentionGAN.

Exploring the weaknesses of the network also contributes, to some degree, to the understanding of the interpretation of the model, as the mistakes made by the networks

reveal the lack of ability for semantic understanding of the neural network. For example, the lack of ability to separate objects not intended for style-transfer in horse-to-zebra application and the lack of ability, in general, to correctly identify cells in microscope-style-transfer applications. Knowing such limitations help us correctly determine how cycleGAN can be applied as part of the real-life applications, and what additional components are necessary to help along with it.

References

J. Elson, J. J. Douceur, J. Howell, and J. Saul. Asirra: A captcha that exploits interest-aligned manual image categorization. In *Proceedings of 14th ACM Conference on Computer and Communications Security (CCS)*. Association for Computing Machinery, Inc., October 2007.

ISBI. 2d+time datasets. `http://celltrackingchallenge.net/2d-datasets/`.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015a.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. `https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/`, 2015b.

H. Tang, D. Xu, N. Sebe, and Y. Yan. Attention-guided generative adversarial networks for unsupervised image-to-image translation. In *International Joint Conference on Neural Networks (IJCNN)*, 2019.

H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. Self-Attention Generative Adversarial Networks. *arXiv e-prints*, art. arXiv:1805.08318, May 2018.