AN EXPERT-DRIVEN DATA GENERATION PIPELINE FOR HISTOLOGICAL IMAGES

Roberto Basla, Loris Giulivi, Luca Magri, Giacomo Boracchi

{name.surname}@polimi.it   DEIB, Politecnico di Milano, Italy

ABSTRACT

Deep Learning (DL) models have been successfully applied to many applications including biomedical cell segmentation and classification in histological images. These models require large amounts of annotated data which might not always be available, especially in the medical field where annotations are scarce and expensive. To overcome this limitation, we propose a novel pipeline for generating synthetic datasets for cell segmentation. Given only a handful of annotated images, our method generates a large dataset of images which can be used to effectively train DL instance segmentation models. Our solution is designed to generate cells of realistic shapes and placement by allowing experts to incorporate domain knowledge during the generation of the dataset.

***Index Terms*—** Instance Segmentation, Data Generation, Deep Learning.

1. INTRODUCTION

In the medical domain, the effectiveness of Deep Learning (DL) methods can be hindered by data scarcity since annotated data may be hard or expensive to obtain. This issue is also accentuated by the variety of scenarios DL models have to face in medical imaging, as specific datasets need to be prepared to train DL models on new tissues, when image acquisition techniques change, or to address different goals. In this work, we propose a novel pipeline (Fig. 1) that enables training instance segmentation models in the histopathological imaging domain in very low data regimes.

Previous works have addressed the issue of data scarcity by means of Data Augmentation (DA) techniques [1]. The idea behind DA is to manipulate images in a realistic manner to increase the amount of annotated data during training, thus improving performance. These techniques have been extensively used for training classifiers, where supervision is at the image level, as far as they do not change the image's label. For tasks like histological instance segmentation, where annotations are provided at pixel level, the very same transformations often need to be applied both to the image and to the pixel-wise Ground Truth (GT). Common DA strategies include geometric and photometric transformations such as rotations, crops and contrast variations. Unfortunately, they can only be applied to already-existing samples resulting in a limited increase of variability. Image generation, instead, has the potential to obtain a large amount of diverse data, enabling more effective model training. On the flip side, this also requires generating annotations (here also referred to as *blobs*) that are pixel-wise consistent with generated samples.

A few efforts [2, 3] have been made towards generating both image and GT. These rely on DL models like Generative Adversarial Networks (GANs) [4] that, while providing good results, do not enable to steer the image generation towards images featuring desired properties like the cell distribution and spacing. Other works break down the generation problem to make it more controllable, but are limited to re-using cell masks from real data [2], or generating blobs at random [5], yielding potentially unrealistic results. Other approaches extract blobs from real images and place them over an empty canvas to create the image mask [6, 7]. Lastly, works such as [8] perform style transfer to transform a generated GT into a realistic image in a fully-DL framework.

None of these methods guarantee that the generated blobs and their positions are coherent with the target tissue, in particular in low-data regimes. We overcome this limitation by proposing a modular pipeline, illustrated in Fig. 1, that can incorporate domain knowledge into the blob generation and placement processes. Our pipeline also allows experts to steer them towards realistic samples even at very low data regimes. Our method is composed of three major steps: ⓐ Blob Generation, where pairs or randomly selected blobs – representing different sections of 3D cells – are interpolated to generate realistic contours, ⓑ Blob Placement, where each generated blob is placed in a GT mask by a greedy algorithm that constrains blob density and distance to satisfy expert-driven criteria, and ⓒ Image Generation via a style transfer Neural Network to generate visually realistic images from the computed GT. Experiments demonstrate that our method generates realistic images even starting from a single image, enabling training instance segmentation models with performance comparable to models trained on larger datasets.

2. PROBLEM FORMULATION

The Instance Segmentation problem is typically formulated as follows: given an histological image $I \in \mathbb{R}^{h \times w \times 3}$, find all objects of a particular class (typically cells or part thereof),

Code available at https://github.com/rb-sl/ExpertDrivenNuclei.

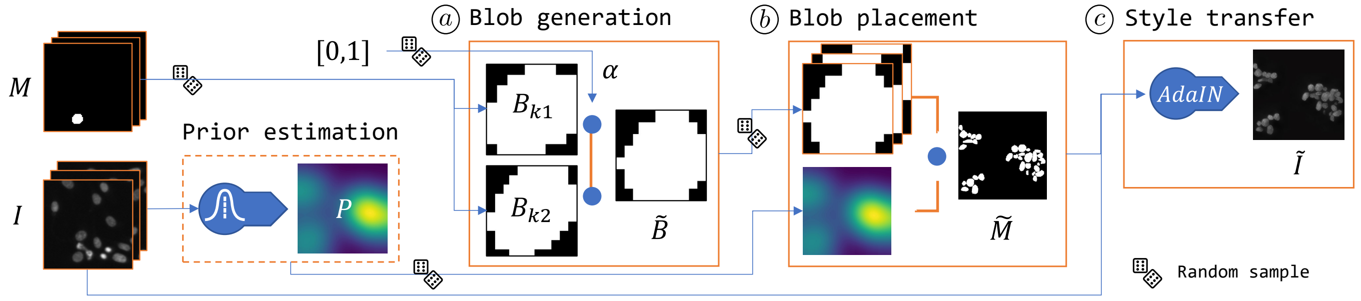


Fig. 1: Our generation pipeline. Our first phase is *Blob Generation* ⓐ, which creates a set of new blobs $\{\widetilde{B}_l\}_L$ by interpolating existing ones. We then perform *Blob Placement* ⓑ to generate the GT $\widetilde{M}$ following a prior distribution $\mathcal{P}$ estimated from the few annotated images. Finally, the *Image Generation* ⓒ phase performs style transfer to transform $\widetilde{M}$ into the new image $\widetilde{I}$.

thus returning binary masks $\widehat{M} \in \{0,1\}^{h\times w\times q}$ where each channel contains a segmentation mask for one of q objects within the image. Instance segmentation is typically solved by Convolutional Neural Networks (CNNs) trained on large datasets where each image I_j is paired with its GT annotation M_j. We address the problem of generating a training set composed of a large number (N) of synthetic images, $\{\widetilde{I}_n\}_N$, starting from a small training set of real images $\{(I_j, M_j)\}_J$. In particular, each generated image $\widetilde{I}_n$ is associated to a generated GT $\widetilde{M}_n$, which can be used to train an instance segmentation network. Training set generation can be summarized as:

$$\{(I_j, M_j)\}_J \to \{(\widetilde{I}_n, \widetilde{M}_n)\}_N, \quad N \gg J.$$

3. METHOD

Fig. 1 and Algorithm 1 describe the proposed generation pipeline that is composed of three parts. First, we leverage homotopy-based interpolation to generate blobs representing cells in a principled way; then, we generate GT segmentation masks by optimizing the positioning of blobs satisfying constraints on the blob distribution; lastly, we generate realistic images that match the generated GTs.

3.1. Blob Generation

The first step consists in generating a new set of L synthetic blobs $\{\widetilde{B}_l\}_L$ by interpolating pairs of cell masks randomly selected from real ones $\{B_k\}_K$ (Fig. 1, ⓐ):

$$\{B_k\}_K \to \{\widetilde{B}_l\}_L, \quad L \gg K.$$

The rationale behind our procedure is that blobs $\{B_k\}_K$ in histopathology correspond to projections of 3D cells into an image plane. If we assume cells are 3D convex volumes, any pair of image projections from the same cell are *homotopically equivalent*, thus projections over other planes can be obtained by continuously deforming one cell towards the other, as depicted in Fig. 2a. In non-convex cases, slicing a 3D cell may yield multiple connected components. Yet, homotopic equivalence applies locally to each projection with the same number of connected components. Since blobs in real images are projections of *similar* 3D cells, we generate the contours of new realistic blobs by interpolating the contours of randomly selected pairs of blobs. Algorithm 2 describes the procedure to generate L blobs starting from our set of real blobs $\{B_k\}_K$ extracted from the training set. We first sample a random pair B_{k1}, B_{k2} of real blobs, on which we identify a number E of equally spaced points along their contours p_{k1} and p_{k2}. We then perform ICP registration [9] to align and pair the contour points of the two blobs. Then, as shown in Fig. 2b, we interpolate between point pairs to generate the contour of the new blob:

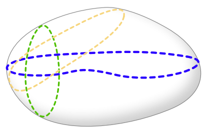
$$\widetilde{p}_l = \{\alpha p_{k1}^{(i)} + (1-\alpha)p_{k2}^{(i)} \quad \forall i \in [1, E]\}, \ \alpha \in [0\,,\,1].$$

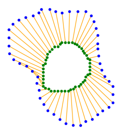
Each generated blob $\widetilde{B}$ is obtained by morphological area closing filling in the interpolated perimeters (Figure 2c). The whole procedure runs in $\mathcal{O}(K \cdot L)$.

3.2. Blob Placement

After obtaining the set $\{\widetilde{B}_l\}_L$ of synthetic blobs, we define realistic image-level GTs $\{\widetilde{M}_n\}_N$ by a greedy blob placement procedure that enables controlling blob spacing and density (Figure 1, ⓑ). To this end, we first define a set of 2D prior nuclei density distributions $\{\mathcal{P}_n\}_N$, $\mathcal{P}_n \in [0,1]^{h\times w}$. Each $\mathcal{P}_n$ is a heatmap (Fig. 3a), mimicking in each point $[i,j]$ the likelihood of a blob covering that region. Second, the univariate distribution $\mathcal{S}$ models the spacing between blobs in real images. Both $\mathcal{P}_n$ and $\mathcal{S}$ can be either estimated from the real available images or crafted by experts.

The proposed blob placement procedure is described in Algorithm 3. First, we initialize $\widetilde{M}$ as an empty binary mask and A as an availability mask which is initially set to ones. Then, we iteratively place blobs by sampling locations in $G_n = \mathcal{P}_n \cdot A$, which is normalized to sum to 1 to mimic the distribution of currently available locations. More specifically we sample a point $[i,j]$ from G_n and a value from $\mathcal{S}$ representing the offset z. We then select the first generated blob in $\{\widetilde{B}_l\}_L$ that fits in at the sampled location. If no blob fits, the algorithm terminates. Otherwise, we update A by setting to 0


(a) Homotopically equivalent blobs.


(b) Interpolation lines between B_{k1} and B_{k2}

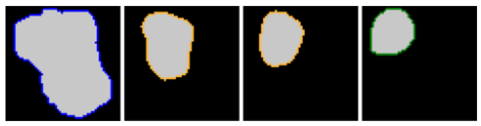

(c) Interpolated blobs between B_{k1} and B_{k2}

Fig. 2: Examples of interpolation between B_{k1} (blue) and B_{k2} (green). New blobs (in orange) are selected at equally spaced intervals along the interpolation lines and can be seen as different views of a 3D nucleus.

all the locations that are either covered by the blob or that are within a radius z from $[i, j]$, enforcing a minimum spacing between blobs. We remove the placed blob from $\{\widetilde{B}_l\}_L$. At the end, the synthetic mask $\widetilde{M}$ is returned. As displayed in Fig. 3, the proposed greedy procedure yields GTs that follow the prior more faithfully w.r.t. a random placement weighted according to $\mathcal{P}$.

Blob placement, although efficiently parallelizable, is the most computationally demanding component as it scales with $\mathcal{O}(N \cdot \frac{h \cdot w}{A_B} \cdot L)$. The fractional term represents the number of blobs that can fit the image given its area and an average blob area A_B.

Algorithm 1 Our generation pipeline

```
1:  Input: Small annotated training set $\{(I_j, M_j)\}_J$
2:  Output: Large synthetic training set $\{(\widetilde{I}_n, \widetilde{M}_n)\}_N$
3:  $\widetilde{D} \leftarrow \emptyset$                                   ▷ Blobs interpolation
4:  $\{\widetilde{B}_l\}_L \leftarrow$ INTERPOLATEBLOBS($\{B_k\}_K, L, E$)
5:  for $n \in [1, N]$ do
6:      $\mathcal{P}, \mathcal{S} \leftarrow$ GETPRIOR($h, w$)                 ▷ Blobs placement
7:      $\widetilde{M}_n \leftarrow$ GREEDYPLACEMENT($\mathcal{P}, \{\widetilde{B}_l\}_L, \mathcal{S}$)
8:      $R \leftarrow$ SAMPLE($\{I_j\}_J$)                          ▷ Style transfer
9:      $\widetilde{I}_n \leftarrow$ ADAIN(FLATTEN($\widetilde{M}_n$), $R$)
10:     $\widetilde{D} \leftarrow \widetilde{D} \cup \{(\widetilde{I}_n, \widetilde{M}_n)\}$
```

Algorithm 2 Blob interpolation

```
1:  procedure INTERPOLATEBLOBS($\{B_k\}_K, L, E$)
2:      for $l \in [1, L]$ do
3:          $B_{k1}, B_{k2} \leftarrow$ SAMPLE($\{B_k\}_K, 2$)
4:          $p_1 \leftarrow$ GETCONTOURPOINTS($B_{k1}, E$)
5:          $p_2 \leftarrow$ GETCONTOURPOINTS($B_{k2}, E$)
6:          $p_1 \leftarrow$ REGISTRATION($p_1, p_2$)
7:          $\alpha \leftarrow$ SAMPLE($[0, 1]$)
8:          $\widetilde{p} \leftarrow$ INTERPOLATE($p_1, p_2, \alpha$)
9:          $\widetilde{B}_l \leftarrow$ CLOSURE($\widetilde{p}$)
10:     return $\{\widetilde{B}_l\}_L$
```

3.3. Image Generation

After obtaining the GT masks $\{\widetilde{M}_n\}_N$, we generate the corresponding images by AdaIN [10], a style transfer framework (Figure 1, ⓒ) that can modify the style (i.e., the texture and appearance) of an image, while preserving its content. In our case, the content is the generated mask $\{\widetilde{M}_n\}_N$ while the style is represented by the collection of real images for training $\{I_j\}_J$. We therefore use AdaIN to create the texture characterizing cells over a mask $\widetilde{M}$, preserving the blob support resulting in pixel-perfect annotations for training segmentation networks. The content and style do not need to be paired, thus the collection of synthetic realistic images $\{\widetilde{I}_n\}_N$ is obtained as follows:

$$\{\widetilde{M}_n\}_N, \{I_j\}_J \rightarrow \{(\widetilde{I}_n, \widetilde{M}_n)\}_N.$$

In our pipeline (Algorithm 1, Line 9), we employ AdaIN by using as content the (flattened) generated mask $\widetilde{M}_n$ and as reference style a patch R from a real image I_j. We train for 30000 epochs using as DA flips and affine transformations for both style and content, and photometric transformations for the style image only. The output of AdaIN is a realistic image $\widetilde{I}_n$ where the displayed cells follow the input $\widetilde{M}_n$. Pairs

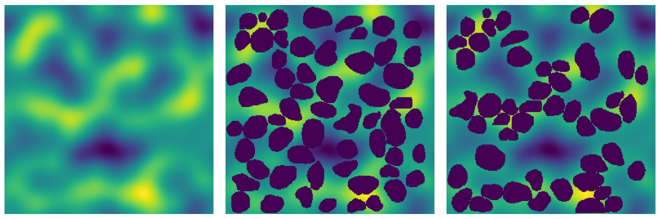

(a) Prior map $\mathcal{P}$. (b) Random. (c) Greedy.

Fig. 3: Examples of blob placement. Given a prior map $\mathcal{P}$ (a), our greedy placement (c) respects much more closely the distribution with respect to a random weighted placement (b).

Algorithm 3 Blob placement

```
1:  procedure GREEDYPLACEMENT($\mathcal{P}_n, \{\widetilde{B}_l\}_L, \mathcal{S}$)
2:      $\widetilde{M} \leftarrow 0^{h \times w \times q}$
3:      $A \leftarrow 1^{h \times w}$
4:      while $found$ do
5:          $G \leftarrow$ NORM($A \cdot \mathcal{P}_n$)
6:          $(y, x) \leftarrow$ SAMPLE($G_n$)
7:          $z \leftarrow$ SAMPLE($\mathcal{S}$)
8:          for $b \in \{\widetilde{B}_l\}_L$ do
9:              $found \leftarrow$ CANHOST($G, b, y, x$)
10:             if $found$ then
11:                 $\{\widetilde{B}_l\}_L \leftarrow \{\widetilde{B}_l\}_L \setminus \{b\}$
12:                 $\widetilde{M} \leftarrow$ ADDMASK($\widetilde{M}, b, y, x$)
13:                 $A \leftarrow$ UPDATEAVAILABLE($A, b, z$)
14:                 break
15:     return $\widetilde{M}$
```

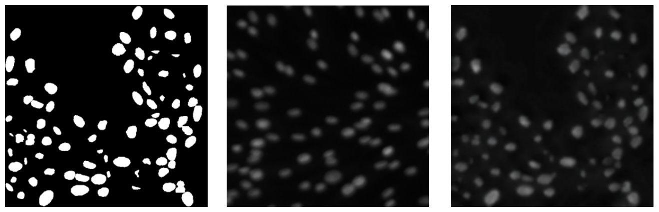

(a) Generated mask $\widetilde{M}$. (b) Reference style R. (c) $\widetilde{I}$ generated from $\widetilde{M}$.

Fig. 4: Example of style transfer.

of generated image and mask can be used as annotated samples to augment the small annotated training set $\{(I_j, M_j)\}_J$. Fig. 4 shows an example of this phase starting from a generated mask $\widetilde{M}$ and using as reference style an image R.

4. EXPERIMENTS

We validate the effectiveness of our pipeline by training HoVerNet [11], a state-of-the-art instance segmentation NN, on our generated dataset. We start by generating training sets from small subsets of fluorescence microscopy images from the Broad Institute Repository (BBBC) [12]. We then assess the instance segmentation performance on real images from the NucleusSegData dataset [13]. As a baseline, we compare the performance of the same network trained on the full BBBC dataset (*Full dataset*) and on the few annotated images used for generation (*Training real dataset*) leveraging standard augmentations. We adopt as metrics those used in HoVerNet, i.e., DICE, DICE2, AJI and AJI+.

During the Blob Placement phase, we model $\mathcal{P}_n$ as realizations of Perlin noise [14] as its complex 2D structures resemble a coarse view of the distribution of nuclei in histological images. We estimate the parameters of Perlin noise to maximize the similarity with blurred masks from training data. Then, we perform a preliminary analysis on the distributions of generated GTs to assess how close they are to real ones. In particular, we consider the distributions of area ($\mathcal{A}$) and aspect ratio (ar), reporting close results in both the median ($\mathcal{A}$: 155px vs. 153px, ar: 1.31 vs. 1.41) and in the Inter-Quartile Range ($\mathcal{A}$: 32px vs. 39px, ar: 0.21 vs. 0.23).

For the image generation phase, we train AdaIN using a tiled version of real images. Then, for each generated mask $\widetilde{M}_n$, we select as reference image the tile having the closest number of blobs to $\widetilde{M}_n$. This choice improves the style transfer procedure, since AdaIN generates images having average value similar to the reference. When the style has too many blobs, artefacts may appear in the image, and when it has too few, the nuclei may be fainter than in real images.

Results shown in Table 1 and in Figure 5 indicate that HoVerNet trained on the full dataset (815 annotated images, $\approx 50\,000$ blobs) achieves an impressive 0.94 DICE score and 0.81 AJI score. These values need to be considered as an ideal standard and are displayed with a dashed line in Figure 5. When trained from a generated dataset starting from only two real images (52 blobs), HoVerNet achieves 0.89 DICE score and 0.49 AJI score, compared to 0.42 and 0.22 when the same architecture is trained only on the same two real images. When increasing the number of real images this gap decreases, and the advantages of image generation are lost starting from 10 real images ($\approx 1\,000$ blobs). Nonetheless, these results show that our pipeline enables to train models effectively even in very low data regimes.

Table 1: Results on our test set.

J	K	**DICE**	**DICE2**	**AJI**	**AJI+**
Full dataset					
815	46016	0.94	0.70	0.81	0.81
Training on real images					
2	52	0.42	0.53	0.22	0.22
5	321	0.89	0.58	0.58	0.58
10	952	0.91	0.67	0.70	0.72
Training on our generated images					
2	52	0.89	0.48	0.49	0.49
5	321	0.89	0.70	0.73	0.74
10	952	0.92	0.69	0.75	0.76

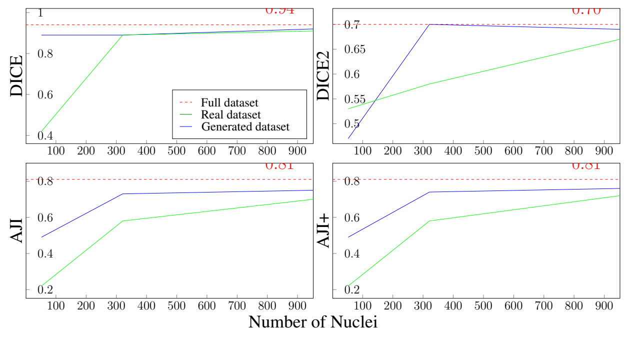


Fig. 5: Our results by metric per number of nuclei instances in the real training set.

5. CONCLUSIONS AND FUTURE WORK

We presented a novel pipeline for generating synthetic and realistic images, paired with their annotation for training instance segmentation networks. Our approach generates realistic blobs by interpolation and enables experts to control the density and spacing of blob placement in GTs. We demonstrated that our data generation pipeline helps in training a NN in low-data regimes.

Our future work consists in assessing the potential of our method to counteract domain shift with very scarce annotations. We will also explore GANs conditioned on mask geometric properties to promote diversity in the nuclei shapes. Additionally, we will analyze how different priors can influence the placement phase, and employ advanced optimization techniques capable of accounting for small overlaps as well.

6. COMPLIANCE WITH ETHICAL STANDARDS

This research study was conducted retrospectively using data made available in open access. Ethical approval was not required as confirmed by the license attached with the open access data.

7. ACKNOWLEDGMENTS

This paper is supported by FAIR (Future Artificial Intelligence Research) project, funded by the NextGenerationEU program within the PNRR-PE-AI scheme (M4C2, Investment 1.3, Line on Artificial Intelligence).

We also would like to thank Ikonisys, Inc. for sponsoring the PhD grant of Roberto Basla and for an insightful discussion.

8. REFERENCES

[1] Connor Shorten and Taghi M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, no. 1, pp. 60, Dec. 2019.

[2] Faisal Mahmood, Daniel Borders, Richard Chen, Gregory N. McKay, Kevan J. Salimian, Alexander Baras, and Nicholas J. Durr, "Deep Adversarial Training for Multi-Organ Nuclei Segmentation in Histopathology Images," Oct. 2018, arXiv:1810.00236 [cs].

[3] Wenyuan Li, Jiayun Li, Jennifer Polson, Zichen Wang, William Speier, and Corey Arnold, "High resolution histopathology image generation and segmentation through adversarial training," *Medical Image Analysis*, vol. 75, pp. 102251, Jan. 2022.

[4] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative Adversarial Networks," June 2014, arXiv:1406.2661 [cs, stat].

[5] Le Hou, Ayush Agarwal, Dimitris Samaras, Tahsin M. Kurc, Rajarsi R. Gupta, and Joel H. Saltz, "Robust Histopathology Image Analysis: To Label or to Synthesize?," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, June 2019, pp. 8525–8534, IEEE.

[6] Jijun Cheng, Zimin Wang, Zhenbing Liu, Zhengyun Feng, Huadeng Wang, and Xipeng Pan, "Deep Adversarial Image Synthesis for Nuclei Segmentation of Histopathology Image," in *2021 2nd Asia Conference on Computers and Communications (ACCC)*, Singapore, Sept. 2021, pp. 63–68, IEEE.

[7] Florian Kromp, Lukas Fischer, Eva Bozsaky, Inge M. Ambros, Wolfgang Dorr, Klaus Beiske, Peter F. Ambros, Allan Hanbury, and Sabine Taschner-Mandl, "Evaluation of Deep Learning Architectures for Complex Immunofluorescence Nuclear Image Segmentation," *IEEE Transactions on Medical Imaging*, vol. 40, no. 7, pp. 1934–1949, July 2021.

[8] Le Hou, Ayush Agarwal, Dimitris Samaras, Tahsin M. Kurc, Rajarsi R. Gupta, and Joel H. Saltz, "Unsupervised Histopathology Image Synthesis," Dec. 2017, arXiv:1712.05021 [cs].

[9] P.J. Besl and Neil D. McKay, "A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, Feb. 1992.

[10] Xun Huang and Serge Belongie, "Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization," July 2017, arXiv:1703.06868 [cs].

[11] Simon Graham, Quoc Dang Vu, Shan E Ahmed Raza, Ayesha Azam, Yee Wah Tsang, Jin Tae Kwak, and Nasir Rajpoot, "Hover-Net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images," *Medical Image Analysis*, vol. 58, pp. 101563, Dec. 2019.

[12] Vebjorn Ljosa, Katherine L Sokolnicki, and Anne E Carpenter, "Annotated high-throughput microscopy image sets for validation," *Nature Methods*, vol. 9, no. 7, pp. 637–637, July 2012.

[13] Gozde Nur Gunesli, Cenk Sokmensuer, and Cigdem Gunduz-Demir, "*AttentionBoost* : Learning What to Attend for Gland Segmentation in Histopathological Images by Boosting Fully Convolutional Networks," *IEEE Transactions on Medical Imaging*, vol. 39, no. 12, pp. 4262–4273, Dec. 2020.

[14] Ken Perlin, "Improving noise," in *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, San Antonio Texas, July 2002, pp. 681–682, ACM.