2023 IEEE 20th International Symposium on Biomedical Imaging (ISBI) | 978-1-6654-7358-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISBI53787.2023.10230593

BiaPy: a ready-to-use library for Bioimage Analysis Pipelines

Daniel Franco-Barranco[1,2,], Jesús A. Andrés-San Román[3,4], Pedro Gómez-Gálvez[3,5,6]*
Luis M. Escudero[3,4], Arrate Muñoz-Barrutia[7], Ignacio Arganda-Carreras[1,2,8]

[1] Dept. of Computer Science and Artificial Intelligence, University of the Basque Country (UPV/EHU)
[2] Donostia International Physics Center (DIPC)
[3] Instituto de Biomedicina de Sevilla (IBiS), Hospital Universitario Virgen del Rocío/CSIC/
Universidad de Sevilla and Dept. de Biología Celular, Facultad de Biología, Universidad de Sevilla
[4] Biomedical Network Research Centre on Neurodegenerative Diseases (CIBERNED)
[5] MRC Laboratory of Molecular Biology
[6] Department of Physiology, Development and Neuroscience, University of Cambridge
[7] Dept. de Bioingenieria, Universidad Carlos III de Madrid
[8] Ikerbasque, Basque Foundation for Science

ABSTRACT

In recent years, technological advances in microscopy have made available large amounts of data to biomedical researchers in the form of images. By learning from such large datasets, deep learning-based methods have successfully addressed previously inaccessible bioimage analysis tasks. However, most available solutions target a particular subset of problems, forcing users to be familiarized with different applications to complete their data analysis. On top of that, other issues, such as reproducibility, lack of documentation, or access to the code, arise. For these reasons, we introduce BiaPy, an open-source ready-to-use all-in-one library that provides deep-learning workflows for a large variety of bioimage analysis tasks, including 2D and 3D semantic and instance segmentation, object detection, super-resolution, denoising, self-supervised learning, and classification. All code and documentation are publicly available at `https://github.com/danifranco/BiaPy`.

***Index Terms*—** Bioimage analysis, deep learning, image segmentation, object detection, denoising, super-resolution.

1. INTRODUCTION

Over the last decade, the complexity and amount of biomedical datasets, especially in the form of images, have considerably increased [1]. Consequently, new technologies, including novel deep learning (DL) methods, have been developed to analyze and process them automatically. Despite the success of DL methods in many computer vision tasks [2], including biomedical applications [3], they often require high-level and up-to-date skills in programming, hindering their usage by researchers without that specific background [4]. The continuous and fast DL evolution, and the wide variety of bioimage analysis applications, compel the user to acquire knowledge in very different and changing frameworks, which becomes cumbersome.

Several open-source efforts have been developed to bridge the gap between computer science and bioimage analysis research. Fiji [5] was a milestone in that sense, introducing a large variety of image processing tools, including machine learning methods, to perform quantitative image analysis of biological microscopy data. Extending its usage to DL was the purpose of deepImageJ [6], which was introduced to plug in pre-trained DL models into Fiji. ZeroCostDL4Mic [7] followed the same direction, providing well-documented Jupyter notebooks using DL to tackle problems such as semantic segmentation, object detection, denoising, super-resolution, and image-to-image translation. ImJoy [8] also simplifies access to this technology through a web interface with plenty of DL solutions. Web-browser based approaches remove the barrier of software installation at the price of reducing flexibility.

As mentioned above, many of the available DL-based tools are task-specific. For instance, Cellpose [9], CellProfiler [10], Ilastik [11], PlantSeg [12] or ZeroCostDL4Mic [7] are commonly used for semantic and instance segmentation. Classification and object detection workflows can also be built in Ilastik [11], CellProfiler [10] and ZeroCostDL4Mic [7]. Remarkably, the core of all these approaches is frequently a convolutional neural network (CNN). Nevertheless, Dosovitskiy *et al.* [13] recently showed the first use case of natural language processing transformers applied to computer vision. From there, new transformer-based architectures have been presented, outperforming the CNN-based state of the art in biomedical applications. This is the case of TransUNet [14],

Corresponding author: daniel_franco001@ehu.eus

```yaml
PROBLEM:
    TYPE: SEMANTIC SEG
    NDIM: 2D
DATA:
    PATCH_SIZE: (256, 256, 1)
    TRAIN:
        PATH: /TRAIN_PATH
        MASK_PATH: /TRAIN_MASK_PATH
    VAL:
        SPLIT_TRAIN: 0.1
    TEST:
        PATH: /TEST_PATH
AUGMENTOR:
    ENABLE: True
    RANDOM_ROT: True
MODEL:
    ARCHITECTURE: unet
TRAIN:
    OPTIMIZER: SGD
    LR: 1.E-3
    BATCH_SIZE: 6
    EPOCHS: 360
TEST:
    POST_PROCESSING:
        Z_FILTERING: True
```

Fig. 1. Example of a BiaPy configuration file to perform state-of-the-art semantic segmentation on 2D data (example taken from [19]).

nnFormer [15] or UNETR [16] to name a few. Currently, transformers are leading bioimaging competitions such as Beyond the Cranial Vault (BTCV) [17] or the Medical Segmentation Decathlon (MSD) [18].

Following these recent advances in the field and to simplify access to them, we introduce BiaPy, an open-source Python library for building bioimage analysis pipelines, also called *workflows*. More specifically, our contributions are:

- A unified framework with ready-to-use implementations of the most demanded types of bioimage analysis pipelines for 2D/3D multi-channel images.
- A wide range of data augmentation techniques, including new specific methods for 2D and 3D biological microscopy image data.
- Many state-of-the-art DL models can be interchangeably used in all pipelines, ranging from popular CNNs to the most recent transformers.

2. METHOD

BiaPy is an open-source library implemented using Python and TensorFlow. All BiaPy pipelines comprise three main steps: data pre-processing, DL model training or inference, and data post-processing (see Fig. 2). Given BiaPy's DL-base core, a machine with a graphics processing unit (GPU) is recommended for fast training and execution.

BiaPy can be installed by simply downloading the source code and creating its corresponding Python or Conda environment as described in the online documentation. Once installed, all that is needed to apply a BiaPy workflow is to edit a text file based on YACS [1]. No coding is required. This configuration file includes information about the hardware to use (i.e., the number of CPUs/GPUs), the downstream task (among those described in Section 2.1), the model name, optional hyperparameters, the optimizer, and the paths to load/store data from/into. An example of a full segmentation pipeline is shown in Fig. 1, summing 25 text lines. Moreover, BiaPy has several ready-to-use templates for different types of workflows that can be easily adapted to their particular applications and data.

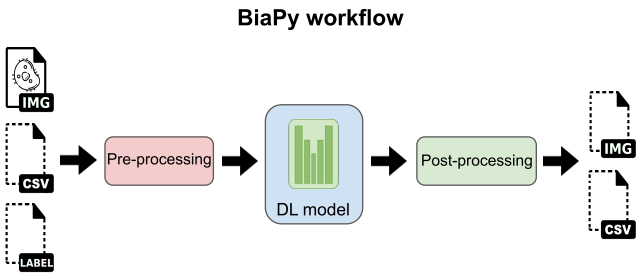

Fig. 2. General scheme of a BiaPy pipeline/workflow. Dash-lined elements are optional.

Finally, BiaPy offers online tutorials, example notebooks, and detailed documentation [2] to guide both users and developers that want to contribute [3].

2.1. Supported tasks

The main idea behind BiaPy is to present a unified framework that includes all the possible components of a bioimage analysis workflow. A comparison with other popular biomedical software analysis tools is shown in Table 2.1. A complete overview of the tasks provided by BiaPy is depicted in Fig. 3. More specifically, BiaPy offers the following types of pipelines in its current version:

Semantic segmentation, that associates a label to every pixel of the input image. BiaPy includes semantic segmentation pipelines for 2D/3D images using a wide range of models [20], from state-of-the-art CNNs to modern transformers.

Instance segmentation, where the goal is to detect, segment, and classify the individual objects of interest in the input image. BiaPy uses a bottom-up approach to instance segmentation in 2D and 3D by learning the binary masks, contours, and (optionally) the distance map of the objects of interest [21]. Next, those representations are combined to create seeds for a marker-controlled watershed transform to output the individual instances.

[1] https://github.com/rbgirshick/yacs
[2] https://biapy.readthedocs.io/en/latest/
[3] https://biapy.readthedocs.io/en/latest/contribute/general.html

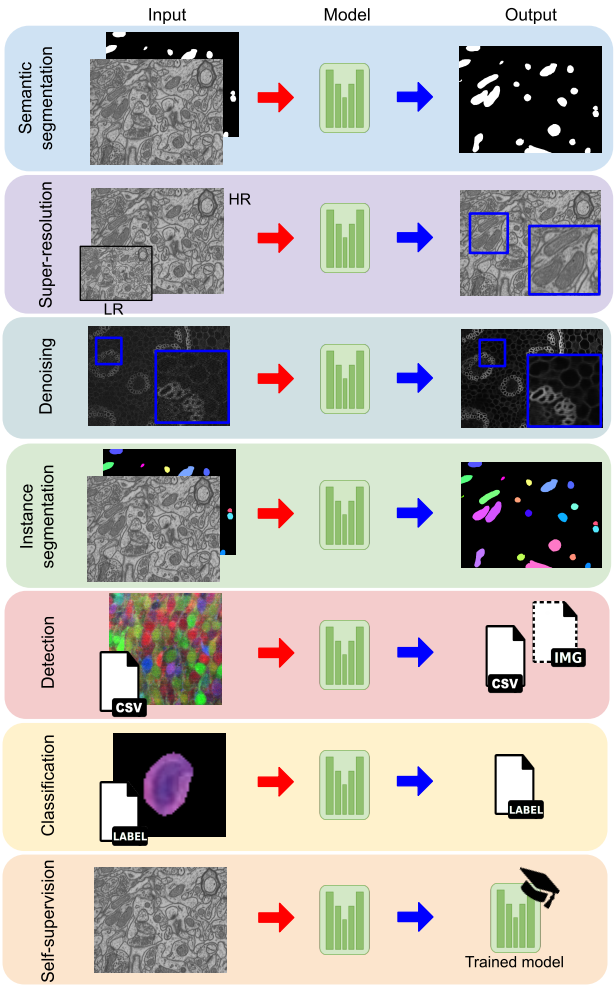

Fig. 3. Overview of the bioimage analysis workflows provided by BiaPy. The red and blue arrows represent pre-processing and post-processing steps, respectively. Blue squares represent zoomed patches of each image.

Object detection, that aims to localize objects in the input image, not requiring a pixel-level class. Common strategies produce bounding boxes containing the objects or individual points at their center of mass [22]. BiaPy adopts the later strategy to detect multi-class objects in 2D and 3D multi-channel images.

Denoising, that removes the noise from the input image. Our library includes Noise2Void [23] using any of the U-Net versions provided. The main advantage of Noise2Void is neither relying on noise image pairs nor clean target images since frequently clean images are simply unavailable.

Super-resolution, that aims at reconstructing high-resolution (HR) images from low-resolution (LR) ones. BiaPy offers pipelines to perform super-resolve LR 2D and 3D images by a factor of $\times 2$, $\times 3$, and $\times 4$.

Self-supervised learning (SSL), where the idea is to pretrain the backbone model by solving a so-called pretext task without labels. This way, the model learns a representation that can be later transferred to solve a downstream task in a labeled (but smaller) dataset.

Method	Class.	S./I. Seg.	Det.	Den.	SR	SSL	Trans.	All
Cellpose [9]	✓	✓	✗	✗	✗	✗	✗	✓
CellProfiler [10]	✓	✓	✓	✗	✗	✗	✗	✓
Ilastik [11]	✓	✓	✗	✗	✗	✗	✗	✓
PlantSeg [12]	✗	✓	✗	✗	✗	✗	✗	✓
ZeroCostDL4Mic [7]	✓	✓	✓	✓	✓	✗	✗	✗
BiaPy (ours)	✓	✓	✓	✓	✓	✓	✓	✓

Table 1. Differences between proposed toolbox (BiaPy) to address the gap between computer science knowledge and bioimage field and the SOTA platforms. Columns, from left to right, classification, semantic/instance segmentation, detection, denoising, super-resolution, self-supervised, transformers, and all-in-one.

Image classification, where full input images are labeled as belonging to a predefined set of classes. For completeness, BiaPy offers some popular image classification pipelines.

2.2. Input data

BiaPy is compatible with a wide range of image modalities. To name a few, it has already been proven useful for electron microscopy [19] and light microscopy [24], isotropic and anisotropic data. To this date, it can handle 2D/3D grayscale, RGB and multi-channel images.

2.3. Data augmentation

BiaPy integrates multiple data augmentation methods common to image classification tasks and adapts them to 3D and multi-channel microscopy images of variable resolution: cutout [25], cutblur [26], cutmix [27], cutnoise (similar to cutblur but adding noise) and gridMask [28], among others. Additionally, new data augmentation methods are provided, mimicking distortions produced during microscope image acquisition, such as misaligned or missing sections (i.e., z-slices). Moreover, BiaPy uses the imgaug library [4], which allows the inclusion of custom augmentations. Examples of data augmentations are depicted in Fig. 4.

2.4. Pre-processing

Before the training or inference stage, BiaPy prepares the input data, usually by creating 2D/3D image patches and normalizing the pixel/voxel intensities. Furthermore, some workflows require adapting the input data into a representation that can be fed to the backbone network. This is the case of the instance segmentation pipelines, where the label images are transformed into binary masks, contours, and distance maps, and also the object detection pipelines, where the point detection lists are transformed into point mask images.

[4] https://github.com/aleju/imgaug

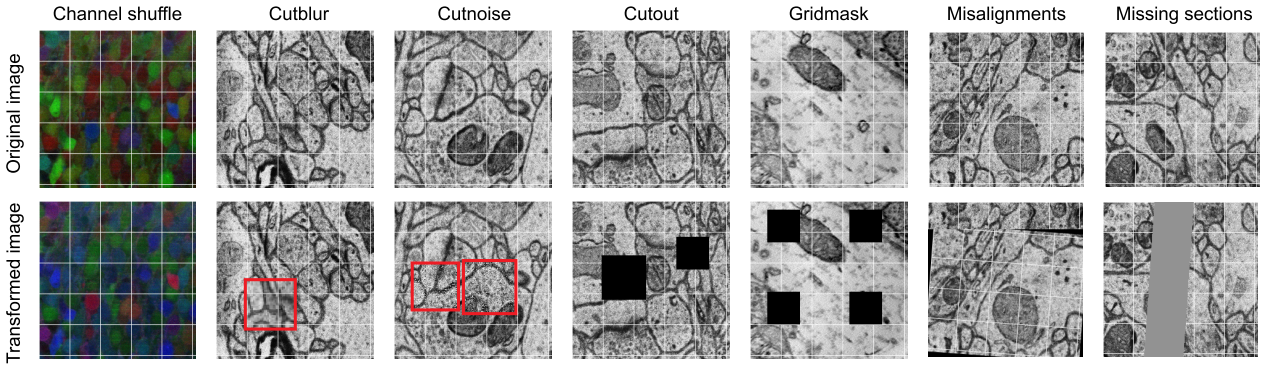

Fig. 4. Examples of data augmentation methods implemented in BiaPy. All methods accept 2D and 3D multi-channel images and take pixel/voxel resolution into consideration.

2.5. Post-processing

Most frequently, the output of the DL models needs to be post-processed to achieve the desired results. The post-processing methods implemented in BiaPy include simple binarization (for semantic segmentation), z-filtering[5] (for 3D data), marker-controlled watershed and Voronoi tessellation [24] (for instance segmentation), and close point suppression (for object detection), among others.

3. CONCLUSION

In this manuscript, we present BiaPy, a ready-to-use, all-in-one library for bioimage analysis using deep learning. BiaPy contains workflows defined by templates that serve to solve a wide variety of bioimaging analysis problems. Moreover, our library provides microscopy-specific data augmentation methods to help improve the performance of the implemented pipelines. This library attempts to address the gap between the computer science and bioimage analysis needs not covered by state-of-the-art Python-based DL toolboxes.

Acknowledgments. This work is supported in part by Ministerio de Ciencia, Innovación y Universidades, AEI, under grants PID2021-126701OB-I00, MCIN/AEI/10.13039/501100011033, PID2019-103900GB-I00 and PID2019-109820RB-I00, MCIN/AEI/ 10.13039/501100011033/, cofinanced by ERDF, "A way of making Europe", European Union - NextGenerationEU, and by grant GIU19/027 funded by the University of the Basque Country UPV/EHU. The authors declare no conflict of interest.

Compliance with Ethical Standards. This work is a study for which no ethical approval was required.

4. REFERENCES

[1] X. Chen, A. E. Gururaj, B. Ozyurt, R. Liu, E. Soysal, T. Cohen, F. Tiryaki, Y. Li, N. Zong, M. Jiang, et al., "Datamed–an open source discovery index for finding biomedical datasets," *Journal of the American Medical Informatics Association*, vol. 25, no. 3, pp. 300–308, 2018.

[2] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational intelligence and neuroscience*, vol. 2018, 2018.

[3] A. Esteva, K. Chou, S. Yeung, N. Naik, A. Madani, A. Mottaghi, Y. Liu, E. Topol, J. Dean, and R. Socher, "Deep learning-enabled medical computer vision," *NPJ digital medicine*, vol. 4, no. 1, pp. 1–9, 2021.

[4] R. F. Laine, I. Arganda-Carreras, R. Henriques, and G. Jacquemet, "Avoiding a replication crisis in deep-learning-based bioimage analysis," *Nature methods*, vol. 18, no. 10, pp. 1136–1144, 2021.

[5] J. Schindelin, I. Arganda-Carreras, E. Frise, V. Kaynig, M. Longair, T. Pietzsch, S. Preibisch, C. Rueden, S. Saalfeld, B. Schmid, et al., "Fiji: an open-source platform for biological-image analysis," *Nature methods*, vol. 9, no. 7, pp. 676–682, 2012.

[6] E. Gómez-de Mariscal, C. García-López-de Haro, W. Ouyang, L. Donati, E. Lundberg, M. Unser, A. Muñoz-Barrutia, and D. Sage, "Deepimagej: A user-friendly environment to run deep learning models in imagej," *Nature Methods*, vol. 18, no. 10, pp. 1192–1195, 2021.

[7] L. von Chamier, R. F. Laine, J. Jukkala, C. Spahn, D. Krentzel, E. Nehme, M. Lerche, S. Hernández-Pérez, P. K. Mattila, E. Karinou, et al., "Democratising deep learning for microscopy with zerocostdl4mic," *Nature communications*, vol. 12, no. 1, pp. 1–18, 2021.

[8] W. Ouyang, F. Mueller, M. Hjelmare, E. Lundberg, and C. Zimmer, "Imjoy: an open-source computational plat-

[5] median filter along the z-dimension

form for the deep learning era," *Nature methods*, vol. 16, no. 12, pp. 1199–1200, 2019.

[9] C. Stringer, T. Wang, M. Michaelos, and M. Pachitariu, "Cellpose: a generalist algorithm for cellular segmentation," *Nature methods*, vol. 18, no. 1, pp. 100–106, 2021.

[10] D. R. Stirling, M. J. Swain-Bowden, A. M. Lucas, A. E. Carpenter, B. A. Cimini, and A. Goodman, "Cellprofiler 4: improvements in speed, utility and usability," *BMC bioinformatics*, vol. 22, no. 1, pp. 1–11, 2021.

[11] S. Berg, D. Kutra, T. Kroeger, C. N. Straehle, B. X. Kausler, C. Haubold, M. Schiegg, J. Ales, T. Beier, M. Rudy, et al., "Ilastik: interactive machine learning for (bio) image analysis," *Nature methods*, vol. 16, no. 12, pp. 1226–1232, 2019.

[12] A. Wolny, L. Cerrone, A. Vijayan, R. Tofanelli, A. V. Barro, M. Louveaux, C. Wenzl, S. Strauss, D. Wilson-Sánchez, R. Lymbouridou, et al., "Accurate and versatile 3d segmentation of plant tissues at cellular resolution," *Elife*, vol. 9, pp. e57613, 2020.

[13] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[14] J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, L. Lu, A. L. Yuille, and Y. Zhou, "Transunet: Transformers make strong encoders for medical image segmentation," *arXiv preprint arXiv:2102.04306*, 2021.

[15] H.-Y. Zhou, Y. Guo, J.and Zhang, Le. Yu, L. Wang, and Y. Yu, "nnformer: Interleaved transformer for volumetric segmentation," *arXiv preprint arXiv:2109.03201*, 2021.

[16] A. Hatamizadeh, Y. Tang, V. Nath, D. Yang, A. Myronenko, B. Landman, H. R. Roth, and D. Xu, "Unetr: Transformers for 3d medical image segmentation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 574–584.

[17] Landman B, Xu Z, Iglesias J, Styner M, Langerak T, and Klein A, "Miccai multi-atlas labeling beyond the cranial vault–workshop and challenge," in *Proceedings of the MICCAI Multi-Atlas Labeling Beyond Cranial Vault—Workshop Challenge*, 2015.

[18] M. Antonelli, A. Reinke, S. Bakas, K. Farahani, A. Kopp-Schneider, B. A. Landman, G. Litjens, B. Menze, O. Ronneberger, R. M. Summers, et al., "The medical segmentation decathlon," *Nature communications*, vol. 13, no. 1, pp. 1–13, 2022.

[19] D. Franco-Barranco, A. Muñoz-Barrutia, and I. Arganda-Carreras, "Stable deep neural network architectures for mitochondria segmentation on electron microscopy volumes," *Neuroinformatics*, 2021.

[20] D. Franco-Barranco, J. Pastor-Tronch, A. González-Marfil, A. Muñoz-Barrutia, and I. Arganda-Carreras, "Deep learning based domain adaptation for mitochondria segmentation on em volumes," *Computer Methods and Programs in Biomedicine*, vol. 222, pp. 106949, 2022.

[21] D. Wei, Z. Lin, D. Franco-Barranco, N. Wendt, X. Liu, W. Yin, X. Huang, A. Gupta, W.-D. Jang, X. Wang, et al., "Mitoem dataset: large-scale 3d mitochondria instance segmentation from em images," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2020, pp. 66–76.

[22] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019.

[23] A. Krull, T.-O. Buchholz, and F. Jug, "Noise2void-learning denoising from single noisy images," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 2129–2137.

[24] P. Gómez-Gálvez, P. Vicente-Munuera, S. Anbari, A. Tagua, C. Gordillo-Vázquez, J. A. Andrés-San Román, D. Franco-Barranco, A. M. Palacios, A. Velasco, C. Capitán-Agudo, C. Grima, V. Annese, I. Arganda-Carreras, R. Robles, A. Márquez, J. Buceta, and L. M. Escudero, "A quantitative biophysical principle to explain the 3d cellular connectivity in curved epithelia," *Cell Systems*, vol. 13, no. 8, pp. 631–643.e8, 2022.

[25] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017.

[26] J. Yoo, N. Ahn, and K.-A. Sohn, "Rethinking data augmentation for image super-resolution: A comprehensive analysis and a new strategy," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8375–8384.

[27] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6023–6032.

[28] P. Chen, S. Liu, H. Zhao, and J. Jia, "Gridmask data augmentation," *arXiv preprint arXiv:2001.04086*, 2020.