THE UNIVERSITY OF CHICAGO

INVESTIGATING INFLAMMATION ON THE CELLULAR LEVEL USING MACHINE LEARNING

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE BIOLOGICAL SCIENCES
AND THE PRITZKER SCHOOL OF MEDICINE
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

COMMITTEE ON MEDICAL PHYSICS

BY
ADAM ROBERT SIBLEY

CHICAGO, ILLINOIS
DECEMBER 2019

To my mother and father

"An important feature of a learning machine is that its teacher will often be very largely ignorant of quite what is going on inside, although he may still be able to some extent to predict his pupil's behaviour. This should apply most strongly to the later education of a machine arising from a child-machine of well-tried design (or programme). This is in clear contrast with normal procedure when using a machine to do computations: one's object is then to have a clear mental picture of the state of the machine at each moment in the computation." – Alan Turing, 1950, Computing Machinery and Intelligence

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

2D	two dimensional
3D	three dimensional
AP	average precision
APC	antigen presenting cell
AR	average recall
AUROC	area under the receiver operating characteristic curve
CD	cluster of differentiation
CDM	cell distance mapping
CI	confidence interval
CMFDA	5-chloromethylfluorescein diacetate
CNN	convolutional neural network
COCO	Common Objects in Context
DAPI	4',6-diamidino-2-phenylindole
DC	dendritic cell
DCNN	deep convolutional neural network
DF	convolutional kernel dilation factor
DIC	differential interference contrast
FM	feature map
FOV	field of view
GPU	graphics processing unit
HPF	high power field
HTRC	University of Chicago Human Tissue Resource Center
IOU	intersection over union
LN	lupus nephritis
mDC	myeloid dendritic cell
MHC	major histocompatibility complex
mTOC	microtubule organizing center
PCC	pigeon cytochrome C peptide
pDC	plasmacytoid dendritic cell
R-CNN	region convolutional neural network
ROC	receiver operating characteristic
ROI	region of interest
SD	standard deviation
SLE	systemic lupus erythematosus
STED	stimulated emission depletion
SVM	support vector machine
Ta	antigen specific T cell
TCR	T cell receptor
TFH	T follicular helper
TII	tubulointerstitial inflammation
TNN	tuned neural network

TPEM	two photon excitation microscopy
Tw	wild type T cell

ACKNOWLEDGMENTS

I would like to thank Maryellen Giger, Marcus Clark, Vladimir Liarksi, Chun-Wai Chan, Madeleine Durkee, Junting Ai, Nicholas van Panhuys, Ronald N. Germain, Maria Merolle, Rebecca Abraham, and the GPMP program.

ABSTRACT

Linked recognition of antigen by T cells and antigen presenting cells (APCs) through direct cell-to-cell contact underlies most adaptive immune responses that both protect from infection and drive tissue damage in autoimmunity. To objectively assess this linked recognition of cells, our research studies the morphology and interaction of individual immune cells relative to inflammatory immune response. We utilize multi-channel immunofluorescence imaging (cellular confocal microscopy) in presentations of lupus nephritis incorporating nuclear stains and cell membrane immunofluorescent stains with deep convolutional neural networks (DCNNs) to classify multiple T cell and dendritic cell types.

Specific to this dissertation, we investigate and develop computerized single-cell segmentation and characterization in cellular confocal microscopy images of lupus nephritis with the goal of quantifying interaction of immune cells to characterize immune response. The feasibility of localizing, segmenting, and classifying individual cells in multi-channel confocal microscopy images is highly dependent on image quality. In certain applications with good image quality and well-separated cells, segmentation can be trivial and accomplished via thresholding, watershed, or a collection of other well-established and studied heuristics; however, at the limit of poor image quality, densely packed cells, and complex image features, these techniques fail. It is at this limit that deep learning approaches excel. Using deep learning region-based segmentation techniques, we enable analysis of challenging image data that previously required laborious hand segmentation, based on multichannel information difficult for the human visual system to parse.

We analyze cellular images from confocal microscopy of (a) fresh frozen tissue in a mouse model of T cell activation by dendritic cells and (b) fresh frozen tissue in kidney biopsies patients with lupus nephritis (c) paraffin embedded kidney biopsies from lupus nephritis patients. Our goal is to segment cells in order to quantify the interaction of immune cells to characterize in situ adaptive immunity. The main challenge of this work is refining segmen-

tation of dense cells in a tissue medium, which is also applicable to general segmentation challenges of histopathological images.

In murine fresh frozen images we achieve a segmentation performance with intersection over union (IOU) of 0.85, sensitivity of 0.88, and specificity of 0.92 across cell types. For human fresh frozen images an IOU of 0.70, sensitivity of 0.72, and specificity of 0.86 is achieved across cell types. Using DCNN segmentation and classification nuclear segmentation output, an area under the receiver operating characteristic curve (AUROC) of 0.82 is achieved for discriminating cell types in murine fresh frozen data and an AUROC of 0.64 is achieved for discriminating cell types in human fresh frozen data.

Overall, we show that by utilizing multi-channel confocal microscopy and our novel DCNN pipeline, we achieve performances similar to two photon excitation microscopy (TPEM) in discriminating stable cognate from non-cognate T cell–dendritic cell interactions in mice and apply it also to human tissue. These data indicate that a quantitative analysis of many static two-dimensional images can approximate much of the information obtained from time-lapse three-dimensional videos of the same phenomenon. Additionally, since our analysis is performed on single-plane images of fixed tissue, we could use it to study human disease and identify important in situ APCs.

CHAPTER 1

INTRODUCTION

The interplay between T cells and APCs has been best studied in murine models in which intravital TPEM has provided a dynamic and quantitative picture of how various cells of the immune system interact to evoke a coordinated immune response in secondary lymphoid organs [1–3]. Although TPEM provides a paradigm for how human adaptive immune responses might proceed, it does not lend itself to the direct study of immunity in humans.

In comparison to TPEM, the techniques usually applied to the study of human tissue are largely non-dynamic and qualitative. Conventional histology can be used to describe tissue morphology, and immunohistochemistry can be used to assess the relative frequencies of individual cell types. However, unlike TPEM, these techniques cannot be used to understand and quantify the spatial or functional relationships between different T cell and APC populations. Functional relationships are often inferred from ex vivo studies of similar peripheral cell populations [4]. However, these studies can only demonstrate that the selected populations of APCs and T cells can respond to antigens under certain experimental conditions. They do not necessarily predict in inflamed tissue at the site of organ destruction.

An earlier cell distance mapping (CDM) method was proposed by Peng et al., 2012 [5] and in Liarski et al., 2014 [6], to quantify spatial relationships between different cell subtypes in tissue. Specifically, they conducted analysis of inflamed human tissues and showed that measurement of internuclear distances between T follicular helper (TFH) cells and B cells could be used to discriminate between apparent cognate and noncognate interactions. They used membrane staining in combination with its respective 4',6-diamidino-2-phenylindole (DAPI) correlate to definitively assign nuclei to specific cell types. All B cell nuclei were then used to construct a distance map in which each pixel value represented the Euclidean distance from the pixel to the closest B cell nucleus edge. T cell nuclei were then overlaid, and they were able to accurately compute minimum distances between the edges of each

B cell nucleus to the edge of the closest T cell nucleus in pixel increments. Inspired by their findings, we aimed to extend CDM to include deep learning region-based segmentation techniques to enable analysis of challenging image data, such as densely-packed cell regions.

The significance of our research is that if it is possible to infer the dynamics of the adaptive cell networks underlying inflammation, then a more mechanistic and quantitative classification of several apparently heterogeneous diseases, such as systemic lupus erythematosus (SLE), could be conducted. Such capabilities would both enhance our understanding of disease pathogenesis and suggest disease-specific therapeutic opportunity [6].

1.1 Innovation

While the study of adaptive immunity is an active area of research with the goal of linked recognition of antigen by T cells and APCs through direct cell-to-cell contact, little progress has been made in objectively conducting such analyses for assessing human inflammatory immune responses. The innovation in our research lies in our machine learning methods to objectively assess this linked recognition of cells. Our research studies the morphology and interaction of individual immune cells, first in a murine model and then in actual human tissue. Also, while use of multi-channel immunofluorescence imaging (cellular confocal microscopy) is a known technique, we uniquely incorporate images obtained with up to 6 stains. Our deep learning segmentation and machine learning characterization methods incorporate these various nuclear stains and cell membrane immunofluorescent stains to automatically classify multiple T cell and dendritic cell types. We believe that, ultimately, our machine learning platforms will handle segmentation challenges in other histopathological images.

1.2 Outline of Thesis

Chapter 2 will review relevant immunology, imaging, segmentation, and analysis techniques. Chapter 3 discusses development of deep convolutional neural networks. Chapter 4 presents the development of a deep segmentation technique for murine fresh frozen tissue and analysis of the resulting data. Chapter 5 presents the development of a deep segmentation technique to human fresh frozen tissue and analysis of the resulting data. Chapter 6 discusses use of Mask R-CNN for segmentation on human fresh frozen and paraffin data. Chapter 7 summarizes the research and gives overall conclusions.

1.3 Technical Notes

An effort was made in the main body of this document to preserve images as their original resolution and embed them with lossless compression where possible within reasonable document memory constraints. Image data included in the main body of the document should render at original image resolution in most PDF viewers without interpolation except where noted. Image label data were included at full image resolution with a few exceptions for very large image matrices beyond the range supported by either the PDF standard or the LaTeX type-setting software used to prepare this PDF document. Images in the appendices were downsampled where noted so that document size could be kept reasonable while still including segmentation results from the different sections. For high resolution figures in the main body of the document, zooming in the PDF viewer may be appropriate.

Appendices are referenced throughout the document and as they are quite long it is recommended to navigate them using the PDF bookmarks and hyperlinking. For best electronic viewing of this document the free software Adobe Acrobat Reader is recommended. In particular figures in the appendices that take up two pages are best viewed with the side by side page setting. Hyperlinking and document navigation also work best in Adobe Acrobat

Reader. Most references to content including figures, sections, references, page numbers, and acronyms are hyperlinked throughout the document.

CHAPTER 2

IMMUNOLOGY AND IMAGING

This chapter covers the biological motivation for developing computational machine learning tools to segment cellular images and background on traditional image segmentation techniques.

2.1 Two Photon Excitation Microscopy

Adaptive immunity depends upon both antigen-restricted cell–cell interactions and environmental niches, which enable and coordinate cellular communication. In mice, TPEM has revolutionized our understanding of immune cell architectures and their contribution to normal immunity. By visualizing cells and structures in live hosts, TPEM provides, as shown in the region of interest (ROI) in figure 2.1, both a quantitative and dynamic picture of immune processes [7–11].

Although it's the gold standard for understanding the organization of immunity, TPEM has several limitations. Cells must be fluorescently labeled to be visualized [12, 13], and, therefore, manipulated systems must be used [14]. Only small volumes of tissue can be assessed, and this must be done over sufficient time to capture cellular dynamics. These restraints limit the number of measurements that can be practically obtained using TPEM. Furthermore, only tissue that can be exposed in live mice is generally amenable to TPEM. Whereas TPEM has a maximal effective depth of 1.6 mm [15], most applications are limited to less than 500 µm. Therefore, immune processes occurring within the interior of some organs cannot be visualized. Finally, with few exceptions [16, 17], TPEM cannot be used to directly study human disease [18]).

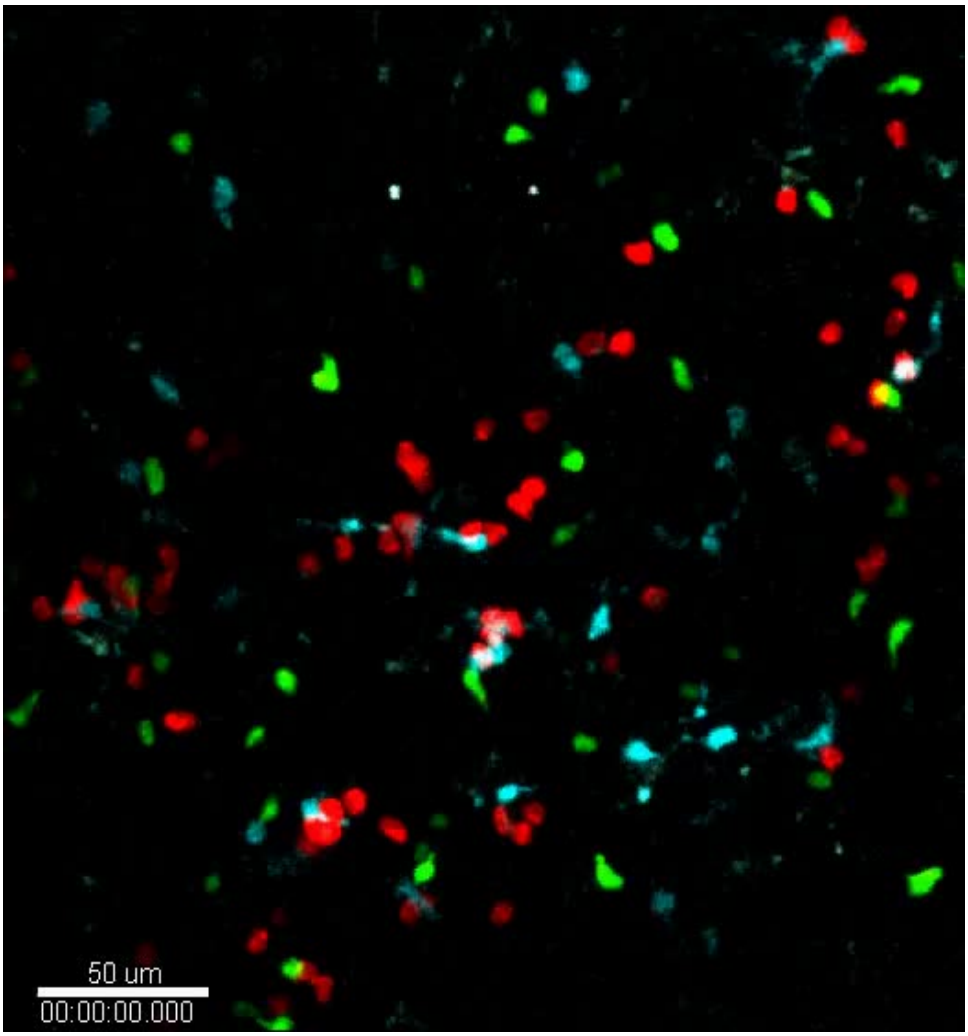

Figure 2.1: **Two photon microscopy example** Two photon microscopy ROI of antigen-specific T cells (red), wild-type T cells (green) and dendritic cells (grey). Viewable as video in electronic version in Adobe Acrobat Reader. (from Liarski and Sibley et al., 2019) (TPEM video acquired by Ronald N Germain and Nicholas van Panhuys)

2.2 Immunofluorescent Confocal Microscopy

Confocal microscopy involves multi-channel imaging of tissue samples that have been stained to enhance the presence of specific cellular and nuclear structures. Confocal microscopy is widely used in many biological science disciplines, from cell biology and genetics to microbiology and developmental biology [19]. Tissue is stained with immunofluorescent antibodies specific to the type of immune cells one aims to identify in both murine and human tissue sections.

Great strides have been made in multiparameter imaging of fixed human tissue such that 36 or more markers can be assayed simultaneously [20–23]. With these and other approaches [24, 25], one can identify infiltrating cell subsets and describe their relative regional behaviors. Such studies have revealed that the cellular constituency of inflammation is very complex [24, 26] and the organization of immune cells can be characteristic of disease states [21] and define prognosis [22]. However, it is difficult to know why different cell populations appear together. Cells such as T cells and APCs can engage in cognate interactions that drive local adaptive immunity and inflammation [6, 27]. Alternatively, cells can just be bystanders of inflammation with different populations coalescing because they are responding to similar environmental cues such as chemokines [28]. There are limited tools to discriminate between these states in human tissue [18].

2.3 Cell Distance Mapping

CDM is a technique to quantitatively assess the distance between T cell nuclei and dendritic cells and how their shapes change relative to one another, which serves as a surrogate of their interaction and potential participation in immune processes [6, 18]. Figure 2.2 shows conceptually how cell distance mapping is done. During the interaction process the T cell reorients itself relative to the dendritic cell surface. This changes the shape of the cell

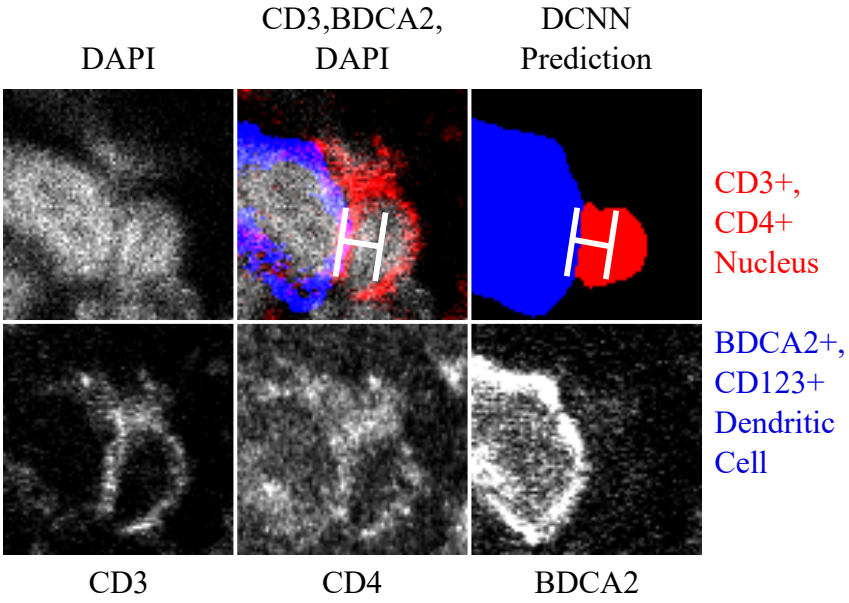

Figure 2.2: **Cell distance mapping diagram:** From left to right, top to bottom, DAPI nuclear stain, 3 color superposition, DCNN segmentation, CD3, CD4, and BDCA2. White lines show conceptually how distance is measured from nuclear center to dendritic cell surface.

membrane as well as the nucleus. This allows us to distinguish between interacting and non-interacting T cells based on their shape. It also allows us to assess T cell and dendritic cell interactivity based solely on the proximity of the whole population of T cells to dendritic cells in tissue. The primary challenge of cell distance mapping is extracting shape information from noisy, multi-channel images of cell populations in dense tissue media.

Previously, we demonstrated that quantitative analysis of human frozen tissue samples, imaged using multicolor confocal microscopy, could be used to characterize interactions between follicular helper T cell populations and B cells [6]. In these investigations, we observed that when follicular helper T cells formed cognate interactions with B cells, their nuclei became tightly apposed. These data indicate that distances between nuclear borders can discriminate between cognate interactions and when T cells and B cells are merely in proximity. Therefore, by mapping relative distances between T cells and B cells in tissue (CDM), we could identify functional relationships [18].

Figure 2.3 is an example of a prior marker-controlled watershed approach to CDM used in

Liarski et al. 2014 [6]. While the techniques used in this approach gave a general indication of cell populations highlighted by staining, they did not localize individual cells well, lacked robustness to noise, and used thresholding techniques which often failed or had to be fine-tuned manually.

The fixed filters and algorithms used in CDM to segment signals within tissue were insufficient for defining positions of larger complex objects such as stains associated with dendritic cells (DCs). Furthermore, CDM did not accurately capture object shape. We postulated that this might be important, as T cells adopt different shapes when scanning for antigen and after recognizing peptides in the context of the major histocompatibility complex (MHC) [29–35]. In the latter case, T cells flatten against the APC to form a stable synapse. In contrast, T cells scanning for antigen or those engaged in brief antigen-specific interactions (kinapses), do not undergo the same changes in T cell shape and polarity [36]. We hypothesized that using computational tools that accurately captured T cell shape features and DC boundaries, we could identify stable synapses and, thus, discriminate between cognate and non-cognate T cell–APC interactions in human tissue.

Therefore, we implemented, as will be detailed throughout this thesis, a deep convolutional neural network (DCNN) that precisely measured both cell position and shape. The DCNN output was then analyzed with a tuned neural network (TNN) to identify the combination of distance and cell shape features that best discriminated between different T cell populations relative to DCs. The use of a TNN allowed us to restrict our analysis to fundamental morphological features of T cell–DC interactions conserved in mice and humans [29–35]. We refer to this analysis pipeline as CDM version 3 (CDM_3). Herein, we demonstrate that in both mice and humans, CDM_3 can discriminate between in situ cognate and non-cognate T cell–DC interactions [37].

This new methodology for CDM, which we recently published (Liarski and Sibley et al., 2019 [18]), is depicted in figure 2.4. It consists of a single custom deep convolutional neural

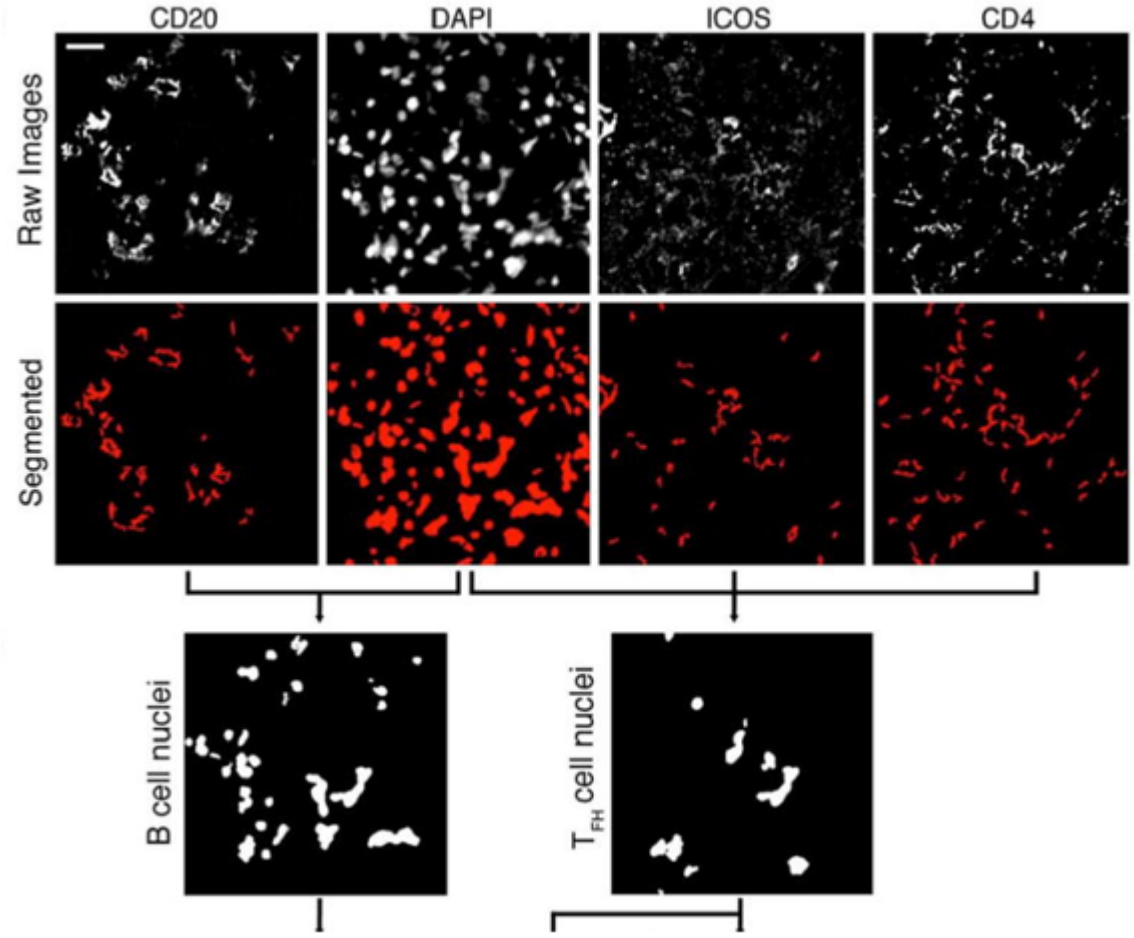

Figure 2.3: **Marker-controlled watershed example** from previous work (Liarski et al. 2014 [6]) using marker-controlled watershed for cell segmentation. (Figure copied from version prepared by Yahui Peng)

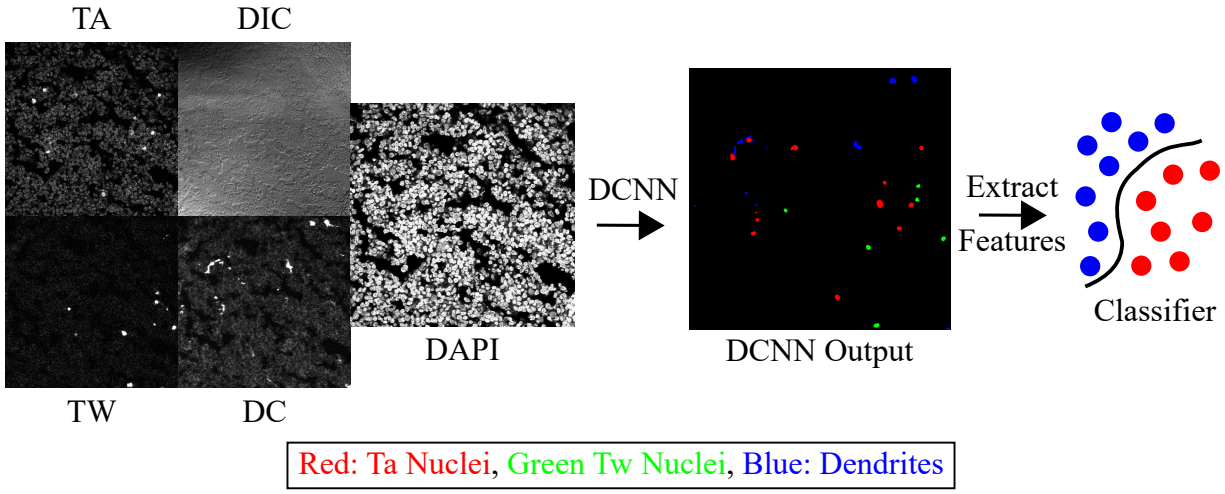

Figure 2.4: **Proposed deep learning pipeline for cell classification:** Multiple images channels are passed through a DCNN which segments and classifies cell types. Segmentations are then post-processed and analytical shape features are extracted for unsupervised classification and analysis.

network which outputs segmentations for multiple cell-types based on the input staining. These segmentations of different classes of cells can then be used to interpret the distribution and shape of different cell types within the tissue. Shape features including perimeter, convex perimeter, area, convex area, circularity, minor axis length, major axis length, eccentricity, equivalent diameter, solidity, major/minor axis ratio, and perimeter to circularity ratio can be computed for each cell. In addition, the distances between all cell types can be computed [18].

2.4 Image Segmentation and Classification

Image segmentation refers to partitioning of an image into different parts, while image classification refers to classifying image data into a set of different classes. Image segmentation can be viewed as a special case of many general data classification algorithms, where the classification operates in the two or three-dimensional spatial domain characteristic of most data commonly referred to as imagery. The challenge the spatial domain introduces is significant. The complexity of this problem was underestimated from the very beginning of computer

vision techniques and is only recently being addressed satisfactorily. Many established techniques exist for image segmentation including thresholding, clustering, region-growing, watershed, variational methods, graph-partitioning methods, model-based segmentation, and others. These techniques often perform well within favorable regimes of image quality, signal strength, and task. However, in recent years they are largely being displaced by methods based on deep convolutional neural networks.

Figures 2.5, 2.6, and 2.7 show some of the difficulties with using traditional image segmentation techniques based on thresholding and mathematical morphology with our images. In figure 2.5 we demonstrate how Otsu thresholding [38] separates the intensity distribution of the image into two classes based on the Otsu method's algorithm of minimizing intra-class variance; however, the fundamental assumption of this thresholding process is that the image can be divided into foreground and background based on the intensity histogram. This assumption fails in figure 2.5 because there are some other bright types of noise in the image we want to ignore. This is relatively easy for a human observer to notice, as the bright spots in the thresholded image that don't overlap with the manually labeled image look different. This is an issue with most thresholding techniques and highlights an important point: interpreting an image as a histogram of intensities is a vast oversimplification. Despite the simplicity of interpreting an image as an intensity histogram, it is still a very valuable technique. This is particularly true when it is possible to quantitatively relate the intensity in an image to whatever you are interested in. Unfortunately this is often very difficult to accomplish in real life imaging situations.

In figure 2.6 we demonstrate selection of a manual intensity threshold on the CD3 (where CD refers to cluster of differentiation in immunophenotyping [39]) image channel such that it selects the $CD3^+$ cell membranes for the nuclei we have already segmented in the image. This highlights a few problems in using traditional image segmentation techniques to process this image. If we were to use marker controlled watershed to process this image, we could use the

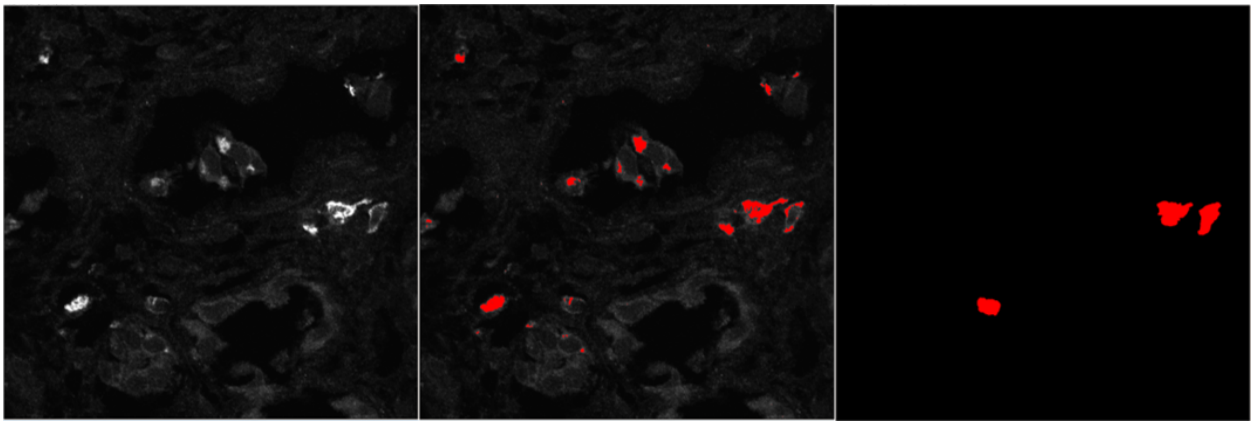

Figure 2.5: **Otsu Threshholding example** on CD123 channel vs. manual labeling in Human data

segmentations from the CD3 channel as markers for the nuclear channel. The nuclei could also be segmented through thresholding. We run into problems when we try to determine with which nuclei the $CD3^+$ segmentations overlap with. When nuclei are crowded in close proximity, they are often difficult to resolve on their own. When we add to that problem the task of assigning membrane segmentations to them, it gets more difficult. In addition, we can see that the CD3 channel is thresholded in the same location as the $BDCA2^+CD123^+$ cell. This is due to either bleed through from fluorescence in one channel to another or non-perfect specificity of the immunofluorescent stain.

In figure 2.7 we show a significant amount of background in the image that Otsu's method interprets as foreground due to the uptake of staining agent by tubules in the kidney. While we can do better with a manual threshold, it is still difficult to completely remove background from the image.

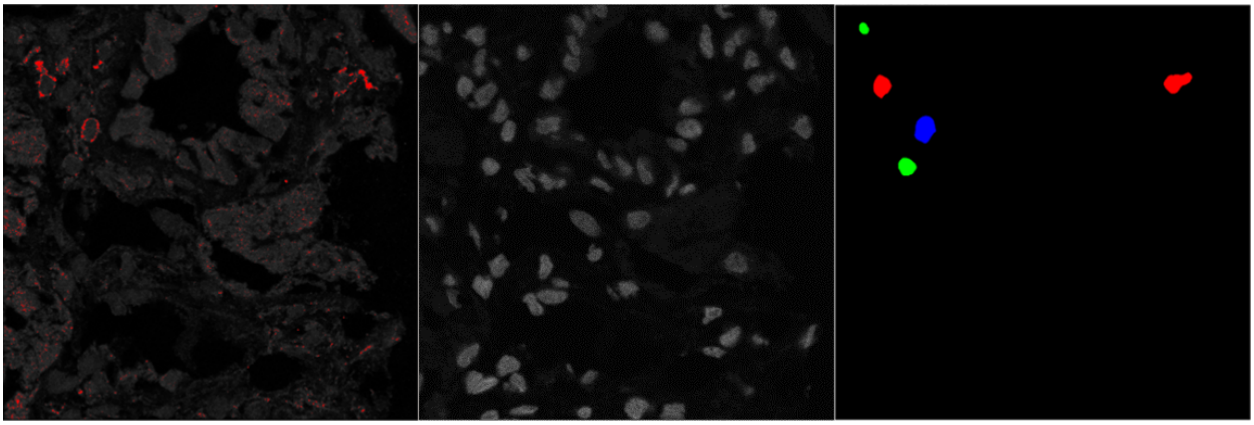

Figure 2.6: **Manual thresholding example** on CD3 channel vs manually labeled in Human data $CD3^{+}CD4^{+}$ (red) $CD3^{+}CD4^{-}$ (green) $BDCA2^{+}CD123^{+}$ (blue).

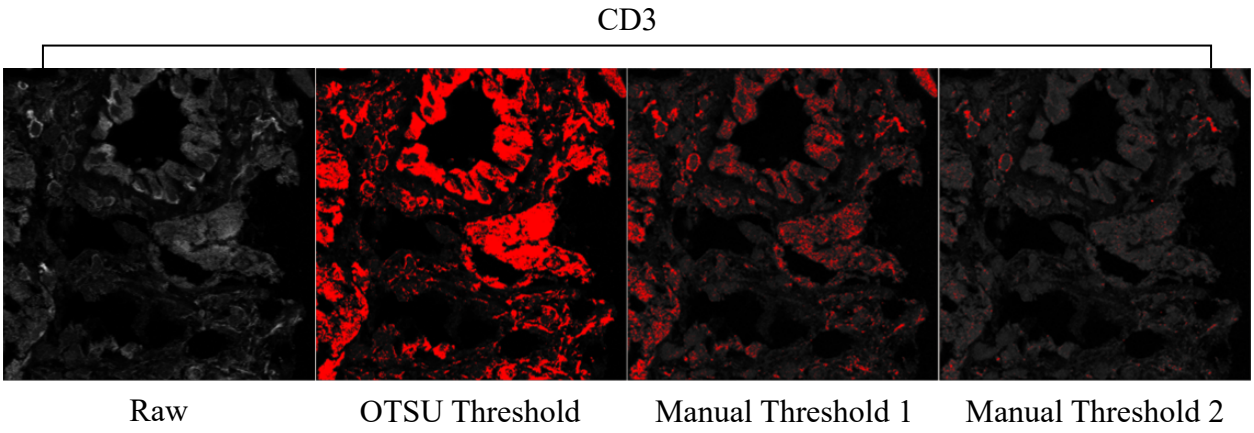

Figure 2.7: **Otsu vs. manual threshold example** on CD3 channel is shown.

CHAPTER 3

DCNN DEVELOPMENT

This chapter covers the details that went into constructing our DCNN including the theoretical motivation and the specifics of the design.

3.1 Background

We began construction of our convolutional neural network (CNN) with the requirements that it accurately reproduced cell shape, accurately classified cell type, and produced an interpretable output. Accurately reproducing cell shape is often a problem for CNNs because they use downsampling and upsampling layers which blur the boundaries of objects. Accurately classifying cell type with a CNN in our imagery is difficult because of image noise, channel bleedthrough, and the fundamental differences between multi-channel fluorescent imagery and the photographic imagery on which most CNN architectures have been based. Interpretability of output is one of the most difficult problems with CNNs [40].

For these reasons, we settled on a paradigm where a supervised deep convolutional neural network produced classifications and segmentations for multiple cell types in an image. The shape features of the segmentation masks and the distances between cells would be used for feature analysis and classification. This second feature analysis and classification step was intended to show that the shape features and distance relationships were different in the different cell types.

The restriction of our second classification step to only shape features and distance relationships between different cell types is justified as our initial goal was to detect these shape changes and distance relationships, which have been shown in in vivo TPEM data. The restriction also serves the important purpose of removing uninterpretable deep-learning based features from the classification process. While deep learning based features can be

very powerful and feature extraction has also been shown to be a very useful tool, they also have a high potential to overfit to data and give misleading results. Overfitting can be ruled out by proper validation and testing, but there are many problems that can occur here as well.

This approach to the problem presented many challenges. Producing accurate segmentation masks based on cell membrane stains and cell nuclear stains that are crowded very close in tissue is difficult. In addition, our images are just two dimensional (2D) slices through tissue that do not capture the complete three dimensional (3D) shape of the cell, only a cross section.

3.2 DCNN Structure

The first important principle when designing DCNNs is to make them only as complex as they need to be for the target problem. Most importantly, the number of parameters (alternatively the weights of the convolutional kernels) should be restricted to the complexity of the problem being addressed. There have been efforts to computationally optimize the size and architecture of DCNNs for specific problems [41], but the most common solutions to design remain trial and error and referring to DCNNs that have already been applied to specific problems. In designing our DCNN, we were firstly inspired by the ZNN [42, 43] deep convolutional neural for segmenting electron microscopy images of Drosophila neurons. ZNN leveraged an image database of approximately several hundred images with image matrix size similar to our own. ZNN was also concerned with producing accurate segmentations at the pixel level, without resolution degradation in the DCNN output. Finally the neurons in the images ZNN was designed for were very dense and often contained ambiguous information. That is the neurons in the images were sometimes at the limit of what was resolvable by human inspection and likely often at the limit of information contained in the image. Some of the early segmentation experiments that motivated this thesis were conducted in the software

program provided by the ZNN authors to the computer vision community. Motivated by ZNN, we designed our own deep convolutional neural network with 10 convolutional layers, 3 maximum pooling layers, and approximately 700,000 trainable parameters.

To design a DCNN appropriate for our problem, we decided that the image input resolution should match the image output resolution of the DCNN. Equal size input and output images were necessary because we wished to perform shape-based analysis on the output segmentation results of the DCNN and also because we wanted the segmentation results to be as accurate as possible. Many DCNN implementations do not produce equal size input and output images and many produce only abstract, high dimensional features that cannot be interpreted as image data. This is done because these high dimensional features are powerful for classification but they are not ideal for segmentation. Converting an image to a hierarchical representation of complex machine learning features is what DCNNs are great at, but this paradigm runs counter to the needs of segmenting and classifying lots of individual objects in an image. Using a DCNN formulation where an entire image is converted into a collection of high dimensional features, one of our $1024 \times 1024 \times 5$ cellular images with hundreds of distinct cells is represented as a single object. This might be fine if the questions you are asking of the image are on the level of the entire image. Questions such as: "Is this image from a region of a kidney with high inflammation or not". However, in our case we are interested in the activity of individual cells in an image and there are many cell types. Encoding an entire image as a single object would mean that the activity of individual cells and specific subsets of cells would become difficult to query. It also leads to a combinatoric complexity problem, where each cell type being considered in the image can classify an image into a unique subset. For example, if we want to ask the question "Are there any A, B, C, or D cells in this image.", the total possible image types are: no cells, A, AB, ABC, AC, etc. for a total number of 2^n or 16 image types. In addition to the combinatoric complexity problem of image type subsets, a large image matrix of $1024 \times 1024 \times 5$ is more complex for

a DCNN to analyze. It is also unnecessary when the objects of interest are approximately $100 \times 100 \times 5$ in the image matrix.

While it is theoretically possible to implement our cellular image analysis despite combinatoric complexity considerations of image labels or large image matrix size, it is entirely unfeasible from a training data perspective. To train the DCNN in such a way would likely require acquisition and labeling of a prohibitive number images to produce satisfactory results compared to the approximately hundreds of images used in this thesis.

The problem of the way in which a DCNN represents an image as a whole without consideration for its constituent parts remains one of the foremost problems in computer vision. It has been most notably recognized and addressed in the idea of capsule neural networks advanced by Geoff Hinton [44]. This method introduces a localization mechanism into the DCNN architecture, but is still an in-development idea. The most common practical solution to this problem is to introduce an object localization network [45] which first evaluates a large number of candidate regions in an image for whether they possess an object of interest or not. Objects of interest are then passed to a DCNN to be processed as a single object inside a bounding box within the image. We implement and discuss a DCNN of this type in chapter 6. However, implementing a DCNN with an object localization network does have drawbacks. The classification of objects during candidate evaluation by the object localization network can be brittle compared to the full DCNN classification architecture. Also, objects in close proximity can become confused and overlapping bounding boxes can cause problems. In addition, bounding boxes can potentially crop out parts of objects that are important or exclude from consideration the area around objects. While this area around objects might not be part of the object itself, it can still be useful for classification.

Another approach to solving the object localization problem is to apply convolutions across the entire image, while restricting the field of view (FOV) of applied convolutions such that the information that goes into classifying the object is restricted to the FOV. This is the

method that we employ in the DCNN architecture discussed in this section and applied in chapter 4 and chapter 5. The concept of expanding field of view with increasing convolutional layers is illustrated in figure 3.1 where each subsequent convolutional layer incorporates more surrounding image information hierarchically. This method has the downside of lacking a specific localization method for cells that identifies them as unique objects; however, a properly trained DCNN should be able to compensate for this problem. The reader can see this is the case by considering a DCNN with a field of view centered around a single cell. If this field of view is wider than the size of the cell, the whole cell is in consideration of the classification machinery of the DCNN. The DCNN can then exclude parts of other cells beyond it's consideration since it possesses information about the entire cell and its surroundings. In practice, borders between can indeed be indistinct and localization can suffer. In fact one of the primary weakness of this method is joining together adjacent cells when it is not appropriate; however, the ability of the DCNN to localize and separate cells without a specific localization method in this application is still quite robust.

Since the field of view in a DCNN is the result of the application of specific sized convolutional kernels across all convolutional layers in the network, we chose to use a field of view that was on the order of the size of an individual cell ($85 \times 85 \times 5$ for the murine data) by adjusting the sparsity of convolutions in the DCNN based on the convolutional layer. Sparsity of the convolutions throughout the network was adjusted such that earlier convolutional layers had smaller kernels and later convolutional layers had larger (or sparser) kernels.

There is a distinction between a large kernel and sparse kernel in a CNN. In a large kernel of size 7×7 all values are considered in a 7×7 square (49 values total). In a 3×3 sparse kernel of the same 7×7 size, only 9 values are considered. This is achieved by spreading the values in the kernel out across the image matrix. A 3×3 kernel would skip two values on either side of the center pixel to reach a total size of 7×7. The size of a dilated kernel is determined by equation 3.1 where $\hat{k}$ is the effective dilated kernel size, d is the convolutional

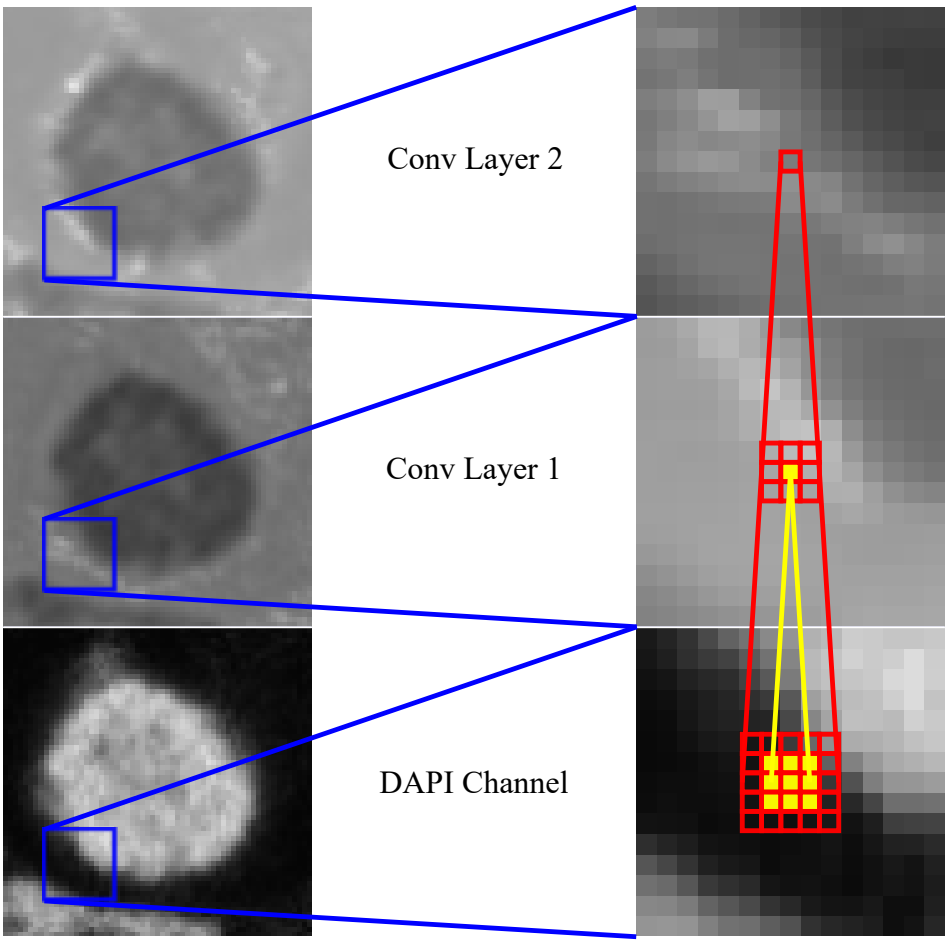

Figure 3.1: **Receptive field illustration** showing how lower convolutional layers combine features from higher convolutional layers throughout the DCNN.

kernel dilation factor (DF), and k is the original kernel size. Final chosen DFs for the DCNN are shown in figure 3.3.

$$\hat{k} = k + \frac{k-1}{d-1} \tag{3.1}$$

While dilated convolutional kernels are used to capture larger scale features in an image, a more common approach is to reduce the dimension size of the image matrix (typically by a factor of 2) in the maximum pooling layers within the DCNN. While this approach is effective for classification, it may not be ideal for segmentation. By reducing the size of the image matrix, small scale features are lost and segmentations aren't as precise. Upsampling

of the image and deconvolutional layers are often used to overcome this problem, but these operations are known to reduce the precision of the boundaries in the segmentation outputs of a DCNN. Therefore, in this DCNN we choose to use dilated convolutions instead of upsampling and deconvolutional layers. The DCNN implementation discussed in chapter 6 does include versions of these operations.

Regarding the convolutional kernels in our DCNN, we also chose to scan $3 \times 3 \times 3$ sized kernels across the $1024 \times 1024 \times 5$ image matrix, treating the channels dimension as an extra spatial dimension, in a certain sense. In many applications of DCNNs, the channel dimension of the image is treated as fully connected when applying convolutional kernels. For example, a photographic image has 3 color channels for an image matrix size of $x \times y \times 3$. DCNNs designed to analyze photographic images typically scan $3 \times 3 \times 3$ convolutional kernels across the image in the x and y dimensions, but they do not scan across the channel dimension. This is related to the principle of translation invariance in DCNNs. Based on translation invariance the assumption made that features computed from convolutions at one x y position in an image are useful at other areas in an image. This makes sense in photographic images because the color channel contains a different type of information than the spatial channel. We choose to scan convolutional kernels across the channel dimension in our application because there are important variations between the channels in the image. For example, many cells are only present in one channel in the image. Also the stains outlining the borders of cell membranes in some channels are very different than the DAPI stain that highlights nuclei and the differential interference contrast (DIC) channel.

Scanning a $3 \times 3 \times 3$ convolutional kernel across the spatial dimensions and the channel dimension also reduces the size of the convolutional kernel from $3 \times 3 \times 5$ which reduces the complexity of training the DCNN. Also, finding common feature interactions between channel dimensions is likely more computationally efficient than finding rigid features restricted to specific channels that are aligned with the weights in the convolutional kernel.

We theorize that finding features localized to smaller regions within the channel dimension of the DCNN may help compensate for noise in the image and help the DCNN to train more effectively. Application of a single $3 \times 3 \times 3$ convolutional kernel at a single point within the $1024 \times 1024 \times 6$ image matrix for the human data is shown in figure 3.2. In this figure the weights in the $3 \times 3 \times 3$ convolutional kernel transform information from 3 adjacent image channels into a single prediction in the output feature map (FM).

The final architecture of our DCNN applied in chapter 4 and chapter 5 is shown in figure 3.3 from two perspectives. Following the 10 convolutional layers the DCNN features are integrated in a fully connected layer which integrates the 600 final features maps into 4 class probability maps for the different cell types.

3.3 Labeling for DCNN

Manual labeling was conducted by using Icy Bio Image Analysis Suite along with ImageJ and Fiji. The Icy software makes syncing axes between multiple image channels easy as shown in figure 3.4. It also has flexible image annotation options with easily interpretable .xml metadata and native integration with ImageJ. This makes it easy to manipulate the data and incorporate it into image processing paradigms in a variety of programming languages and software platforms. These features of the software were essential to making multichannel annotation of our images feasible. Using the wrong tools, the laborious manual segmentation we did on these images would take far longer and manipulation of the segmented data more difficult, if not infeasible. For our immunofluorescent confocal microscopy images, truth was hand-segmented by volunteers including medical students, undergraduates, and summer students. For hand segmentation, it was important to both accurately segment and classify cells. All truth was validated by Dr. Vladimir Liarski by carefully reviewing hand segmentations and correcting outlines and classifications. Prior to manual segmentation image channels were rigidly registered using the 'multimodel' configuration of the imregister

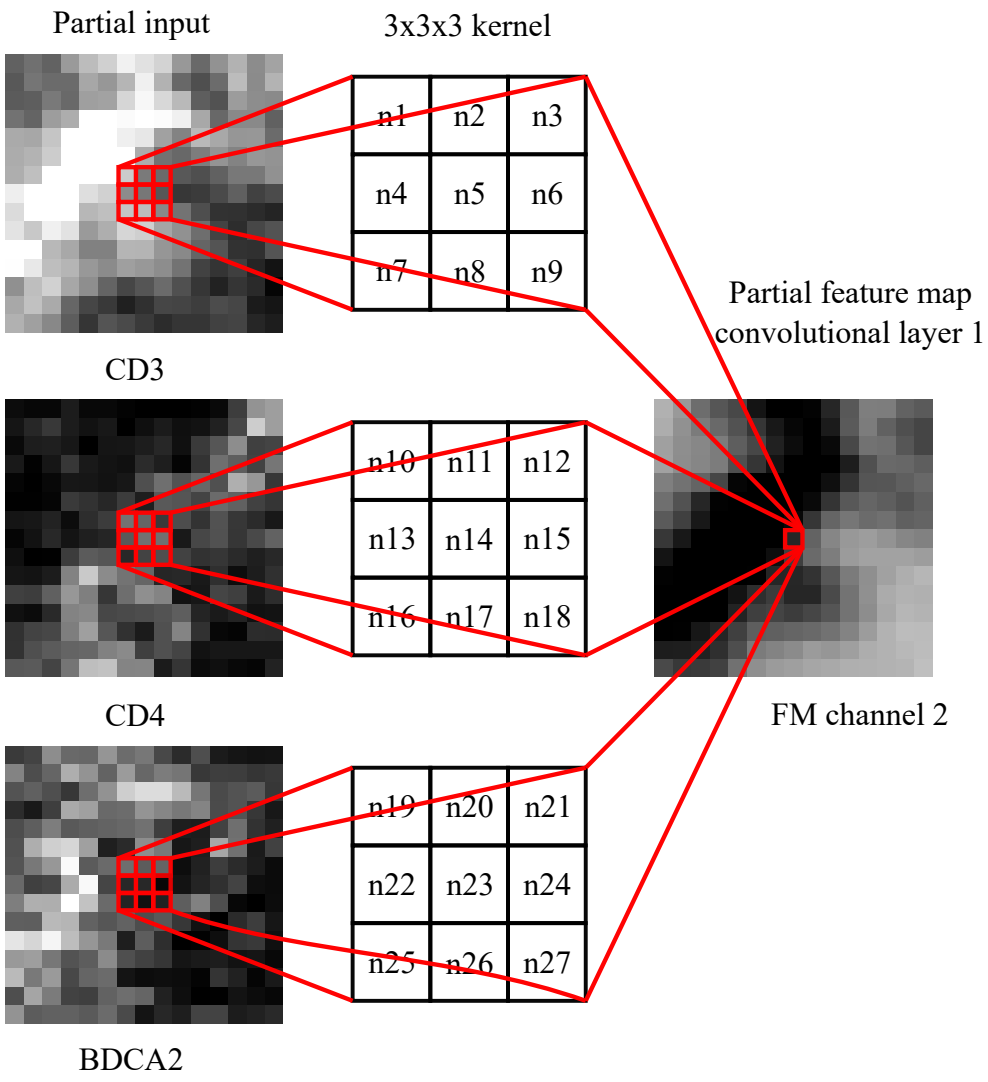

Figure 3.2: **Multi-dimensional convolutional kernel illustration:** The way in which the $3 \times 3 \times 3$ convolutional kernel is applied across multiple channel dimensions is shown for three select channels. Information is combined from multiple channels into a single point in the output feature map for each application of the convolutional kernel. The weights of the convolutional kernel are multiplied by the pixel values and summed together. A rectified linear unit (RELU) activation function [46] of the form $f(x) = max(0, x)$ is applied to this value to produce the final feature map.

function in Matlab [47] for human image data. This was necessary for human image data due to suspected vibration misaligning the microscope stage during the image acquisition process.

While labeling for murine data chapter 4 and human data in chapter 5 were conducted in

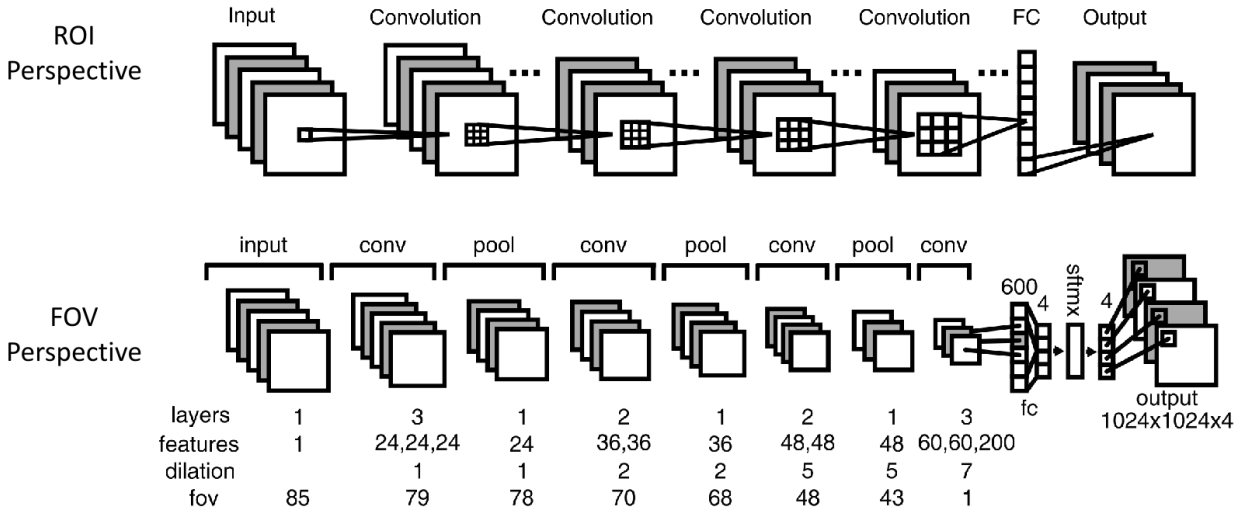

Figure 3.3: **Diagram of DCNN architecture** shown from both a region of interest and field of view perspective. The ROI perspective shows how the total image matrix size remains constant throughout the DCNN, while the sparsity of the convolutional kernel increases. The FOV perspective shows the number of feature maps, number of convolutional layers in each section, the dilation factor of the convolution kernel sparsity, and the FOV that passes on to subsequent layers in the DCNN.

Icy. Further image segmentation for project discussed in chapter 6 was implemented using image overlays in ImageJ. This turned out to be more flexible than annotating in Icy and was achieved by embedding overlays in the image .tif metadata that ImageJ can interpret natively. In addition these image overlays can be retrieved from an .tif image file with python and rewritten to the image file metadata. This was achieved using a modification of tifffile.py [48] implemented in [49].

3.4 Training of DCNN

The z-score (standard score, equation 3.2) of the ROIs was taken for every individual channel within every image channel independently before training. In equation 3.2, x stands for the value of an individual pixel, μ is the mean value of all pixels within a single channel in a single roi, and σ is the standard deviation (SD) of all pixels within a single channel in

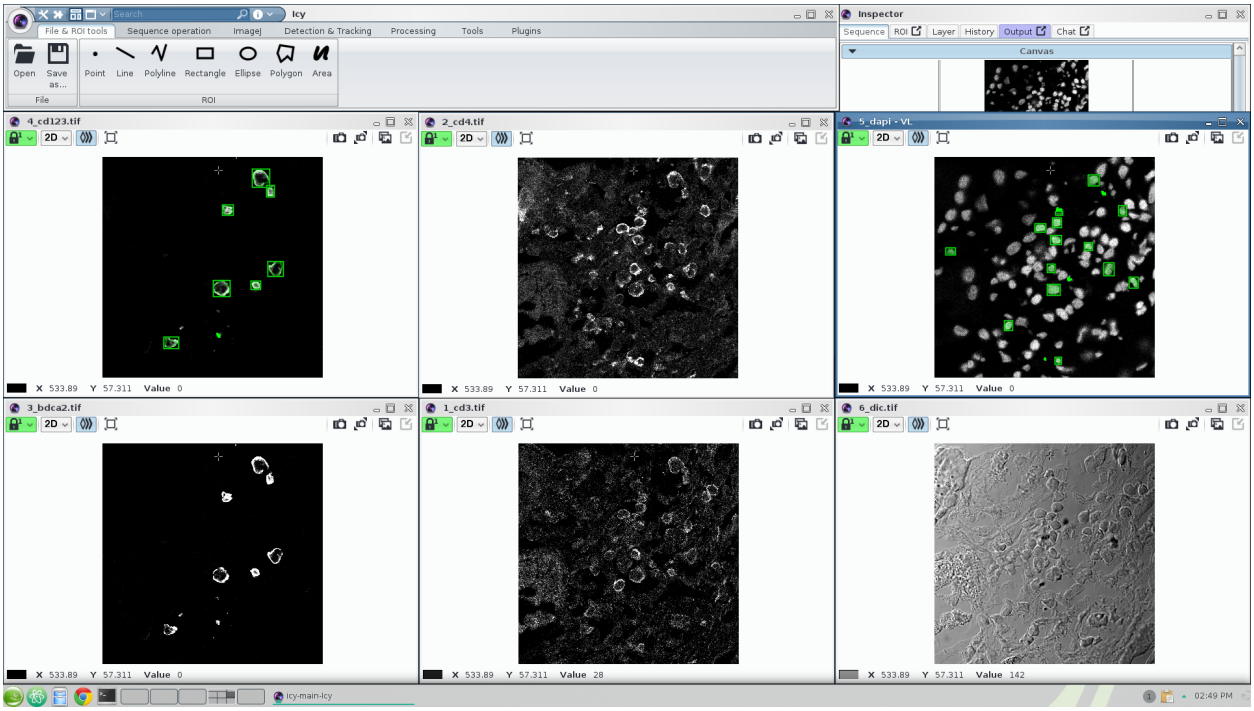

Figure 3.4: **Icy interface screenshot** capturing manual labeling process in ICY Bio Image Analysis software.

a single roi. All training and inference was performed on z-score transformed ROIs. The DCNN was trained with a batch size of four $184 \times 184 \times 6$ image patches distributed across four Tesla graphics processing units (GPUs) with 12 GB memory per card, system memory of 128 GB with two Intel E5-2680v4 CPUs at 2.4 GHz on the University of Chicago Research Computing Center GPU2 nodes on Midway2 [50]. Image patches were sampled from the entire labeled dataset. Each of the four patches for a training iteration represented one of the four classes to be segmented. Class membership of a patch was determined by the class of the center pixel. Where this 184×184 patch around pixels extended beyond the ROI border, mirror padding was used. Sampling was implemented in Tensorflow [51] using a list of all pixel locations by class, stored in system memory.

$$\text{z-score} = \frac{x - \mu}{\sigma} \tag{3.2}$$

The image, label, and data for each ROI were stored in binary TFrecord format in 32-bit float format and accessed by Tensorflow queue runners for active data augmentation (rotation and mirroring) of training examples onto a queue. Error was computed for all classes within the patch. Softmax binomial cross-entropy was used to compute neural network error over an output image patch of 100×100 pixels, reduced input size of 184×184. This dense output was important because it reduces training redundancy and increases training stability. All convolutional and pooling layers were allowed to shrink the input by filter overlap at each layer in x and y, known as valid padding. Convolutional layers were padded in the channel dimension to keep channel dimensions the same. All weights were initialized with Xavier initialization [52], and all biases were initialized at zero. Gradients were averaged across GPUs for each variable at each iteration. Stochastic gradient descent was used for optimization, with a learning rate of 0.001 and no decay of the learning rate. The DCNN was trained for $200,000$ iterations at which point cross-entropy error was stable and small (figure 3.5).

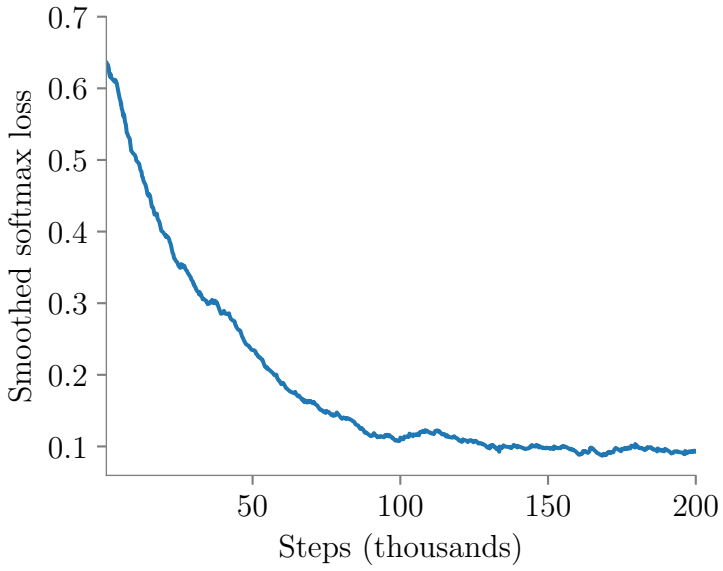

Figure 3.5: **Smoothed softmax cross entroy error** as a function of training iterations. Error is smoothed since the error is highly variable for each batch of 4 training example patches. Smoothing algorithm is presented in figure 3.6.

```
def smooth(scalars, weight):
    last = scalars[0]
    smoothed = list()
    for point in scalars:
        smoothed_val = last * weight + (1 - weight) * point
        smoothed.append(smoothed_val)
        last = smoothed_val
    return smoothed
```

Figure 3.6: **Python algorithm for smoothed softmax loss:** Weight of 0.993 used in figure 3.5. This is the smoothing algorithm used by the Tensorflow Tensorboard (6.1) visualization tool for time-series performance data.

In the post-processing step, each pixel is assigned the class with the maximum predicted probability. The post-processing step can also include removing objects with anomalous shapes and dropping segmentations with low predicted probability.

Due to GPU memory limitations during training, we were limited to processing a patch of the entire image at a time. For the murine data we chose an input patch size to the DCNN of $184 \times 184 \times 5$ from a ROI with matrix size $1024 \times 1024 \times 5$. If images are broken into patches for processing by a DCNN naively, information can be lost. This is because of the way in which the DCNN field of view overlaps with the border of the an image. For example, if we extract a patch of $184 \times 184 \times 5$ from a large two dimensional image and pass it through a DCNN, the image is going to shrink because of the field of view. This effect is illustrated in figure 3.7. This happens because there is no information beyond the border of the image, so the image shrinks by kernel size - 1 for each convolution. This problem can be addressed by padding the edge of the image with zeros or a mirrored version of the border during every convolution operation. This is a fine solution if we don't have information about what lies beyond the border of an image or if the information that lies beyond the border of an image is irrelevant to classification. However, in the context of extracting a small image patch from a larger image and passing it through a DCNN, padding before convolutions with false data is a bad solution. What we do instead in our DCNN is extract a patch from our ROI and allow the patch to shrink throughout the DCNN. In addition, these patches are chosen to

be overlapping such that the whole image is processed in patches without introducing any false data, except for at the borders of the actual larger ROI of course. In fact, using this method the result from processing the image in patches in this way is identical to the result of processing the whole ROI simultaneously.

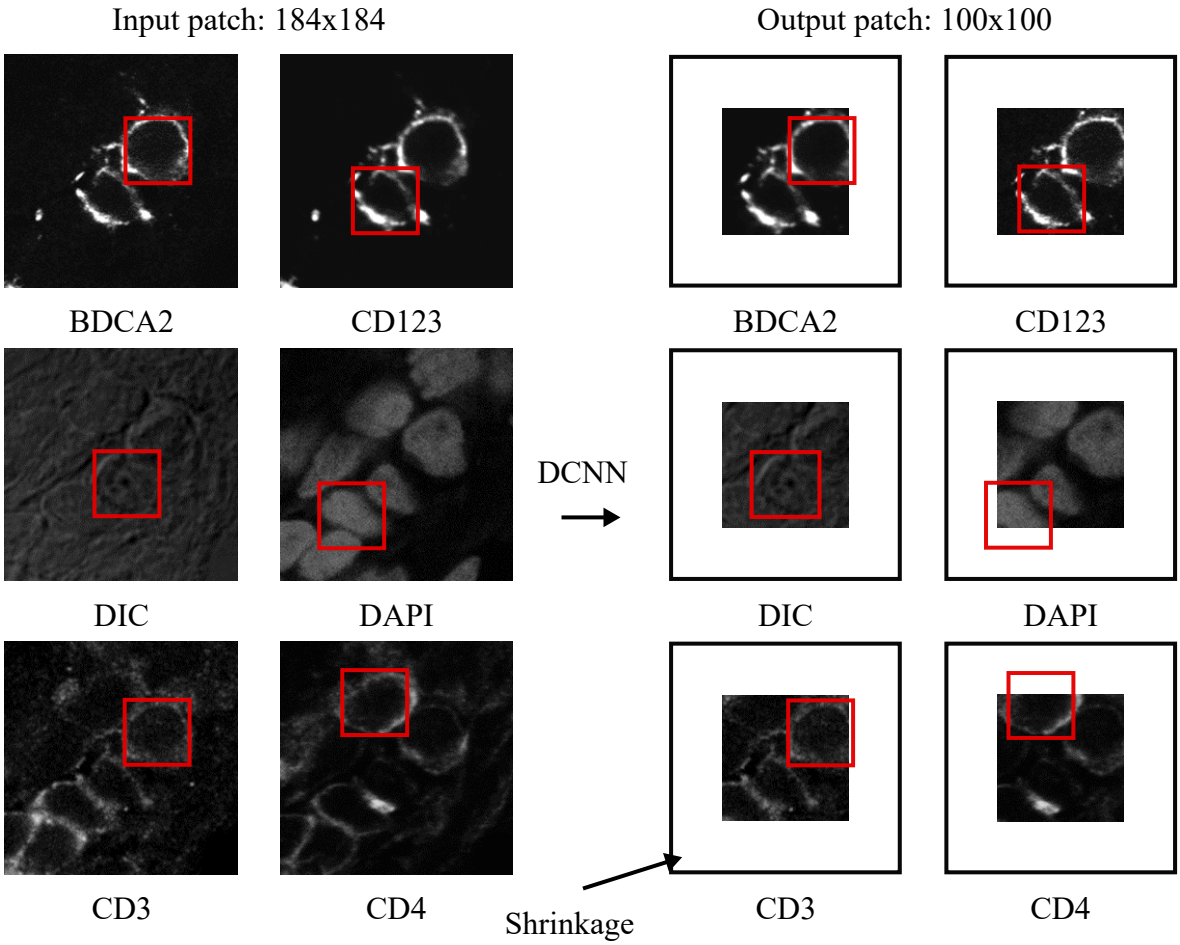

Figure 3.7: **Shrinking field of view implications:** The field of view shrinks as the image matrix moves through the DCNN. This happens because filters at the edge of the matrix reduce the size of the total matrix. If identical size output is desired in the output of the DCNN, the image matrix can be padded with zeros or a mirrored version of the image edge to eliminate shrinkage. Otherwise, the image matrix can be allowed to shrink throughout the DCNN.

The primary concern during training was to limit overfitting of the DCNN to the training images and train it to generalize well to new images. Progress of training was judged both with the performance on a held-out test set (typically 20% of data) and also generalization

of the DCNN to unlabeled images, judged by visual inspection.

3.5 Testing

Prediction with the DCNN was performed on $1024 \times 1024 \times 5$ matrix size ROIs with boundary mirroring to fit the field of view. The final segmentation result is taken by assigning each pixel the label of the class with the max predicted probability from the softmax output. Segmentation performance was assessed using 5-fold cross validation, in which a full DCNN model is trained on 4/5 folds of the dataset and tested on the fifth fold, for all five folds. We assessed the sensitivity and specificity of our cell detection and computed the mean intersection over union, defined in equation 3.3, for each class of detected cells. All object analysis in this work was conducted using scikit-image [53] and custom code. For the purposes of computing sensitivity and specificity, a cell was considered to be detected if the IOU with the truth, based on manual segmentation, was ≥ 0.5. Feature maps for a trained DCNN with outputs are shown in figure 3.8.

$$IOU = \frac{|A \cap B|}{|A \cup B|} \tag{3.3}$$

3.6 DCNN Feature Classification Algorithms

Following segmentation by the DCNN, minimum and mean minimum distances in between cells or nuclei of interest, the convex and regular areas, circularity and eccentricity, convex and regular perimeters, equivalent diameter, major and minor axis lengths, aspect ratio, pixel size, solidity, perimeter/circularity ratio, and probability of belonging to the designated class for each object were computed for detected $CD3^{+}CD4^{+}$ and $CD3^{+}CD4^{-}$ T cells (figure 3.9). Additionally, for every T cell the minimum distance to the nearest DC was computed in two ways: the first was simply the minimum Euclidean distance found between all the pixels in

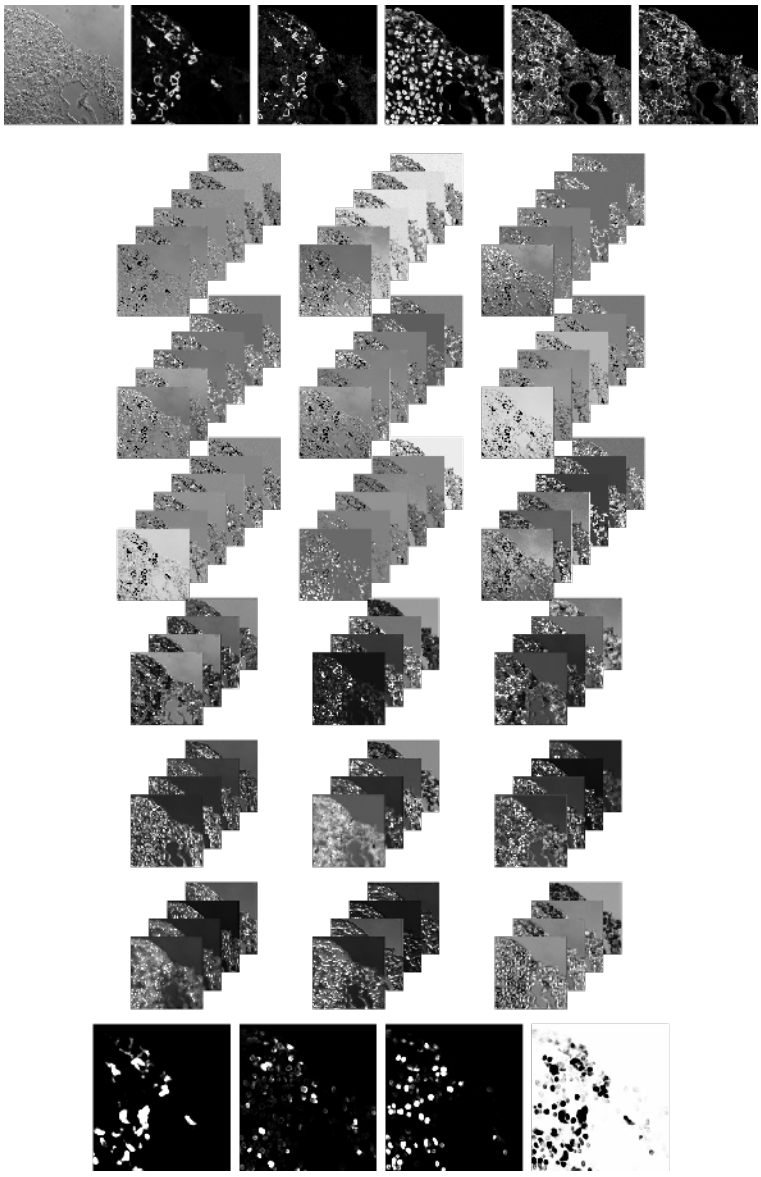

Figure 3.8: **Visualized feature map layers for DCNN** with some of the middle layers omitted. Input image data shown at top and output softmax predictions for multiple cell types shown at bottom.

the T cell and the nearest DC pixel; an additional mean minimum distance was computed by averaging the distances from all pixels in the T cell to the nearest dendrite.

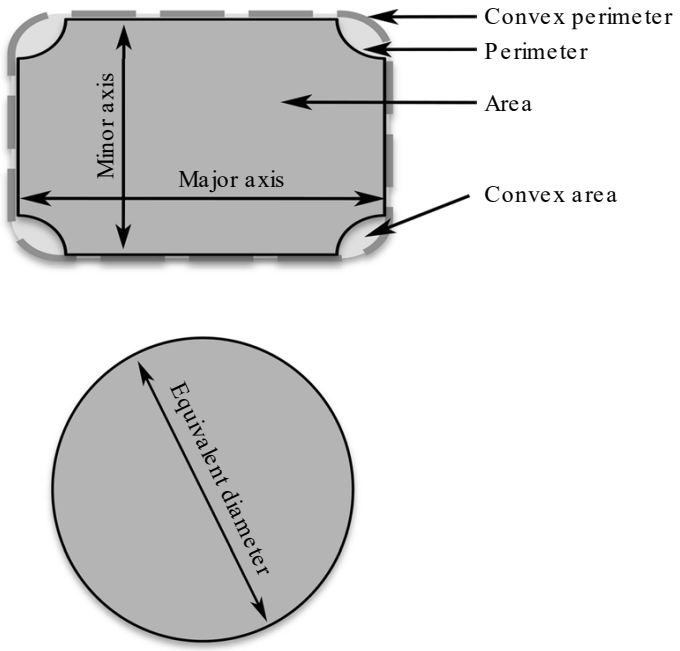

Figure 3.9: **Independent cell shape parameters:** Convex measurements can be thought of as applying an imaginary rubber band around an object (dashed gray line) and are important in identifying local concave shape changes. The equivalent diameter represents the diameter of a circle, possessing an identical area as that of a non-circular object. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

Original CDM analysis based on minimum distance [6] and preliminary TPEM was performed with GraphPad Prism 5.0a software for Mac (GraphPad Software). All subsequent dataset analysis was performed using R statistical software (version 3.4.1, The R Foundation for Statistical Computing) and RStudio (version 1.0.153, RStudio), running on MacOS 10.13.1 High Sierra, powered by a 3.5 GHz 6-core Intel Xeon E5 CPU with 32 GB of RAM. Mouse adoptive transfer and lupus nephritis data were subjected to multivariate logistic regression, support vector machine (SVM), random forest, and neural network analyses. The following R packages were used for modeling: randomforest (ver 4.6-14), e1071 (1.7-0), xgboost (0.71.2), rpart (4-1.13) and rpart.plot (3.0.4), neuralnet (1.33), glmnet (2.0-16). The caret package was utilized when multiple instances or tunes were required to be generated

and compared. Each instance of SVM, neural network, and random forest model generation was preceded by explicitly invoking a specific kernel seed to allow for result reproducibility. Modeling was performed on log-transformed and normalized input data with a binary outcome variable, representing classification (either wild-type or 5CC7, or $CD3^{+}CD4^{+}$ or $CD3^{+}CD4^{-}$ cells). To control for incomplete or incorrect segmentation, all objects with areas < 3 and ≥ 100 square pixels were removed. Test and train datasets were defined as random 1/3:2/3 selections of input data. All n-fold cross-validations were performed with n=5. SVM analysis was subjected to linear, radial, and sigmoid kernels when comparing among models. Random forest analysis was used to define the relative importance of predictors as follows: a default of 500 trees were generated for each datapoint using the randomforest package, and the optimal Cp parameter was selected based on minimum square error optimization. The resultant RF plots were visualized, and the relative importance of each split versus mean decrease in accuracy was recorded. The features across each experiment were compared to determine predictor hierarchy. Results of logistic regression were cross-referenced to ensure data consistency and agreement. ROCR and pROC packages were used to generate AUROC curves and confidence interval on all included ROC plot figures with the following parameters: 10,000 bootstrap replicates, stratification, curve smoothing, and a confidence interval alpha of 0.90, corresponding to a type I error of 0.05. P values for comparison of neural network algorithm output with the null hypothesis (AUROC= 0.5) were obtained with the verification R package using the roc.area function and are separate from the indicated confidence intervals. Mann-Whitney U test (unpaired Wilcoxson rank sum test) was utilized whenever group comparisons were performed. A Benjamini-Hochberg correction was applied for large datasets exceeding 5,000 observations with a false discovery rate set to 5 %, which was applicable for human lupus nephritis analysis.

After segregating the data based on minimum distance cutoffs, neural network models (figure 3.10) were generated using the following independent predictors: convex and regular

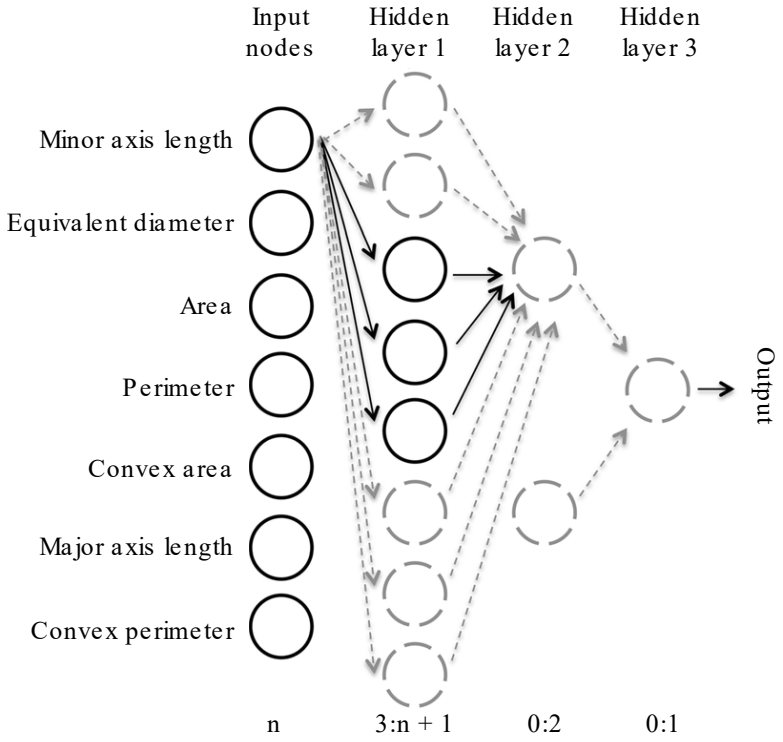

Figure 3.10: **Outline of classification neural network model** used to analyze lymphocyte distance and cell shape data. After segregation by minimum distance, the indicated seven measures of cell shape were scaled, normalized, and used as input into three neural network models (simple, tuned, and linear output) in R statistical software, as described in Methods. The arrows and circles indicate data flow from a single input node throughout the network (applied weights omitted). The maximum potential number of hidden layers and nodes, used for active tuning, is indicated by dashed gray circles, with data flows in between steps denoted by dashed gray lines. Dark circles and lines denote obtained optimal parameters used at the completion of network tuning. n denotes the number of input nodes. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

areas, convex and regular perimeters, equivalent diameter, and major and minor axis lengths. Three different models of neural networks were generated, and their performances were compared: a single-layer model with n+1 nodes in the hidden layer, where n is the number of predictors; an actively tuned model, with 3:(n–1) nodes in the first hidden layer, with the addition of 0:2 additional hidden layers, consisting of a maximum of 2 and 1 additional hidden nodes, respectively; and a linear output neural network model with the specification of no hidden layers, constant weights, and linear output. In order to avoid result skewing due to unbalanced numbers of cell types in relation to distance, each instance of neural network

analysis was performed by taking the minimum number of cells (wild-type and 5CC7 cells for mouse, and $CD3^{+}CD4^{+}$ and $CD3^{+}CD4^{-}$ cells for lupus nephritis) and randomly sampling an equal number of cells from the second population. Every network analysis performed was specified with a threshold of 0.1, stepmax of 1×10^{8}, default (logistic) activation function, cross-entropy error differentiable function, and otherwise default package parameters for learning rate, starting weights, and number of repetitions. When possible, the performance of each algorithm was compared based on the parameters of total error, classification accuracy, and receiver operator curve performance of correctly predicting cell type [18].

CHAPTER 4

MURINE FRESH FROZEN TISSUE

In this chapter we discuss the application of our DCNN segmentation and cell-type classification analysis to murine fresh-frozen data.

4.1 Introduction

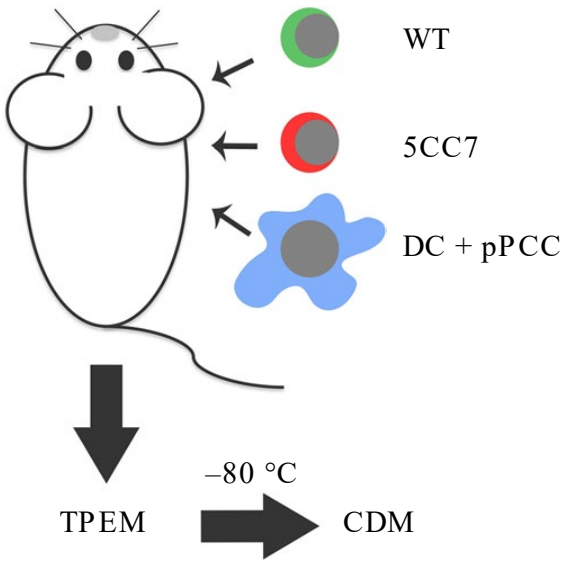

Figure 4.1: **Illustration of Differences between CDM and TPEM:** Indicated T cells (wild-type (WT) or 5CC7) and antigen-pulsed pigeon cytochrome C peptide DCs were transferred into B10.A2 CD45.2^{-} mice and, after 12 h, popliteal lymph nodes were first imaged using TPEM, frozen, and then imaged by confocal microscopy for CDM3. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

To develop better computational tools to study immunity in fixed tissue, we first used an established murine model of T cell and dendritic cell interactions [37]. Briefly, three cell populations were prepared (figure 4.1): CD11c^{+} DCs from CD45.1^{+} mice stimulated in vitro with lipopolysaccharide, loaded with pigeon cytochrome C peptide (PCC) peptide (10 µM), and labeled with the fluorescent dye CMF2HC (blue); polyclonal CD45.1^{+} CD4^{+} T cells (wild-type) labeled with 5-chloromethylfluorescein diacetate (5-chloromethylfluorescein

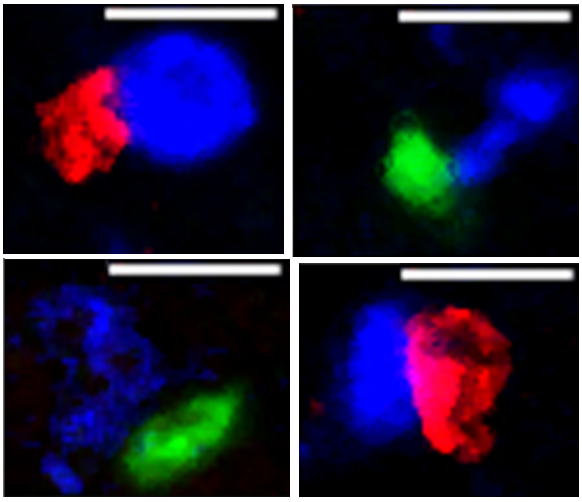

Figure 4.2: **Confocal microscopy cell interaction closeup:** Confocal microscopy examples of WT cells (green) interacting with DCs (blue) and Examples of 5CC7 cells (red) interacting with DCs (blue). Scale bars, 10 µm. $n = 3$ independent experiments. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

diacetate (CMFDA),green); and T cell receptor (TCR)-transgenic 5CC7 CD45.2$^+$ CD4$^+$ T cells labeled with CMTPX (red). Previous studies have identified a peptide recognition rate of 0.1 % to 0.3 % for wild-type cells in this model system [54, 55]. Cells were then transferred into wild-type recipient mice and, after 12 h, popliteal lymph nodes were imaged using TPEM by Nicholas van Panhuys and Ronald N. Germain. These same lymph nodes were then frozen, sectioned, stained for cell nuclei with TOPRO-3, and imaged using confocal microscopy by Junting Ai. TPEM revealed clear differences in the behavior of 5CC7 antigen-specific and wild-type T cells relative to antigen-pulsed DCs (figure 2.1). 5CC7 cells are alternatively referred to as antigen specific T cell (Ta) cells in this document. Wild-type T cells are alternatively referred to as Tw. Many of the PCC-specific 5CC7 T cells engaged in prolonged interactions with DCs, whereas wild-type T cells were more motile and only interacted briefly. Furthermore, wild-type T cells were spherical when interacting with DCs, whereas 5CC7 T cells tended to flatten against DCs (figure 4.2).

Quantitative analysis by Vladimir Liarski of TPEM data (supplementary table B.1) revealed that measures of cellular motion discriminated between wild-type and 5CC7 T cells

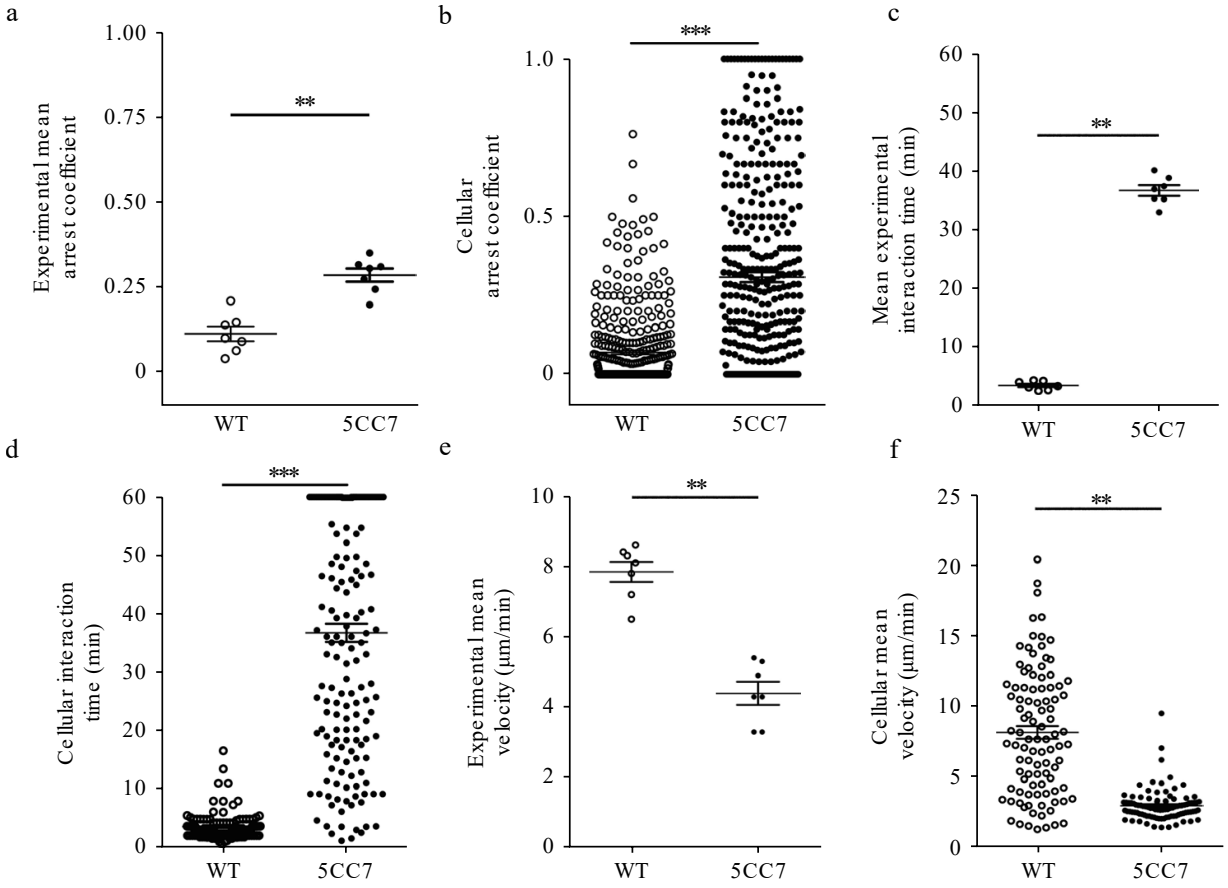

Figure 4.3: **TPEM data: a**,**b**, Arrest coefficient plotted either as mean per mouse ($n = 7$) (**a**) or for individual cells, all experiments (**b**). **c**,**d**, Interaction time plotted per mouse (**c**) or per cell, all experiments (**d**). **e**,**f**, Cellular velocity plotted either per mouse (**e**) or per cell, all experiments (**f**). $^{**}P < 0.005$, $^{***}P < 0.0005$, two-sided Mann–Whitney U test. All center values denote the mean, and error bars denote s.e.m. $n = 2$ independent experiments for **a**-**f**. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

relative to DCs. Plotting the mean T cell arrest coefficient per mouse (figure 4.3a) revealed that the motility of 5CC7 cells was less than that of wild-type cells. However, when plotting values per cell across mice, there was substantial overlap between the two populations (figure 4.3b). Mean T cell interaction time provided better separation with larger relative differences between wild-type and 5CC7 T cells on a per-mouse basis and less overlap when individual cells were plotted (figure 4.3c,d). The cellular mean velocity of T cells, plotted both per mouse and per cell, provided intermediate separation between groups with moderate overlap between individual cell values (figure 4.3e,f). In contrast, in response to very

low doses of antigen (DCs pulsed with 0.01 µM PCC), there was relatively little difference between wild-type and 5CC7 cells by most TPEM measures (supplementary figure A.1). These data indicate that TPEM measures can accurately discriminate between wild-type and antigen-specific T cell populations based on their behavior relative to antigen-pulsed DCs.

B10.A CD45.2^{-} and B10.A CD45.2^{+} 5CC7 TCR-transgenic Rag2$^{-/-}$ mice were obtained from Taconic Laboratories through a special NIAID contract. All animal experiments were conducted under a protocol approved by the NIAID Animal Care and Use Committee (LSB-1E) and the University of Chicago Animal Resource Center. All animal experiments were conducted in compliance with all relevant ethical regulations.

For the adoptive tranfer performed by Nicholas Panhuys and Ronald Germain, dendritic cells were purified from mouse spleens using anti-CD11c beads (Miltenyi). Purified dendritic cells were activated in vitro with lipopolysaccharide $1\,\mu g\,mL^{-1}$ and pulsed with pigeon cytochrome c peptide (Bachem, sequence corresponding to amino acids $88-104$) at high (10 µM) or low (0.01 µM) concentration for 4 h at 37 °C. Activated DCs were labeled with Cell Tracker Blue (CMF2HC, Invitrogen), then injected (1×10^6 per recipient) into the right rear footpad of recipient mice. Polyclonal and 5CC7 TCR-transgenic T cells were isolated from the lymph nodes of B10.A CD45.2^{-} wild-type and B10.A CD45.2^{+} 5CC7 TCR-transgenic Rag2$^{-/-}$ mice, respectively, and purified using a CD4^{+} T cell isolation kit (Miltenyi). Polyclonal T cells were then labeled with Cell Tracker Green (CMFDA, Invitrogen), and 5CC7 T cells were labeled with Cell Tracker Red (CMTPX, Invitrogen). 2×10^6 of each T cell population were then co-injected IV into recipient mice 18 h post transfer of DCs. 12 h post T cell transfer, mice were subjected to TPEM studies as previously described [37]. Isoflurane was used to anesthetize mice prior to exposure of popliteal lymph nodes (Baxter; 2.5 % for induction, ∼1 % to 1.5 % for maintenance, vaporized in an 80:20 mixture of O2 and air), and subsequent TPEM was performed as described [37]. Briefly, imaging was conducted

on a Bio-Rad/Zeiss Radiance 2100MP, configured with a Nikon 600FN upright microscope equipped with a 203 water immersion lens (NA 0.95, Olympus) and LaserSharp acquisition control software. Anesthetized mice were maintained in environmental chambers warmed by heated air with the surgically exposed lymph node kept at 36 °C to 37 °C with warmed PBS. Upon completion, mice were euthanized, and draining popliteal lymph nodes were isolated, cured overnight in 30 % sucrose, and frozen at – 80 °C. The tissue was subsequently sectioned at 5 µm thickness and prepared for confocal microscopy.

Confocal imaging of mouse tissue was performed by Junting Ai. Mouse tissue sections were prepared and stained with TO-PRO-3 Iodide (Invitrogen) to visualize nuclei and avoid interference with the fluorescence spectrum of transferred cell trackers. The TO-PRO-3 nuclear stain is alternatively referred to as DAPI (another nuclear stain) in throughout this document. This is not technically correct. In mouse imaging TO-PRO-3 was used and in human fresh frozen and paraffin embedded DAPI was used. Single-fluorochrome controls were utilized to ensure no cross-bleeding was present in between fluorescent channels. Images were acquired at 12-bit depth, 1024 × 1024 matrix size, at 400× and 630× magnifications utilizing either the SP5 Tandem Scanner Spectral two-photon confocal microscope or the SP8 3D three-color stimulated emission depletion (STED) laser scanning confocal microscope with time gating (Leica). Each ROI was 144.74 µm or 1024 pixels wide, corresponding to an average absolute resolution size of 0.28 µm, based on Nyquist sampling. Regions of interest, containing all three transferred cell populations, were selected for acquisition. Raw images were stored in manufacturer-specified .lif format. Lif files were converted to multichannel .tif images and used as input for DCNN analysis [18].

4.2 Dataset Overview

Total murine imaged data set is tabulated in table 4.1. For experiment 1 and experiment 2 all nuclei with positive staining were segmented from all ROIs for a total of 295 segmented

ROIs. Table 4.2 shows the number of Tas and wild type T cells (Tws) that were segmented. In 10 of these ROIs, all the nuclei in the field were segmented. Care was taken to ensure that the 10 wholly segmented ROIs were representative of the larger dataset. Some representative murine ROIs are shown in figure 4.4. Finally, a 3rd experiment with 233 ROIs was conducted. There was no manual labeling for this set. A representative ROI for the segmentation and classification performed in this section is shown in figure 4.4 with both manual segmentation and automatic classification from a 5-fold model-based cross validation run of the DCNN algorithm (discussion to follow).

	Matrix size	Channels	ROI #	Pixel Size
Exp 1	1024×1024	5	253	$0.14\,\mu m \times 0.14\,\mu m$
Exp 2	1024×1024	5	42	$0.23\,\mu m \times 0.23\,\mu m$
Exp 3	1024×1024	4	233	$0.23\,\mu m \times 0.23\,\mu m$
Total	–	–	528	–

Table 4.1: **Murine fresh frozen imaged dataset:** showing image matrix size number of channels, and pixel size for 3 experiments.

	Ta # nuclei	Tw # nuclei	Dendritic cells
Exp 1	611	598	2016
Exp 2	335	269	440
Exp 3	–	–	–
Total	946	867	2456

Table 4.2: **Murine fresh frozen manually segmented dataset:** showing number of Ta, Tw, and DC cells labeled across 3 experiments

4.3 Binary Nuclei Segmentation

We want to extract shape information from the DAPI stained nuclei in our images but we also care about the class of nuclei based on the staining information in the other image channels. One approach to analyzing the data is to segment all nuclei based only on the information contained in the DAPI nuclear channel, and then classify each nucleus based

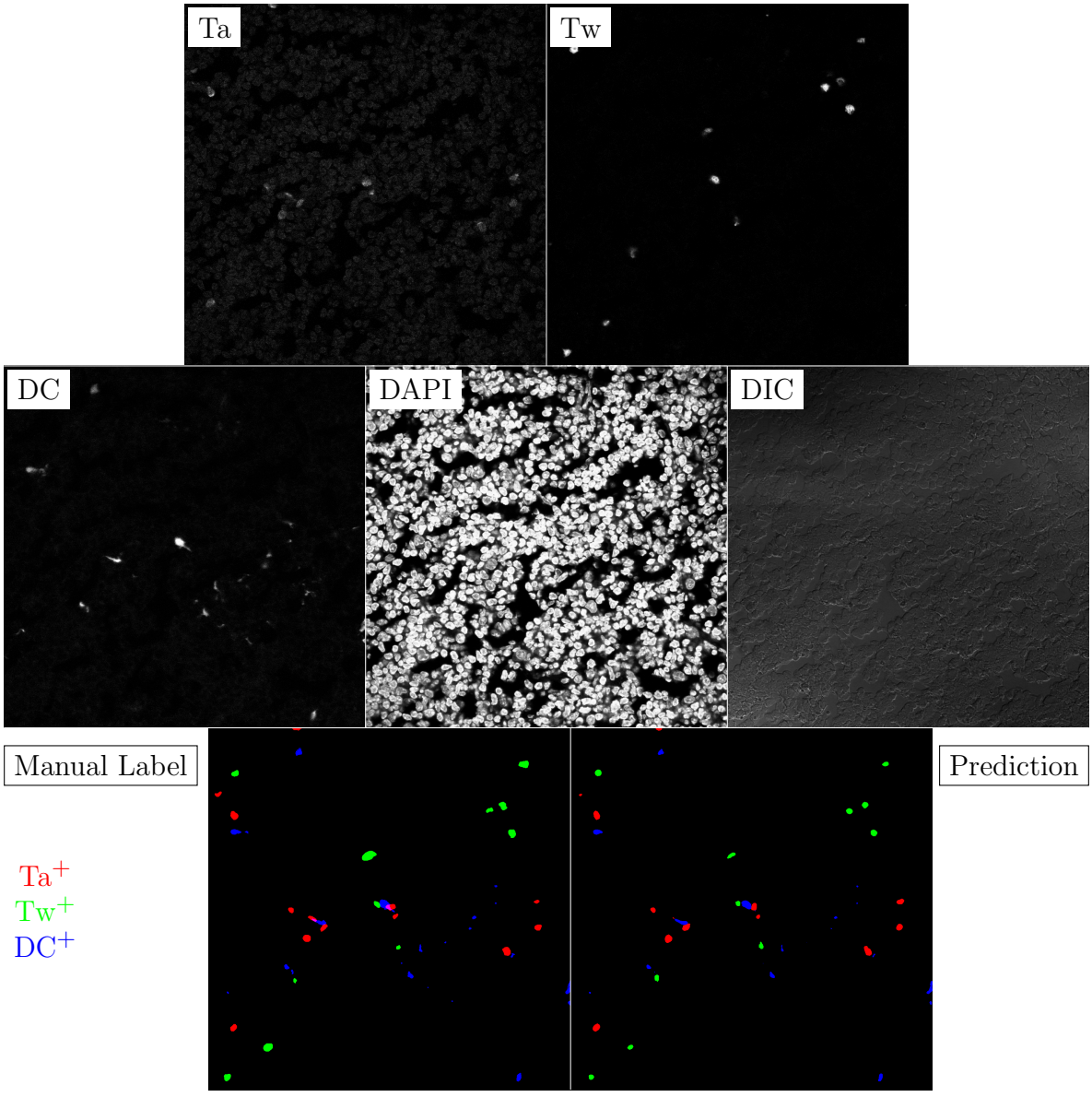

Figure 4.4: **Murine fresh frozen ROI:**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 1024×1024.

on the information in the other channels. The problem with this approach is that the nuclear channel is often not sufficient to segment nuclei accurately in our images, even with deep learning. Figure 4.5 shows results from our deep learning model trained on 7 DAPI nuclear images with binary labels. These three images were not contained in the training set. Traditional image analysis techniques were unable to segment nuclei in these

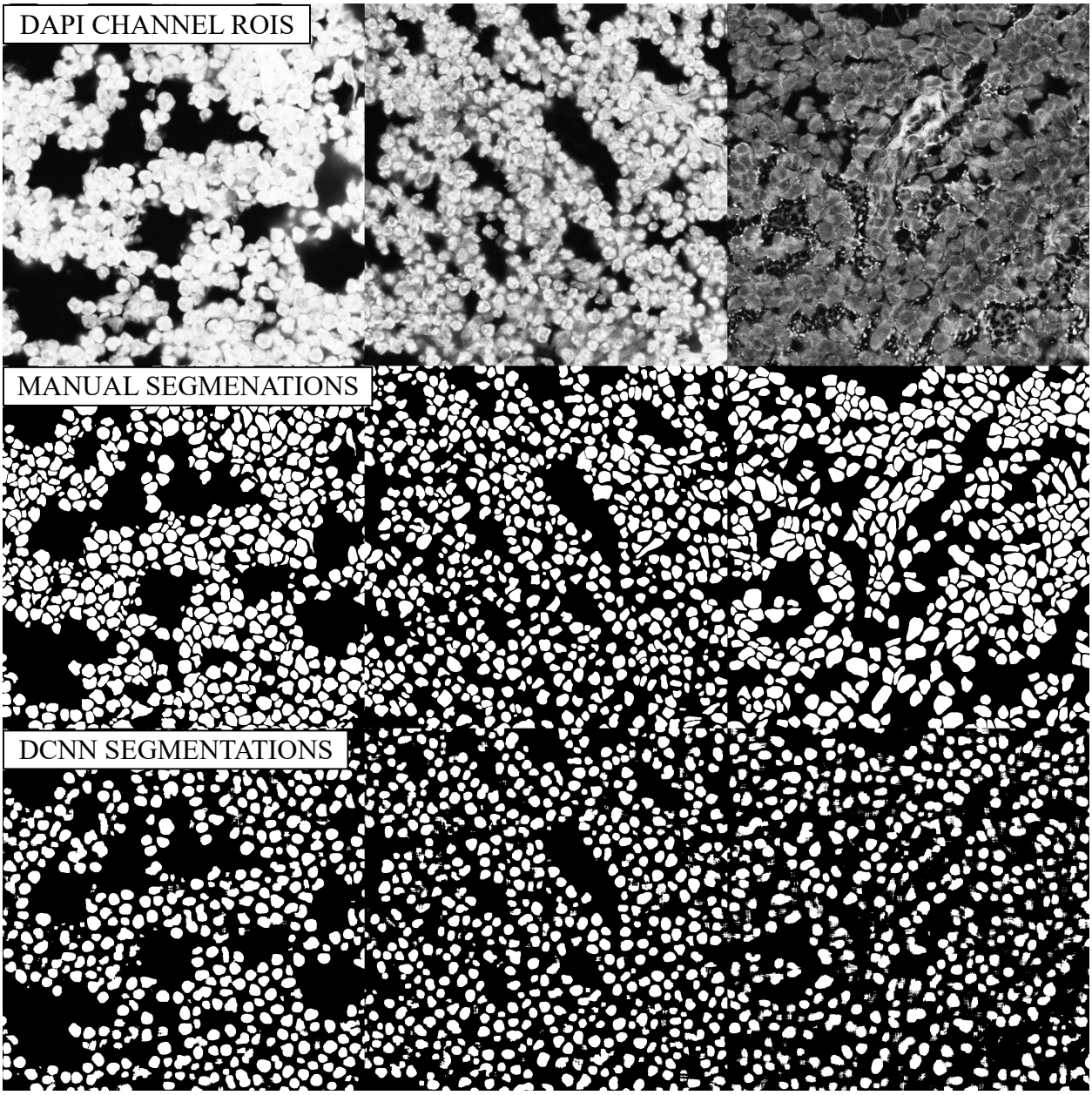

Figure 4.5: **Binary nuclei segmentation examples:** (**top row**) The DAPI channel from 3 ROIs with image quality variation. (**middle row**) Manual segmentations for these ROIs. (**bottom row**) Object segmentations from DCNN trained on manual segmentations from other ROIs. A single DCNN model produced these segmentations from images in the validation set, meaning the neural network was not trained on these images. Pixel size is $0.14\,\mu\text{m} \times 0.14\,\mu\text{m}$. True image matrix size is 1024×1024. Depicted image matrix size is 1024×1024. Images are losslessly compressed with DEFLATE compression algorithm. Bit-depth is 8.

DAPI nuclear ROIs with any reasonable accuracy. The model used to segment these ROIs is identical to the model depicted in figure 4.6 with the exception that the DAPI nuclear channel is the only input channel, 3D kernels ($3 \times 3 \times 3$) are instead 2D (3×3), and there is not pooling in the nonexistent channel dimension. These images are representative of the image quality variation found in the DAPI nuclear channel within our image sets and illustrate this model's ability to generalize across image quality for a single channel. Since the method of segmenting all nuclei in a field and assigning class membership based on the other image channels afterwards was not viable for our data, we investigated using multiclass deep learning segmentation methods. The manual segmentations used to train the DCNN here are shown in appendix C.

4.4 Multiclass Nuclei Segmentation

To produce usable segmentations from our data, we developed and implemented a 3D, multiclass deep convolutional neural network to incorporate both spatial and classification information into a single framework. Since the boundaries of a nucleus are often difficult to resolve, the cell membrane stain helps to resolve the boundaries of the nucleus, while also providing classification information. In addition, noise in one channel can be compensated for by information in another.

The DCNN used 10 convolutional layers, three maximum pooling layers, and a fully connected layer (figure 4.6). Rather than down-sampling our images with each max-pooling layer, we kept our feature maps at original resolution and increased the sparsity of subsequent convolutions. We used a receptive field of view of 85 pixels $\times$ 85 pixels $\times$ 5 channels. Receptive field of view refers to the total amount of information from the surrounding image that goes into a prediction at a single pixel. The trainable weights and biases across all the feature maps in the convolutional layers of the DCNN and the fully-connected layer resulted in $\sim$700k trainable parameters, shown to be reasonable in similar studies [42, 43].

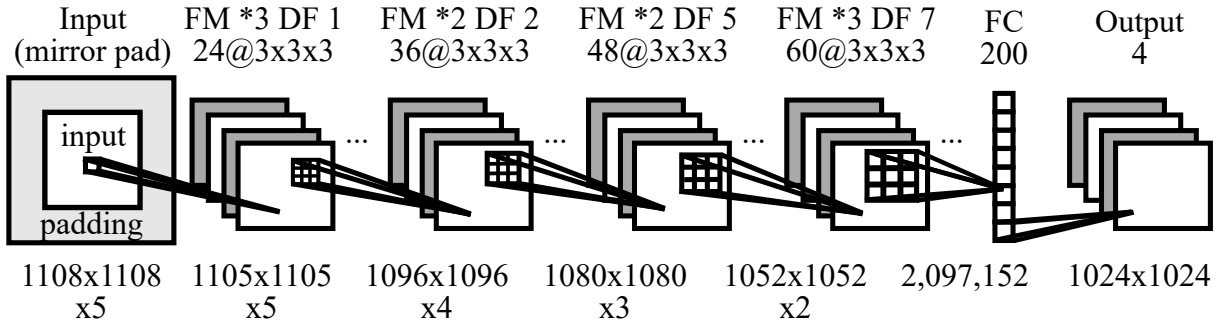

Figure 4.6: **Abbreviated DCNN architecture:** 5 channel input is padded and fed through 4 sets of convolutional layers. Each of the first 3 sets of convolutional layers is followed by a pooling layer. The final set of convolutional layers is followed by a fully connected layer. The fully connected layer is fed into a softmax layer to produce the final prediction. Top row shows feature maps (FMs)×number of layers and convolution dilation factor (DF) of the kernels in the layer. 2nd row shows number of filters per layer with filter kernel size. Bottom row shows input volume size as it passes through DCNN.

To train the DCNN, we manually segmented confocal images using ICY Bio Image Analysis software and ImageJ. All segmentations were independently validated by a blinded observer. For the murine experiment described above, the total dataset of 295 randomly collected high power fields (HPFs), corresponding to ROIs containing all three cell types, was segmented for wild-type T cells, 5CC7 T cells, DCs, and corresponding cell nuclei (table 4.2). Training batches consisted of four $184 \times 184 \times 5$ image patches drawn randomly from the entire dataset, each belonging to four different classes (5CC7 T cells, wild-type T cells, DCs, and background). The DCNN was trained for $200,000$ iterations at which point cross-entropy error was stable and small.

As depicted in figure 4.7, the 5 image channels for each ROI which include the antigen specific T cell (Ta) channel, the wild type T cell channel (Tw), the DIC channel), the DC channel, and the DAPI channel are put into the DCNN. The DCNN outputs a 4-class prediction for antigen specific T cells (Ta), wild type T cells (Tw), dendritic cells (DC), and background. In the post-processing step, each pixel is assigned the class with the maximum predicted probability. The post-processing step can also include removing objects with anomalous shapes and dropping segmentations with low predicted probability.

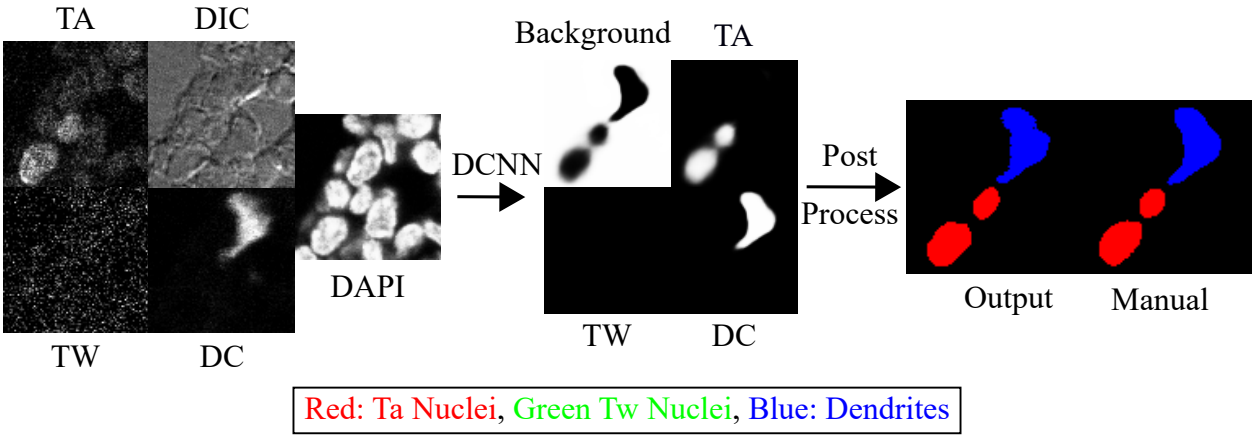

Figure 4.7: **DCNN segmentation pipeline:** Multi channel input produces a multi cell class probability prediction. Cell class is assigned to each pixel based on maximum predicted probability. Finally objects below a certain size are eliminated to produce final segmentation. Predictions shown in red (Ta), green (Tw), and blue (DC). Pixel size is 0.23 µm × 0.23 µm. True image matrix size is 90×90. Depicted image matrix size is 90×90. All ROIs embedded in document with lossless DEFLATE compression. Bit-depth is 8.

Figure 4.8 depicts the total segmentation and image classification pipeline.

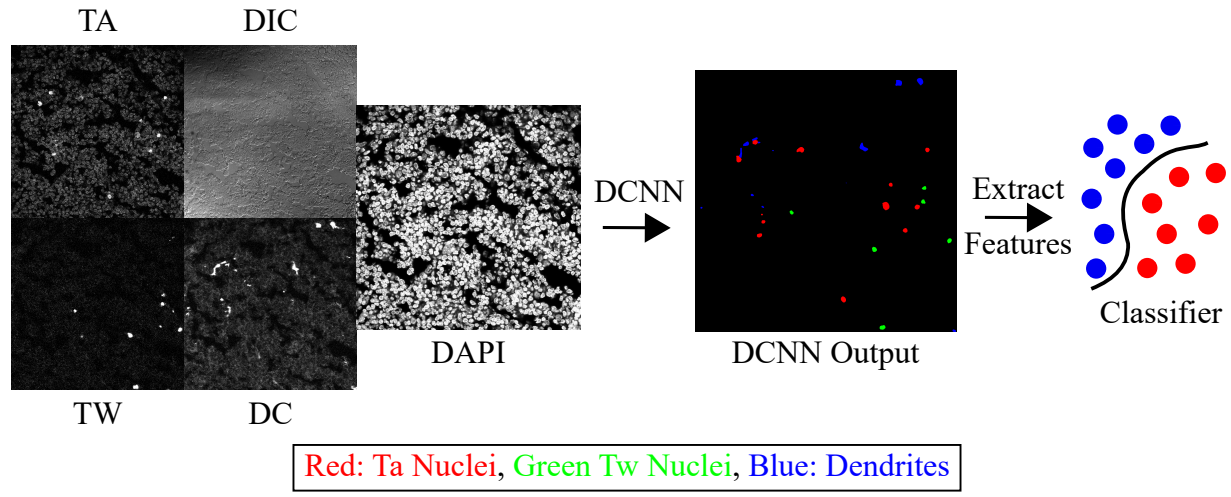

Figure 4.8: **Image analysis pipeline for murine data:** Pixel size is 0.23 µm × 0.23 µm. True image matrix size is 1024 × 1024. Depicted image matrix size is 1024 × 1024. Images were losslessly compressed with DEFLATE compression algorithm. Bit-depth is 8.

4.5 Segmentation Results

A segmentation performance example with overlap with manual truth is shown in figure 4.9. We assessed the performance of the DCNN segmentation by computing sensitivity and specificity of cell detections for automatic segmentation vs. manual truth and IOU for automatic segmentation vs. manual truth. These metrics are computed from a 5-fold model-based cross validation. That is, a model is trained on 4/5 of manually segmented data and used to predict on the other 1/5 for all 5 permutations. To assess the sensitivity and specificity of localization and segmentation, a segmented cell was considered detected if the IOU of the manually segmented cell with the automatically segmented cell was greater than or equal to 0.5. Overall, across all cells, the DCNN had an average sensitivity of 88 %, specificity of 92 %, and a mean IOU of 0.85 (table 4.3). Finally 6 selected performance examples are shown in appendix D and all ROIs from the murine cross validation dataset are shown in appendix F

	Ta # nuclei	Tw # nuclei	Dendritic cells
Average IOU score ± SD	0.81 ± 0.11	0.76 ± 0.10	0.90 ± 0.06
Sensitivity	0.80	0.90	0.93
Specificity	0.88	0.91	0.97

Table 4.3: **Murine fresh frozen DCNN performance statistics:** Average IOU score, sensitvity, and specificity are show for Ta, Tw, and Dendritic cells.

4.6 Classification Results

Initial classification results on mouse tissue were performed by Adam Sibley. Results presented below were prepared by Vladimir Liarski. From the DCNN output, we extracted relative distances between T cell populations and DCs, as well as features of T cell shape. For the latter, we used seven independent measures of two-dimensional shape that include major and minor axis lengths, convex and regular perimeters, convex and regular areas,

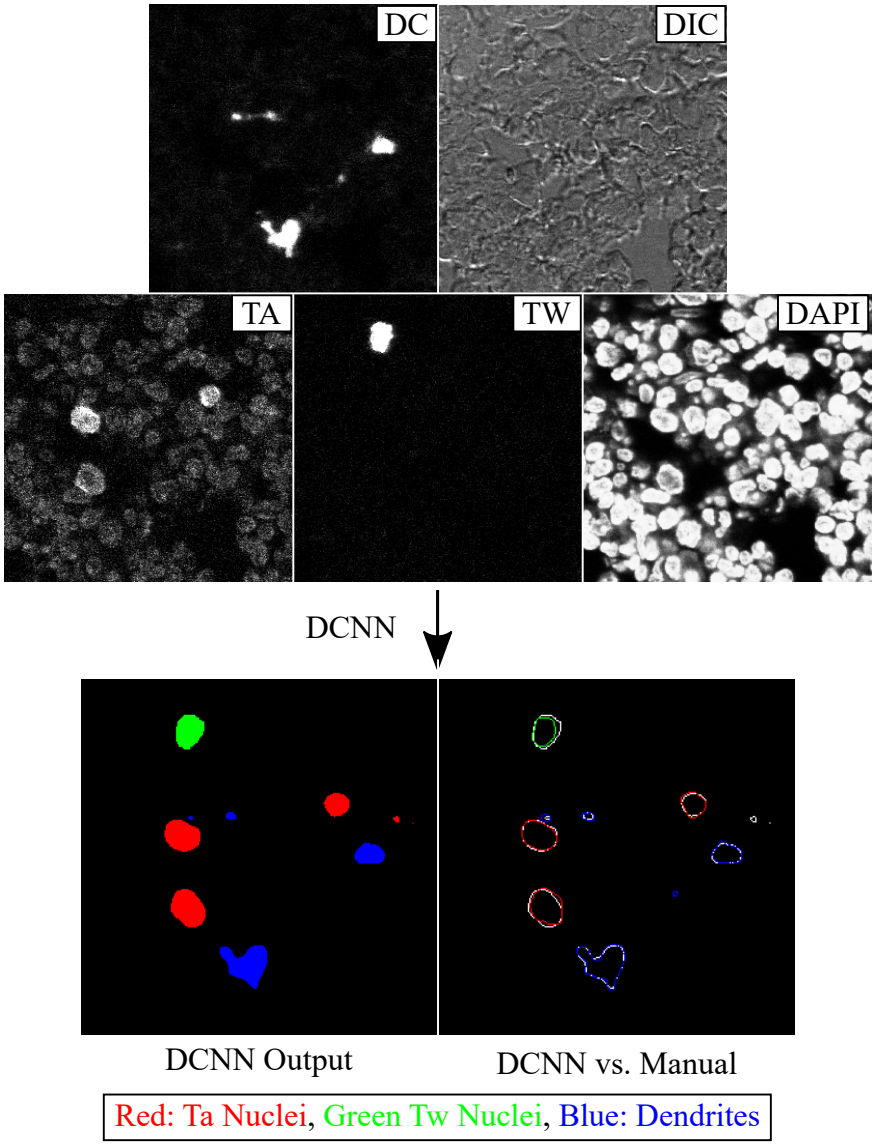

Figure 4.9: **DCNN performance illustration:** Manual label borders are shown for Tw (red), Ta (green), and DC (blue) overlaying the DCNN output in white. Pixel size is 0.23 µm × 0.23 µm. True image matrix size is 278 × 278. Depicted image matrix size is 278 × 278. All ROIs embedded in document with lossless DEFLATE compression. Bit-depth is 8.

and equivalent diameter [56] (figure 3.9). These data were imported into R statistical software [57] and analyzed to determine which combination of variables and variable weights best discriminated between the 5CC7 and wild-type T cell populations relative to DCs. Our approach included the use of the following algorithms: logistic regression, SVM, and

neural networks. For the latter, three different neural network models (simple, tuned, and linear output) were generated. The performance of each algorithm was assessed as measured by parameters of classification accuracy, error, and receiver operating characteristic (ROC) curve performance for correctly predicting cell type (supplementary table B.2a). A TNN (figure 3.10) consistently displayed the best performance among neural network models (supplementary table B.2b) at the expense of increased number of steps and computation time (supplementary figure A.2a–c). Therefore, we used a DCNN followed by a TNN in the CDM_3 pipeline.

Random forest analysis revealed that minimum distance to a DC provided the best discrimination between 5CC7 and wild-type T cells (figure 4.10a). Simply plotting relative distance to closest DC provided excellent discrimination between wild-type and 5CC7 T cells, with 5CC7 T cells being, on average, much closer to the nearest DC (figure 4.10b). We next plotted the true-positive rate (sensitivity) versus the false-positive rate ($1 - specificity$) in a ROC curve to determine how distance performed as a test to discriminate between wild-type and 5CC7 T cells (figure 4.10c). This analysis revealed that cellular distance provides good discrimination with an AUROC of 0.70 (95 % confidence interval (CI): 0.62–0.74, $5 < 5 \times 10^{-5}$).

Among the T cell shape variables, minor-axis length was the most promising for discriminating wild-type from 5CC7 T cells. Comparison of T cell minor-axis length at distances of less than 5 µm versus 75 µm or greater from DCs, revealed that 5CC7 T cells had a longer minor axis compared to wild-type T cells at close distances (figure 4.11a). Plotting individual cell minor axes as a function of distance (figure 4.11b and supplementary figure A.3a) clearly revealed a sub-population of 5CC7 cells displaying high values, especially when contacting a DC (0 µm). This difference was diminished by 25 µm and largely lost at distances greater than 50 µm. These changes are consistent with the 5CC7 cells flattening against the antigen-pulsed DCs, resulting in an increase in minor cell axis for some cells in the 2D plane

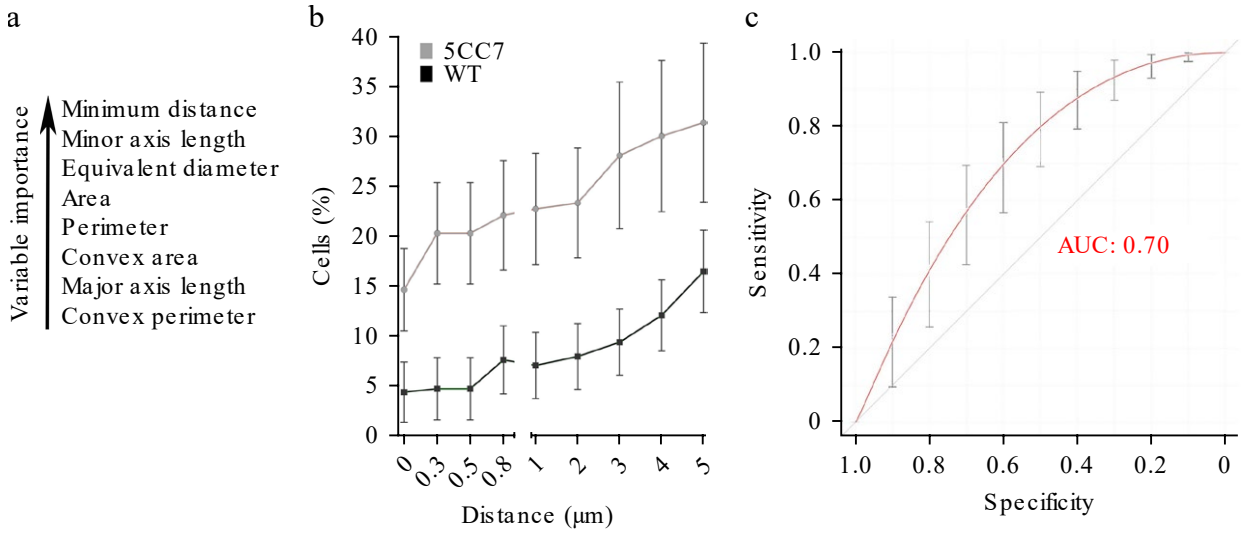

Figure 4.10: **Feature contribution hierachy (a), relative distance plot (b), and distance-based AUROC (c): a**, Hierarchy of contribution of distance and T cell shape parameters to accuracy, as determined by means of random forest analysis. **b**, Cumulative frequency of either 5CC7 (gray) or WT (black) T cells as a function of distance from antigen-pulsed DCs ($P < 0.005$). **c**, Plot of sensitivity and specificity of distance alone in discriminating between 5CC7 and WT T cells (AUROC 95 % CI: 0.62–0.74, $P < 5 \times 10^{-5}$). Diagonal lines in **c** denote AUROC of 0.5, which represents a random probability($P = 0.5$). $n = 2$ independent experiments. Center values denote the mean in (**b**), and error bars denote SD (**b**) or cross-validation error (**c**) (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

of the confocal micrograph. In contrast, the minor axis of wild-type T cells in some cells decreased as a function of distance from DCs. This latter trend is consistent with wild-type T cells becoming more spherical upon contacting DCs.

Plotting T cell cross-sectional area as a function of distance revealed similar results (figure 4.11c,d and supplementary figure A.3b) with a subset of 5CC7 T cells having relatively large cross-sectional areas when very close to or abutting DCs. Wild-type T cell area decreased at close DC distances. In contrast, 5CC7 and wild-type T cells have similar shape characteristics at distances of 75 µm or greater from DCs. These data suggest that there were no substantial intrinsic differences between the T cell populations in shape or size. Rather, antigen-specific 5CC7 and wild-type cells diverge in their shape properties in proximity to antigen-pulsed DCs. These data suggest that CDM_3 can capture changes in T cell shape

that occur upon recognition of antigen presented by DCs.

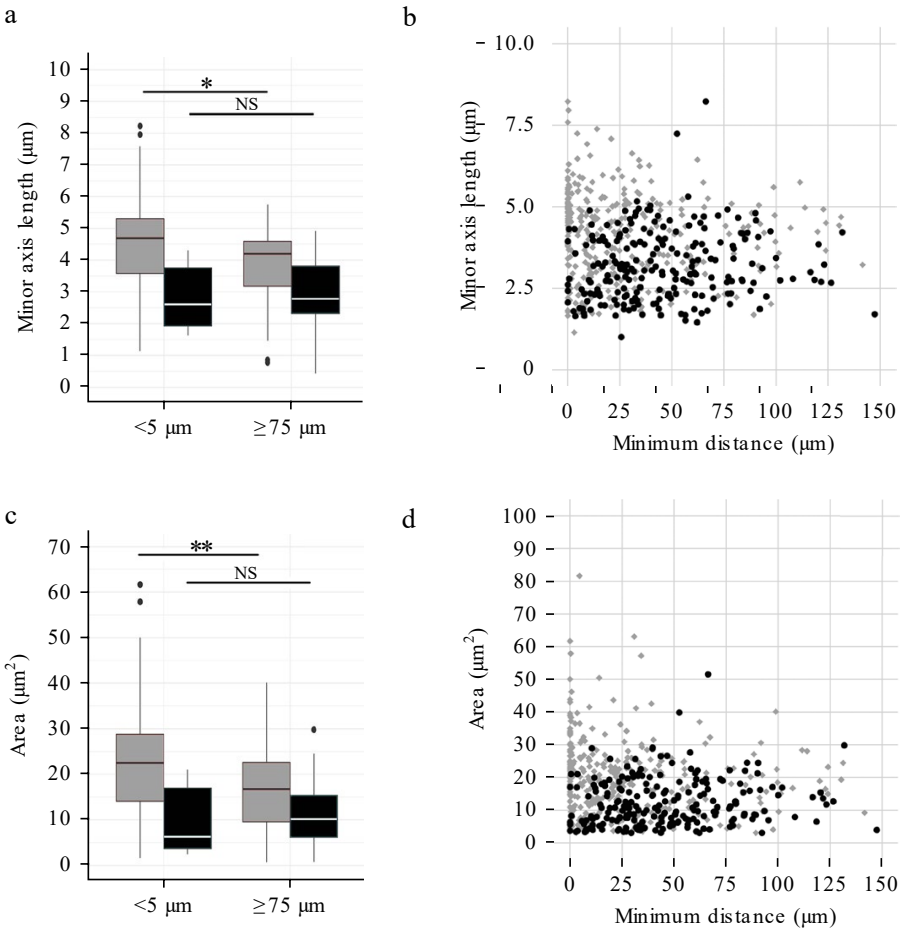

Figure 4.11: **Box and scatter plots for minor axis length and area features: a**, Comparison of minor axis length for 5CC7 (gray) or WT (black) T cells at indicated distances. **b**, Minor axis length for each cell plotted as a function of distance from DCs. **c**,**d**, 5CC7 (gray) and WT (black) T cell cross-sectional area as an average at indicated distances (**c**) or for each cell as a function of distance (**d**). Midlines in box plots (**a**,**c**) denote median value, with upper and lower hinges denoting first (Q1) and third (Q3) quartile values, respectively, vertical bars corresponding to values 1.5 × the interquartile range (IQR) for Q1 and Q3, respectively, and dots representing outlier values, not included in the above. $n = 2$ independent experiments. Center values denote the mean in **a**,**c**. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

We next determined how well CDM_3 could discriminate between the 5CC7 and wild-type T cell populations by simultaneously incorporating both distance relationships and T cell shape. We focused our analysis to T cell populations within 25 μm of DCs, as this was

the distance at which we observed substantial differences in T cell shape. Furthermore, we examined close distances because we were interested in discriminating between T cells that recognize antigen from those that are scanning peptide–MHC class II complexes, looking for antigen. The full CDM_3 output, which integrates distance and T cell shape variables, provided an AUROC of 0.84 (95 % CI 0.80–0.90, $P < 5 \times 10^{-5}$) and was substantially better than distance alone for all measurements (figure 4.12a). In contrast, at distances of greater than 75 µm, the two T cell populations were indistinguishable (figure 4.12b). Within 25 µm, the use of the minimum distance variable by itself could also discriminate between 5CC7 and wild-type cells (data not shown). However, differences between populations were less robust (AUROC = 0.65, 95 % CI: 0.59–0.72, $P = 0.008$). These data indicate that CDM_3, by combining measurements of both cell distance and T cell shape, provides excellent discrimination between T cell populations that are scanning for antigen versus those that have recognized antigen.

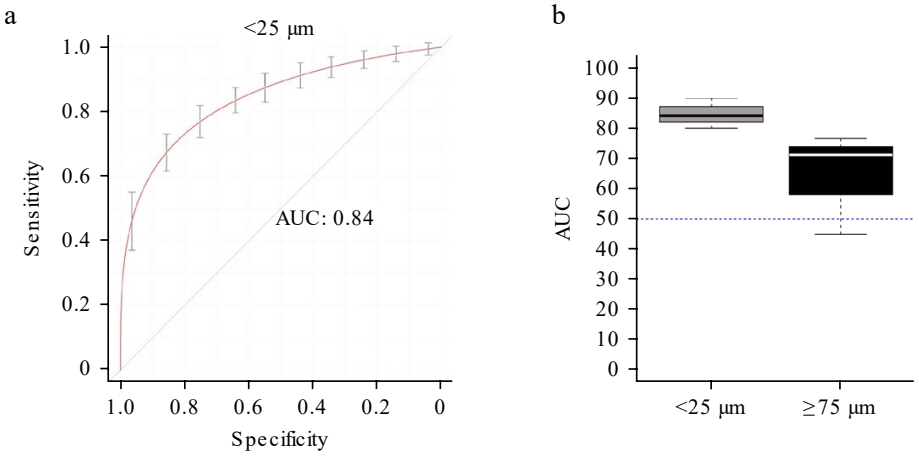

Figure 4.12: **ROC curve and AUROC for full CDM_3 output feature classification:** **a**, Plot of sensitivity and specificity of CDM_3 for discriminating between 5CC7 and WT T cells (AUROC 95 % CI: 0.80–0.90, $P < 5 \times 10^{-5}$). **b**, Comparison of AUROC and 95 % confidence interval performance shown in **a**, with values derived from analysis of cells at distances $\leq$ 75 µm. Diagonal lines in **a** denote AUROC of 0.5, which represents a random probability($P = 0.5$). $n = 2$ independent experiments. Center values denote the mean in (**a**,**b**) and and error bars denote s.e.m. (**b**) (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

We next examined how well measures obtained by TPEM discriminated between 5CC7 and wild-type cells interacting with antigen-pulsed DCs. Therefore, we took the TPEM outputs described above (figures 4.1 and 4.3, and supplementary table B.1) and subjected them to the same statistical modeling by plotting their true-positive rate versus false-positive rate. The arrest coefficient provided good discrimination with an AUROC of 0.74 (95 % CI: $0.72 - 0.82$ $P < 5 \times 10^{-5}$; figure 4.13a). Cell velocity was more robust with an AUROC of 0.86 (95 % CI: $0.78 - 0.90$, $P < 5 \times 10^{-5}$, figure 4.13b). Only cell interaction time, with an AUROC of 0.95 (95 % CI: $0.94 - 0.97$, $P < 5 \times 10^{-5}$), substantially outperformed CDM_3 (figure 4.13c). These data indicate that CDM_3 performs as well as many TPEM measures in identifying antigen-specific T cell interactions with DCs.

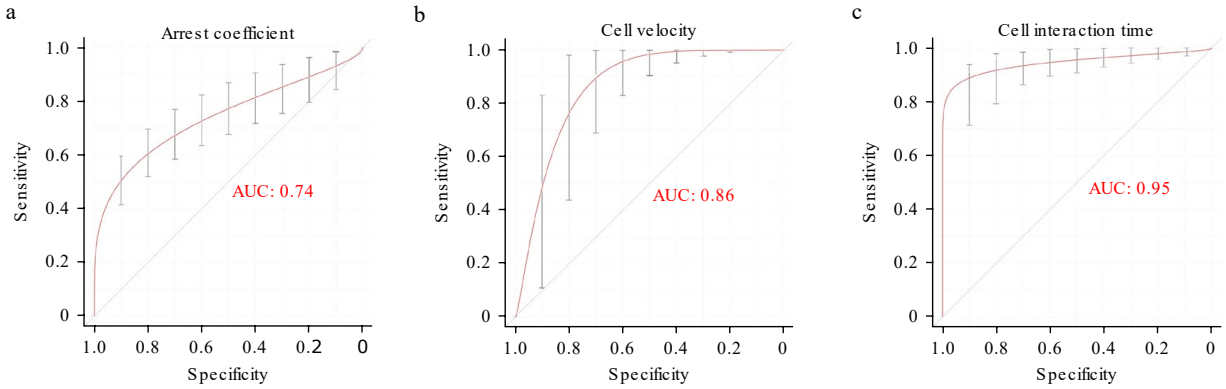

Figure 4.13: **ROC curves for TPEM features:** **a**–**c**, Plot of sensitivity and specificity of TPEM measures for discriminating between 5CC7 and WT T cells including cellular arrest coefficient (**a**, AUROC 95 % CI: 0.72–0.82 $P < 5x10^{-5}$), cellular velocity (**b**, AUROC 95 % CI: 0.78–0.90, $P < 5 \times 10^{-5}$), and cellular interaction time (**c**, AUROC 95 % CI: 0.94–0.97, $P < 5 \times 10^{-5}$). $^{*}P < 0.05$, $^{**}P < 0.005$, two-sided Mann-Whitney U test. NS, not significant. Diagonal lines in **a**–**c** denote AUROC of 0.5, which represents a random probability($P = 0.5$). $n = 2$ independent experiments. Error bars denote cross-validation error (**a**–**c**) (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

We next sought to apply CDM_3 to the analysis of multichannel confocal images of human tissue. However, immunofluorescence with antibodies to surface markers is often inadequate for identifying the exact boundaries of lymphocytes in dense infiltrates. In our original studies using CDM, nuclear stains were necessary to define lymphocyte position [6]. As the

nucleus constitutes the majority of a lymphocyte's volume [58], we postulated that nuclear shape would approximate cell shape. Therefore, 5CC7 and wild-type T cell nuclei were segmented and analyzed using CDM_3. Evaluation of representative images of cell trackers compared with to TO-PRO-3 nuclear staining revealed extensive overlap between the two staining signatures (figure 4.14a). Analysis of individual shape parameters revealed similar relation-ships between 5CC7 and wild-type T cell nuclear shape and distance to closest DCs as observed for their cell tracker counterparts (figure 4.14b–d, supplementary figure A.3d,e and supplementary table B.1). That is, in a subset of 5CC7 cell nuclei, shape parameters increased close to DCs, whereas in wild-type nuclei, they did not. Interestingly, a small population of 5CC7 nuclei displayed a decrease in two-dimensional T cell shape parameters close to DCs. This distance-dependent increase in nuclear shape variability is consistent with 5CC7 cells becoming more irregular (less spherical) upon contacting DCs. This is expected when a cell (sphere) flattens against a DC (surface). Application of the composite distance and T cell nuclear shape scores revealed similar discrimination between 5CC7 and wild-type T cell interactions with DCs as that observed for whole cells (figure 4.14e, AUROC = 0.82, 95 % CI: 0.77–0.91, $P < 0.005$). Similarly, at distances greater than 25 µm, the two T cell nuclei populations were indistinguishable (data not shown, AUROC = 0.52, 95 % CI: 0.45–0.72). These data indicate that nuclear shape alone can be used to approximate T cell shape for the purpose of discriminating cognate from non-cognate T cell–DC interactions.

Finally, to validate our mouse experiments, we independently repeated the adoptive cell transfer experiment in mice and obtained 233 additional ROIs. Here, DC and T cell nuclei were subjected to the same segmentation and analysis using CDM_3 as described for the original experiment. These data revealed similar discrimination as before between 5CC7 and wild-type T cell nuclei, with an AUROC of 0.82 (95 % CI: 0.72–0.87, $P < 5 \times 10^{-5}$) at distances less than 25 µm from DCs. Likewise, for distances greater than 25 µm, the two T cell nuclei populations were similar (AUROC = 0.54, 95 % CI: 0.45–0.58) [18].

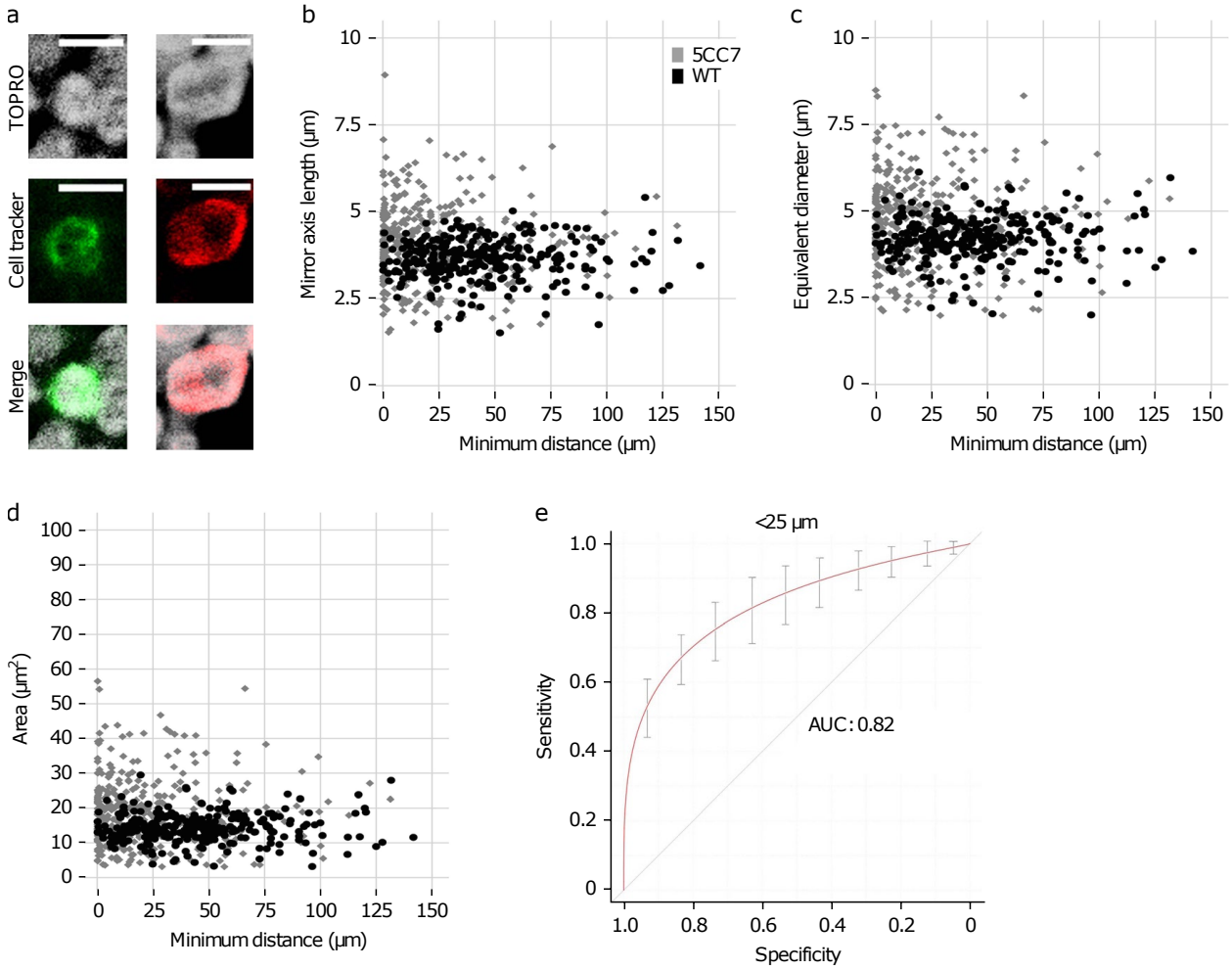

Figure 4.14: **CDM$_3$ nuclear segmentation analysis: a**, Representative images of fluorescent cell-tracker-labeled WT (left) and 5CC7 (right) T cells stained with TO-PRO-3 iodide (TO-PRO). **b**–**d**, Plots of 5CC7 (gray) and WT (black) T cell nuclear minor axis length (**b**), equivalent diameter (**c**), and nuclear area (**d**) as functions of minimum distance from antigen-pulsed DCs. **e**, Curves denoting sensitivity and specificity of CDM$_3$ for discriminating between 5CC7 versus WT cell nuclei at distances $< 25\,\mu m$ (AUROC 95 % CI: 0.77–0.91, $P < 0.005$). Scale bars, 5 µm. Diagonal lines in e denote AUROC of 0.5, which represents a random probability ($P = 0.5$). $n = 3$ independent experiments for **a**, and $n = 2$ independent experiments for **b**–**e**. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

4.7 Discussion and Conclusions

Our application of a DCNN to identify multiple cell types in a mouse model of T cell interaction with dendritic cells showed strong peformance in classification of cell types and

segmentation of dendritic cell bodies and T cell nuclei. Segmenting and classifying cell types in multi channel images is in general a very difficult problem. Our approach for these mouse fresh frozen confocal microscopy images was to identify all the subsets of cells in the images we were interested in, label the images manually, and use a custom DCNN to segment and classify the cell types. In this method, the DCNN treated each cell as having multiple channel dimensions, and classification and segmentation was drawn from that information.

This approach worked well for a somewhat specific question: querying a set of cell subsets and extracting shape and classification information from their nuclei or cell body. Other approaches are possible but were judged to be impractical for this work due to technical limitations of software, man-power, and time. For example, the two primary types of information we have in our images are cell membrane stains and nuclear stains. For the mouse images, cell membranes can either belong to T cells or dendritic cells. Nuclear stains are non-specific and may belong to any cell with a nucleus. Instead of labeling and designing a DCNN that identified specific cell nuclei belonging to membranes that presented in a specific channel (or cell membranes that present in a certain channel in the case of the DCs), We could have labeled all cell membranes or nuclei non-specifically, and then taught the DCNN to segment those. While it is less apparent in the context of these mouse images (it will become more apparent in the sections discussing human fresh frozen and paraffin embedded images), segmenting all cell membranes non-specifically generally leads to much more human labeling work. In the case of nuclei, the amount of work required is even greater.

Continuing with the idea of non-specific cell membrane and nuclear segmentation, segmenting these non-specifically would also require an additional classification step. This could be accomplished in a way similar to Mask-RCNN (discussed further in chapter 6), but this would require a large amount of training data and some additional technical innovation. Other implementations are possible. All cell nuclei could first be segmented non specifically and then cell membranes could be associated with those nuclei. Segmenting

all nuclei is a difficult task that is still being tackled in the general case by the biological sciences community. It was recently addressed in the Kaggle Nuclei Data Science Bowl in 2018 (https://www.kaggle.com/c/data-science-bowl-2018) with a large nuclear dataset, but reliably segmenting all nuclei still seems out of reach, particularly when image quality, characteristics, and magnfication can vary so much between data sets. In addition, associating a cell membrane with a nucleus is not a trivial problem. Often the staining agents have a wide range of presentation in cell membranes (from very bright to very faint). Most traditional computer vision techniques don't do well in trying to divide up such data. The problem of doing so becomes even more difficult when noise is introduced. Techniques for doing so include marker controlled watershed techniques and graph partitioning methods.

Segmenting all nuclei and then associating membrane stains with the nuclei is a top down approach to this problem. The bottom up approach would be segmenting all cell membranes and then associating nuclei. This is difficult for reasons discussed above, but might be more practical for the simple reason that cell membranes surround nuclei. Segmenting nuclei with information that they are contained in a specific region becomes much easier.

Continuing on future segmentation possibilities with our data, non-specific object detectors based on DCNNs also have the potential to be useful. For example, a segmentation method to generate object segmentation masks where only bounding boxes were present was presented in [59]. This method used a novel weight transfer function to approach the problem of segmentation mask generation in a partially supervised fashion.

On the subject of manual segmentation, while the task of segmenting by drawing lines around objects is very labor intensive, it can be accelerated using appropriate software and computer (and machine learning) aided methods. We took great care to leverage appropriate software and interfaces to conduct our segmentations, but the future holds many possibilities for machine-aided segmentation. This topic will be discussed further in chapter 7.

In regard to acquisition of images, the work in this thesis was conducted on static 2D

images. These same techniques could be applied to 3D images, though manual labeling might be more difficult without the right tools. When trying to analyze cells with 2D images, you only have a cross-section of the cell. With 3D images you get full information about the shape of the cell. 3D imaging also brings much more information to bear on the problem. Many of the issues with indistinct cell membrane or nuclear borders might be resolved with 3D imaging, simply because you have more information to use. This topic will be discussed further in chapter 7.

Following classification and segmentation of desired cell types within our images, we extracted shape and distance features from the individual cells to show we could discriminate between the different cell types. This paradigm of segmenting and classifying cells using a DCNN and then discriminating between them with simpler and more traditional machine learning methods is different than an approach based entirely on a DCNN. We chose this divided approach both because we wanted the data and shape and distance relationships to remain human interpretable and because our specific interest in the activity of many individual cells was not a suitable application for a DCNN. DCNNs excel at hierachlly representing individual objects with collections of features, but they are not good at analyzing relationships between many individual objects.

The work in this chapter motivates the following chapter, which investigates the application of this DCNN model and classification pipeline to human fresh frozen data from inflamed human kidney biopsies.

CHAPTER 5

HUMAN FRESH FROZEN TISSUE

In this chapter, we discuss the application of our DCNN segmentation and cell-type classification analysis to human fresh-frozen data.

5.1 Introduction

While live tissue imaging using two-photon imaging is possible with mice, it is not possible in human subjects. Labeled cells cannot be ethically transferred into a live human lymph node, and lymph nodes cannot be dissected and imaged over time in a live human. Use of the established two-photon imaging in live mice, allows for functional imaging – allowing one to see different cell types and their interactions. However, our goal is to use CDM to obtain implied function from tissue slides. Thus, while we confirmed the use of CDM on mouse images using the known truth on cell types (Ta or Tw) in chapter 4, we now extend that to the analysis to human tissue slides. We use our multichannel images to segment candidate T cell nuclei as we hypothesize a change in shape when they are in cognate interaction with immune dendritic cells. This analysis relies on labeled human data for supervised neural network training and traditional computer vision methods for image labeling and feature analysis. In our deep learning method, we take as input ROIs of with many cell types and output segmented T cell nuclei and dendritic cells. Then we apply machine-learning methods to quantitatively extract characteristics (e.g., area, shape, distance of T cells from dendritic cells) that are subsequently merged using classifiers to yield a likelihood of the cell belonging to different subtypes. The relative amounts of these two cells could potentially yield values that estimate the tubulointerstitial inflammation (TII) grade.

This study used 25 renal biopsies from deidentified patients, obtained from the University of Chicago Human Tissue Resource Center (HTRC), Department of Pathology. The tissue

was fresh frozen in OCT Tissue Plus (Thermo-Fischer) and stored at –80 °C. The study protocol was approved by the University of Chicago Institutional Review Board (IRB#15-0727) and did not require informed consent, as no patient data were used. All human experiments were conducted in compliance with all relevant ethical regulations. Confirmation of the diagnosis of lupus nephritis as well as grading of the severity of TII was performed by a blinded reading nephropathologist (A.C.) as previously described [60]. Additionally, deidentified tonsil samples were utilized from the University of Chicago Pathology Core Facility for antibody testing and validation. Two distinct antibody panels were utilized to stain 3 µm to 4 µm-thick tissue sections; for plasmacytoid dendritic cell (pDC) analysis - CD3 (clone SP7, Abcam or clone CD3-12, AbD Serotec), CD4 (clone YNB46.1.8, Abcam), BDCA2 (clone AC144, Miltenyi), and CD123 (clone 6H6, eBioscience); myeloid dendritic cell (mDC) analysis - CD3, CD4, BDCA1 (clone L161, Beckman Coulter), and CD11c (clone EP1347Y, Abcam). DAPI (Hoechst 33342, Invitrogen) was used with the above to visualize tissue nuclei. Images were acquired using a SP5 or SP8 confocal microscope as described above. In addition to selection of individual ROIs for analysis, selected biopsies underwent tiling, wherein the entire available tissue was imaged and a composite stitched image was obtained based on default manufacturer settings (SP8). All images were stored in .lif format. Lif files were converted to multichannel tif images and used as input for DCNN analysis.

For microtubule organizing center (mTOC) localization quantification discussed below, Three 3 µm-thick severely inflamed lupus nephritis biopsies (TII grade 3) were stained with antibodies to CD3, CD4, BDCA2, and DAPI as per the above protocol. Anti-tubulin (clone YL1/2, Abcam) was added to visualize the mTOC. Automated z-stack protocol images were obtained from regions containing pDCs using the Leica SP8 laser scanning confocal microscope. The images were processed in Fiji using the 3D viewer plug-in [61] to obtain a maximal projection in two dimensions. Manual counting of mTOCs was then performed in a blinded fashion by a single observer (Junting Ai). All $CD3^{+}CD4^{-}$ and $CD3^{+}CD4^{+}$ T

cells abutting a pDC were quantified, along with those featuring an mTOC at the junction between the T cell and APC [18].

5.2 Dataset

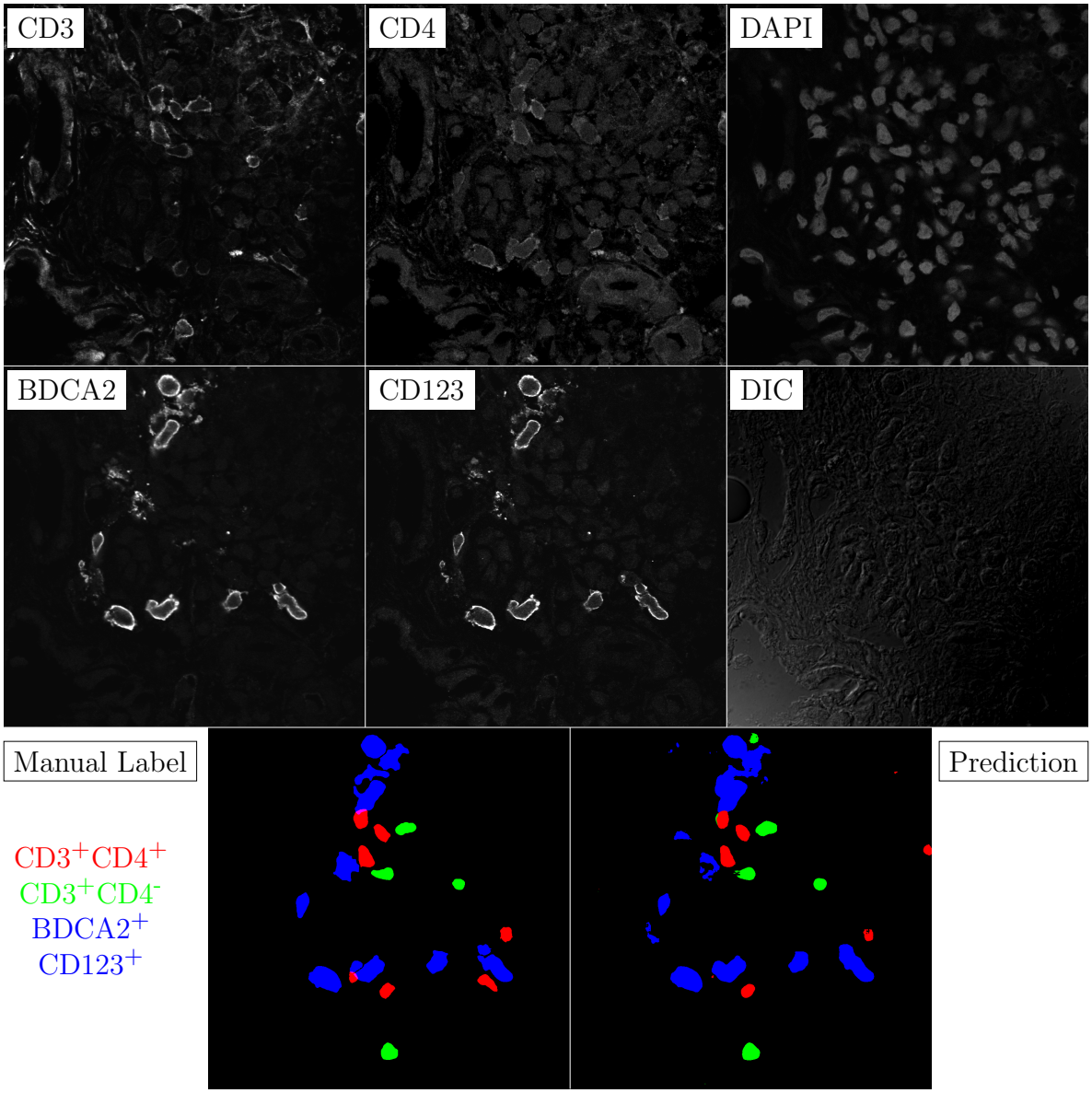

Figure 5.1: **Human fresh frozen ROI:** True ROI matrix size is 1024 × 1024. Depicted ROI matrix size is 1024 × 1024. for images and 1024 × 1024 for labels.

Our human dataset was obtained from renal lupus nephritis (LN) biopsies using con-

focal microscopy with immunofluorescent antibody staining and DAPI staining. Several Immunofluorescent stains were used to stain dendritic cells and T cells in tissue. CD3 and CD4 stains highlighted subtypes of T cells in tissue. BDCA1 and CD11c stains highlighted mDCs in tissue. BDCA2 and CD123 stains highlighted pDCs in tissue. The DAPI stain was used to stain nuclei in the tissue. In addition, the DIC channel was used. Staining and imaging for pDC and mDC dendritic cells were conducted separately. An example of a pDC ROI is shown in figure 5.1. This provided 6 channels of information for analysis for each ROI. Here we want to discriminate between [$CD4^+$, $CD3^+$] T cells and [$CD4^-$, $CD3^+$] T cells in the presence of [$BDCA2^+$, $CD123^+$] pDCs or [$BDCA1^+$, $CD11c^+$] mDCs. We expect these two types of T cells to behave differently in proximity to pDCs and mDCs. Our human dataset has a total of 24 cases with imaging of pDCs and mDCs overlapping for most cases. There are a total of 364 ROIs corresponding to pDC acquisitions and 323 corresponding to mDC acquisitions. Images and cells are tallied in table 5.1 and table 5.2. Manual labels of [$CD4^+$, $CD3^+$] T cells, [$CD4^-$, $CD3^+$] T cells, [$BDCA1^+$, $CD11c^+$] mDCs, and [$BDCA2^+$, $CD123^+$] pDCs were produced for 246 ROIs by a technologist and students in the Giger and Clark labs and validated by Dr. Vladimir Liarski [18].

Segmenting our human data presented similar challenges to our murine data; however, the overall image characteristics of the two datasets was quite different. The presence of two dendritic cell stains was a major difference between the two sets. Also, generally the T cell staining and dendritic cell staining was less definitive. In addition, there was significant background staining of convoluted tubules within the kidney. The cells in the cell walls of these convoluted tubules tend to look similar to the interstitial T cells we want to Identify. Finally, there is significant erroneous cross-staining in our human data. For example, the cell membrane of dendritic cells will often take up the CD3 or CD4 staining agent for T cells. This is not obvious by looking at only the CD3 or CD4 channel, but is obvious when viewing the dendritic stains along with the CD3 or CD4 stains. This is because, while the

cell membrane of the dendrite may uptake CD3 or CD4 as well as the dendritic staining agent, the converse never occurs. T cells never uptake dendritic staining agent. In addition, pDCs express CD4. A human observer viewing all 6 image channels simultaneously is able to make sense of these subtleties. It is more difficult for analytical algorithms to process these images. Deep learning models are suited to this task, however. We design and train our deep convolutional neural network with the goal of the network learning both small-scale and large-scale features in the images and across the image channels, like a human observer does.

	pDC		mDC	
TII Score	Case #	ROI #	Case #	ROI &
0	2	7	2	25
1	5	36	6	62
2	7	42	6	54
3	8	279	8	182
Total	22	364	22	323

Table 5.1: **Human fresh frozen imaged dataset** showing TII grade, case number, ROI number, for pDC and mDC ROIs.

pDC					
TII grade	Case #	ROI #	$CD3^+CD4^+$	$CD3^+CD4^-$	$CD123^+BDCA2^+$
0	0	-	-	-	-
1	1	9	65	22	10
2	1	8	46	20	15
3	5	155	1924	1077	1151
Total	7	172	2035	1119	1176
mDC					
TII Score	Case #	ROI #	$CD3^+CD4^+$	$CD3^+CD4^-$	$CD11c^+BDCA1^+$
0	0	-	-	-	-
1	0	-	-	-	-
2	0	-	-	-	-
3	2	74	473	249	256
Total	2	74	473	249	256

Table 5.2: **Human fresh frozen manually labeled dataset:** showing TII grade, case number, ROI number, and cell counts for $CD3^+CD4^+$, $CD3^+CD4^-$, $CD11c^+BDCA1^+$ cells in mDC ROIs, and $CD123^+BDCA2^+$ in pDC ROIs.

5.3 Multiclass Nuclei Segmentation

To produce usable segmentations from our data, we used a 3D, multiclass deep convolutional neural network to incorporate both spatial and classification information into a single framework. Since the boundaries of a nucleus are often difficult to resolve, the cell membrane stain helps to resolve the boundaries of the nucleus, while also providing classification information. In addition, noise in one channel can be compensated for by information in another. The DCNN used 10 convolutional layers and three maximum pooling layers and a fully connected layer (figure 5.2).

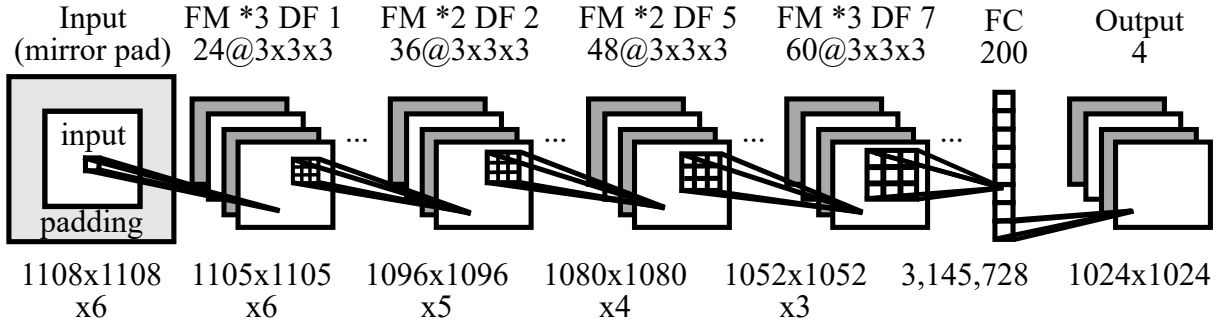

Figure 5.2: **Abbreviated DCNN architecture:** 6 channel input is padded and fed through 4 sets of convolutional layers. Each of the first 3 sets of convolutional layers is followed by a pooling layer. The final set of convolutional layers is followed by a fully connected layer. The fully connected layer is fed into a softmax layer to produce the final prediction. Top row shows feature maps (FM) $\times$ number of layers and convolution dilation factor (DF) of the kernels in the layer. 2nd row shows number of filters per layer with filter kernel size. Bottom row shows input volume size as it passes through DCNN.

Rather than down-sampling our images with each max-pooling layer, we kept our feature maps at original resolution and increased the sparsity of subsequent convolutions. We used a receptive field of view (FOV) of 85 pixels $\times$ 85 pixels $\times$ 6 channels. Receptive FOV refers to the total amount of information from the surrounding image that goes into a prediction at a single pixel. The trainable weights and biases across all the feature maps in the convolutional layers of the DCNN and the fully-connected layer resulted in $\sim$700k trainable parameters, shown to be reasonable in similar studies [42, 43]. As depicted in figure 5.3, the 6 image

channels for each ROI are put into the DCNN, which outputs a 4-class prediction for ($CD3^+$, $CD4^+$) T cells, ($CD3^+$,$CD4^-$) T cells, and ($BDCA2^+$, $CD123^+$) or ($BDCA1^+$, $CD11c^+$) dendritic cells, and background.

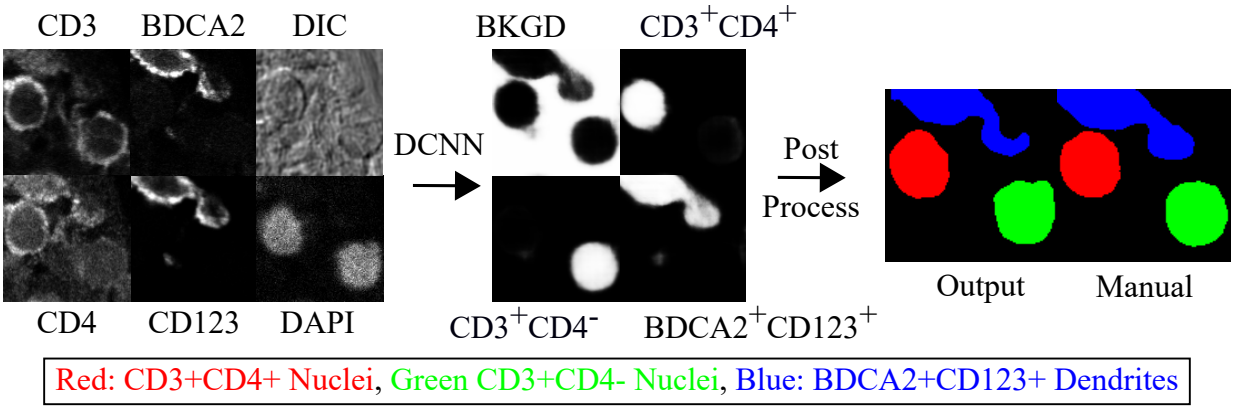

Figure 5.3: **DCNN segmentation pipeline:** (a) DCNN segmentation pipeline: Multi channel input produces a multi cell class probability prediction. Cell class is assigned to each pixel based on maximum predicted probability. Finally objects below a certain size are eliminated to produce final segmentation. Predictions shown in red ($CD3^+CD4^+$), green ($CD3^+CD4^-$), and blue ($CD123^+BDCA2^+$). Pixel size is 0.14 µm × 0.14 µm. True image matrix size is 115 × 115. Depicted image matrix size is 115 × 115. All ROIs embedded in document with lossless DEFLATE compression. Bit-depth is 8.

Following segmentation of the T cell populations and dendritic cells, we compute shape features for every T cell including perimeter, convex perimeter, area, convex area, circularity, minor axis length, major axis length, eccentricity, equivalent diameter, solidity, major/minor axis ratio, and perimeter to circularity ratio. In addition, we compute the distance between each T cell and the closest dendritic cell. This distance is computed by finding the Euclidean distance from every pixel in the T cell to the nearest dendritic cell, and taking the mean of these values. These features are used to train a classifier with the task of discriminating between T cells close to dendrites and T cells far away based on the shape features. The pipeline is depicted in figure 5.4 for human data. The performance of the classifier is evaluated using ROC analysis [62]. Since only a portion of our dataset is manually segmented, we can train and evaluate a classifier on the manually segmented portion and/or the automatically segmented portion. This allows us to assess whether classification results hold for

automatically segmented version of the manual data. This is possible since the entire set of manual data is segmented automatically using 5-fold cross-validation. This avoids training and testing on the same manual data.

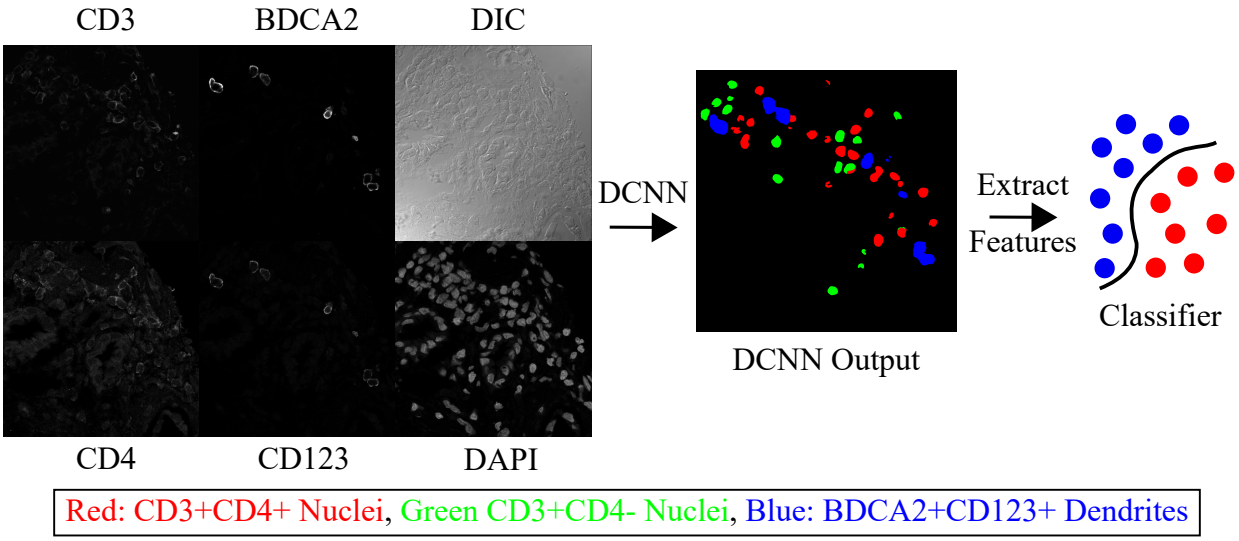

Figure 5.4: **Image analysis pipeline for human data** from raw images to classification is shown. Pixel size is 0.14 µm × 0.14 µm. True image matrix size is 1024 × 1024. Depicted image matrix size is 1024 × 1024. All ROIs embedded in document with lossless DEFLATE compression. Bit-depth is 8.

5.4 Segmentation Results

A single DCNN model was trained on pDC and mDC ROIs to produce predictions for both. While there was substantial image quality difference between the pDC and mDC imaging sets, the DCNN prediction was robust for both sets of images. In the post-processing step, each pixel is assigned the class with the maximum predicted probability. The post-processing step can also include removing objects with anomalous shapes and dropping segmentations with low predicted probability. We assess the performance of the DCNN segmentation by computing sensitivity and specificity of cell detections for automatic segmentation vs. manual truth and IOU for automatic segmentation vs. manual truth. These metrics are computed from a 5-fold model-based cross validation. That is, a model is trained on 4/5 of

manually segmented data and used to predict on the other 1/5 for all 5 permutations. These results are tallied in table 5.3. Finally 6 selected performance examples are shown in nd all examples from the total murine cross validation dataset are shown in appendix F. Figure 5.5 shows results for a larger patch of an ROI, along with an overlap of the DCNN output with the manual truth.

	$CD3^+CD4^+$	$CD3^+CD4^-$	Dendritic cells
Average IOU score ± SD	0.72 ± 0.17	0.68 ± 0.19	0.70 ± 0.20
Sensitivity	0.79	0.53	0.84
Specificity	0.78	0.89	0.90

Table 5.3: **Human fresh frozen DCNN performance statistics** showing average IOU, sensitivity, and specificity for $CD3^+CD4^+$, $D3^+CD4^-$, and dendritic cells.

5.5 Classification Results

Initial classification results on human tissue were performed by Adam Sibley. Results presented below were prepared by Vladimir Liarski. We next sought to understand the relationships between DCs and T cells in human tissue. Specifically, we examined the relative abilities of mDCs and pDC to present antigen to T cells in lupus nephritis TII. Conventionally, mDCs are considered a professional APC [63, 64], and the function of pDCs is thought to be the secretion of interferon-α and other cytokines [65]. However, some subpopulations of pDCs can present antigen to $CD4^+$ T cells [66, 67]. Single longitudinal sections were captured by either tiling HPF across the entire renal biopsy ($n = 10$ biopsies) or capturing those with at least one pDC or mDC ($n = 12$ biopsies). A total of 243 ROIs, from a dataset of 687 ROIs, were manually segmented and used for training the DCNN, as described in (supplementary table B.3). Cells that could be categorically assigned with 90 % or greater confidence were used for the subsequent analysis. 25 lupus nephritis biopsies, scored for degree of TII (integer scale: 0–3 ref. [60]), were stained with antibodies specific for CD11c, BDCA1, BDCA2, CD123, CD3, CD4, and DAPI. In individual longitudinal biopsy sections,

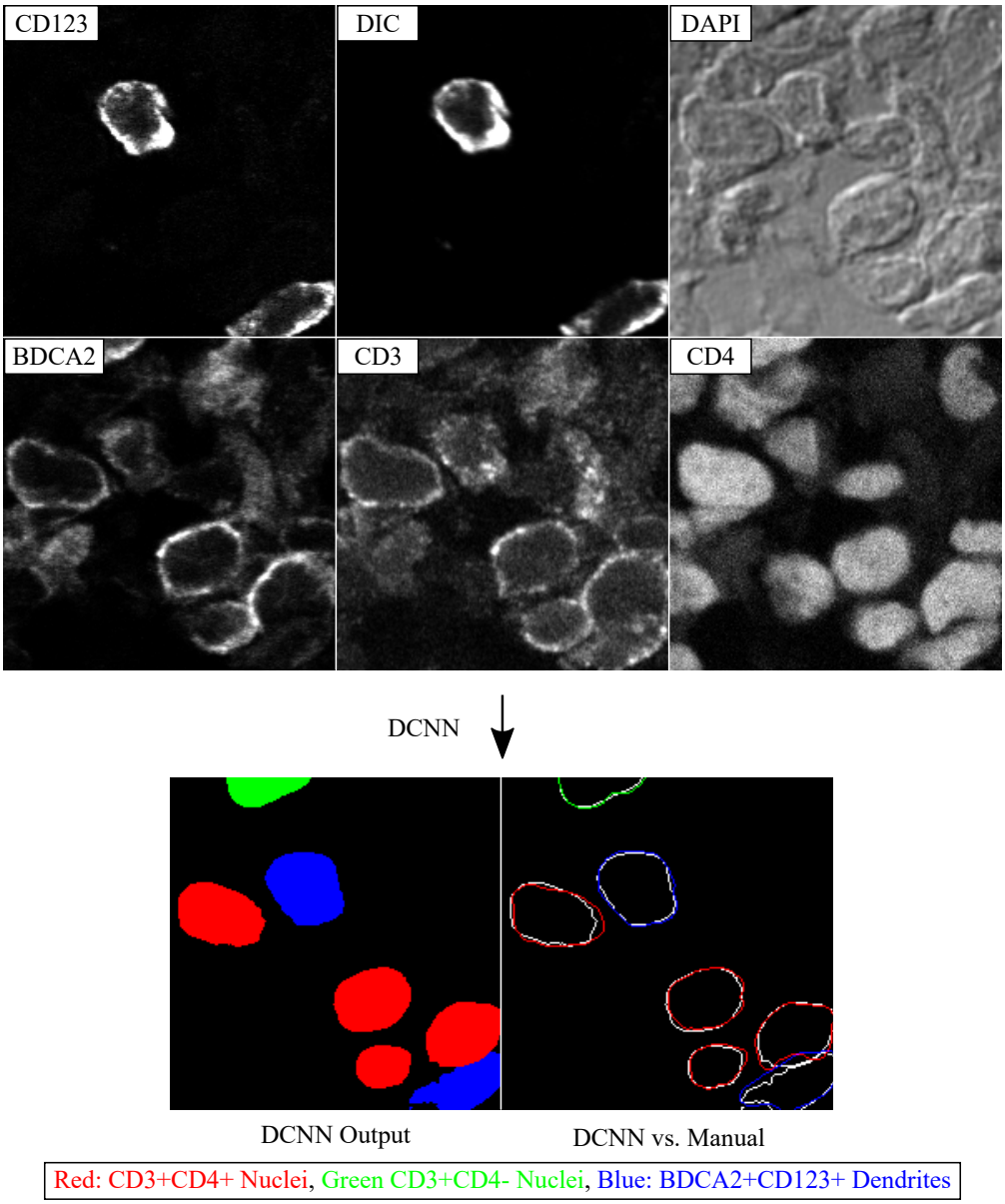

Figure 5.5: **DCNN performance illustration:** Manual label borders are shown for $CD3^{+}CD4^{+}$ (red), $CD3^{+}CD4^{-}$ (green), and $BDCA2^{+}CD123^{+}$ (blue) overlaying the DCNN output in white. Pixel size is 0.14 µm × 0.14 µm. True image matrix size is 200 × 200. Depicted image matrix size is 200 × 200. All ROIs embedded in document with lossless DEFLATE compression. Bit-depth is 8.

the numbers of mDCs ($CD11c^{+}BDCA1^{+}$) and pDCs ($CD123^{+}BDCA2^{+}$) were determined using CDM_3. mDCs were present in all degrees of TII with no statistically significant differences among the groups. In contrast, the majority of pDCs occurred with severe TII

($P < 0.05$)(figure 5.6 and supplementary table B.4). $CD4^+$ T cells appeared more frequent around pDCss than mDCs; $CD4^-$ T cells exhibited similar behavior, although less frequent (figure 5.7 and supplementary figure A.4). However, the number of $CD4^+$ and $CD4^-$ T cells per either DC population was similar when considering whole biopsies or HPFs (supplementary table B.3). These data indicate that pDCs are more common than mDCs with severe TII and are associated with a proportional increase in local T cell infiltrates.

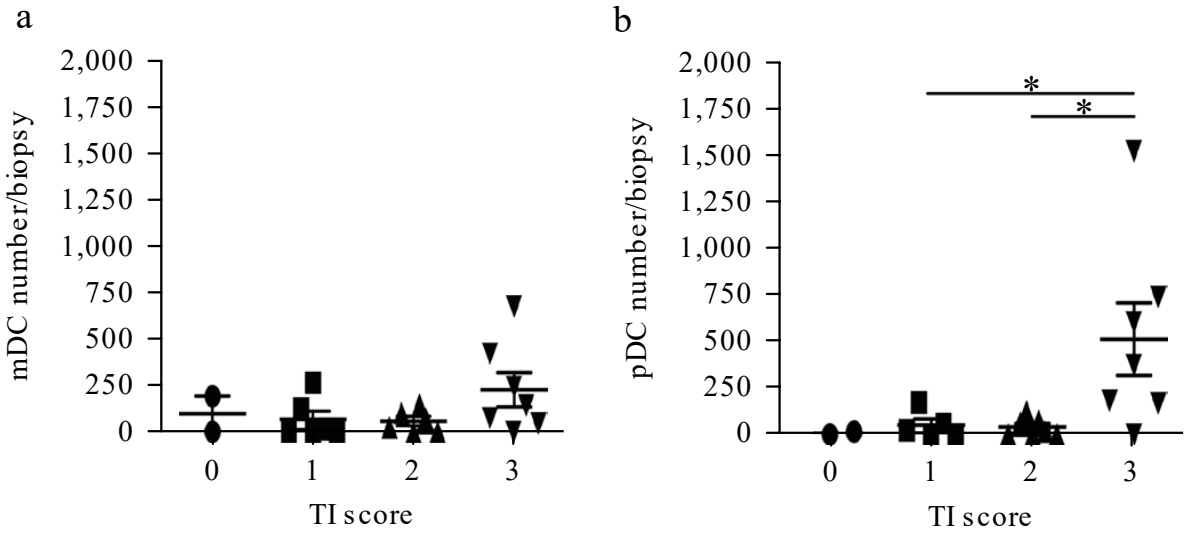

Figure 5.6: **Frequency of mDCs (a) and pDCs (b)** per biopsy by TII (0=none, 1=mild, 2=moderate, 3=severe [60]). Center values denote the mean, and error bars denote s.e.m. $n = 2$ independent experiments. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

Both pDCs and mDCs could just be cosegregated with T cells in areas of active inflammation or they could be contributing to inflammation by locally presenting antigen to $CD4^+$ T cells. To discriminate between these two possibilities, we analyzed the distance and shape characteristics of $CD3^+CD4^+$ T cell nuclei relative to each DC population in biopsies with severe TII ($n = 8$). Local $CD3^+CD4^-$ T cells provide a non-MHC class II restricted bystander control population (figure 5.8).

Analysis of relative distances revealed that $CD3^+CD4^+$ T cells were, on average, closer to both pDCs and mDCs than $CD3^+CD4^-$ T cells (figure 5.9a and supplementary figure A.5).

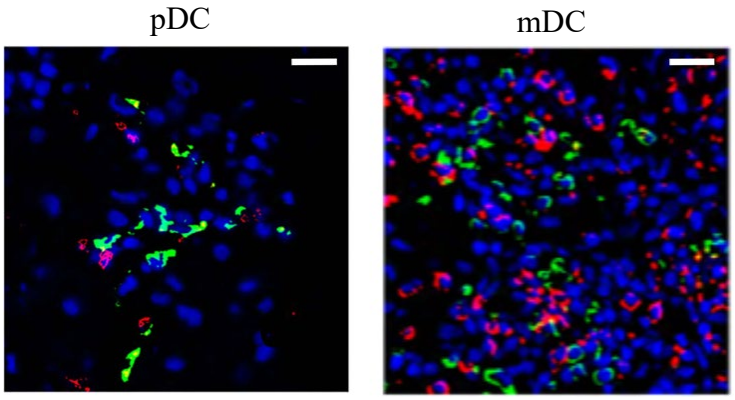

Figure 5.7: **Representative images of pDCs and mDCs** (green) with $CD4^+$ T cells (red) in lupus. Nuclei are blue. Scale bars,40 µm. $n = 2$ independent experiments. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

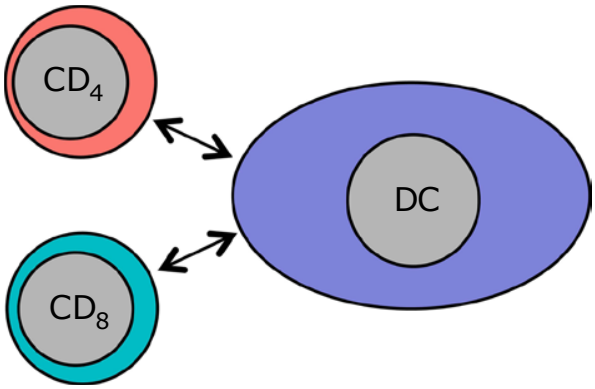

Figure 5.8: **T cell interaction schematic** of $CD3^+CD4^+$ and $CD3^+CD4^-$ T cells relative to DCs. $n = 2$ independent experiments. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

Analysis of T cell nuclear shape revealed differences between $CD3^+CD4^+$ and $CD3^+CD4^-$ cells relative to each DC population, with different measures of T cell shape making differential contributions to accuracy relative to either pDCs or mDCs (figure 5.9b). Examination of T cells relative to pDCs for both convex area and equivalent diameter revealed that some of those $CD3^+CD4^+$ cells close to pDCs tended to be larger on cross-section (figure 5.10a,b and supplementary figure A.3f,g). Similar relationships were observed for T cell equivalent diameter and major axis relative to mDCs (figure 5.105c,d and supplementary figure A.3h,i). In contrast, some $CD3^+CD4^-$ cells tended to get smaller in cross-section as a function of dis-

tance from either DC population. These same trends were observed for 5CC7 and wild-type murine T cells relative to antigen pulsed DCs. However, in contrast to the murine experiment, $CD3^+CD4^-$ cells were larger than $CD3^+CD4^+$ cells when not in proximity to a DC (supplementary figure A.3f–i). These data demonstrate that by examining changes in T cell shape as a function of distance, one can compare T cell populations that are intrinsically different in size.

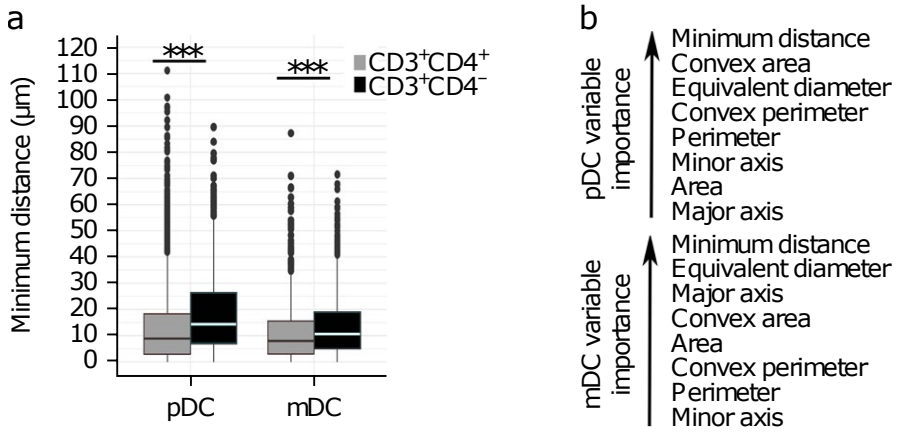

Figure 5.9: **Minimum distance and parameter importance: a**, Minimum distances between indicated DC populations and $CD3^+CD4^+$ (gray) and $CD3^+CD4^-$ (black) T cells. **b**, Hierarchy of contribution of distance and T cell shape parameters to accuracy based on random forest analysis of pDC and mDC datasets. Midlines in box plot (**g**) denote median value, with upper and lower hinges denoting first (Q1) and third (Q3) quartile values, respectively, vertical bars corresponding to values 1.5× the IQR for Q1 and Q3, and dots representing outlier values, not included in the above. $^*P < 0.05$, $^{***}P < 0.0005$, two-sided Mann–Whitney U test. $n = 2$ independent experiments. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

Plotting the composite CDM_3 output of distance and T cell shape features revealed clear discrimination between $CD3^+CD4^+$ and $CD3^+CD4^-$ T cells within 25 µm for both mDCs (figure 5.11a, AUROC = 0.63, 95 % CI: 0.55–0.71, $P < 0.01$) and pDCs (figure 5.11b, AUROC = 0.65, 95 % CI: 0.61–0.69, $P < 0.0005$). Use of minimum distance alone yielded inferior results in cell type discrimination (data not shown; AUROC = 0.63, 95 % CI: 0.50–0.69, $P = 0.05$, and AUROC = 0.57, CI: 0.52–0.61, $P = 0.03$ for $CD3^+CD4^+$ and $CD3^+CD4^-$ T cells within 25 µm of mDCs and pDCs, respectively). These data are consistent with dis-

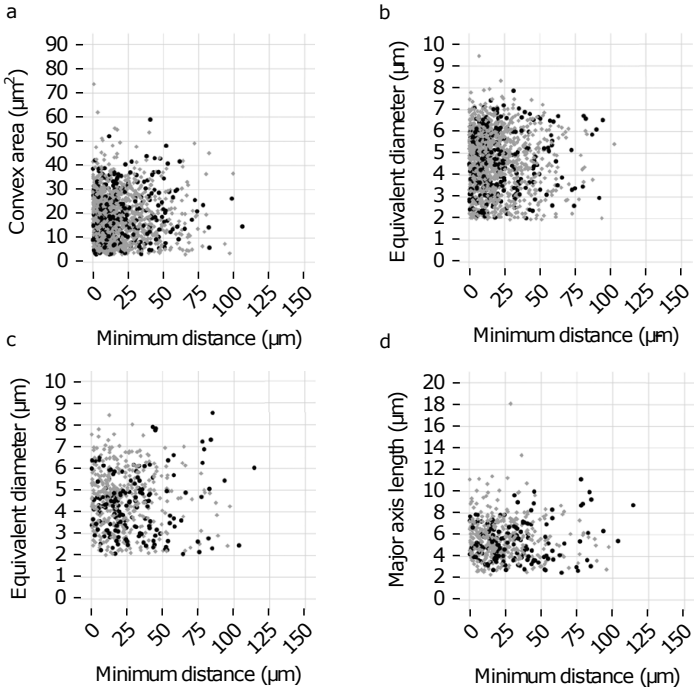

Figure 5.10: **Dot plots of select features: a**,**b** Plot of convex area (**a**) and equivalent diameter **b** per T cell ($CD3^+CD4^+$ (gray) and $CD3^+CD4^-$ (black)) as a function of distance from pDCs ($P < 0.0005$ for both groups at all distances). Random 10 % of total values are plotted for visualization. **c**,**d**, Equivalent diameter (**c**, $P < 0.05$) and major axis (**d**, $P < 0.005$) per T cell as a function of distance from mDCs. Random 10 % of total values plotted for visualization. Benjamini–Hochberg correction with FDR of 5 % was used. $n = 2$ independent experiments. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

tance and T cell shape relationships observed in murine cells and suggest that both pDCs and mDCs present antigen to $CD4^+$ T cells. However, as there are more pDCs than mDCs with severe TII, pDCs appear to make a larger contribution to in situ $CD4^+$ T cell activation.

In peripheral blood, a subpopulation of pDCs expressing the surface markers AXL and SIGLEC6 can function as APCs [67]. However, the lupus intrarenal pDCs are $AXL^-SIGLEC6^-$ (data not shown). Therefore, to confirm that intrarenal pDCs were important APCs in vivo, lupus nephritis biopsies were stained with antibodies specific for CD3, CD4, CD43, BDCA2, and tubulin to visualize the microtubule organizing center, as well as DAPI. We then performed three-dimensional confocal imaging on representative T cell-pDC conjugates. As seen, there is polarization of the TCR-CD3 complex toward the interface with the pDC

(figure 5.11c,d). Likewise, the T cell mTOC is oriented toward the interface with the pDC. In contrast, CD43 is accumulated at the distal pole complex, consistent with a canonical, mature T cell-APC synapse [68, 69]. To quantify mTOC polarization, lupus renal biopsies with pDCs were stained with antibodies specific for CD3, CD4, BDCA2, tubulin, and DAPI, then subjected to confocal microscopy to obtain z-stack images. We scored the number of $CD3^{+}CD4^{+}$ and $CD3^{+}CD4^{-}$ T cells abutting pDCs (186 pDCs across three biopsies) and how many of these T cells had their mTOCs polarized toward the interface with pDCs. $CD3^{+}CD4^{+}$ were almost six times more likely to be abutting pDCs (a two to threefold enrichment compared with total T cell numbers; figure 5.11e and supplementary table B.4). Of these, 40 % had a mTOC polarized toward pDCs, whereas only 10 % of abutting $CD3^{+}CD4^{-}$ cells did. Therefore, there was an overall 24-fold difference in conjugate rate between $CD4^{+}$ and $CD4^{-}$ T cells ($P < 0.0005$). These data confirm that CDM_3 accurately detects T cell-pDC conjugates [18].

5.6 Discussion and Conclusions

Our application of a DCNN to identify multiple cell types in human tissue samples stained for multiple T cell and dendritic cell types showed strong peformance in classification of cell types and segmentation of dendritic cell bodies and T cell nuclei. Segmenting and classifying cell types in multi channel images is in general a very difficult problem. Our approach for these human fresh frozen confocal microscopy images was to identify all the subsets of cells in the images we were interested in, label the images manually, and use a custom DCNN to segment and classify the cell types. In this method, the DCNN treated each cell as having multiple channel dimensions, and classification and segmentation was drawn from that information.

Following classification and segmentation of desired cell types within our images, we extracted shape and distance features from the individual cells to show we could discriminate

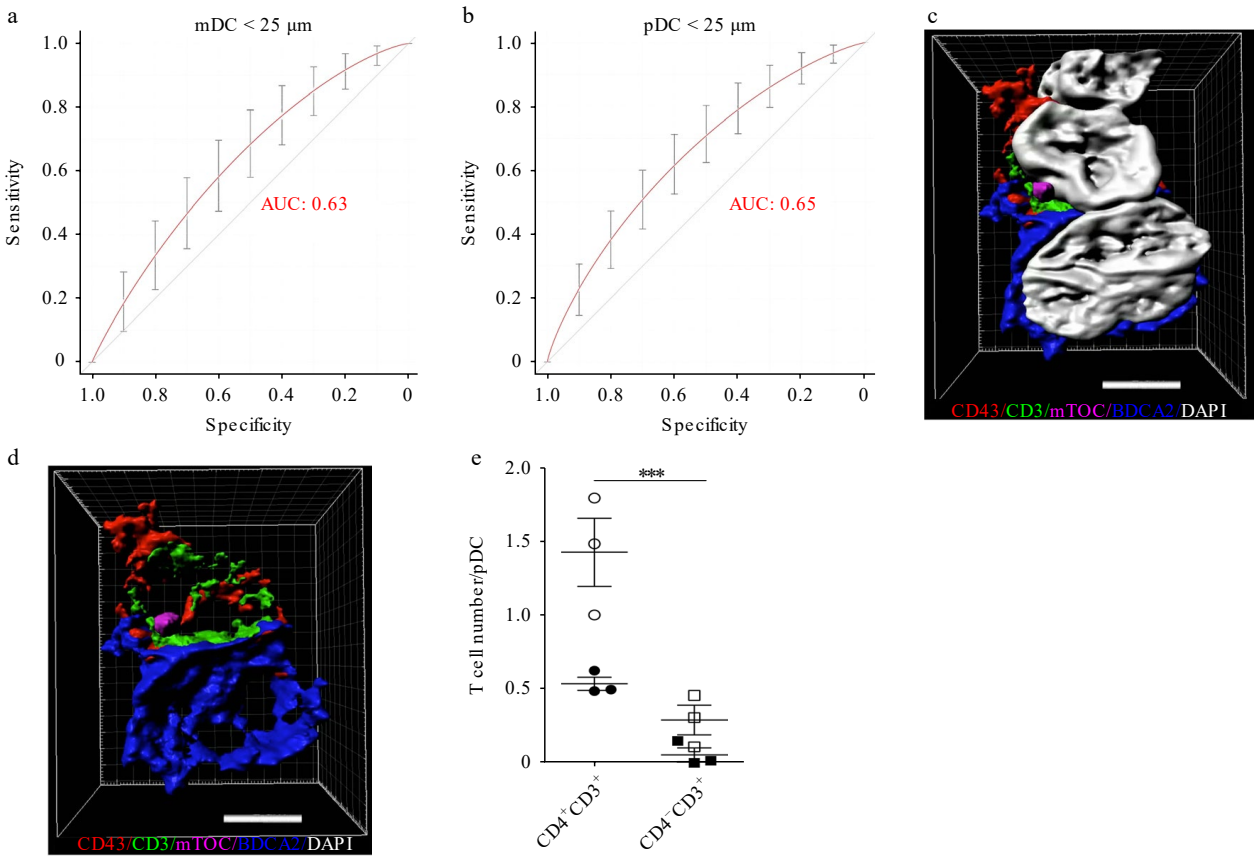

Figure 5.11: **CDM$_3$ nuclear segmentation analysis:** **a**,**b**, Sensitivity and specificity of CDM$_3$ for discriminating between CD3$^+$CD4$^+$ and CD3$^+$CD4$^-$ cells for mDC (**a**, 95 % CI: 0.55–0.71, $P < 0.005$) and pDC (**b**, 95 % CI: 0.61–0.69, $P < 0.0005$) datasets at a minimum distance cutoff of ¡25 µm. **c**,**d**, Three-dimensional surface reconstructions from lupus TII, utilizing Imaris software. T cell (top) abutting a pDC is shown with (**c**) (left) and without (**d**) (right) nuclei, as stained by DAPI. Immunofluorescent antibody staining is indicated. Scale bars, 4 µm. **e**, Analysis of mTOC localization in three highly inflamed (score: 3) lupus nephritis biopsies. Z-stack acquisition was performed with examination of pDC:CD3$^+$CD4$^+$ and pDC:CD3$^+$CD4$^-$ T cell pairs to determine mTOC localization. Open circles denote all counted T cells, and black circles signify the number of T cells with mTOCs polarized toward the abutting DC, as indicated. ***$P < 0.0005$, two-sided Mann–Whitney U test. Diagonal lines in **a**,**b** denote AUROC of 0.5, which represents a random probability ($P = 0.5$). Error bars denote cross-validation error (**a**,**b**) or s.e.m. (e). $n = 2$ independent experiments for all panels. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

between the different cell types. This paradigm of segmenting and classifying cells using a DCNN and then discriminating between them with simpler and more traditional machine learning methods is different than an approach based entirely on a DCNN. We chose this

divided approach both because we wanted the data and shape and distance relationships to remain human interpretable and because our specific interest in the activity of many individual cells was not a suitable application for a DCNN. DCNNs excel at hierachically representing individual objects with collections of features, but they are not good at analyzing relationships between many individual objects.

Our application of a DCNN to segment multiple cell types in fresh frozen human kidney biopsies showed strong classification and segmentation performance. We were also able to take these segmented cells and accurately discriminate between them using unsupervised classification techniques. This technique opens up new ways for interrogating tissue samples at the individual cell level, using the power of deep convolutional neural networks.

CHAPTER 6

MASK R-CNN

This chapter details our implementation of Mask region convolutional neural network (R-CNN), the motivations for doing so, and the results of our implementation.

6.1 Introduction

The DCNN implementation discussed in chapter 3, chapter 4, and chapter 5 and published in Liarski et al [18] performed strongly, but it had a number of issues that are synonymous with deep convolutional neural networks for object recognition. To improve performance we adapted Mask R-CNN [70] for training and testing on our images.

6.2 Mask R-CNN Implementation

Mask R-CNN was chosen to adapt to segment and classify our cell datasets because it has become the tool of choice for object segmentation and classification in recent years and much research surrounds it. It is an extension of Faster R-CNN [71] with an independent branch of the DCNN for predicting masks only. This has the effect of substantially increasing segmentation results for objects by decoupling object segmentation and object classification. During the course of this thesis work was attempted to design a similar mask branch to a DCNN, but is was unsuccessful. The authors note in the Mask R-CNN paper [70] that proper design of the mask branch is critical to achieving high performance.

The code framework for training the Mask R-CNN was obtained from the Tensorpack Neural Net Training Interface [72] and the Mask R-CNN example [73] was adapted to our data. The ResNet 101 backbone network [74] was used with a feature pyramid network [75] and Cascade R-CNN [75].

Labeling of truth for the DCNN was conducted as described in chapter 3. Image labels were processed using the Common Objects in Context (COCO) dataset Python API [76] which stores images labels as run length encoded objects. Operations such as determining intersection over union can be performed directly on the run length encoded objects, enhancing computational efficiency and decreasing computer memory requirements. This is especially important in the case of large image arrays with thousands of cells, which are processed in this work. Training was conducted on Uchicago Midway2 cluster with 1-20 GPUs using the distributed deep learning distributed training library Horovod [77]. Training performance was monitored using the Tensorboard interface in Tensorflow (figure 6.1). During training extensive data augmentation was used including channel independent brightness augmentation, image rotation, image flips, and image scaling.

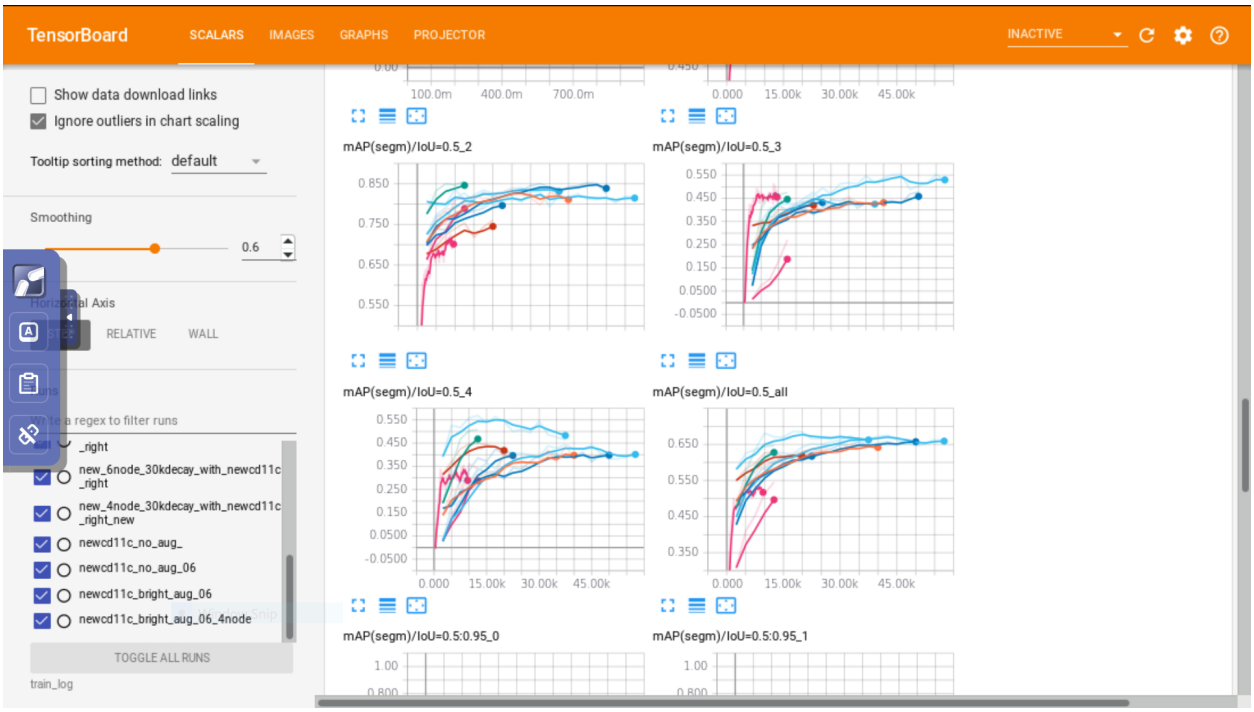

Figure 6.1: Tensorboard training monitoring interface showing performance metrics during training

6.3 Application to Paraffin Embedded Images

We developed and tested our Mask R-CNN implementation on paraffin embedded images from lupus biopsies. These images have some different properties from the fresh frozen images discussed previously, but were largely similar to the human fresh frozen data. These images were stained for DAPI, CD3, CD4, CD20, BDCA2, and CD11c similarly to the human fresh frozen data, but with one additional dendritic cell stain. Similar to the human fresh frozen data, we were interested in identifying T cell nuclei with the CD3 and CD4 stains and their proximity to dendritic cells stained by CD20, BDCA2, and CD11c. We desired to classify and segment $CD3^+CD4^+$ and $CD3^+CD4^+$ T cell nuclei and $CD20^+$, $BDCA2^+$, and $CD11c^+$ dendritic cells. Some image channels presented with signal intensity and image quality that was better than the fresh frozen human data (DAPI, CD3, CD4, CD20) and others poorer (BDCA2, CD11c). The paraffin dataset also had large tiled biopsies (up to $18488 \times 8364 \times 6$) which presented increased computer processing challenges than smaller ($1024 \times 1024 \times 6$) ROIs used in the human fresh frozen data.

While tiling of individually acquired image ROIs often leads to image artifacts that hurt analysis, the quality of the tiling for these images was very strong. For this reason we chose to perform analysis on the entire tiled biopsies. When trying to determine the activity of cell networks, analyzing as large a region as possible is obviously preferable. When looking at smaller regions of tissues, you lose information about what is happening at the border of the region. A cell at the border of the imaged region might be influenced by a cell just beyond the border, without any way of the observer knowing. Particularly, recalling the name of our analysis method, cell distance mapping, it is apparent that effective judgements about the distance at which cells have an effect on one another are limited by the size of an image ROI. If it is assumed that an ROI has a certain number of cell types, distributed in a certain density, and that the surrounding ROIs have a similar set of cell types and density, it is probably only reasonable to make judgements about the distance relationships between

cells if the distance being considered is an appropriately small fraction of an ROI.

However, this is not always the case in real imaging situations where much of the tissue surrounding an ROI may be empty and have no or limited bearing on the contents of a ROI. If an ROI can be imaged such that it captures most relevant information in the tissue neighborhood, it should be sufficient for analysis. Also, while more (accurate) information is always better, most cell interaction and recognition likely happens in close cell proximity. While information about cells at far distances may not be that useful, knowledge about a cell's location in the tissue macro structure might be. Analyzing images at the level of entire biopsies makes information about a cell's location in the larger structure of an organ possible. So in the spirit of gleaning as much information as possible from our data, we developed analysis software while analyzing the paraffin data to determine the shape of all cells identified in a tiled biopsy image, as well as all the distance relationships between all cells in an image, at all distances. We hope that in the future this analysis paradigm might allow analysis of information at various scales in the images, from the individual cell level to the organ level.

6.3.1 Dataset

A total dataset of 59 variously-sized paraffin embedded tiled biopsies was imaged and stained for DAPI, CD3, CD4, CD20, BDCA2, and CD11c. The full (downsampled) set of tiled biopsies is shown in appendix H. While the size of the individual biopsies varied, the equivalent number of 1024×1024 ROIs was 718.23 by total pixel count. The tiled biopsies are composed of individual 1024×1024 ROIs but since the edges of these ROIs overlap in the tiling process, the total count of these ROIs is inflated. A total of 289 $1024 \times 1024 \times 6$ paraffin embedded ROIs were labeled for $CD3^+CD4^+$ (2422 cells), $CD3^+CD4^+$ (1880 cells), $CD20^+$ (1710 cells), $BDCA2^+$ (820 cells), and $CD11c^+$ (596 cells) (labeled individual ROIs are not shown).

6.3.2 Training

For training and testing the DCNN for paraffin, of the 289 total ROIs, 90% of the ROIs were used for training and 10% were used for testing. The DCNN model was tested during training periodically (generally every 2,500 training iterations) on the full testing set to observe performance on Tensorboard. All cell types were independently monitored for average precision (AP) and average recall (AR) to monitor for overfitting of the DCNN on the testing set. Generally the degree to which overfitting occurred on the testing set was highly dependent on the learning rate and the degree of data augmentation taking place during training.

Augmenting of the brightness values in the training images particularly dramatically increased the length of time it took to train the DCNN; however, brightness augmentation seemed to increase performance. This is likely due to the extreme nature of the brightness augmentation that took place during training. A less extreme brightness augmentation scheme that more accurately modeled variations in the image acquisition process would likely lead to better performance in a faster fashion; however, it is difficult to simply model such variations and we took a naive approach. This naive approach (discussed in 6.2) explored extremes of image brightness augmentation and likely resulted in the model being presented with unrealistic image samples for much of its training time. Despite these unrealistic examples, it seems that enough realistic variations in image brightness were introduced to increase segmentation and classification performance.

$CD3^+CD4^+$, $CD3^+CD4^-$, and $CD20^+$ cells were most consistent in segmentation performance. $BDCA2^+$ and $CD11c^+$ cells were less consistent in their segmentation performance during training. This is not surprising as these cell subsets consisted of substantially fewer cells than the other subsets. Also, the BDCA2 and CD11c channels also generally had lower signal intensity and labeling of these cell types was less certain. The $BDCA2^+$ and $CD11c^+$ types seemed to often peak in segmentation and classification performance and then decline during training. This is likely because the model was overfitting to the more prevalent

$CD3^+CD4^+$, $CD3^+CD4^-$, and $CD20^+$, thus decreasing performance on the less prevalent ones.

For the testing and results to follow, we selected the trained model with the best performance across all cell types at the point during its training where performance was maximized. The DCNN model was chosen at 15,000 iterations for the data presented below. This is substantially less than the approximately 100,000 iterations the DCNN models were often trained to. This point was chosen to maximize $BDCA2^+$ and $CD11c^+$ at their peak. Generally $CD3^+CD4^+$, $CD3^+CD4^-$, and $CD20^+$ cell types plateaued in performance and increased minimally beyond that plateau.

6.3.3 Testing

To produce segmentations for the large tiled biopsy images, patches were extracted for each tiled image and then stitched back together. First, tiled biopsy images by extracting overlapping $1024 \times 1024 \times 6$ patches from the tiled images with a patch overlap region of 100 pixels. Tiled images were padded with zeros on the right and bottom edges such that an integer number of overlapping patches could be extracted. Image patch pixel values were cast to a 32 bit floating point value and divided by 2^{16}; however, a value of 2^{12} might have been more appropriate as this was the maximum integer value acquired by the imaging sensor. This was an oversight as the images were stored in 16 bit unsigned integer format. This should not affect results since the trained DCNN model was treated in the same way and the small loss of numerical precision should be insignificant.

Image patches were then passed through the DCNN using the same settings discussed in the training section, except that no data augmentation was necessary. Object predictions for the image patches were stitched back together by considering all objects in mutually overlapping regions and removing redundant objects that overlapped with an IOU> 0.3. No special consideration was taken for which object to remove. Simply all overlapping objects

except for one were removed. Finally all objects with a classification score below 0.5 were removed from the final results.

6.3.4 Results

Image segmentation results were assessed using AP and AR at an IOU threshold of 0.5. While this is in contrast to our previous assessment of sensitivity, specificity, and IOU for detection of cell types, it is more in line with metrics used by the general computer vision community. Results from the testing set for AP were $CD3^{+}CD4^{-}$ (0.72), $CD3^{+}CD4^{+}$ (0.85), $CD20^{+}$ (0.82), $BDCA2^{+}$ (0.42), and $CD11c^{+}$ (0.54) for an unweighted average of 0.67 across cell types. Results from the testing set for AR were $CD3^{+}CD4^{-}$ (0.59), $CD3^{+}CD4^{+}$ (0.67), $CD20^{+}$ (0.56), $BDCA2^{+}$ (0.33), and $CD11c^{+}$ (0.33) for an unweighted average of 0.50 across cell types. Results for a single large tiled paraffin embedded biopsy are shown in figures 6.2 and 6.3. Testing results for the full dataset of 59 paraffin tiled images are shown in appendix H.

6.4 Mask R-CNN Comparison with Fresh Frozen Results

Also in order to compare the aforementioned CDM_3 method for human fresh frozen labeled data data (table 5.2) against the new Mask R-CNN method, sensitivity, specificity, and IOU were assessed for both and compared by Madeleine Durkee. Performance is tallied in table 6.1. CDM_3 is clearly outperformed by Mask R-CNN in most of the metrics.

6.5 Discussion and Conclusions

Use of Mask R-CNN to analyze cell distance relationships shows great promise in our paraffin tiled biopsy images. Comparison with our previous DCNN segmentation technique shows improved results. Mask R-CNN and original CDM_3 DCNN are at different ends of the

IOU Threshold=0.25	$CD3^+CD4^+$	$CD3^+CD4^-$	DC	All (avg)
Sensitivity				
CDM_3 Net	0.79	0.53	0.84	0.72
Mask R-CNN: 130k	0.79	0.67	0.81	0.76
Mask R-CNN: 200k	0.80	0.67	0.80	0.76
Specificity				
CDM_3 Net	0.78	0.89	0.90	0.86
Mask R-CNN: 130k	0.91	0.88	0.95	0.91
Mask R-CNN: 200k	0.91	0.91	0.96	0.93
IOU				
CDM_3 Net	0.72 ± 0.17	0.68 ± 0.19	0.70 ± 0.20	0.71 ± 0.18
Mask R-CNN: 130k	0.81 ± 0.12	0.82 ± 0.13	0.79 ± 0.14	0.81 ± 0.13
Mask R-CNN: 200k	0.82 ± 0.11	0.82 ± 0.12	0.80 ± 0.14	0.81 ± 0.12

Table 6.1: **Fresh frozen CDM_3 vs. Maskrcnn performance:** Sensitivity, specificity, and IOU are shown for three cell types for CDM_3 DCNN and for Mask R-CNN trained to 130k and 200k iterations. (Comparison performed by Madeleine Durkee)

spectrum when it comes to DCNN design. Mask R-CNN has many different stages and deconstruction of the segmentation and classification problem into different parts, while the CDM_3 architecture is much simpler and self-contained in its design. This technique furthers our goal of interrogating tissue samples at the individual cell level, using the power of deep convolutional neural networks.

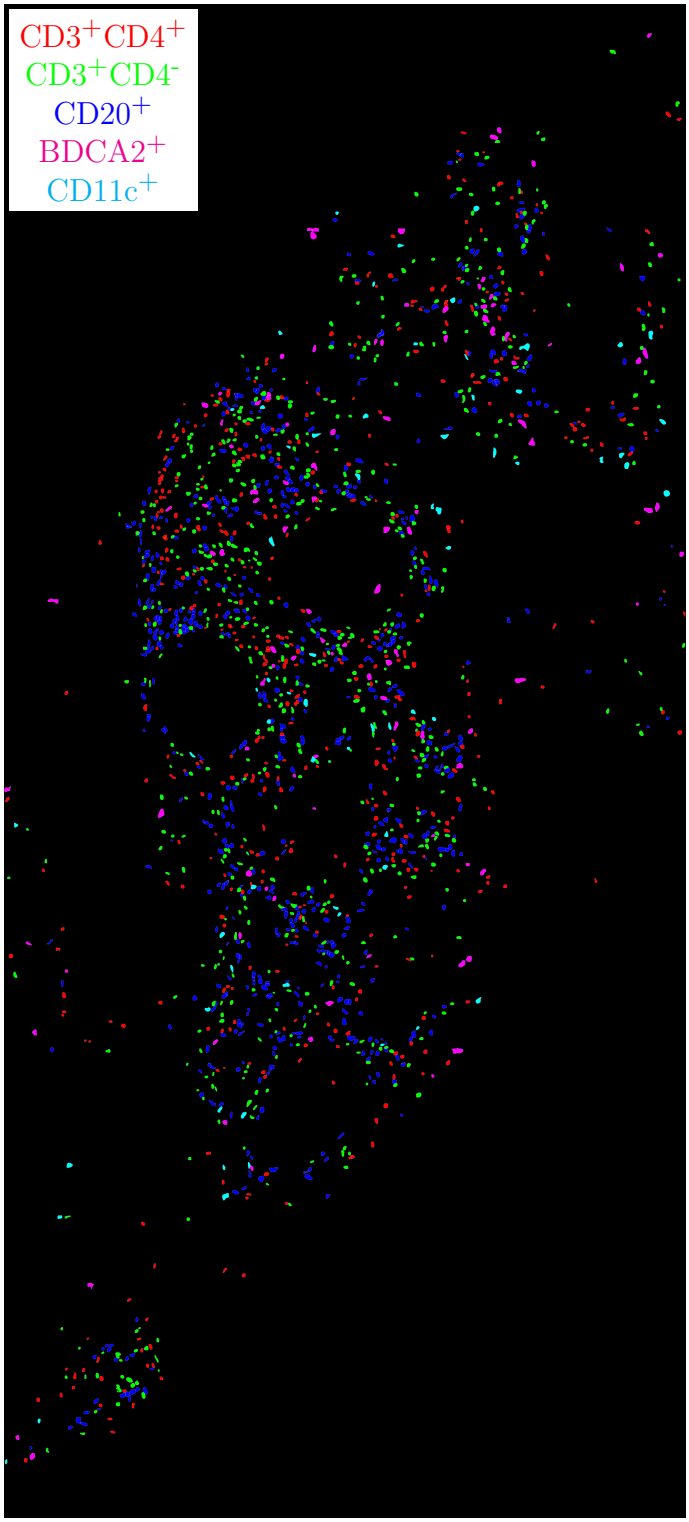

Figure 6.2: **Predicted paraffin cell labels for 6.3:** True label matrix size is 18488×8364. Depicted label matrix size is 10000×4524. Pixel size is $0.14\,\mu\text{m} \times 0.14\,\mu\text{m}$

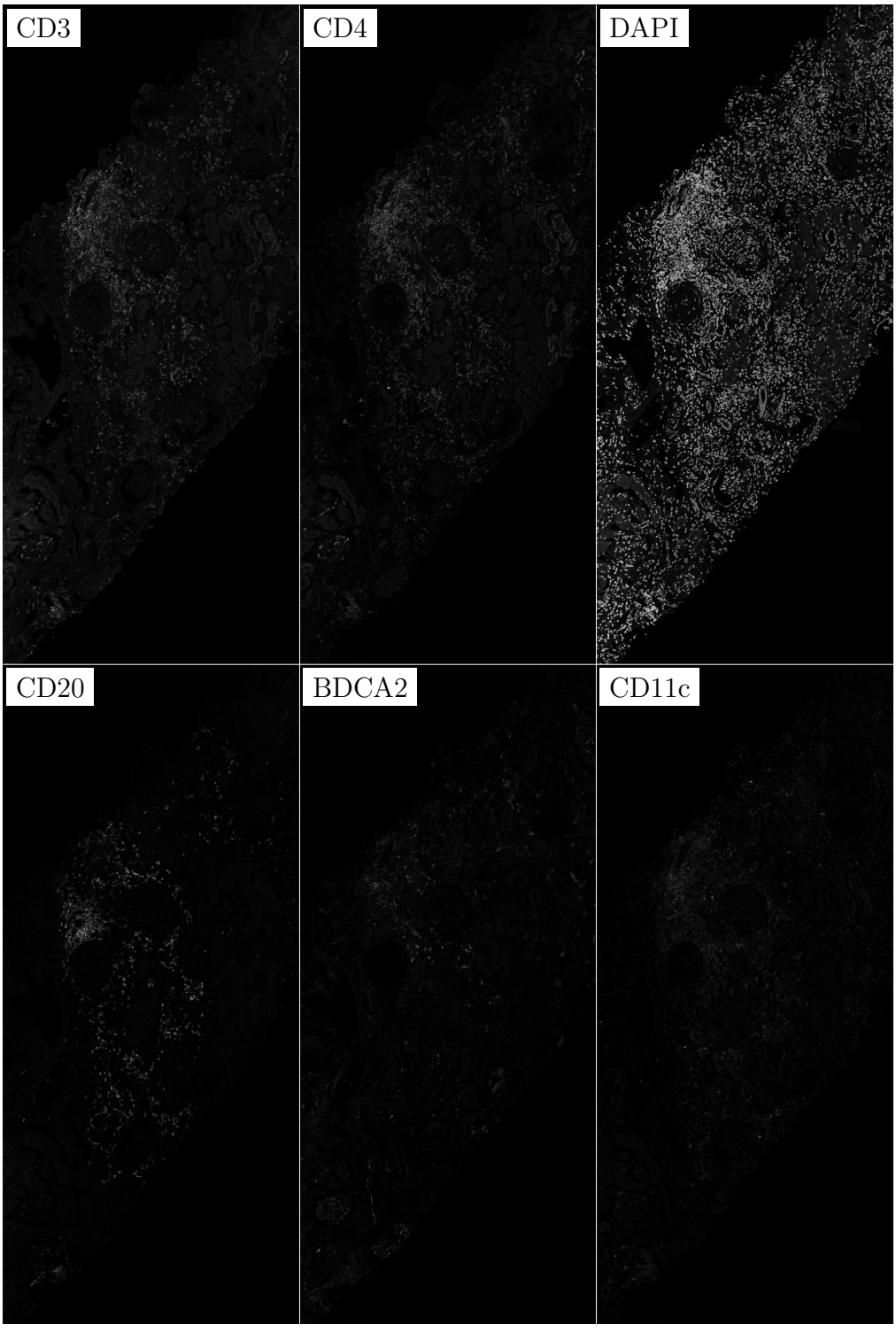

Figure 6.3: **Image channels for 6.2:**. True image channel matrix size is 18488×8364. Depicted image channel matrix size is 2000×904. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$

CHAPTER 7

SUMMARY AND CONCLUSIONS

This thesis has discussed the implementation of a cellular analysis method that bridges supervised and unsupervised machine learning methods. While deep convolutional neural networks are the tool du jour in science and visual recognition, we have attempted in thesis to drive home the point that DCNNs have their place in visual recognition systems and they are not a one-size-fits-all method for visual problems or otherwise. By restraining DCNNs to their domain of competence in this work, we have shown that they can produce results that are both accurate and interpretable to a human being.

The study of immune cell dynamics in tissue is challenging and presen ts several trade-offs. Techniques such as TPEM provide direct visualization of precisely labeled and characterized cell populations and quantifies their interactions with other cells over time. However, TPEM can only be used on some tissues at limited organ depths. It is also difficult to apply directly to the study of human disease [15, 17]. In this work, we demonstrate that by utilizing multi-channel confocal microscopy and a novel analytic pipeline that we term CDM_3, we approach the performance measures of TPEM in discriminating stable cognate from non-cognate T cell–DC interactions in mice. These data indicate that a quantitative analysis of many static two-dimensional images can approximate much of the information obtained from time-lapse three-dimensional videos of the same phenomenon. Additionally, as CDM_3 is performed on single-plane images of fixed tissue, we could use CDM_3 to study human disease and identify important in situ APCs.

Beyond applicability to the study of both human disease and animal models, not readily accessible to TPEM, CDM_3 offers several additional advantages. It provides higher throughput, which enables larger sample sizes and robust statistical confidence. CDM_3 does not require experimental manipulations to label cells and, therefore, can be performed on native systems, minimizing experimental artifact. Furthermore, it can be performed on any

tissue and at any depth. CDM_3 cannot assess the kinetics of cellular interactions and, thus, cannot be used to address some questions. Regardless, for many experimental applications in animal models, CDM_3 might be a preferred approach.

However, the major advantage of CDM_3 is that it can be performed on single sections from frozen tissue. Therefore, it is ideally suited to human studies and can be applied to biopsies that are routinely obtained as part of clinical care. In lupus nephritis, we were able to identify putative in situ cognate T cell–DC interactions and, furthermore, assign relative importance of different DC populations in presenting antigen to $CD4^+$ T cells. Overall sensitivities and specificities in human disease were lower than those observed in transgenic mice. This could reflect inherent differences in the imaging approaches used in mice and humans. More likely, the lower signal observed in human disease reflects underlying heterogeneity in the cells and antigens driving in situ adaptive immunity. However, by capturing more events than is practical with TPEM, we overcame this heterogeneity to make statistically robust conclusions.

The sequential use of a DCNN followed by a TNN in CDM_3 is critical for application to human disease. In mice, we can dictate the antigen specificity of T cell populations and, therefore, use a DCNN to best discriminate between antigen-specific and non antigen-specific interactions with DCs. We have an ideal training set. In humans, we cannot control T cell antigen specificity. However, by using mice to learn how to extract fundamental features of how T cells interact with DCs, which are conserved across mice and humans [29–35], we can apply the CDM_3 pipeline to human samples and identify stable cognate interactions. Thus, our sequential pipeline helps overcome major limitations of machine learning, including transparency, in-depth understanding of how machine-learning algorithms function, and what newly created predictors or intermediate variables represent. These ambiguities often make it difficult to relate machine learning outputs to meaningful biological variables or behavior [78, 79].

In CDM_3, we designed and trained a highly customized DCNN from scratch because our application was so different from main-stream applications designed for photographic images. Particularly, we were interested in making predictions on multichannel images where there was significant bleed-through between stain channels. We also wanted to classify nuclei based on surface stains that were not spatially colocalized between image channels. To do this, we used 3D convolutional kernels to relate information across the channel dimension. This not only allowed us to relate non-spatially colocalized information between image channels, but it also allowed the DCNN to use features from one image channel to mitigate noise or ambiguity in another. Additional customization of our DCNN included selecting an appropriate number of neural network parameters for the CNN such that it was complex enough to learn features in our labeled image dataset, but not so complex that it overfit and simply memorized the training set. Training images were also carefully selected to be representative of the whole dataset and representative of variations in image quality. The classes of training examples used were balanced to ensure equal performance in classifying the different cell types. The scale of the dilated convolutions used in the layers of the neural network were selected such that they acted on spatial scales where features relevant to classifying the image data were most likely to exist. Data were normalized using the standard score (z-score) applied separately to each channel of each ROI. These design decisions were made so that learned features from the training data were robust and transferrable to other experimental data. Our studies were limited to single-plane confocal images and simple pairwise cell–cell interactions. However, CDM_3 is adaptable to three-dimensional images that would provide more definitive measures of in situ APC function. Furthermore, the general approach illustrated by CDM_3 is applicable to the study of complex cellular networks containing three or more cell types. A quantitative understanding of the cellular architectures of in situ adaptive immunity and inflammation in human disease will provide new insights into the pathogenic mechanisms of autoimmunity and features of immunity effective against cancer [18].

Following the development of the CDM_3 machine learning pipeline and DCNN, we also adapted Mask R-CNN to analyze our images and applied them to new tiled paraffin-embedded biopsies from inflamed human kidney biopsies. This approach showed great promise and integrated an object proposal network and a separate branch of the DCNN for predicting object masks.

Future directions for this research will likely be focused on better incorporating unsupervised methods on image data into the machine learning pipeline. While DCNNs perform well on explicitly labeled image data, they do less well at predicting on image data in an unsupervised fashion. In addition, more advanced interfaces for human labeling of image data will likely play a role by making the generation of label data for images easier. The following sections will discuss some of these future possibilities.

7.1 Data Labeling And Interface

Explicit data labeling for DCNN segmentation of individual cells remains a very time consuming task. Some ways to ease the task of segmentation in future work would be better computer interfaces, iterative data labeling, and algorithm aided segmentation.

7.1.1 Interface

The interface is critical to the task of manual segmentation. It should allow the user to view the information in multiple channels easily, outline cells, and assign cells to specific groups. Icy and ImageJ software were used for this work but both required significant file and software manipulation to produce our labeled datasets. Preference for the two interfaces seemed to be split among data labelers. There were positives and negatives to both softwares. Icy's primary strength was allowing seamless zooming and syncing of axes between image channels. ImageJ has similar functionality but this functionality seems to have bugs in ImageJ and we judged it unusable. Labeling by toggling between image channels in an image stack in

ImageJ seemed to work well for data labelers. This is different from viewing the channels side-by-side, but the action of toggling between channels was generally preferred by the data labelers for visualizing the data.

ImageJ and Icy both lack an easy way to assign cell segmentations to different classes and to visualize these classes. Due to this limitation, we simply outlined the cell segmentations for different classes on different channels in the image. This was sufficient for segmenting the handful of cell classes we were interested in, but might not be sufficient if we had wanted to look at more combinations of cell classes.

A more specialized interface for object classification and segmentation could make data labeling for our different cellular classes significantly easier. There are many open source tools for the problem of object segmentation interfaces. One prominent open source tool we experimented with using but never fully implemented is the OpenCV Computer Vision Annotation Tool (CVAT,https://github.com/opencv/cvat). If properly customized to our labeling task this would likely be superior to ImageJ and Icy as an image segmentation tool.

7.1.2 Iterative Data Labeling

Iterative data labeling is a promising approach to easing the manual segmentation task. Iterative data labeling can work because easy segmentation tasks are learned quicker than hard ones. For example, a DCNN might learn to correctly segment 90% of cells in all images with only a few training images, while getting the remaining 10% of cells correct might take significantly more labeled training images. By training on a few labeled images, predicting on a few images, correcting the predictions, and then training on the original labeled images plus the corrected predictions over many iterations, a large set of labeled images can be built up. This pardigm was implemented during the course of this work with ImageJ, but final results were not presented in this thesis.

Iterative labeling is not without drawbacks. Particularly there is a strong possibility that

the data labels produced by the DCNN will influence the decisions of human labelers. More advanced formulations of iterative data labeling are possible, including actively correcting segmentation predictions during the DCNN training process. Such an implementation would be technically complex, however, and would require appropriate software.

7.1.3 Interactive Algorithm Aided Segmentation

Using algorithm aided manual segmentation is another way to ease the burden of the manual segmentation task. For example, a user could draw a box around the relevant cell to be segmented, and a DCNN or other image processing algorithm could propose a segmentation mask for the cell. The user could then either accept the automatic segmentation or correct it. There are also potentially interactive ways in which the user could correct the cell segmentation mask, instead of having to redraw the whole mask by hand.

7.1.4 Data Aided Manual Segmentation

Data-aided decision making has seen much interest in image processing, including radiology. The basic idea is to offer quantiative data to a decision maker based on image data.

All of the segmentation and classification work in this thesis has relied exclusively on an experienced labeler manually drawing the cell or nucleus border and then assigning the cell or nucleus to a specific class based on the intensity of the different cell membrane stain (and also knowledge of tissue structure and image noise characteristics). While the task of outlining a cell or nucleus border is fairly atomic (meaning a reasonable determination can be made based on the edges and borders of an individual cell), the task of determining the class of a cell based on cell membrane staining intensity is much more subjective.

A data-aided approach to this manual segmentation task would incorporate consideration of quantitative image features in the manual classification task. The most obvious quantitative feature is a normalized intensity along the cell border for each cell in the la-

beled data set; however, incorporating quantitative features into the data labeling decision making process is a lofty goal and likely to be very difficult to achieve in practice. Quantitative image features would need to be robust between images. Achieving robustness of intensity based features is very difficult as intensity can vary signficantly between different image ROIs due to variations in imaging conditions. In addition, extracting the intensity of the border around each segmented cell would require an extremely exact determination of the cell border. Significant deviation of the border between cells would throw off the quantitative feature.

From a computer's perspective, human manual labelers are very bad at drawing consistent borders around objects. This is particularly true for the character of data in stained cellular imaging. A consistent border around a stained cell border might lie at the peak of the intensity distribution of the border, inside of the border or outside of the border. The border might also lie at the half maximum of the peak intensity value of the border on the inside or outside of the border. Or the border could lie where the peak intensity of the cell membrane drops to zero or to the background value of the image. For a simple computer algorithm to produce usable output, this determination must be made very consistently. This issue with human border determination also highlights another strength of DCNNs for segmentation: they can make sense of inconsistent and variable data. The machinery of the DCNN is making statistical decisions, so the final border determination it makes for all cells should be some consistent average of the human manual labeler's inconsistent decisions.

7.2 Divorcing Segmentation Tasks

Our DCNN implementations in this work relied on explicitly labeling and focusing in on the cell classes we cared about. This paradigm allowed us to reduce the burden of manual labeling and unite cell classification and segmentation into a single framework. It also allowed us to leverage the image information from all channels simultaneously for the segmentation

and classification tasks. Despite these advantages, there are other possible solutions to the problem including segmenting all cell membranes non-specifically or segmenting all nuclei non-specifically.

7.2.1 Segmenting All Cell Membranes Non-Specifically

Segmenting all cell membranes in an image non-specifically would involve combining the information from all cell membrane stain channels to produce segmentations for all visible cell membranes. This approach has the advantage of being very generalizable: It could potentially be applied to any number of different cell membrane stains, depending on how the information is combined. The main disadvantage with this approach is it requires segmenting all cell membranes in an image, including those that may not be very interesting to the task at hand. While this does increase the amount of manual labeling substantially per ROI, it also potentially decreases the total number of ROIs that need to be segmented.

This approach also has another advantage: cell membranes surround nuclei. If cell membranes could be non-discriminately segmented, nuclei could then be segmented fairly easily in a second step. A disadvantage of this formulation is that nuclei without a visible cell border would be excluded from the image segmentation process. This is probably not an issue with many analyses, as nuclei that do not belong to a specific cell membrane stain may not be of interest.

The combination of information from different cell membrane stained channels is not trivial and would greatly affect the generalizability of the method to different staining tasks. For example, if a conventional DCNN were trained to segment an image with 6 cell membrane channels, it would then only be applicable to images with 6 cell membrane channels. In addition, the trained DCNN model would be exquisitely sensitive to the image characteristics of each image channel. If the image characteristics were significantly different in any subsequent imaging, the conventional DCNN model would certainly fail to perform on this

new data. This sensitivity to image channel characteristics could be ameliorated with a comprehensive data augmentation scheme, but such a scheme is generally very computationally intensive and difficult to implement in practice. It is important to note that these limitations only apply to a conventional or naive DCNN model where the input data is treated as a multi-channel stack of images with filters applied across the channel dimension. There are other DCNN formulations that may be appropriate to this task including multi-task learning, weight sharing between different branches of the DCNN, and parameter sharing between different channels of the DCNN.

The simplest way to combine a number of different cell membrane channels is to add them all together before processing them with the DCNN. Again, this may cause issues with differences in brightness values between different channels. So a more reasonable solution may be to normalize them with the standard score on a per channel basis and then add them together. This could also cause problems if one type of cell is positive in (say) only a single channel and another type of cell is positive in very many channels. The best solution may be to take the maximum value across all channel dimensions to produce a final image that only display the brightest standard score normalized values. This single channel image could then be passed to a DCNN and all cell membranes could be segmented.

Alternatively, the full stack of image channels could be passed to a DCNN which uses channel-wise parameter sharing in the first convolutional layer. This would mean that the first convolutional layer would use the same 2D convolutional kernel across all image channels. This would highlight the same features across all channels to produced a summed 2D feature map in the output.

Following the non-specific segmentation of all cell membranes in the image, the individual cells could then be separated into different classes using a secondary classifcation DCNN (or a branch of the same DCNN) that uses the multi channel stain information to separate cells into different classes based on all stain channels.

7.2.2 Segmenting Nuclei Non-Specifically

Similar to the task of segmenting all cell membranes non-specifically, cell nuclei could all be segmented non-specifically. This approach has the advantage of being a very general biological problem that a lot of people are interested in. Large scale labeled nuclei datasets are also becoming available. Non-specific nuclei segmentation also does not need to be done using only the nuclear image channel. Additional image stain channels can also be leveraged to discriminate ambigious borders between nuclei. Additional stain channels can be joined in a similar fashion as described above in the cell membrane non-specific segmentation subsection.

Segmenting nuclei in this way would not be as useful in determining the segmentation mask of the cell membrane as cell membrane masks are in determing the nuclear segmentation mask. Nuclei are obviously contained by cell membranes, but cell membranes do not contain nuclei. It might not be necessary to segment the cell membranes all applications, however. Segmented nuclei could still be separated into different classes using a classification DCNN. This classification DCNN would take as input the image matrix around a nucleus, which contains the cell membrane.

7.3 Large Scale Public Datasets

Large public datasets are becoming available for a wide variety of segmentation tasks across disciplines, but are somewhat less prominent in biological imaging. Variations in imaging techniques and biological questions can make it difficult to generalize datasets to different problems. The Kaggle Nuclei Data Science Bowl in 2018 (https://www.kaggle.com/c/data-science-bowl-2018) addressed the problem of segmenting nuclei in images and provided a large labeled dataset. Many of these nuclear images are not directly applicable to the type of nuclei segmentation tasks tackled in this work. Particularly the Kaggle labeled nuclei dataset contained nuclear images that were much easier to segment the the general character

of the images used in this work. However, due to the very general nature of segmenting cells in biological imagery, it can be expected larger datasets will become available, particularly for the relatively straightforward task of image segmentation. DCNN datasets and DCNN models for segmenting biological datasets across a wide range of tasks will likely become available in the near future. They will certainly be commoditized in a similar way to the trajectory that photographic image datasets for deep learning have followed.

7.4 Semi-Supervised and Unsupervised Object Segmentation and Classification

Most of the deep learning applications for image segmentation and classification discussed in this document are fully supervised formulations of machine learning. For example, a segmentation method to generate object segmentation masks where only bounding boxes were present was presented in [59]. This method used a novel weight transfer function to approach the problem of segmentation mask generation in a partially supervised fashion. A method for fully unsupervised clustering of deep learning based features is presented in [80] and could useful for segmentation of cellular imagery as well.

7.5 3D Imaging and Video

Most of the data presented in this work involved 2D static confocal microscopy imagery. While 3D imaging or video can only be acquired under certain circumstances, it brings more information to bear on the problem of determining cell shape and distinguishing different cells from one another. A limitation of 2D static imaging for shape analysis is that tissue slices are cross sections of a cell and don't give you and idea of the entire cell shape. Most of the techniques in this work are applicable to 3D imaging and would likely perform better on 3D imaging due to the additional information available in the 3D format.

REFERENCES

[1] Abul Abbas, Andrew H. Lichtman, and Shiv Pillai. *Cellular and Molecular Immunology*. Elsevier, 2012. URL https://www.elsevier.com/books/cellular-and-molecular-immunology/abbas/978-0-323-47978-3.

[2] Tanja A Schwickert, Randall L Lindquist, Guy Shakhar, Geulah Livshits, Dimitris Skokos, Marie H Kosco-Vilbois, Michael L Dustin, and Michel C Nussenzweig. In vivo imaging of germinal centres reveals a dynamic open structure. *Nature*, 446:83, jan 2007. URL https://doi.org/10.1038/nature05573.

[3] Christopher D C Allen, Takaharu Okada, and Jason G Cyster. Germinal-Center Organization and Cellular Dynamics. *Immunity*, 27(2):190–202, aug 2007. ISSN 1074-7613. doi: 10.1016/j.immuni.2007.07.009. URL https://doi.org/10.1016/j.immuni.2007.07.009.

[4] Rimpei Morita, Nathalie Schmitt, Salah-Eddine Bentebibel, Rajaram Ranganathan, Laure Bourdery, Gerard Zurawski, Emile Foucat, Melissa Dullaers, SangKon Oh, Natalie Sabzghabaei, Elizabeth M Lavecchio, Marilynn Punaro, Virginia Pascual, Jacques Banchereau, and Hideki Ueno. Human Blood CXCR5+CD4+ T Cells Are Counterparts of T Follicular Cells and Contain Specific Subsets that Differentially Support Antibody Secretion. *Immunity*, 34(1):108–121, jan 2011. ISSN 1074-7613. doi: 10.1016/j.immuni.2010.12.012. URL https://doi.org/10.1016/j.immuni.2010.12.012.

[5] Yahui Peng, Yulei Jiang, Vladimir M Liarski, Natalya Kaverina, Marcus R Clark, and Maryellen L Giger. Computerized image analysis of cell-cell interactions in human renal tissue by using multi-channel immunoflourescent confocal microscopy. In *Proc.SPIE*, volume 8315, feb 2012. URL https://doi.org/10.1117/12.910373.

[6] Vladimir M Liarski, Natalya Kaverina, Anthony Chang, Daniel Brandt, Denisse Yanez, Lauren Talasnik, Gianluca Carlesso, Ronald Herbst, Tammy O Utset, Christine Labno, Yahui Peng, Yulei Jiang, Maryellen L Giger, and Marcus R Clark. Cell distance mapping identifies functional T follicular helper cells in inflamed human renal tissue. *Science translational medicine*, 6(230):230ra46, apr 2014. ISSN 1946-6242 (Electronic). doi: 10.1126/scitranslmed.3008146. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4129446/.

[7] Mark J Miller, Sindy H Wei, Ian Parker, and Michael D Cahalan. Two-Photon Imaging of Lymphocyte Motility and Antigen Response in Intact Lymph Node. *Science*, 296 (5574):1869 LP – 1873, jun 2002. doi: 10.1126/science.1070051. URL http://science.sciencemag.org/content/296/5574/1869.abstract.

[8] Mark J Miller, Olga Safrina, Ian Parker, and Michael D Cahalan. Imaging the Single Cell Dynamics of CD4+ T Cell Activation by Dendritic Cells in Lymph Nodes. *The Journal of Experimental Medicine*, 200(7):847 LP – 856, oct 2004. doi: 10.1084/jem.20041236. URL http://jem.rupress.org/content/200/7/847.abstract.

[9] Thorsten R Mempel, Sarah E Henrickson, and Ulrich H von Andrian. T-cell priming by dendritic cells in lymph nodes occurs in three distinct phases. *Nature*, 427(6970): 154–159, 2004. ISSN 1476-4687. doi: 10.1038/nature02238. URL https://doi.org/10.1038/nature02238.

[10] Sabine Stoll, Jérôme Delon, Tilmann M Brotz, and Ronald N Germain. Dynamic Imaging of T Cell-Dendritic Cell Interactions in Lymph Nodes. *Science*, 296(5574): 1873 LP – 1876, jun 2002. doi: 10.1126/science.1071065. URL http://science.sciencemag.org/content/296/5574/1873.abstract.

[11] Ronald N Germain, Ellen A Robey, and Michael D Cahalan. A decade of imaging cellular motility and interaction dynamics in the immune system. *Science (New York, N.Y.)*, 336(6089):1676–1681, jun 2012. ISSN 1095-9203. doi: 10.1126/science.1221063. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3405774/.

[12] Andrius Masedunskas, Oleg Milberg, Natalie Porat-Shliom, Monika Sramkova, Tim Wigand, Panomwat Amornphimoltham, and Roberto Weigert. Intravital microscopy. *BioArchitecture*, 2(5):143–157, sep 2012. ISSN 1949-0992. doi: 10.4161/bioa.21758. URL https://doi.org/10.4161/bioa.21758.

[13] Judith Secklehner, Cristina Lo Celso, and Leo M Carlin. Intravital microscopy in historic and contemporary immunology. *Immunology & Cell Biology*, 95(6):506–513, jul 2017. ISSN 0818-9641. doi: 10.1038/icb.2017.25. URL https://doi.org/10.1038/icb.2017.25.

[14] Sixian You, Haohua Tu, Eric J Chaney, Yi Sun, Youbo Zhao, Andrew J Bower, Yuan-Zhi Liu, Marina Marjanovic, Saurabh Sinha, Yang Pu, and Stephen A Boppart. Intravital imaging by simultaneous label-free autofluorescence-multiharmonic microscopy. *Nature Communications*, 9(1):2125, 2018. ISSN 2041-1723. doi: 10.1038/s41467-018-04470-8. URL https://doi.org/10.1038/s41467-018-04470-8.

[15] Demirhan Kobat, Nicholas G Horton, and Chris Xu. In vivo two-photon microscopy to 1.6-mm depth in mouse cortex. *Journal of Biomedical Optics*, 16(10):1–5, oct 2011. URL https://doi.org/10.1117/1.3646209.

[16] Elijah Yew, Christopher Rowlands, and Peter T C So. Application of Multiphoton Microscopy in Dermatological Studies: a Mini-Review. *Journal of innovative optical health sciences*, 7(5):1330010, jan 2014. ISSN 1793-5458. doi: 10.1142/S1793545813300103. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4112132/.

[17] Daniel T Fisher, Jason B Muhitch, Minhyung Kim, Kurt C Doyen, Paul N Bogner, Sharon S Evans, and Joseph J Skitzki. Intraoperative intravital microscopy permits the study of human tumour vessels. *Nature Communications*, 7:10684, feb 2016. URL https://doi.org/10.1038/ncomms10684.

[18] Vladimir M Liarski, Adam Sibley, Nicholas van Panhuys, Junting Ai, Anthony Chang, Domenick Kennedy, Maria Merolle, Ronald N Germain, Maryellen L Giger, and Marcus R Clark. Quantifying in situ adaptive immune cell cognate interactions in humans. *Nature Immunology*, 2019. ISSN 1529-2916. doi: 10.1038/s41590-019-0315-3. URL https://doi.org/10.1038/s41590-019-0315-3.

[19] Juan Carlos Stockert and Alfonso Blazquez-Castro. Chapter 6: Fluorescence Instrumental Techniques. In *Fluorescence Microscopy in Life Sciences*, pages 180–184. Bentham Science Publishers, 2017.

[20] Michael Y Gerner, Wolfgang Kastenmuller, Ina Ifrim, Juraj Kabat, and Ronald N Germain. Histo-cytometry: a method for highly multiplex quantitative tissue imaging analysis applied to dendritic cell subset microanatomy in lymph nodes. *Immunity*, 37 (2):364–376, aug 2012. ISSN 1097-4180. doi: 10.1016/j.immuni.2012.07.011. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3514885/.

[21] Yury Goltsev, Nikolay Samusik, Julia Kennedy-Darling, Salil Bhate, Matthew Hale, Gustavo Vazquez, Sarah Black, and Garry P Nolan. Deep Profiling of Mouse Splenic Architecture with CODEX Multiplexed Imaging. *Cell*, 174(4):968–981.e15, aug 2018. ISSN 1097-4172. doi: 10.1016/j.cell.2018.07.010. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6086938/.

[22] Leeat Keren, Marc Bosse, Diana Marquez, Roshan Angoshtari, Samir Jain, Sushama Varma, Soo-Ryum Yang, Allison Kurian, David Van Valen, Robert West, Sean C Bendall, and Michael Angelo. A Structured Tumor-Immune Microenvironment in Triple Negative Breast Cancer Revealed by Multiplexed Ion Beam Imaging. *Cell*, 174(6): 1373–1387.e19, sep 2018. ISSN 0092-8674. doi: 10.1016/j.cell.2018.08.039. URL https://doi.org/10.1016/j.cell.2018.08.039.

[23] Charlotte Giesen, Hao A O Wang, Denis Schapiro, Nevena Zivanovic, Andrea Jacobs, Bodo Hattendorf, Peter J Schüffler, Daniel Grolimund, Joachim M Buhmann, Simone Brandt, Zsuzsanna Varga, Peter J Wild, Detlef Günther, and Bernd Bodenmiller. Highly multiplexed imaging of tumor tissues with subcellular resolution by mass cytometry. *Nature Methods*, 11:417, mar 2014. URL https://doi.org/10.1038/nmeth.2869.

[24] Elham Azizi, Ambrose J Carr, George Plitas, Andrew E Cornish, Catherine Konopacki, Sandhya Prabhakaran, Juozas Nainys, Kenmin Wu, Vaidotas Kiseliovas, Manu Setty, Kristy Choi, Rachel M Fromme, Phuong Dao, Peter T McKenney, Ruby C Wasti, Krishna Kadaveru, Linas Mazutis, Alexander Y Rudensky, and Dana Pe'er. Single-Cell Map of Diverse Immune Phenotypes in the Breast Tumor Microenvironment. *Cell*, 174 (5):1293–1308.e36, aug 2018. ISSN 0092-8674. doi: 10.1016/j.cell.2018.05.060. URL https://doi.org/10.1016/j.cell.2018.05.060.

[25] Arnon Arazi, Deepak A Rao, Celine C Berthier, Anne Davidson, Yanyan Liu, Paul J Hoover, Adam Chicoine, Thomas M Eisenhaure, A Helena Jonsson, Shuqiang Li, David J Lieb, Edward P Browne, Akiko Noma, Danielle Sutherby, Scott Steelman,

Dawn E Smilek, Patti Tosta, William Apruzzese, Elena Massarotti, Maria Dall'Era, Meyeon Park, Diane L Kamen, Richard A Furie, Fernanda Payan-Schober, Jill P Buyon, Michelle A Petri, Chaim Putterman, Kenneth C Kalunian, E Steve Woodle, James A Lederer, David A Hildeman, Chad Nusbaum, David Wofsy, Matthias Kretzler, Jennifer H Anolik, Michael B Brenner, Nir Hacohen, and Betty Diamond. The immune cell landscape in kidneys of lupus nephritis patients. *bioRxiv*, page 363051, jan 2018. doi: 10.1101/363051. URL `http://biorxiv.org/content/early/2018/07/07/363051.abstract`.

[26] Stéphane Chevrier, Jacob Harrison Levine, Vito Riccardo Tomaso Zanotelli, Karina Silina, Daniel Schulz, Marina Bacac, Carola Hermine Ries, Laurie Ailles, Michael Alexander Spencer Jewett, Holger Moch, Maries van den Broek, Christian Beisel, Michael Beda Stadler, Craig Gedye, Bernhard Reis, Dana Pe'er, and Bernd Bodenmiller. An Immune Atlas of Clear Cell Renal Cell Carcinoma. *Cell*, 169(4):736–749.e18, may 2017. ISSN 1097-4172. doi: 10.1016/j.cell.2017.04.016. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5422211/`.

[27] Qianqian Zhang, Hehua Dai, Karim M Yatim, Khodor Abou-Daya, Amanda L Williams, Martin H Oberbarnscheidt, Geoffrey Camirand, Christopher E Rudd, and Fadi G Lakkis. CD8+ Effector T Cell Migration to Pancreatic Islet Grafts Is Dependent on Cognate Antigen Presentation by Donor Graft Cells. *The Journal of Immunology*, 197(4):1471 LP – 1476, aug 2016. doi: 10.4049/jimmunol.1600832. URL `http://www.jimmunol.org/content/197/4/1471.abstract`.

[28] Yannick Simoni, Etienne Becht, Michael Fehlings, Chiew Yee Loh, Si-Lin Koo, Karen Wei Weng Teng, Joe Poh Sheng Yeong, Rahul Nahar, Tong Zhang, Hassen Kared, Kaibo Duan, Nicholas Ang, Michael Poidinger, Yin Yeng Lee, Anis Larbi, Alexis J Khng, Emile Tan, Cherylin Fu, Ronnie Mathew, Melissa Teo, Wan Teck Lim, Chee Keong Toh, Boon-Hean Ong, Tina Koh, Axel M Hillmer, Angela Takano, Tony Kiat Hon Lim, Eng Huat Tan, Weiwei Zhai, Daniel S W Tan, Iain Beehuat Tan, and Evan W Newell. Bystander CD8+ T cells are abundant and phenotypically distinct in human tumour infiltrates. *Nature*, 557(7706):575–579, 2018. ISSN 1476-4687. doi: 10.1038/s41586-018-0130-2. URL `https://doi.org/10.1038/s41586-018-0130-2`.

[29] Noa Beatriz Martín-Cófreces, Francesc Baixauli, and Francisco Sánchez-Madrid. Immune synapse: conductor of orchestrated organelle movement. *Trends in cell biology*, 24(1):61–72, jan 2014. ISSN 1879-3088. doi: 10.1016/j.tcb.2013.09.005. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4347664/`.

[30] Michael L Dustin and Jay T Groves. Receptor signaling clusters in the immune synapse. *Annual review of biophysics*, 41:543–556, 2012. ISSN 1936-1238. doi: 10.1146/annurev-biophys-042910-155238. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4000727/`.

[31] Rémi Lasserre and Andrés Alcover. Microtubule dynamics and signal transduction at the immunological synapse: new partners and new connections. *The EMBO journal*,

31(21):4100–4102, nov 2012. ISSN 1460-2075. doi: 10.1038/emboj.2012.276. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3492735/.

[32] Colin R F Monks, Benjamin A Freiberg, Hannah Kupfer, Noah Sciaky, and Abraham Kupfer. Three-dimensional segregation of supramolecular activation clusters in T cells. *Nature*, 395(6697):82–86, 1998. ISSN 1476-4687. doi: 10.1038/25764. URL https://doi.org/10.1038/25764.

[33] Michael L Dustin, Michael W Olszowy, Amy D Holdorf, Jun Li, Shannon Bromley, Naishadh Desai, Patricia Widder, Frederick Rosenberger, P.Anton van der Merwe, Paul M Allen, and Andrey S Shaw. A Novel Adaptor Protein Orchestrates Receptor Patterning and Cytoskeletal Polarity in T-Cell Contacts. *Cell*, 94(5):667–677, sep 1998. ISSN 0092-8674. doi: 10.1016/S0092-8674(00)81608-6. URL https://doi.org/10.1016/S0092-8674(00)81608-6.

[34] Marie Tourret, Sarah Guégan, Karine Chemin, Stéphanie Dogniaux, Francesc Miro, Armelle Bohineust, and Claire Hivroz. T Cell Polarity at the Immunological Synapse Is Required for CD154-Dependent IL-12 Secretion by Dendritic Cells. *The Journal of Immunology*, 185(11):6809 LP – 6818, dec 2010. doi: 10.4049/jimmunol.1001501. URL http://www.jimmunol.org/content/185/11/6809.abstract.

[35] Nicolas Blanchard, Maud Decraene, Kun Yang, Francesc Miro-Mur, Sebastian Amigorena, and Claire Hivroz. Strong and Durable TCR Clustering at the T/Dendritic Cell Immune Synapse Is Not Required for NFAT Activation and IFN-γ Production in Human CD4+ T Cells. *The Journal of Immunology*, 173(5):3062 LP – 3072, sep 2004. doi: 10.4049/jimmunol.173.5.3062. URL http://www.jimmunol.org/content/173/5/3062.abstract.

[36] Michael L Dustin. The cellular context of T cell signaling. *Immunity*, 30(4):482–492, apr 2009. ISSN 1097-4180. doi: 10.1016/j.immuni.2009.03.010. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2905632/.

[37] Nicholas van Panhuys, Frederick Klauschen, and Ronald N Germain. T-cell-receptor-dependent signal intensity dominantly controls CD4(+) T cell polarization In Vivo. *Immunity*, 41(1):63–74, jul 2014. ISSN 1097-4180. doi: 10.1016/j.immuni.2014.06.003. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4114069/.

[38] N Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979. ISSN 0018-9472 VO - 9. doi: 10.1109/TSMC.1979.4310076.

[39] J K C CHAN, C S NG, and P K HUI. A simple guide to the terminology and application of leucocyte monoclonal antibodies. *Histopathology*, 12(5):461–480, may 1988. ISSN 0309-0167. doi: 10.1111/j.1365-2559.1988.tb01967.x. URL https://doi.org/10.1111/j.1365-2559.1988.tb01967.x.

[40] Zachary C Lipton. The Mythos of Model Interpretability. *Queue*, 16(3):30:31—-30:57, jun 2018. ISSN 1542-7730. doi: 10.1145/3236386.3241340. URL http://doi.acm.org/10.1145/3236386.3241340.

[41] Barret Zoph and Quoc V Le. Neural Architecture Search with Reinforcement Learning. *CoRR*, abs/1611.01578, 2016. URL http://arxiv.org/abs/1611.01578.

[42] Aleksandar Zlateski, Kisuk Lee, and H Sebastian Seung. ZNN - A Fast and Scalable Algorithm for Training 3D Convolutional Networks on Multi-Core and Many-Core Shared Memory Machines. *CoRR*, abs/1510.0, 2015. URL http://arxiv.org/abs/1510.06706.

[43] Kisuk Lee, Aleksandar Zlateski, Ashwin Vishwanathan, and H Sebastian Seung. Recursive Training of 2D-3D Convolutional Networks for Neuronal Boundary Detection. *CoRR*, abs/1508.0, 2015. URL http://arxiv.org/abs/1508.04843.

[44] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic Routing Between Capsules. *CoRR*, abs/1710.09829, 2017. URL http://arxiv.org/abs/1710.09829.

[45] Ross B Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013. URL http://arxiv.org/abs/1311.2524.

[46] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010. URL https://www.cs.toronto.edu/{~}hinton/absps/reluICML.pdf.

[47] MATLAB and Signal Processing Toolbox, 2016.

[48] tifffile.py. URL https://www.lfd.uci.edu/{~}gohlke/code/tifffile.py.html.

[49] csachs tifffile.py. URL https://github.com/csachs/imagej-tiff-meta.

[50] Midway2. URL https://rcc.uchicago.edu/support-and-services/midway2.

[51] Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Devin Matthieu, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Jia Yangqing, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. URL http://tensorflow.org/.

[52] Xavier Glorot, , and Yoshua Bengio B T Proceedings of the Thirteenth International Conference on Artificial Intelligence Statistics. Understanding the difficulty of training deep feedforward neural networks, mar 2010. URL http://proceedings.mlr.press/v9/glorot10a/glorot10a.pdfhttp://proceedings.mlr.press/v9/glorot10a.html.

[53] Stéfan van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in Python. *PeerJ*, 2:e453, 2014. ISSN 2167-8359. doi: 10.7717/peerj.453. URL https://doi.org/10.7717/peerj.453.

[54] Laurent Malherbe, Linda Mark, Nicolas Fazilleau, Louise J McHeyzer-Williams, and Michael G McHeyzer-Williams. Vaccine adjuvants alter TCR-based selection thresholds. *Immunity*, 28(5):698–709, may 2008. ISSN 1097-4180. doi: 10.1016/j.immuni.2008.03.014. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2695494/.

[55] Christina K Baumgartner, Andrea Ferrante, Mika Nagaoka, Jack Gorski, and Laurent P Malherbe. Peptide-MHC class II complex stability governs CD4 T cell clonal selection. *Journal of immunology (Baltimore, Md. : 1950)*, 184(2):573–581, jan 2010. ISSN 1550-6606. doi: 10.4049/jimmunol.0902107. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2975033/.

[56] E. Olson. Particle shape factors and their use in image analysis-part 1: theory. *J. GXP Compl.*, 15:85–90, 2011.

[57] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017. URL http://www.r-project.org/.

[58] A Polliack, N Lampen, B D Clarkson, E de Harven, Z Bentwich, F P Siegal, and H G Kunkel. IDENTIFICATION OF HUMAN B AND T LYMPHOCYTES BY SCANNING ELECTRON MICROSCOPY. *The Journal of Experimental Medicine*, 138(3):607 LP – 624, sep 1973. doi: 10.1084/jem.138.3.607. URL http://jem.rupress.org/content/138/3/607.abstract.

[59] Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross B Girshick. Learning to Segment Every Thing. *CoRR*, abs/1711.10370, 2017. URL http://arxiv.org/abs/1711.10370.

[60] Anthony Chang, Scott G Henderson, Daniel Brandt, Ni Liu, Riteesha Guttikonda, Christine Hsieh, Natasha Kaverina, Tammy O Utset, Shane M Meehan, Richard J Quigg, Eric Meffre, and Marcus R Clark. In situ B cell-mediated immune responses and tubulointerstitial inflammation in human lupus nephritis. *Journal of immunology (Baltimore, Md. : 1950)*, 186(3):1849–1860, feb 2011. ISSN 1550-6606. doi: 10.4049/jimmunol.1001983. URL https://www.ncbi.nlm.nih.gov/pubmed/21187439https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3124090/.

[61] Fiji 3D Viewer. URL https://imagej.net/3D{_}Viewer.

[62] Nancy A Obuchowski. Receiver Operating Characteristic Curves and Their Use in Radiology. *Radiology*, 229(1):3–8, oct 2003. ISSN 0033-8419. doi: 10.1148/radiol.2291010898. URL https://doi.org/10.1148/radiol.2291010898.

[63] F Sallusto and A Lanzavecchia. Efficient presentation of soluble antigen by cultured human dendritic cells is maintained by granulocyte/macrophage colony-stimulating factor plus interleukin 4 and downregulated by tumor necrosis factor alpha. *The Journal of experimental medicine*, 179(4):1109–1118, apr 1994. ISSN 0022-1007. doi: 10.1084/jem.179.4.1109. URL https://www.ncbi.nlm.nih.gov/pubmed/8145033https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2191432/.

[64] Pierre Guermonprez, Jenny Valladeau, Laurence Zitvogel, Clotilde Théry, and Sebastian Amigorena. Antigen Presentation and T Cell Stimulation by Dendritic Cells. *Annual Review of Immunology*, 20(1):621–667, apr 2002. ISSN 0732-0582. doi: 10.1146/annurev.immunol.20.100301.064828. URL https://doi.org/10.1146/annurev.immunol.20.100301.064828.

[65] Melissa Swiecki and Marco Colonna. The multifaceted biology of plasmacytoid dendritic cells. *Nature reviews. Immunology*, 15(8):471–485, aug 2015. ISSN 1474-1741. doi: 10.1038/nri3865. URL https://www.ncbi.nlm.nih.gov/pubmed/26160613https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4808588/.

[66] José A Villadangos and Louise Young. Antigen-Presentation Properties of Plasmacytoid Dendritic Cells. *Immunity*, 29(3):352–361, sep 2008. ISSN 1074-7613. doi: 10.1016/j.immuni.2008.09.002. URL https://doi.org/10.1016/j.immuni.2008.09.002.

[67] Alexandra-Chloé Villani, Rahul Satija, Gary Reynolds, Siranush Sarkizova, Karthik Shekhar, James Fletcher, Morgane Griesbeck, Andrew Butler, Shiwei Zheng, Suzan Lazo, Laura Jardine, David Dixon, Emily Stephenson, Emil Nilsson, Ida Grundberg, David McDonald, Andrew Filby, Weibo Li, Philip L De Jager, Orit Rozenblatt-Rosen, Andrew A Lane, Muzlifah Haniffa, Aviv Regev, and Nir Hacohen. Single-cell RNA-seq reveals new types of human blood dendritic cells, monocytes, and progenitors. *Science (New York, N.Y.)*, 356(6335):eaah4573, apr 2017. ISSN 1095-9203. doi: 10.1126/science.aah4573. URL https://www.ncbi.nlm.nih.gov/pubmed/28428369https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5775029/.

[68] Jérôme Delon, Kozo Kaibuchi, and Ronald N Germain. Exclusion of CD43 from the Immunological Synapse Is Mediated by Phosphorylation-Regulated Relocation of the Cytoskeletal Adaptor Moesin. *Immunity*, 15(5):691–701, nov 2001. ISSN 1074-7613. doi: 10.1016/S1074-7613(01)00231-X. URL https://doi.org/10.1016/S1074-7613(01)00231-X.

[69] Eric J Allenspach, Patrick Cullinan, Jiankun Tong, Qizhi Tang, Amanda G Tesciuba, Judy L Cannon, Stephenie M Takahashi, Renell Morgan, Janis K Burkhardt, and Anne I Sperling. ERM-Dependent Movement of CD43 Defines a Novel Protein Complex Distal to the Immunological Synapse. *Immunity*, 15(5):739–750, nov 2001.

ISSN 1074-7613. doi: 10.1016/S1074-7613(01)00224-2. URL https://doi.org/10.1016/S1074-7613(01)00224-2.

[70] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B Girshick. Mask R-CNN. *CoRR*, abs/1703.0, 2017. URL http://arxiv.org/abs/1703.06870.

[71] Shaoqing Ren, Kaiming He, Ross B Girshick, and Jian Sun. Faster {R-CNN:} Towards Real-Time Object Detection with Region Proposal Networks. *CoRR*, abs/1506.01497, 2015. URL http://arxiv.org/abs/1506.01497.

[72] Tensorpack, . URL https://github.com/tensorpack/tensorpack.

[73] Tensorpack: Mask R-CNN, . URL https://github.com/tensorpack/tensorpack/tree/master/examples/FasterRCNN.

[74] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

[75] Tsung-Yi Lin, Piotr Dollár, Ross B Girshick, Kaiming He, Bharath Hariharan, and Serge J Belongie. Feature Pyramid Networks for Object Detection. *CoRR*, abs/1612.03144, 2016. URL http://arxiv.org/abs/1612.03144.

[76] COCO Dataset. URL http://cocodataset.org.

[77] Horovod. URL https://github.com/horovod/horovod.

[78] L. Hutton. Using statistics to assess the performance of neural network classifiers. *Johns Hopkins APL Tech. Dig.*, 13:291–299, 1992.

[79] Muhammad A. Razi and Athappilly Kuriakose. comparative predictive analysis of neural networks (NNs), nonlinear regression and classification and regression tree (CART) models. *Expert Syst. Appl*, 29:65–74, 2005.

[80] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep Clustering for Unsupervised Learning of Visual Features. *CoRR*, abs/1807.05520, 2018. URL http://arxiv.org/abs/1807.05520.

Appendices

APPENDIX A

SUPPLEMENTARY FIGURES

This appendix contains primary supplemental figures mentioned in the main body of the text.

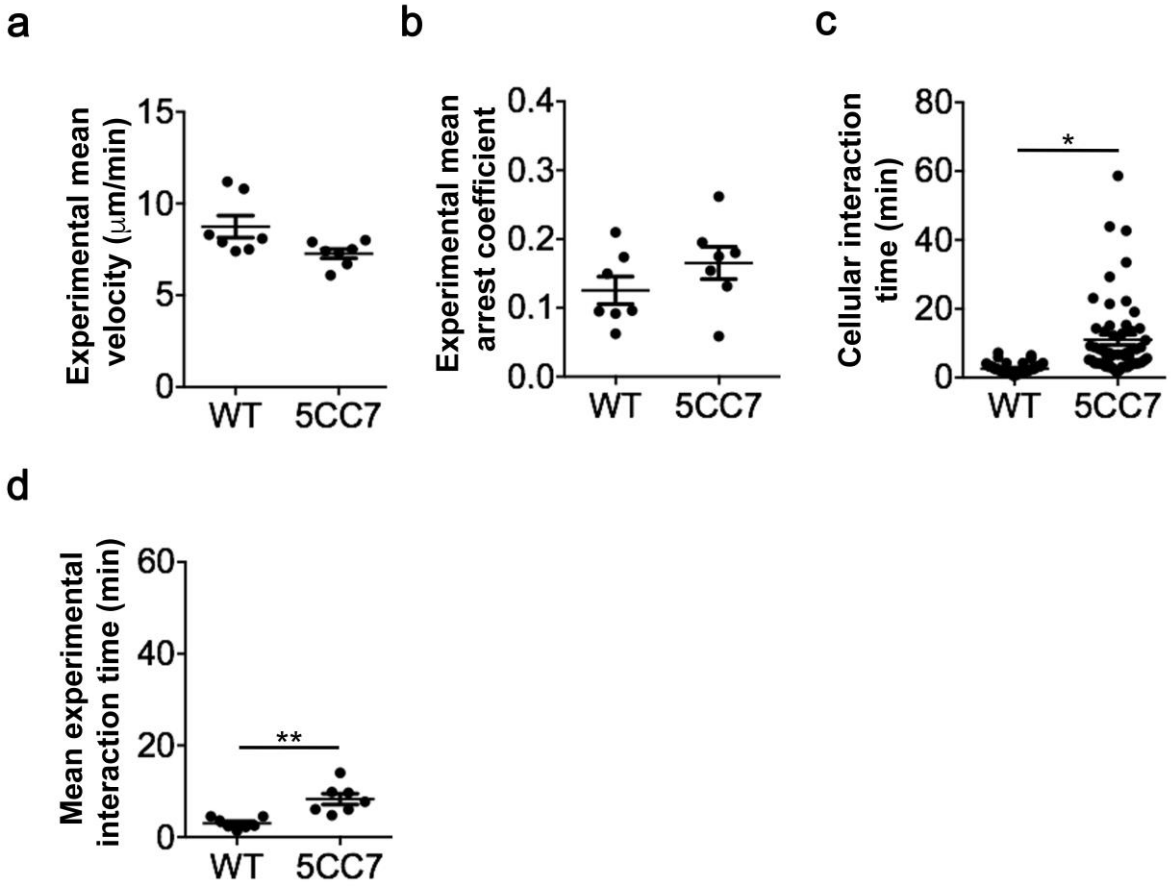

Figure A.1: **TPEM measurements of interactions of antigen-specific and wild-type CD4$^+$ T cells with DCs under low antigen conditions (**0.01 µM **PCC)**. Indicated T cells (WT or 5CC7) and antigen-pulsed DCs were transferred into B10.A2 CD45.2$^-$ mice and, after 12 h, popliteal lymph nodes were imaged by TPEM, as in Figure 4.3. a-d, TPEM parameters plotted as mean per mouse ($n = 7$): mean velocity (**a**), and mean arrest coefficient (**b**). The cellular interaction time for all experiments is shown in (**c**), and per mouse in (**d**). $^*p < 0.05$, $^{**}p < 0.005$, 2-sided Mann-Whitney U test. All center values denote the mean and error bars denote standard error of the mean. $n = 2$ independent experiments for all panels. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

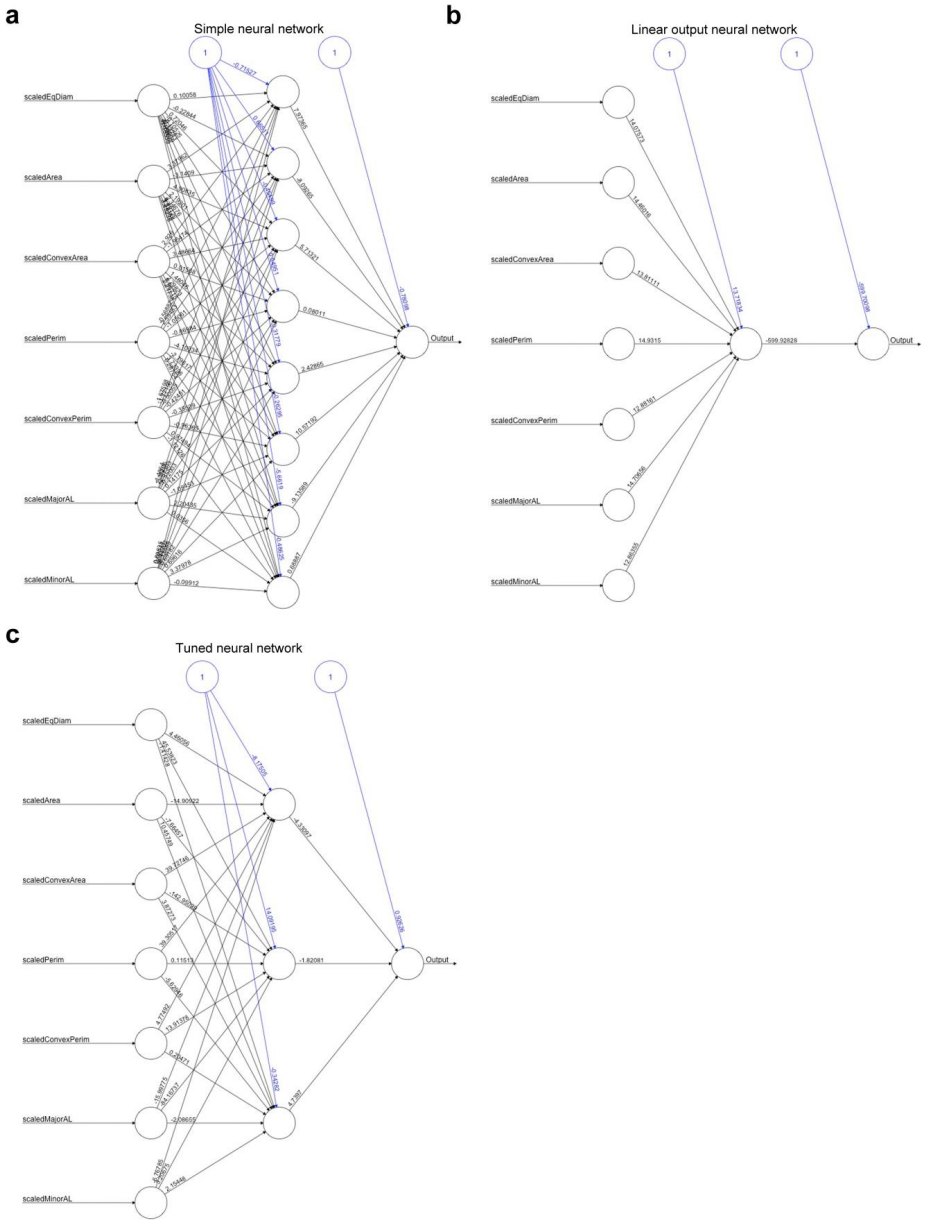

Figure A.2: **TNN Used for Analysis:** Representative plots of the simple (**a**), linear output (**b**), and final tuned (**c**) neural networks with input nodes, hidden layer(s), weights, and output as indicated. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

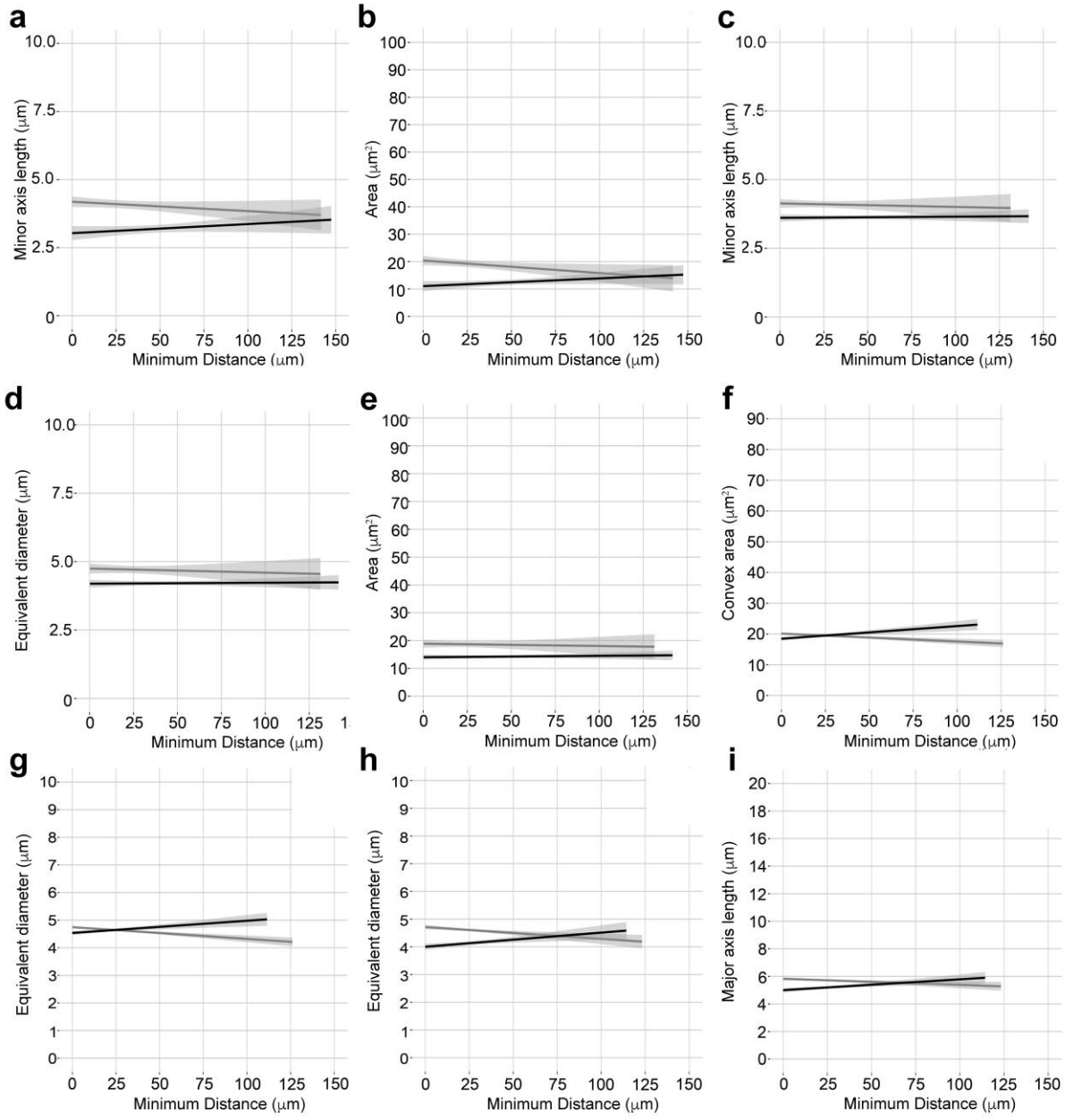

Figure A.3: **Object feature scatter plot trendlines:** Trendlines for all manuscript scatter plots are included for data reference and ease of visualization. Shape parameters as indicated for (**a**-**b**) mouse cell tracker, (**c**-**e**) mouse nuclei, (**f**-**g**) human plasmacytoid dendritic cells, and (**h**-**I**) human monocytic dendritic cells. Grey lines denote 5CC7 (**a**-**e**) or $CD3^{+}CD4^{+}$ (**f**-**i**) T cells; black lines denote WT (**a**-**e**) or $CD3^{+}CD4^{-}$ (**f**-**i**) T cells. Shaded regions denote 95 %. CI. $n = 2$ independent experiments for all panels. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

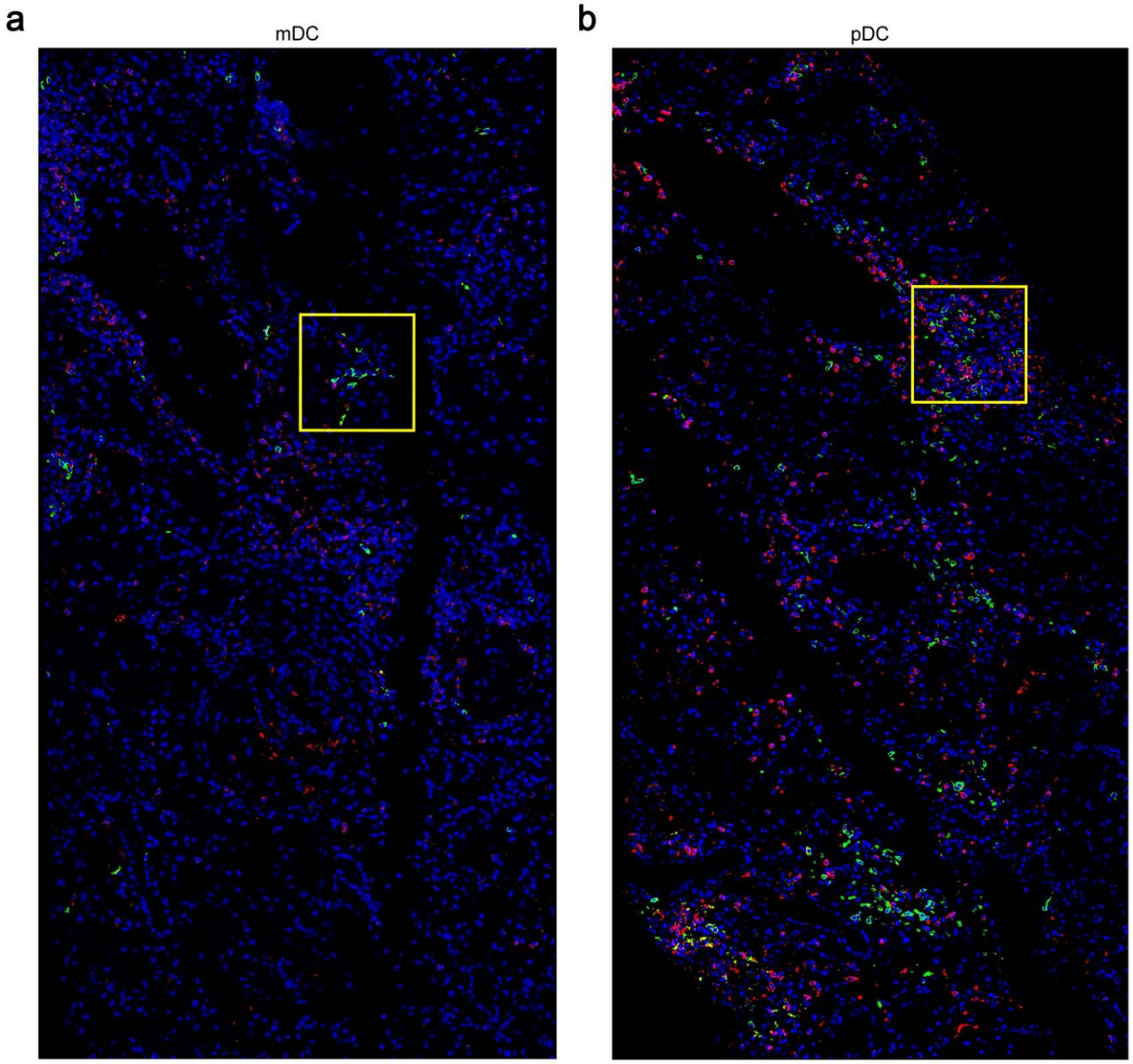

Figure A.4: **Tiled image of whole biopsy section:** Representative tiled images of mDCs and pDC (green) as indicated with $CD4^{+}$T cells (red) in lupus TII. Nuclei are blue. HPFs corresponding to Figure 5e indicated by yellow boxes. $n = 4$ (**a**) and 6 (**b**) independent tiling experiments. (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

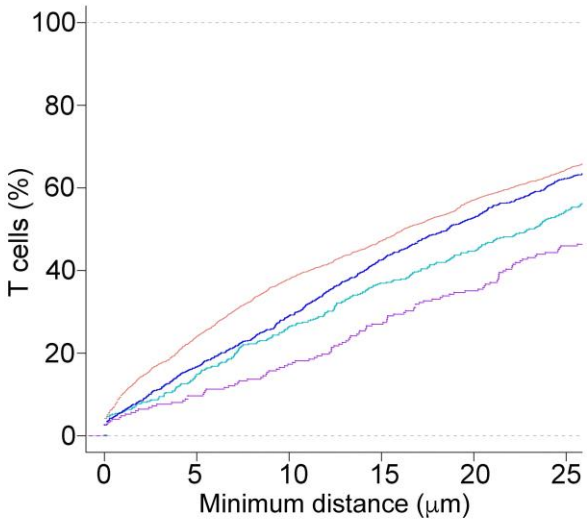

Figure A.5: **Distribution of T cells and DCs in lupus nephritis:** Percentage of T cells versus minimum distance in µm between $CD3^{+}CD4^{+}$ T cells and pDCs (salmon), $CD3^{+}CD4^{-}$ T cells and pDCs (teal), $CD3^{+}CD4^{+}$ T cells and mDCs (blue), and $CD3^{+}CD4^{-}$ T cells and mDCs (purple). (from Liarski and Sibley et al., 2019) (figure prepared by Vladimir Liarski)

APPENDIX B

SUPPLEMENTARY TABLES

This appendix contains primary supplemental tables mentioned in the main body of the text.

Supplemental Table 1

	TPEM		CDM			Independent validation		
Number of lymph nodes	4		4			3		
Number of ROIs			295			233		
Cell type	WT	5CC7	WT	5CC7	DC	WT	5CC7	DC
Cell velocity	99	98						
Arrest coefficient	405	410						
Interactions	111	173						
Manual training (42 ROIs)			268	334	508			
Automatic segmentation			660	659	1794	637	1222	2554
Analyzed dataset			928	993	2302	637	1222	2554

Table B.1: Experiment dataset table for murine data (from Liarski and Sibley et al., 2019) (table prepared by Vladimir Liarski)

a

	Logistic regression	Random forest	SVM	Tuned neural network
AUC	0.84	NA	0.75	0.82
AUC 95% CI	0.63 – 0.93	NA	NA	0.77 – 0.91
Accuracy	0.76	NA	0.77	0.80
Classification error	0.22	0.10 for spilt 1, 0.49 for split 2	0.20	0.18

b

	Simple NN	Linear output NN	Tuned NN
AUC	0.61	NA	0.82
AUC 95% CI	0.37 – 0.81	NA	0.77 – 0.91
Accuracy	0.62	0.65	0.79
Steps	351	1764	11860
Cross entropy error	19.74	16.17	2.31

n as described in Supplemental Table 1.

Table B.2: Performance for different classification algorithms used (from Liarski and Sibley et al., 2019) (table prepared by Vladimir Liarski)

	pDC	mDC
Manual training set		
Biopsy number	7	2
ROI number	172	71
$CD3^+CD4^+$ cell number	2035	473
$CD3^+CD4^-$ cell number	1119	249
Dendritic cell number	1176	256
Full data set		
Biopsy number	22	22
0: no TI	2	2
1: ≤25% TI involvement	6	5
2: ≤50% TI involvement	6	7
3 >50% TI involvement	8	8
Mean TI score (± standard deviation)	1.91 ± 1.02	1.95 ± 1.00
ROI number	364	323
$CD3^+CD4^+$ cell number	12599	4912
$CD3^+CD4^-$ cell number	4056	2347
Dendritic cell number	4121	2518
Mean number of $CD3^+CD4^+$/DC	3.05	1.95
Mean number of $CD3^+CD4^-$/DC	0.98	0.93

Table B.3: Human dataset statistics (from Liarski and Sibley et al., 2019) (table prepared by Vladimir Liarski)

		pDC	mDC
All distances	**TI score 0-2**		
	Dendritic cell number	550	936
	$CD3^+CD4^+$ cell number	2332	1108
	$CD3^+CD4^-$ cell number	495	302
	Mean number of $CD3^+CD4^+$/DC	4.24	1.71
	Mean number of $CD3^+CD4^-$/DC	0.90	0.58
	TI score 3		
	Dendritic cell number	3571	1582
	$CD3^+CD4^+$ cell number	10267	3309
	$CD3^+CD4^-$ cell number	3561	2799
	Mean number of $CD3^+CD4^+$/DC	2.88	2.09
	Mean number of $CD3^+CD4^-$/DC	1.00	1.14
≤ 12.5 ɷm	**TI score 0-2**		
	Dendritic cell number	45	46
	$CD3^+CD4^+$ cell number	953 (40%)	574 (51%)
	$CD3^+CD4^-$ cell number	187 (37%)	189 (63%)
	Mean number of $CD3^+CD4^+$/DC	21.17	12.47
	Mean number of $CD3^+CD4^-$/DC	4.15	4.10
	TI score 3		
	Dendritic cell number	752	142
	$CD3^+CD4^+$ cell number	4577 (44%)	1136 (34%)
	$CD3^+CD4^-$ cell number	1431 (40%)	499 (18%)
	Mean number of $CD3^+CD4^+$/DC	6.08	8.00
	Mean number of $CD3^+CD4^-$/DC	1.90	3.51
≤ 25 ɷm	**TI score 0-2**		
	Dendritic cell number	148	96
	$CD3^+CD4^+$ cell number	1470 (63%)	724 (65%)
	$CD3^+CD4^-$ cell number	284 (57%)	123 (41%)
	Mean number of $CD3^+CD4^+$/DC	9.93	7.54
	Mean number of $CD3^+CD4^-$/DC	1.91	1.28
	TI score 3		
	Dendritic cell number	2656	985
	$CD3^+CD4^+$ cell number	7059 (68%)	2083 (62%)
	$CD3^+CD4^-$ cell number	2072 (58%)	1134 (41%)
	Mean number of $CD3^+CD4^+$/DC	2.65	2.11
	Mean number of $CD3^+CD4^-$/DC	0.78	1.15

Table B.4: Further human dataset statistics (from Liarski and Sibley et al., 2019) (table prepared by Vladimir Liarski)

APPENDIX C

MOUSE NUCLEAR SEGMENTATION

This section presents 7 examples of manual segmentation of all nuclei in a DAPI nuclear images as discussed in Chapter 4.

All images and label ROIs are embedded in document with DEFLATE png compression at full resolution. Images were transformed from native bit depth of 12 by rescaling the image using the standard score (z-score). Z-score images were transformed to an 8-bit range by trucating the distribution of z-scores at 3 standard deviations from the mean and rescaling to the range [0,255]. Bit depth is 8 in embedded images.

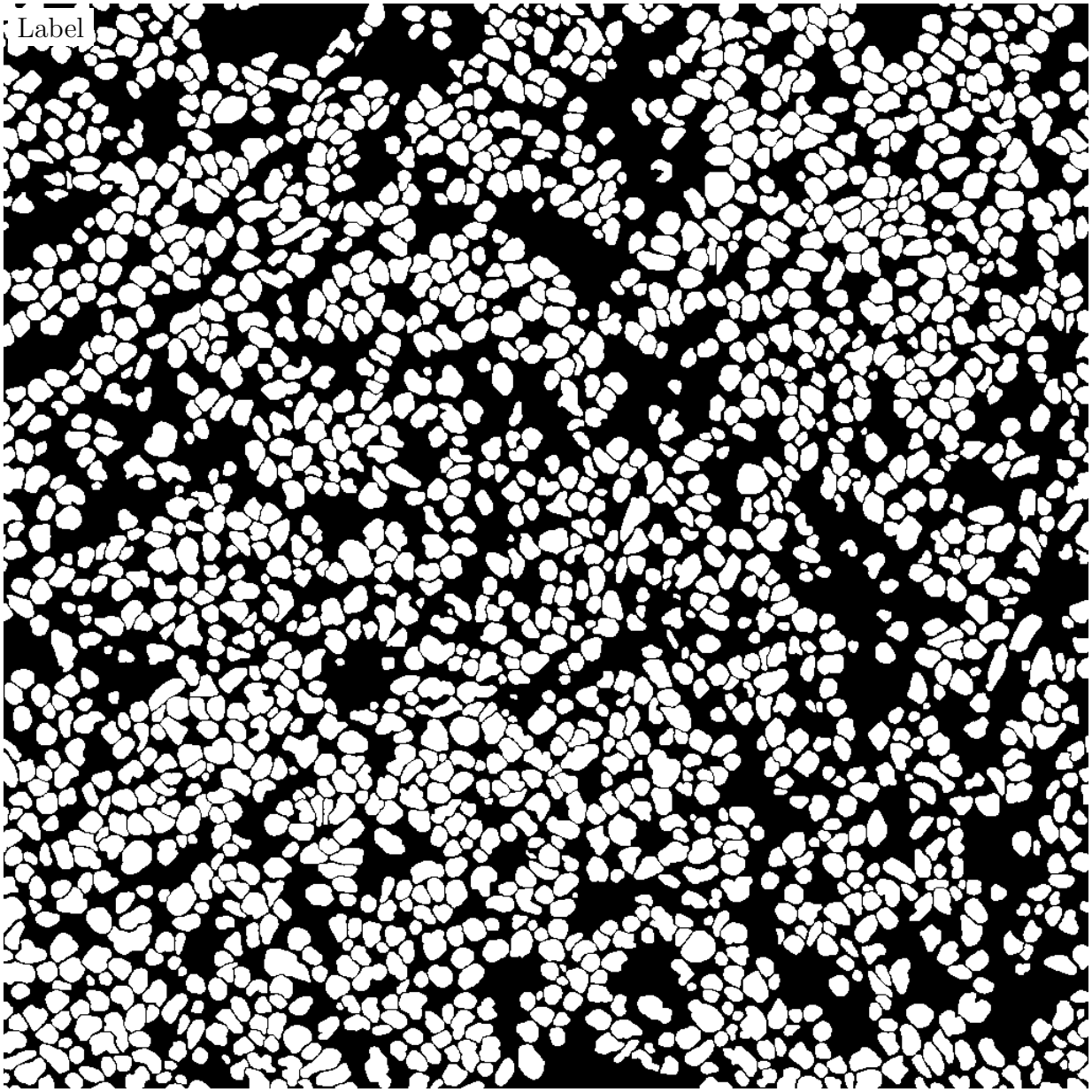

Figure C.1: **Label for C.2**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.23\,\mu m \times 0.23\,\mu m$.

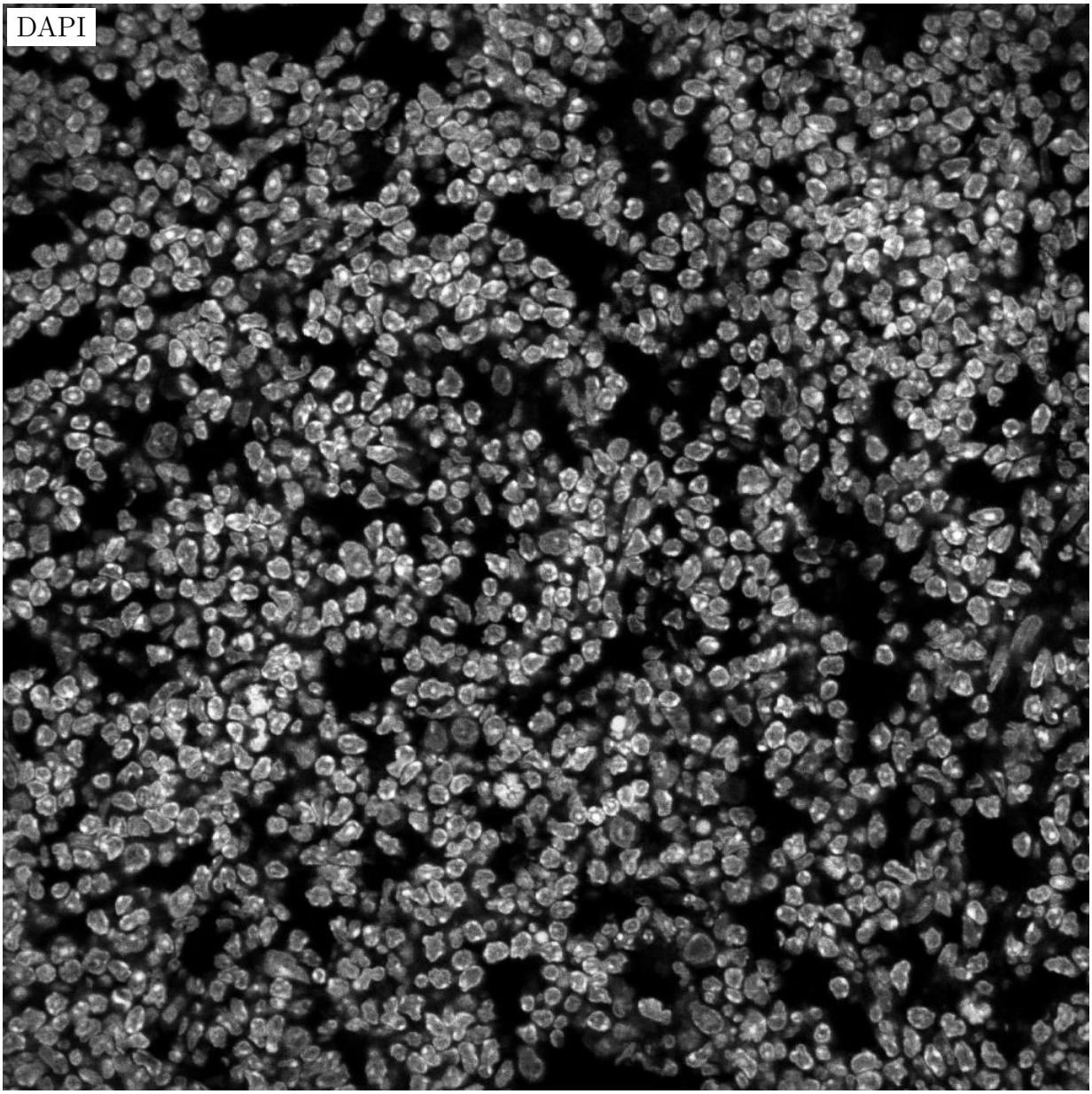

Figure C.2: **DAPI for C.1**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.23\,\mu m \times 0.23\,\mu m$.

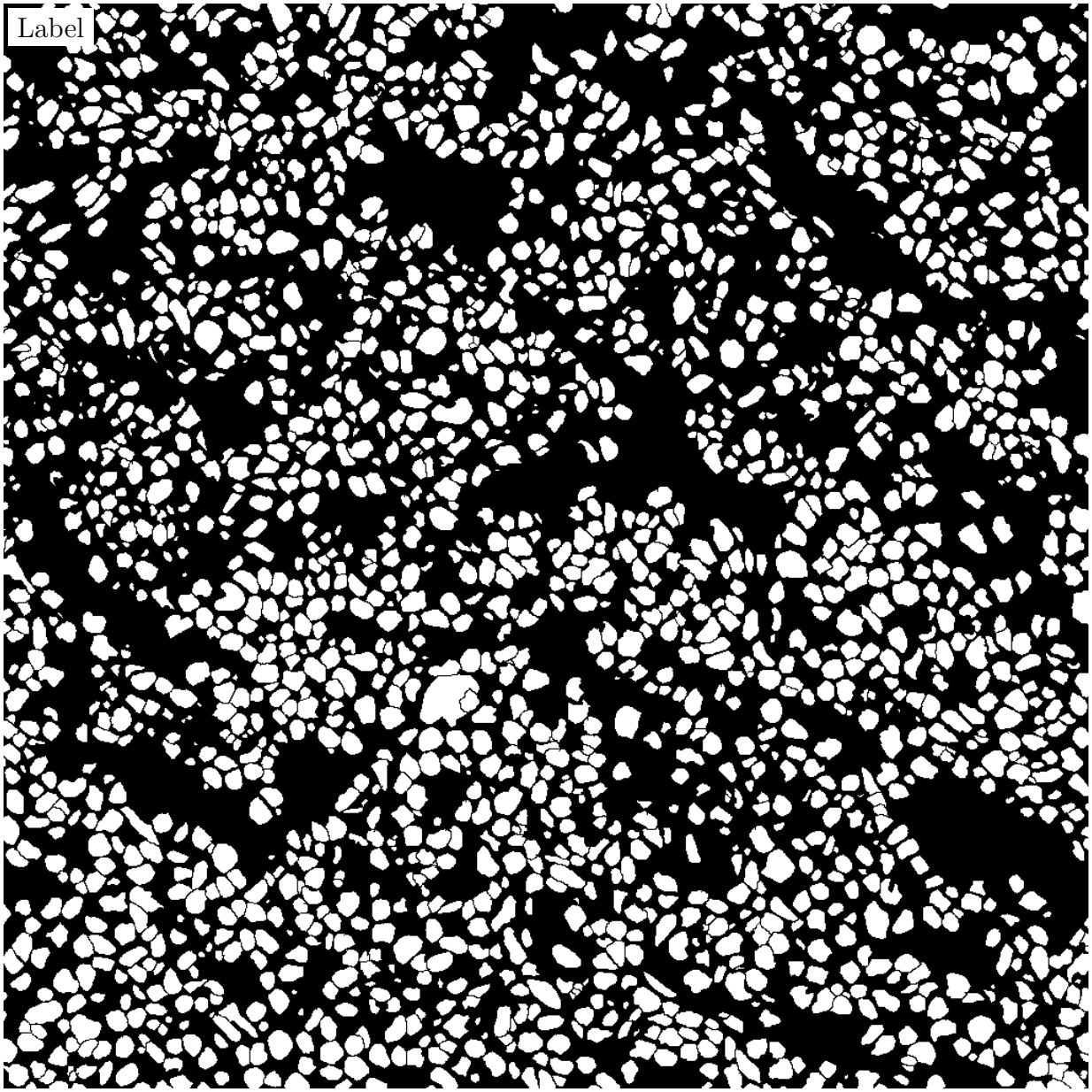

Figure C.3: **Label for C.4**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.23\,\mu m \times 0.23\,\mu m$.

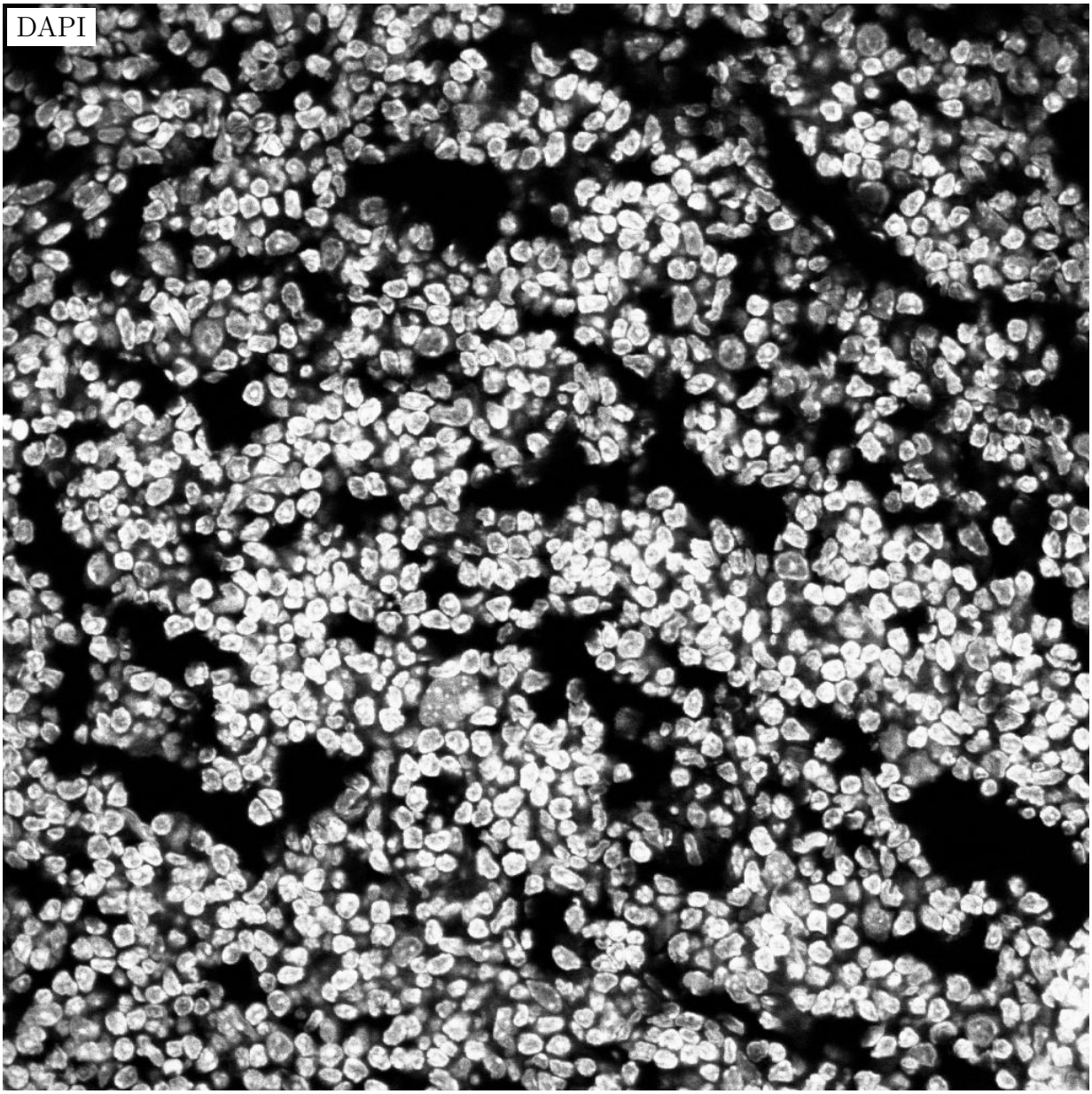

Figure C.4: **DAPI for C.3**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.23\,\mu m \times 0.23\,\mu m$.

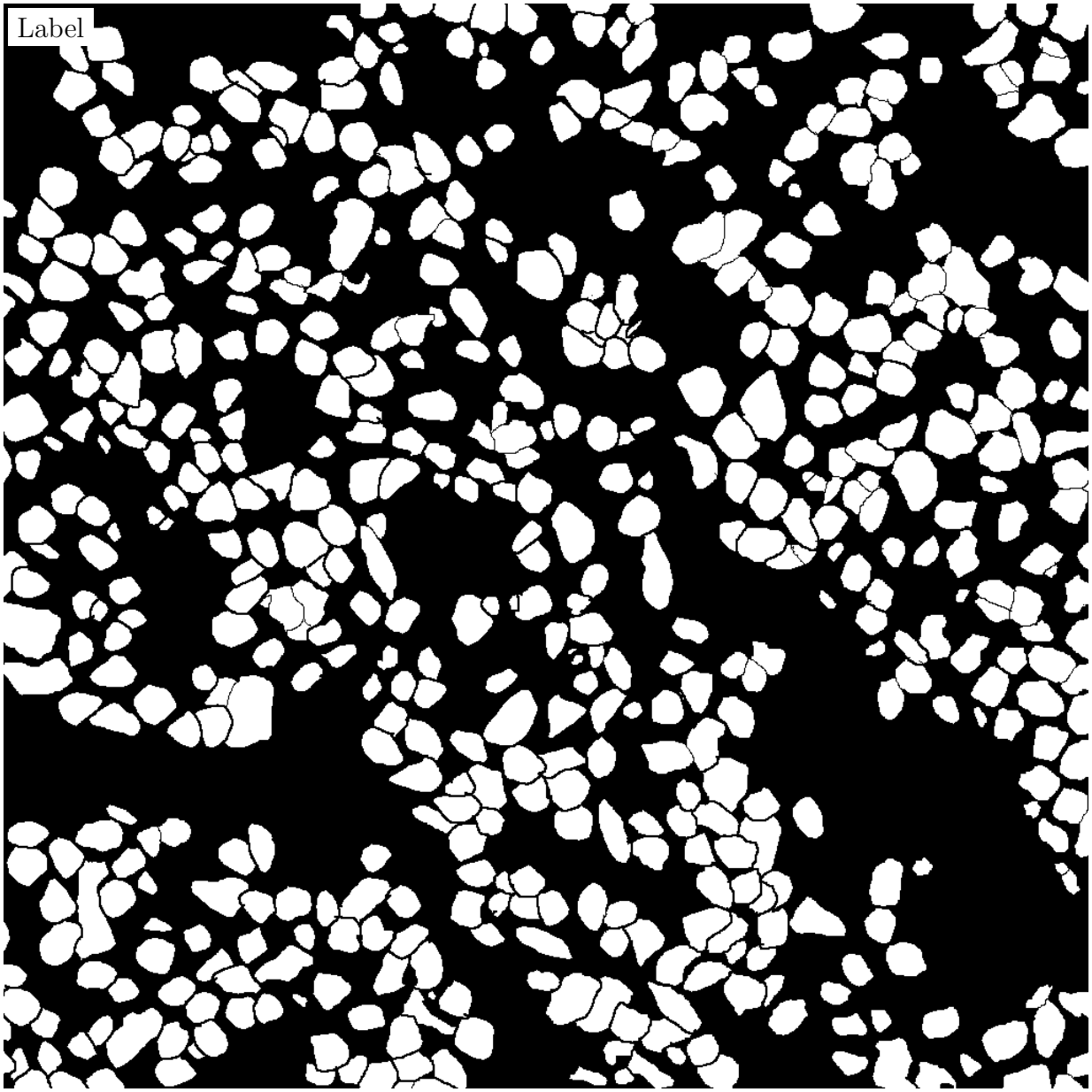

Figure C.5: **Label for C.6**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$.

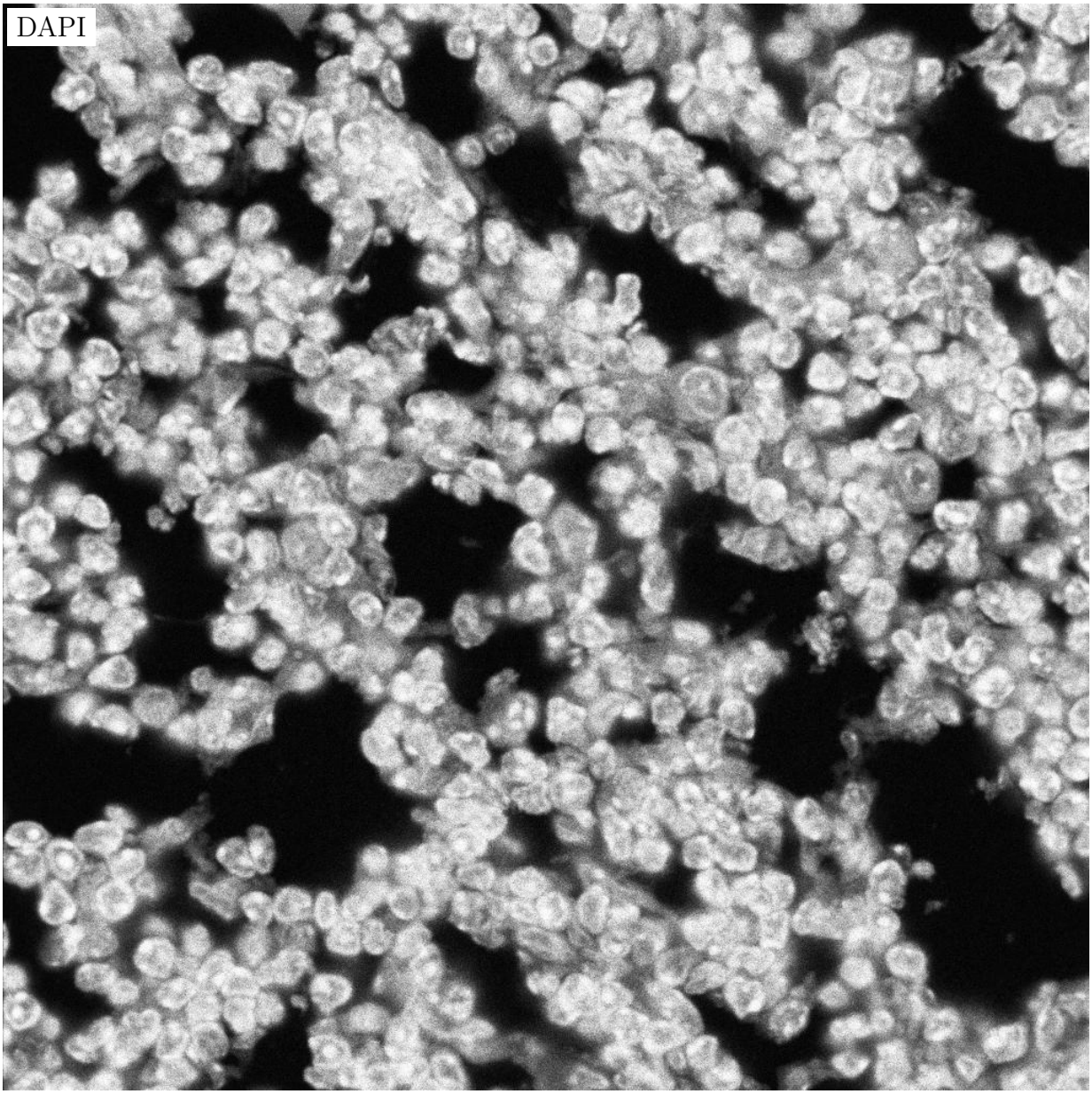

Figure C.6: **DAPI for C.5**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$.

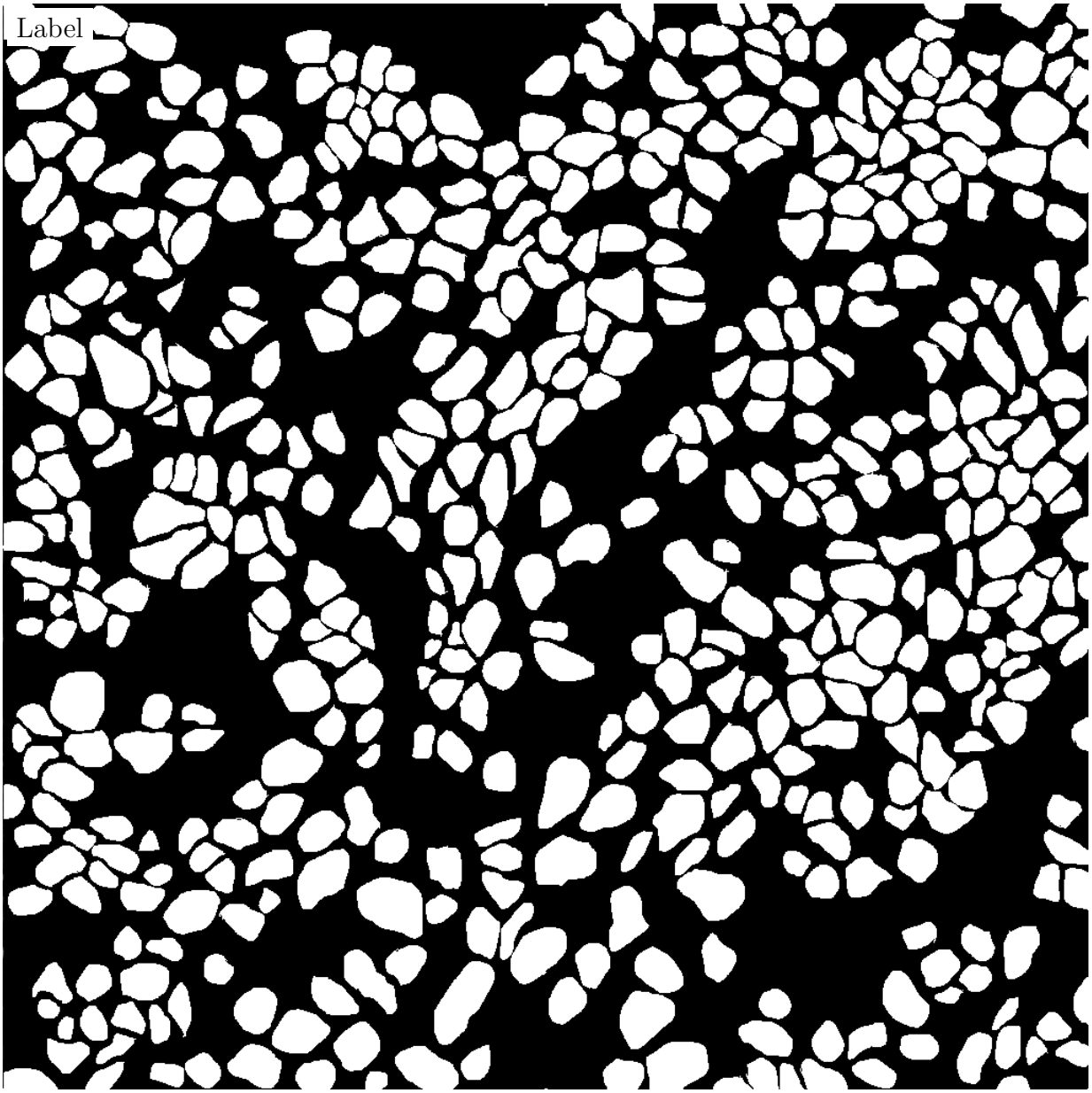

Figure C.7: **Label for C.8**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$.

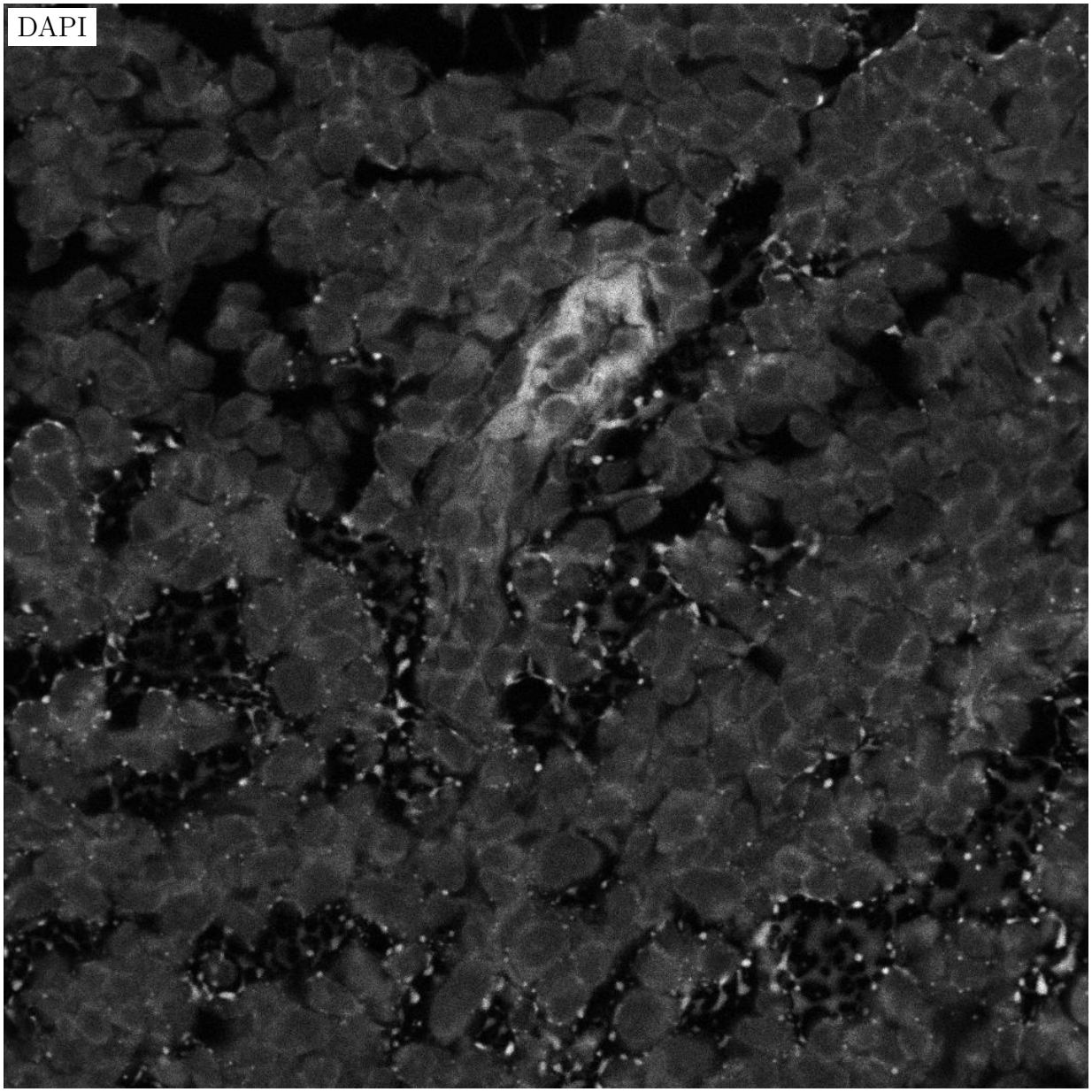

Figure C.8: **DAPI for C.7**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$.

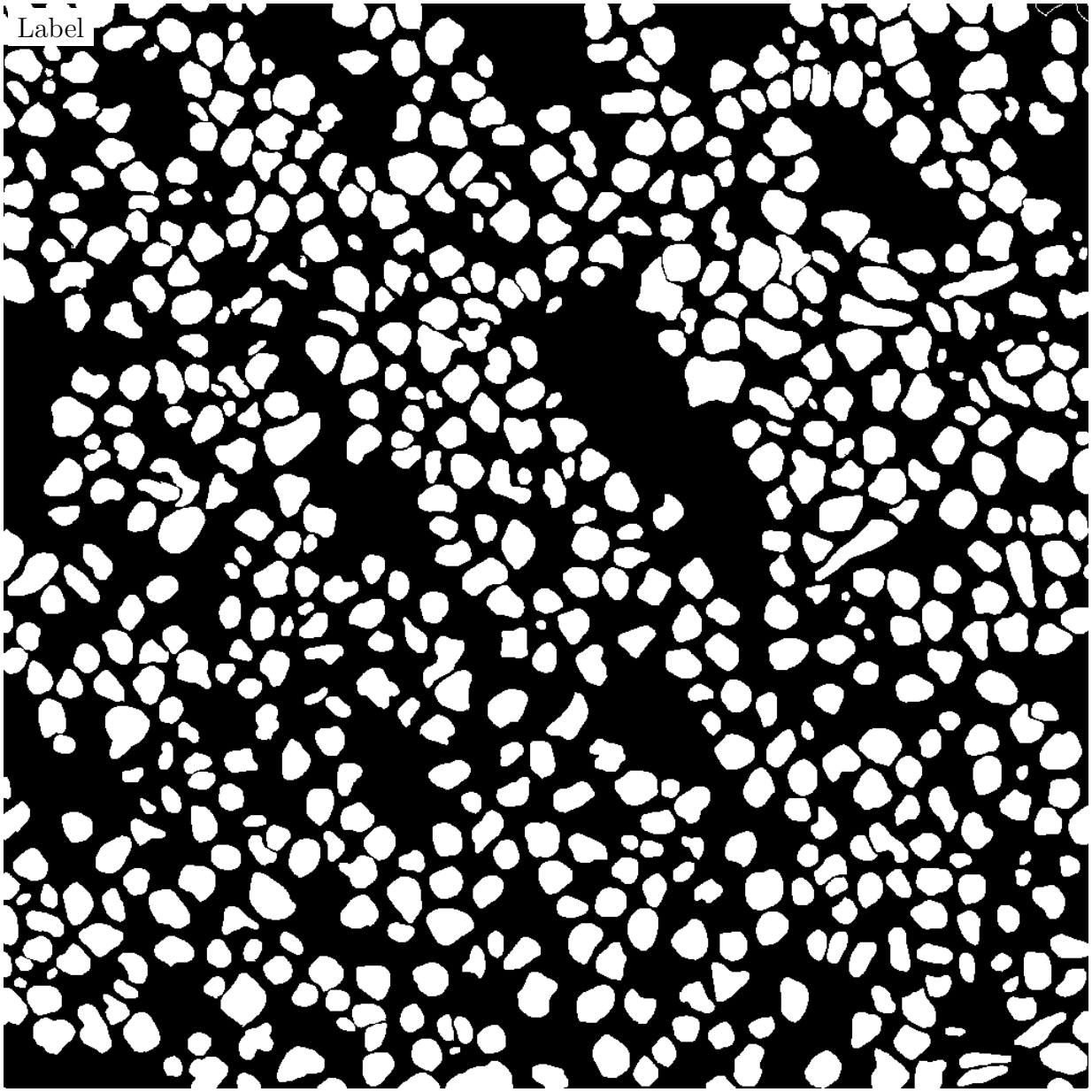

Figure C.9: **Label for C.10**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$.

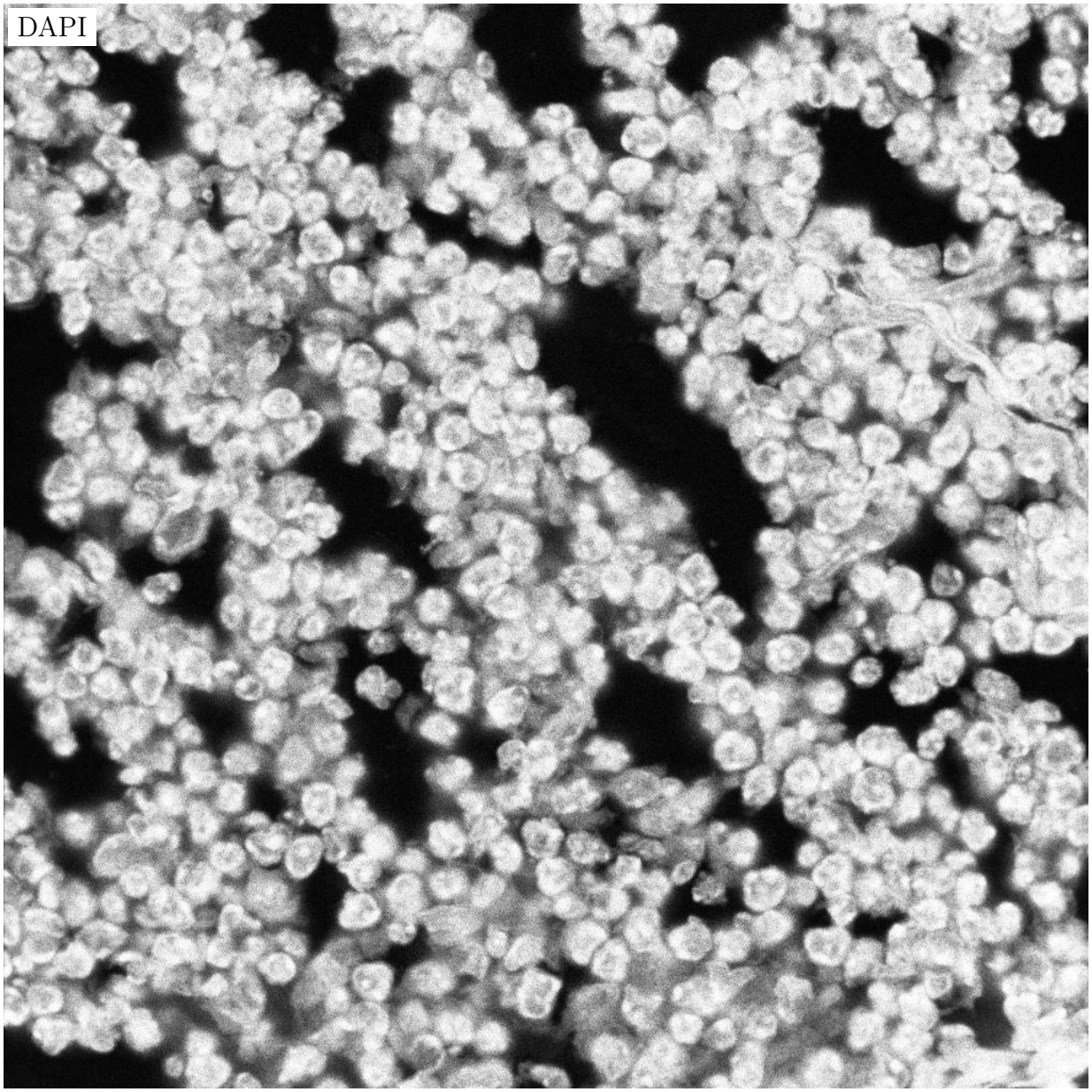

Figure C.10: **DAPI for C.9**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$.

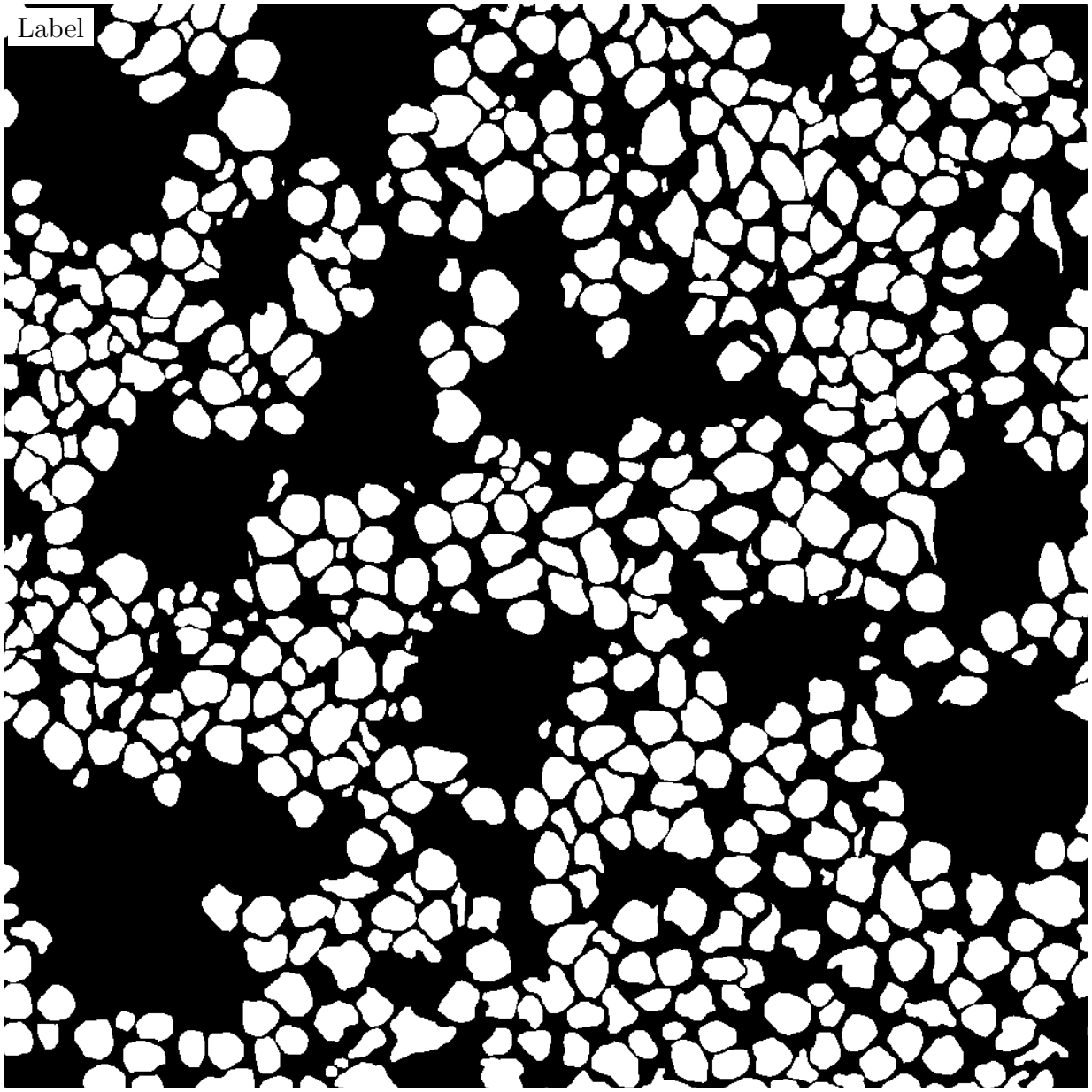

Figure C.11: **Label for C.12**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$.

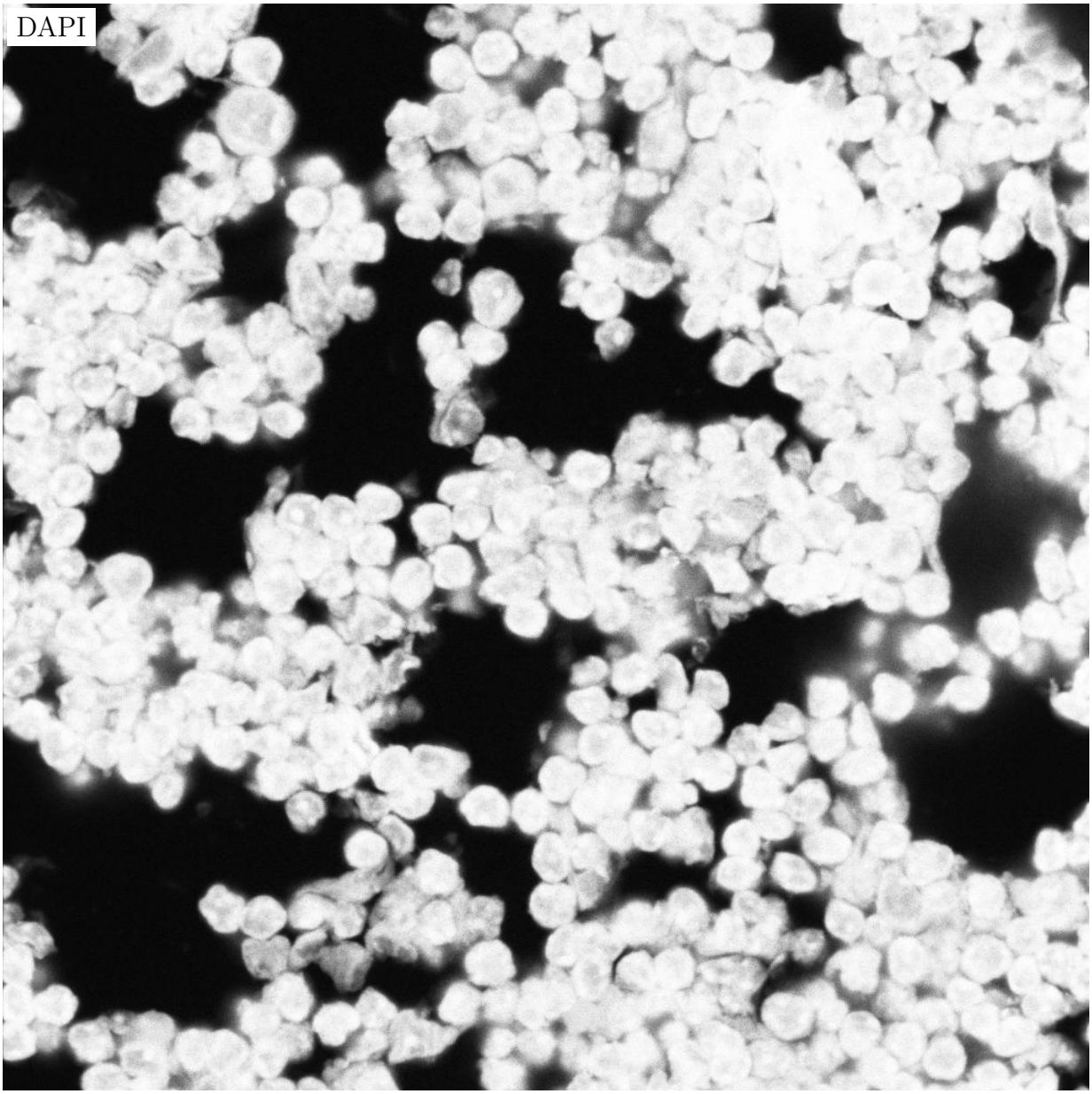

Figure C.12: **DAPI for C.11**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$.

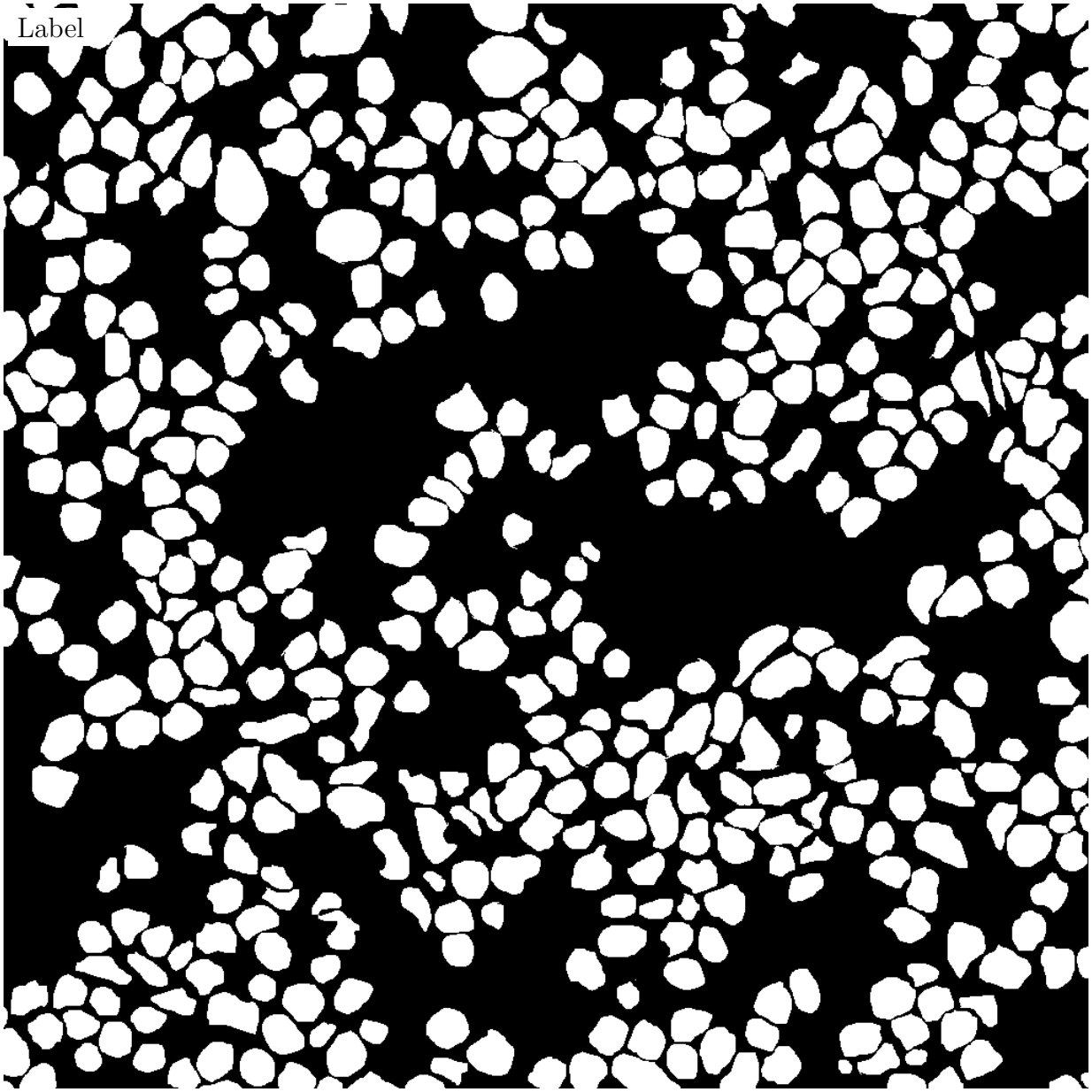

Figure C.13: **Label for C.14**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$.

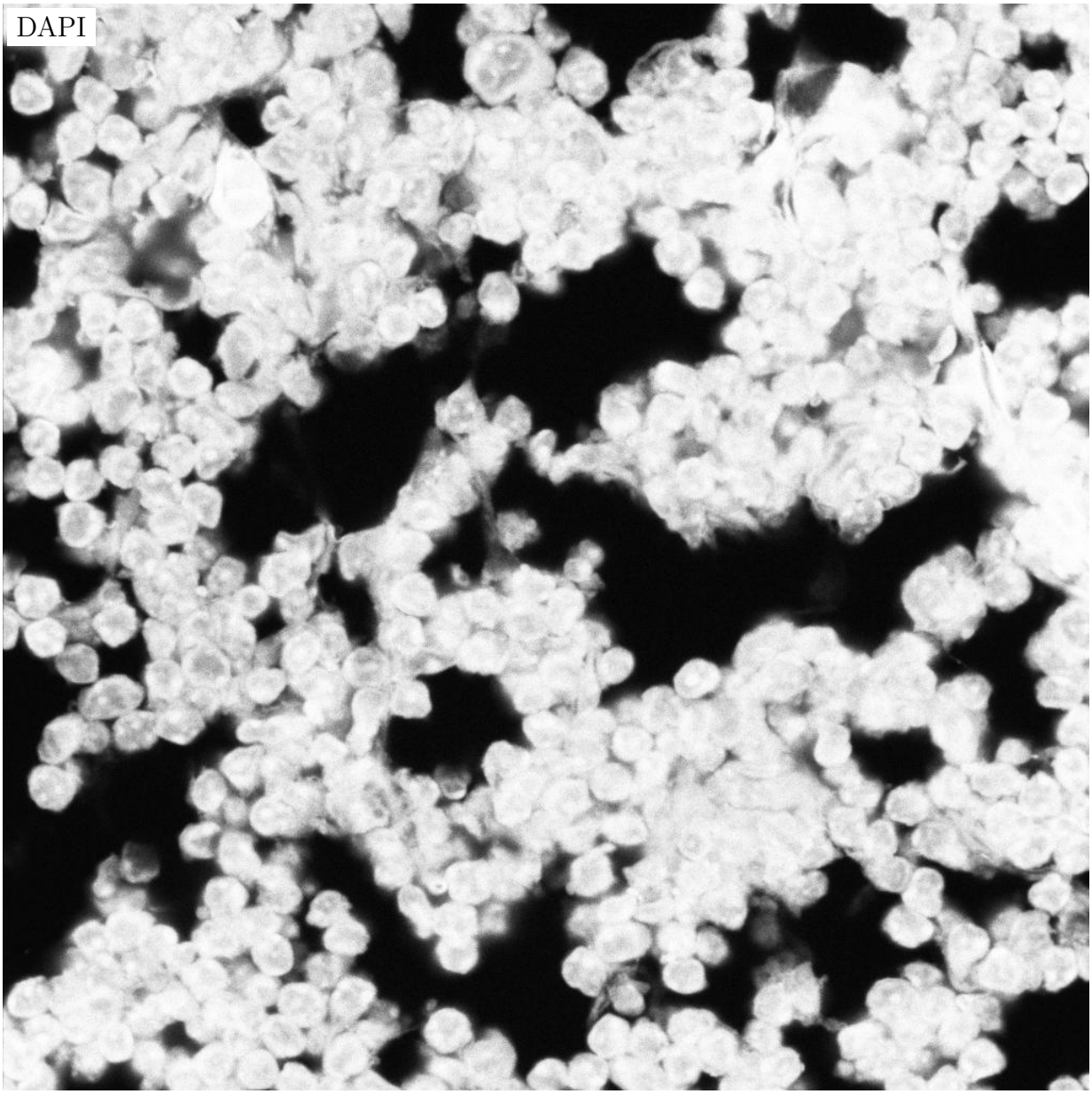

Figure C.14: **DAPI for C.13**. True label matrix size is 1024×1024. Depicted label matrix size is 1024×1024. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$.

APPENDIX D

SELECTED MOUSE PERFORMANCE EXAMPLES

This section presents 3 hand-selected examples from the cross validation run of the murine fresh frozen dataset as discussed in Chapter 4. Several of these examples are zoomed to show specific performance characteristics.

All images and label ROIs are embedded in document with DEFLATE png compression at full resolution. Images were transformed from native bit depth of 12 by rescaling the image using the standard score (z-score). Z-score images were transformed to an 8-bit range by trucating the distribution of z-scores at 3 standard deviations from the mean and rescaling to the range [0,255]. Pixel size is $0.23\,\mu m \times 0.23\,\mu m$. Bit depth is 8 in embedded images.

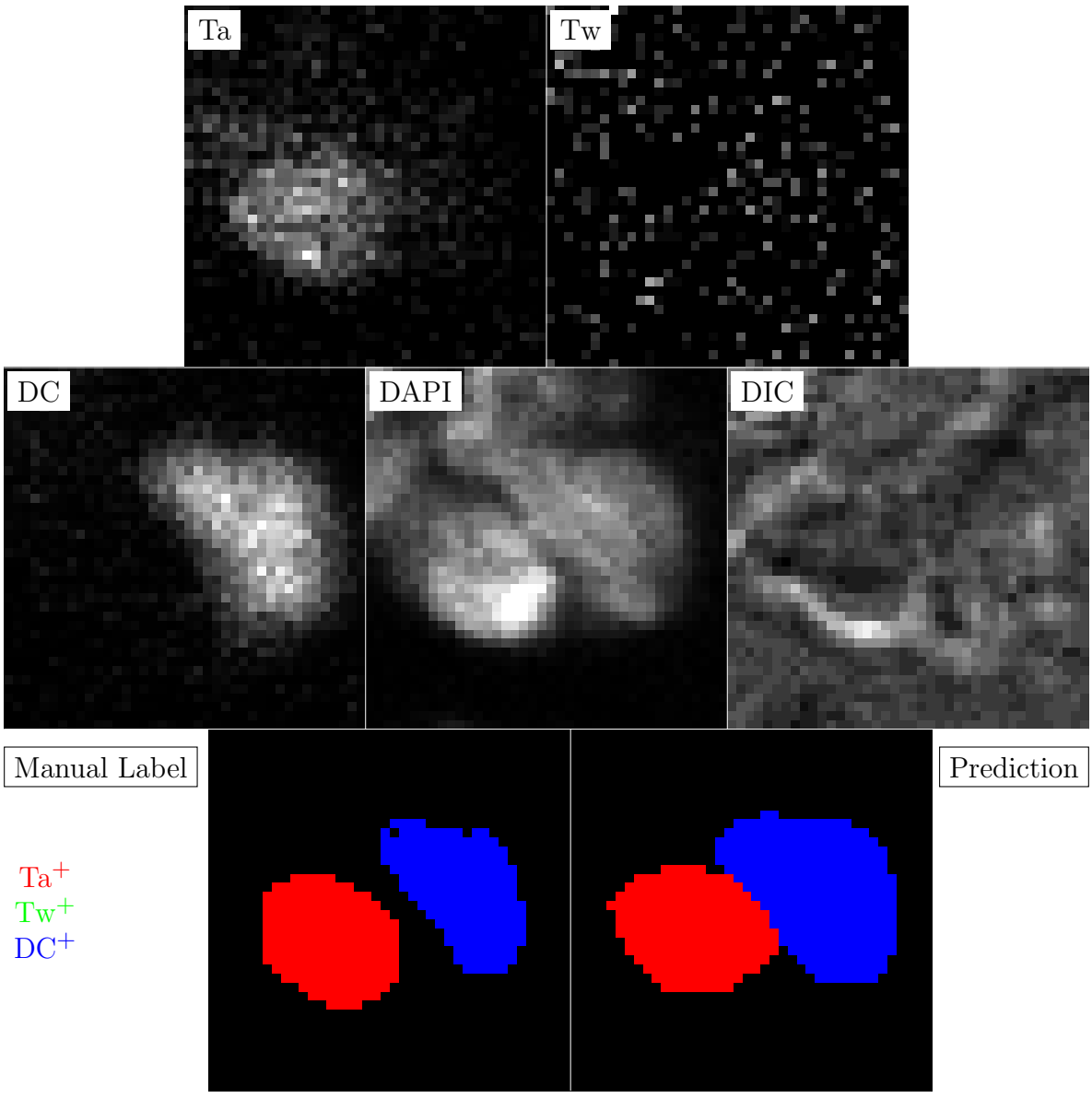

Figure D.1: **Murine FF**. True image channel matrix size is 40×40. Depicted image channel matrix size is 40×40.

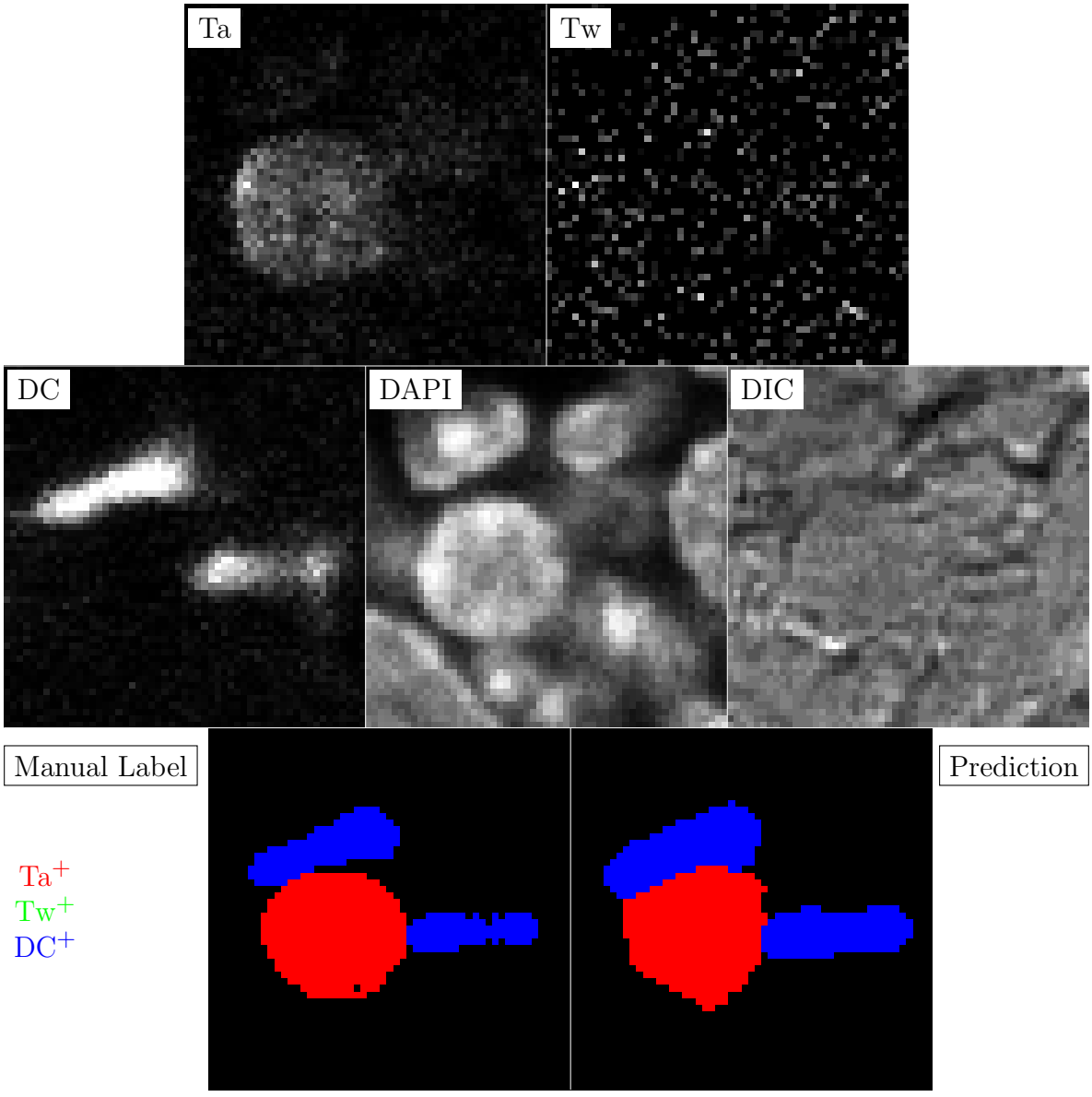

Figure D.2: **Murine FF**. True image channel matrix size is 55×55. Depicted image channel matrix size is 55×55.

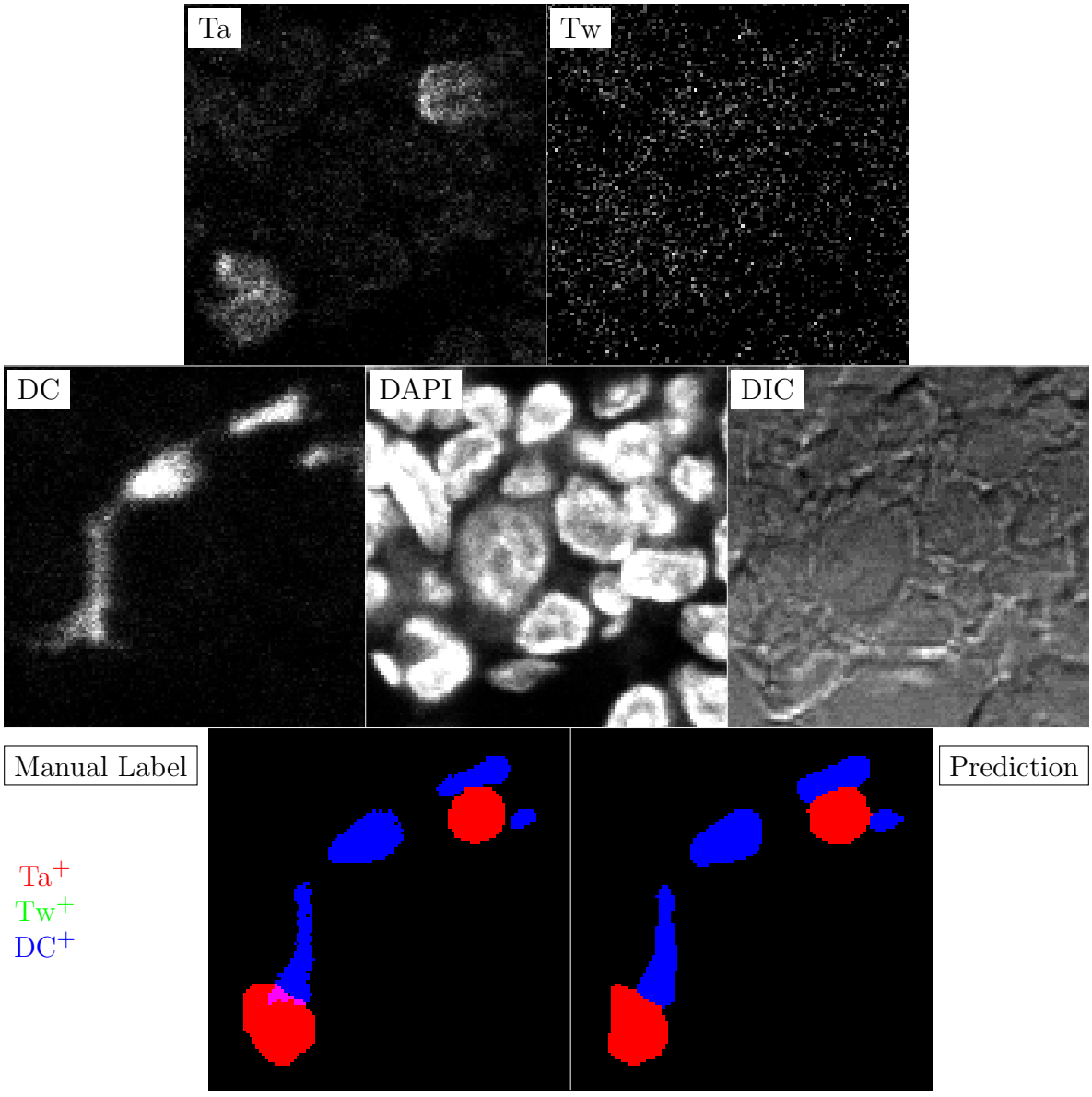

Figure D.3: **Murine FF**. True image channel matrix size is 115×115. Depicted image channel matrix size is 115×115.

APPENDIX E

SELECTED HUMAN PERFORMANCE EXAMPLES

This section presents 3 hand-selected examples from the cross validation run of the human fresh frozen dataset as discussed in Chapter 5. Several of these examples are zoomed to show specific performance characteristics.

All images and label ROIs are embedded in document with DEFLATE png compression at full resolution. Images were transformed from native bit depth of 12 by rescaling the image using the standard score (z-score). Z-score images were transformed to an 8-bit range by trucating the distribution of z-scores at 3 standard deviations from the mean and rescaling to the range [0,255]. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$. Bit depth is 8 in embedded images.

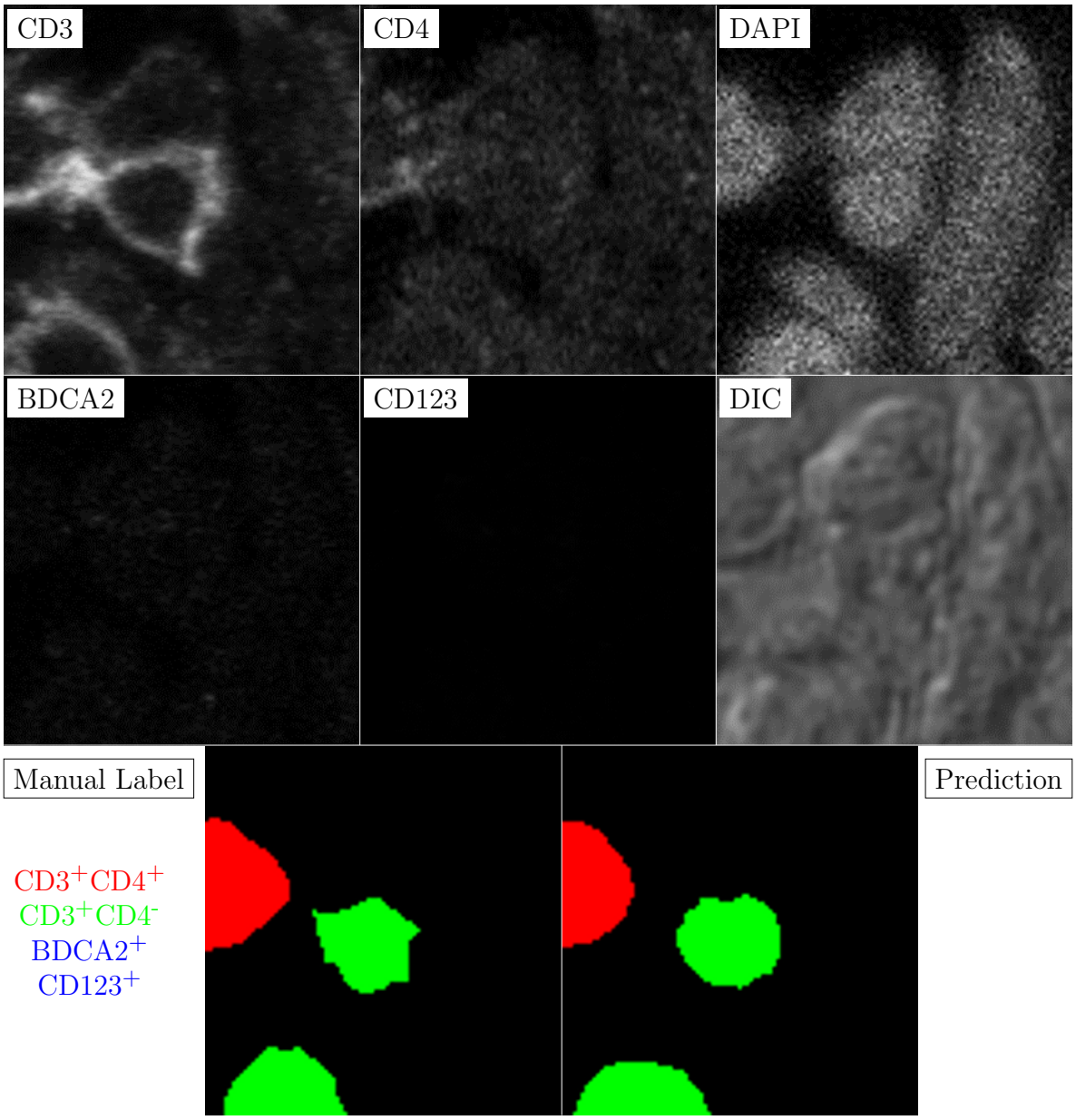

Figure E.1: **Human FF**. True ROI matrix size is 297 × 286. Depicted ROI matrix size is 297 × 286.

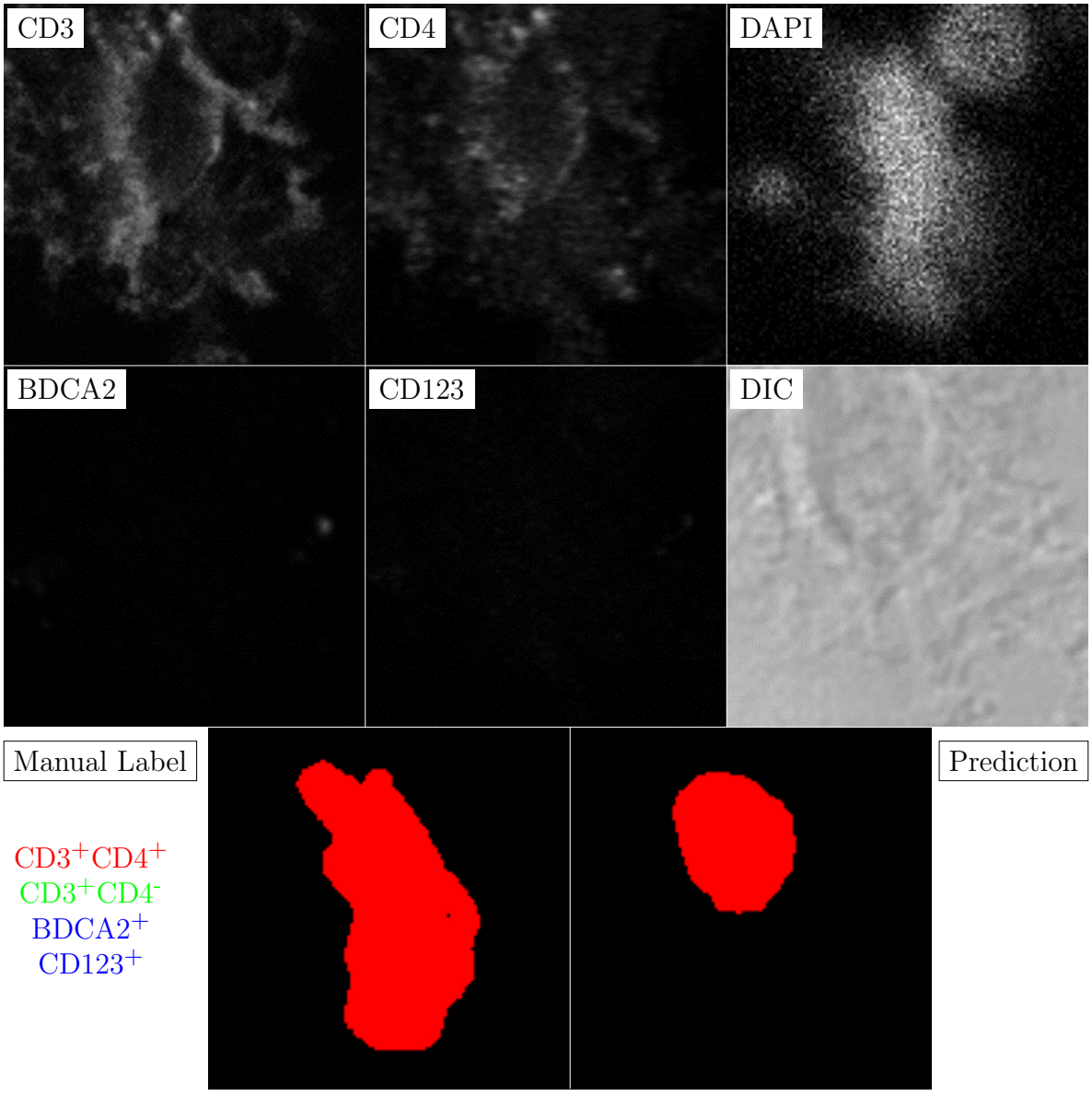

Figure E.2: **Human FF**. True ROI matrix size is 298 × 298. Depicted ROI matrix size is 298 × 298.

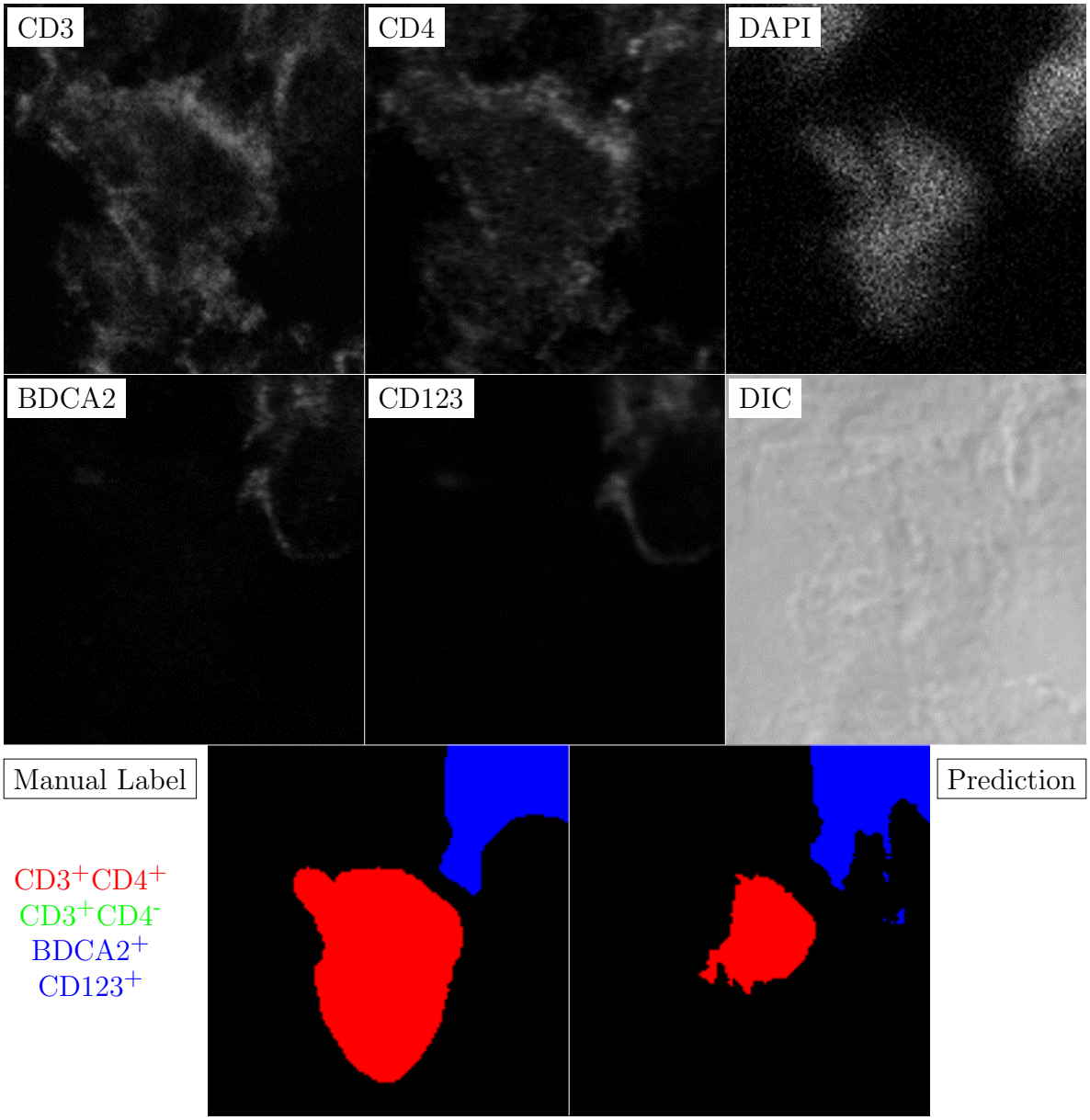

Figure E.3: **Human FF**. True ROI matrix size is 297 × 289. Depicted ROI matrix size is 297 × 289.

APPENDIX F

MOUSE CROSS VALIDATION EXAMPLES

This section presents the full dataset from the cross validation run of the murine fresh frozen dataset as discussed in Chapter 4.

Label ROIs are embedded in document with DEFLATE png compression at full resolution. In the label image matrix, Ta are red, Tw are green, and Tw are blue. All images for the biopsy channel were downsampled with bilinear interpolation such that the maximum x,y dimension of the image was equal to 300 pixels. This was done to keep the memory footprint of this pdf document at a reasonable level while also avoiding the compression artifacts of high level jpeg compression. Images were transformed from native bit depth of 12 by rescaling the image using the standard score (z-score). Z-score images were transformed to an 8-bit range by trucating the distribution of z-scores at 3 standard deviations from the mean and rescaling to the range [0,255]. Pixel size is $0.23\,\mu m \times 0.23\,\mu m$. Bit depth is 8 in embedded images.

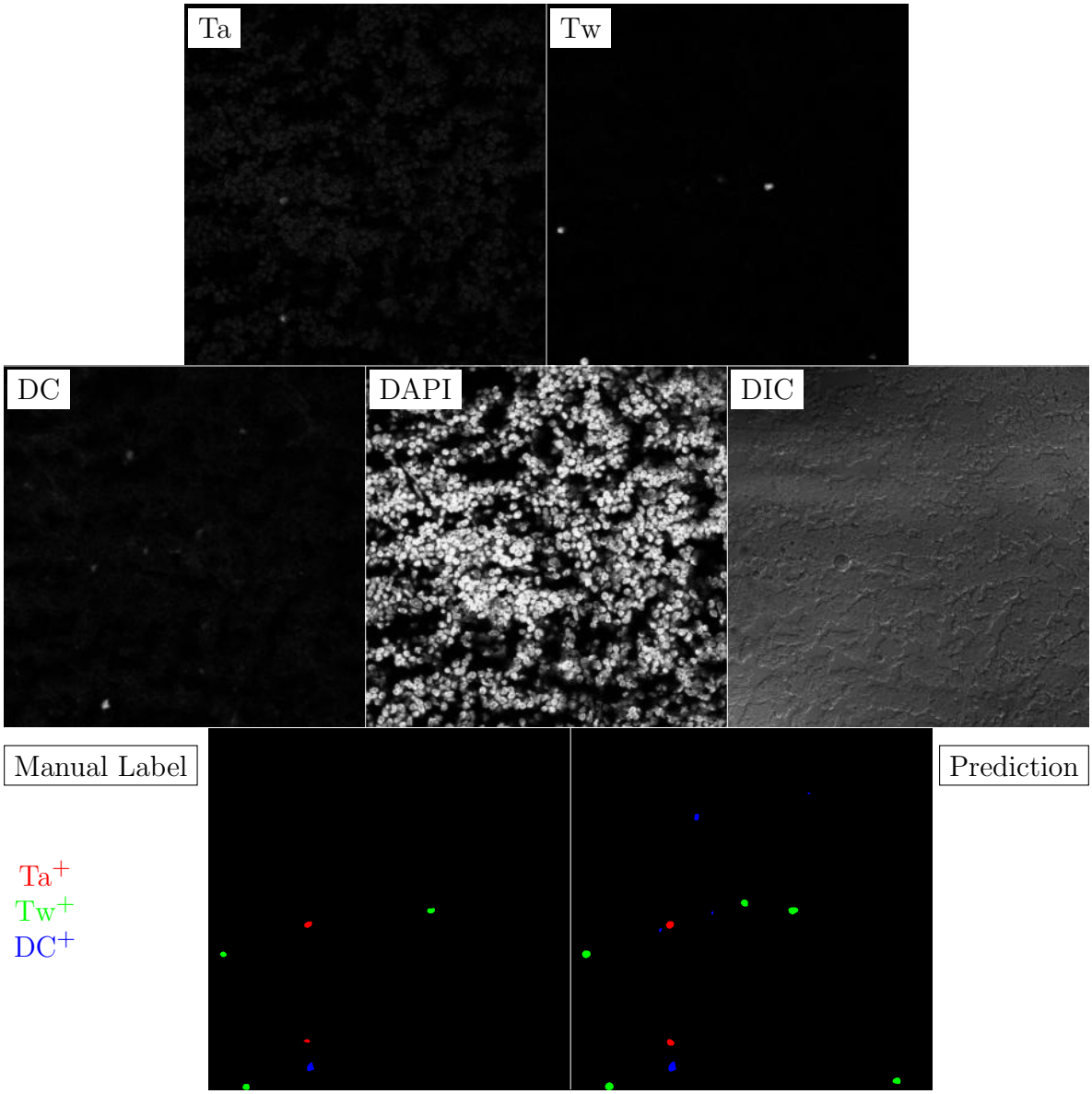

Figure F.1: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

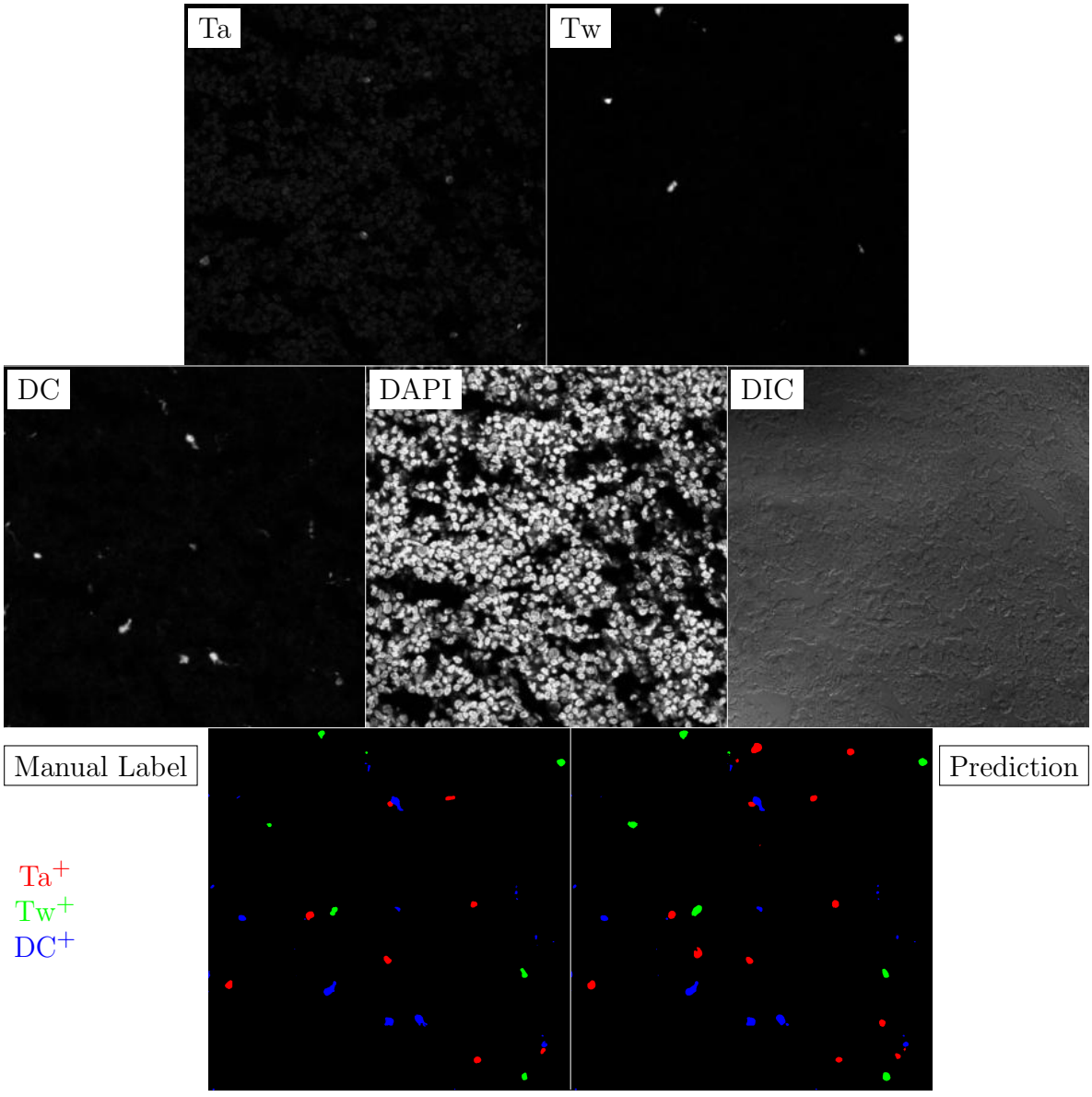

Figure F.2: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

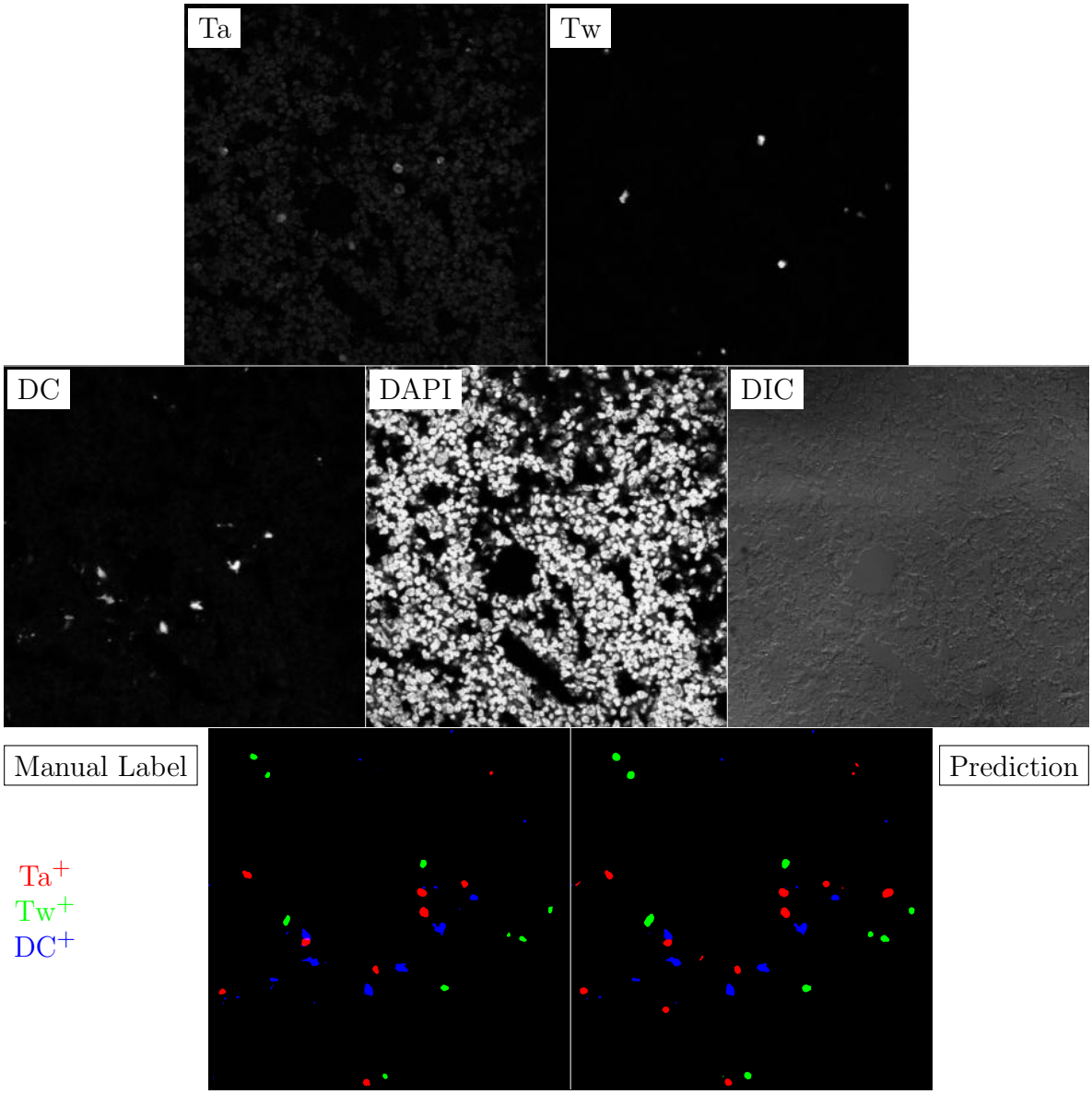

Figure F.3: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

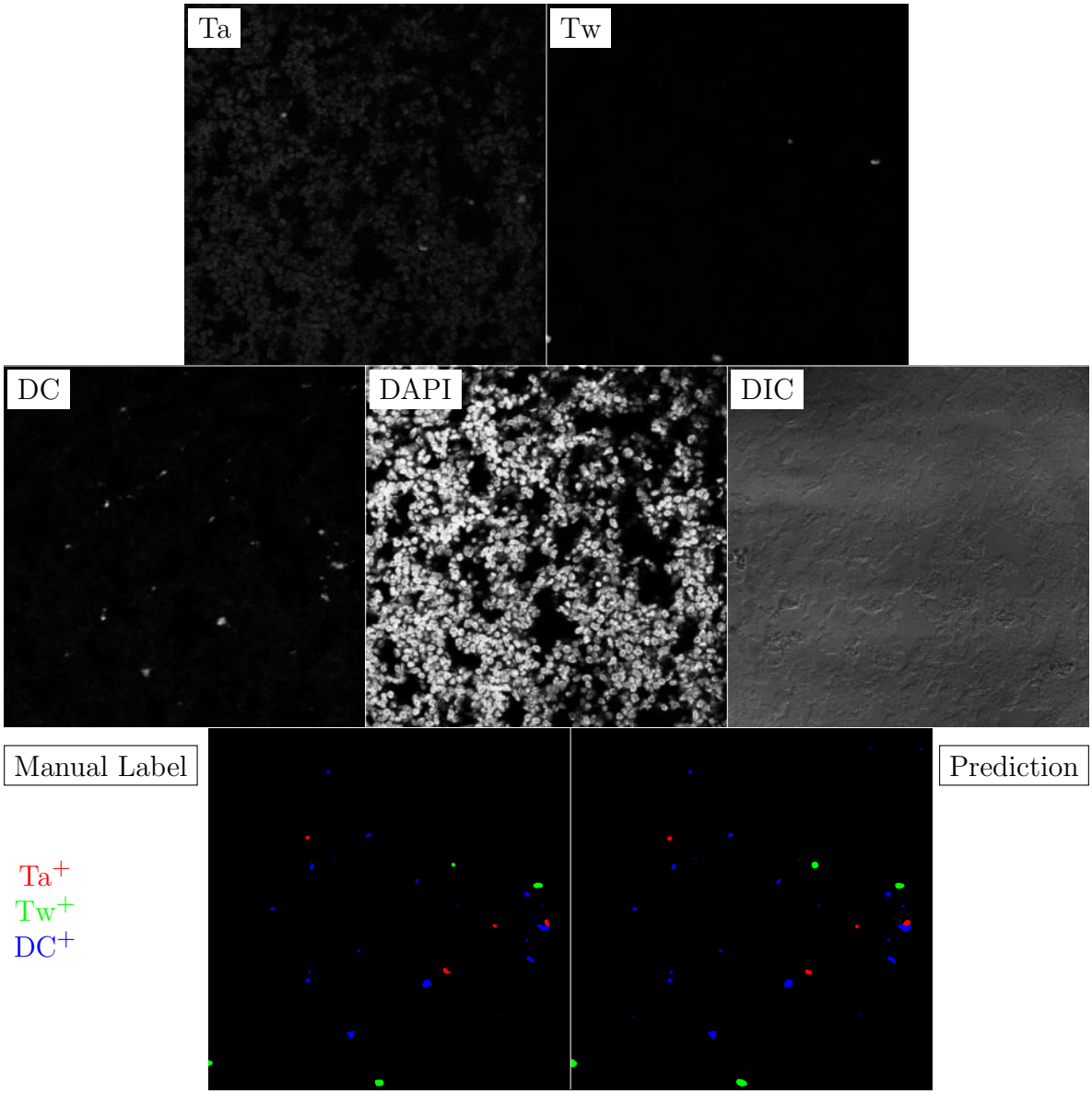

Figure F.4: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

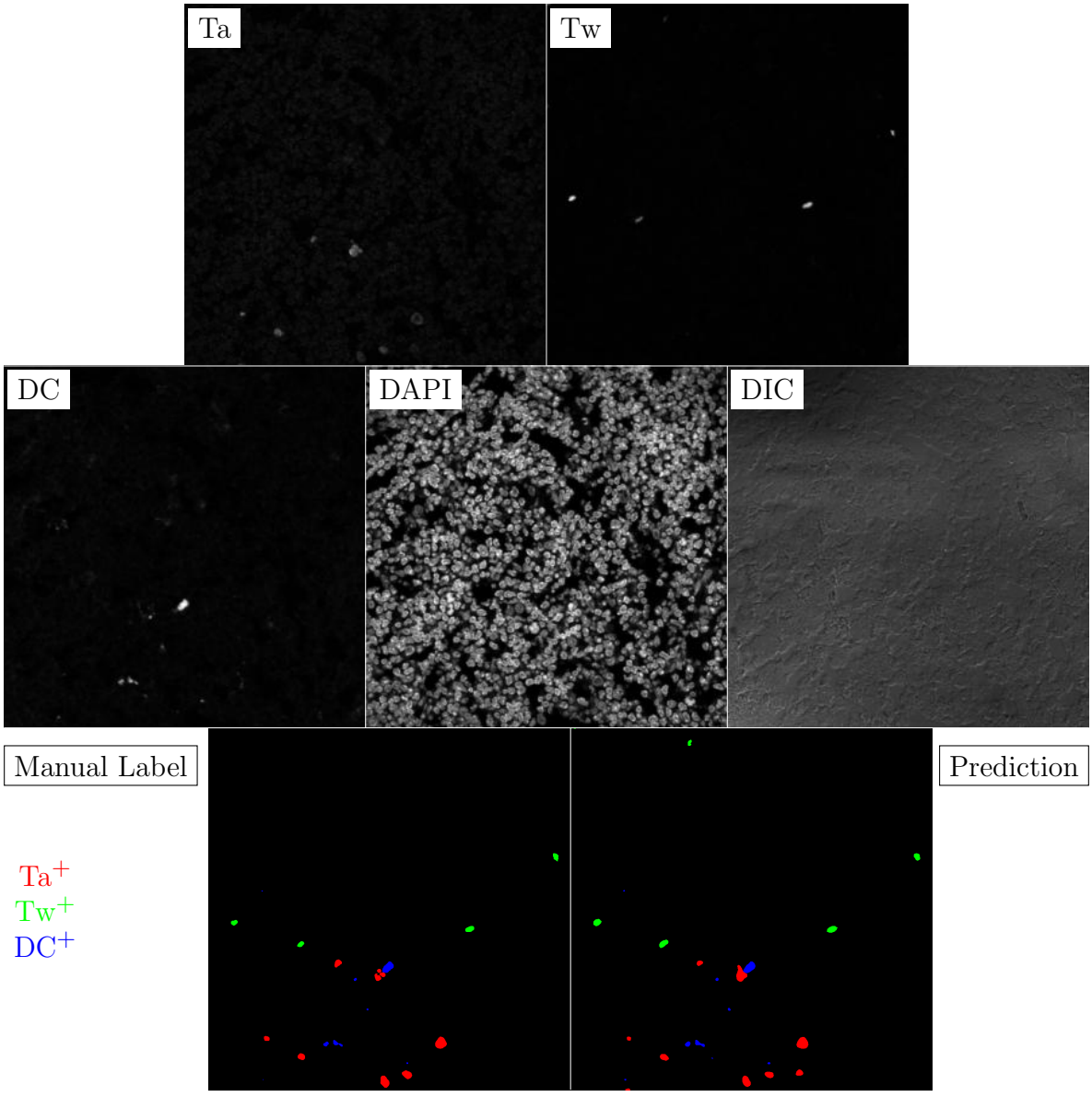

Figure F.5: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

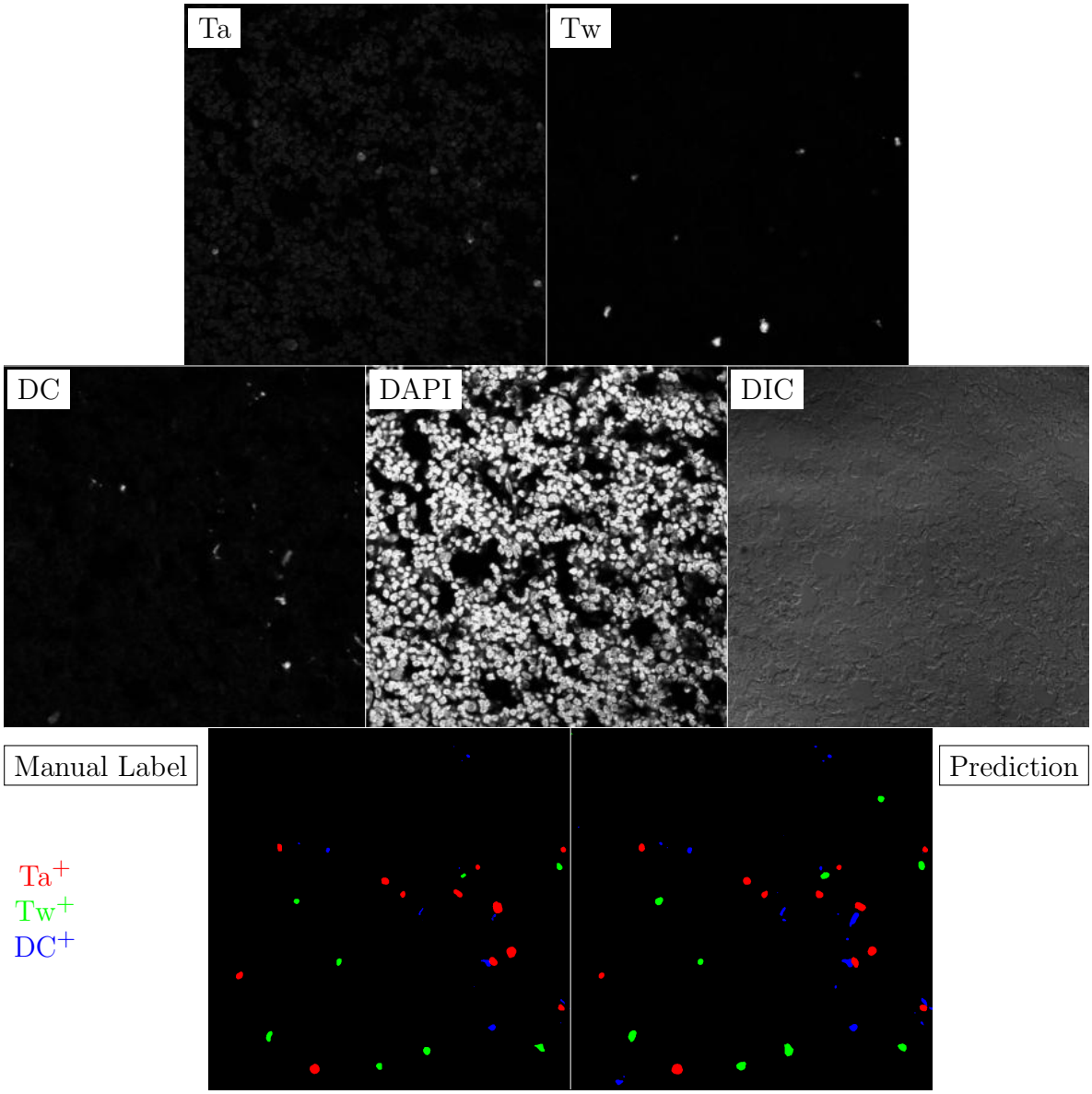

Figure F.6: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

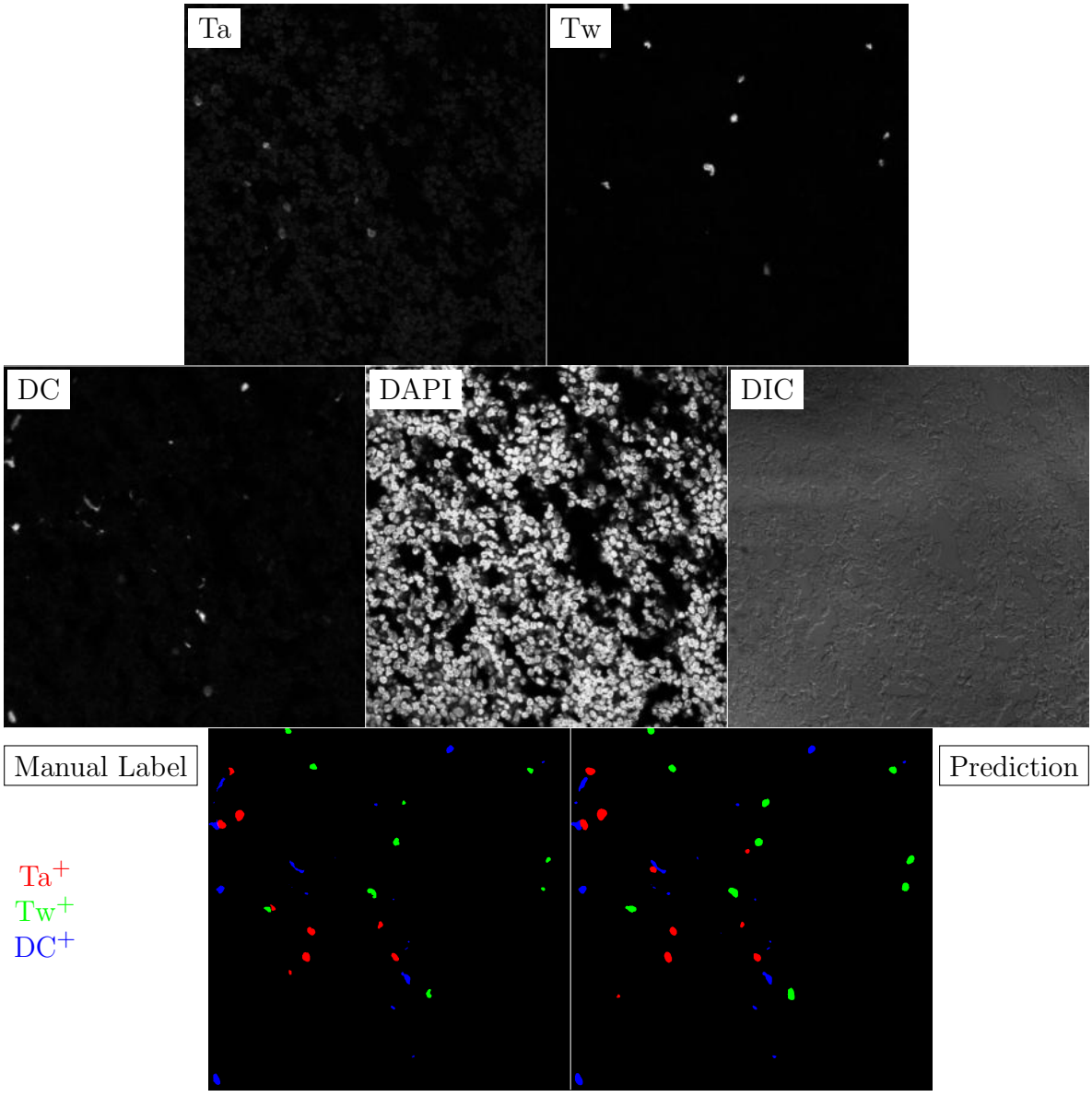

Figure F.7: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

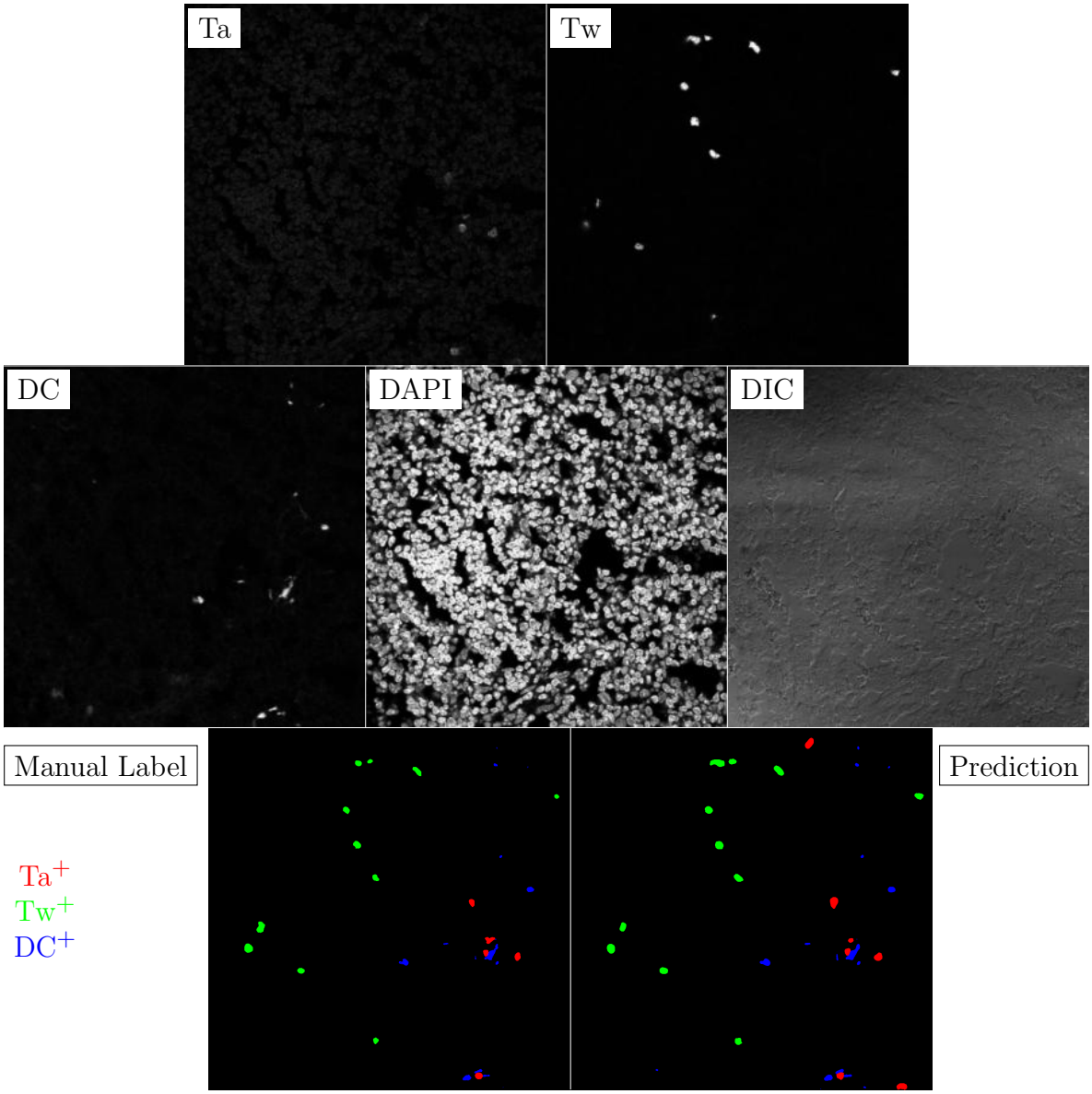

Figure F.8: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

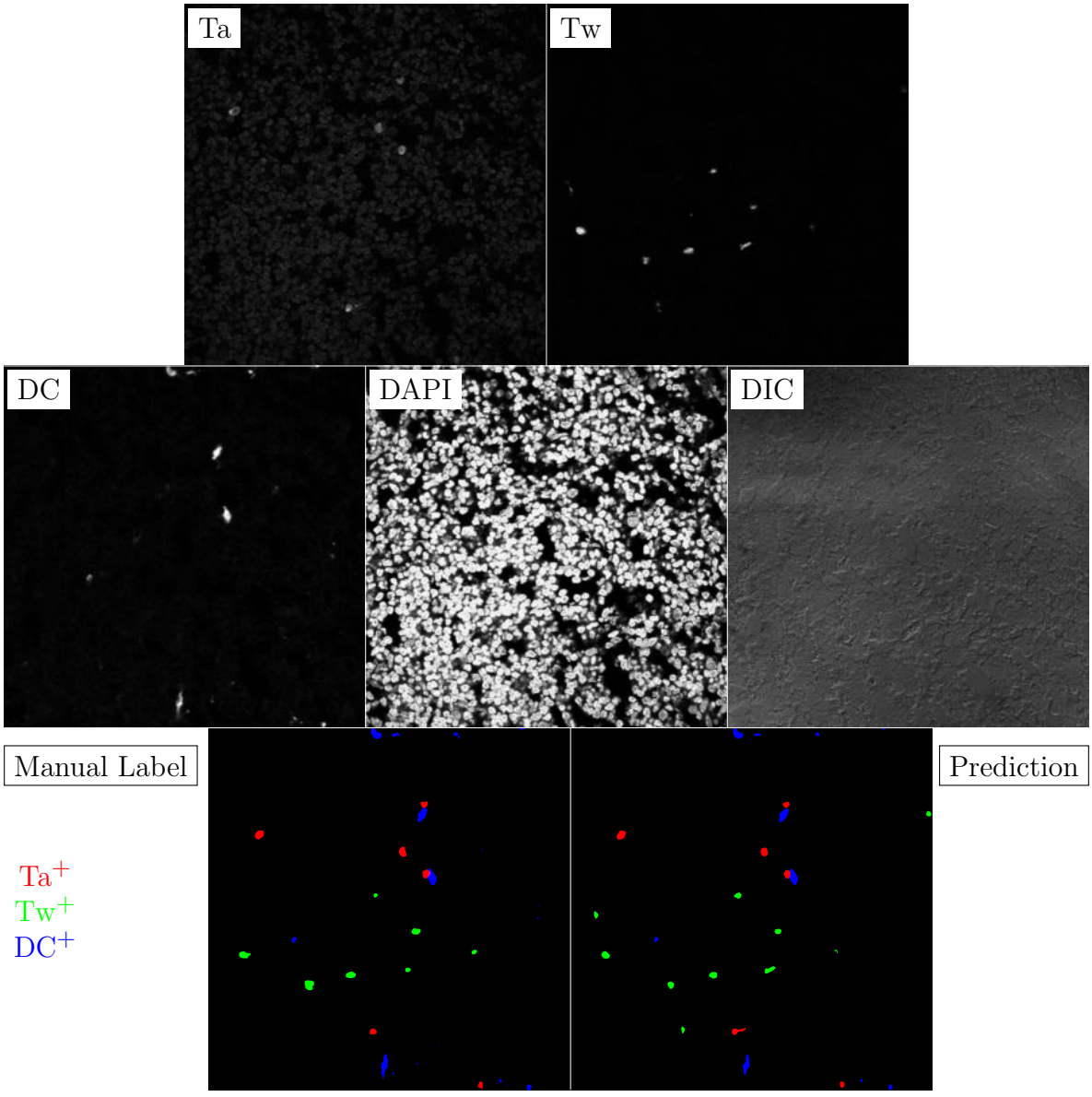

Figure F.9: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

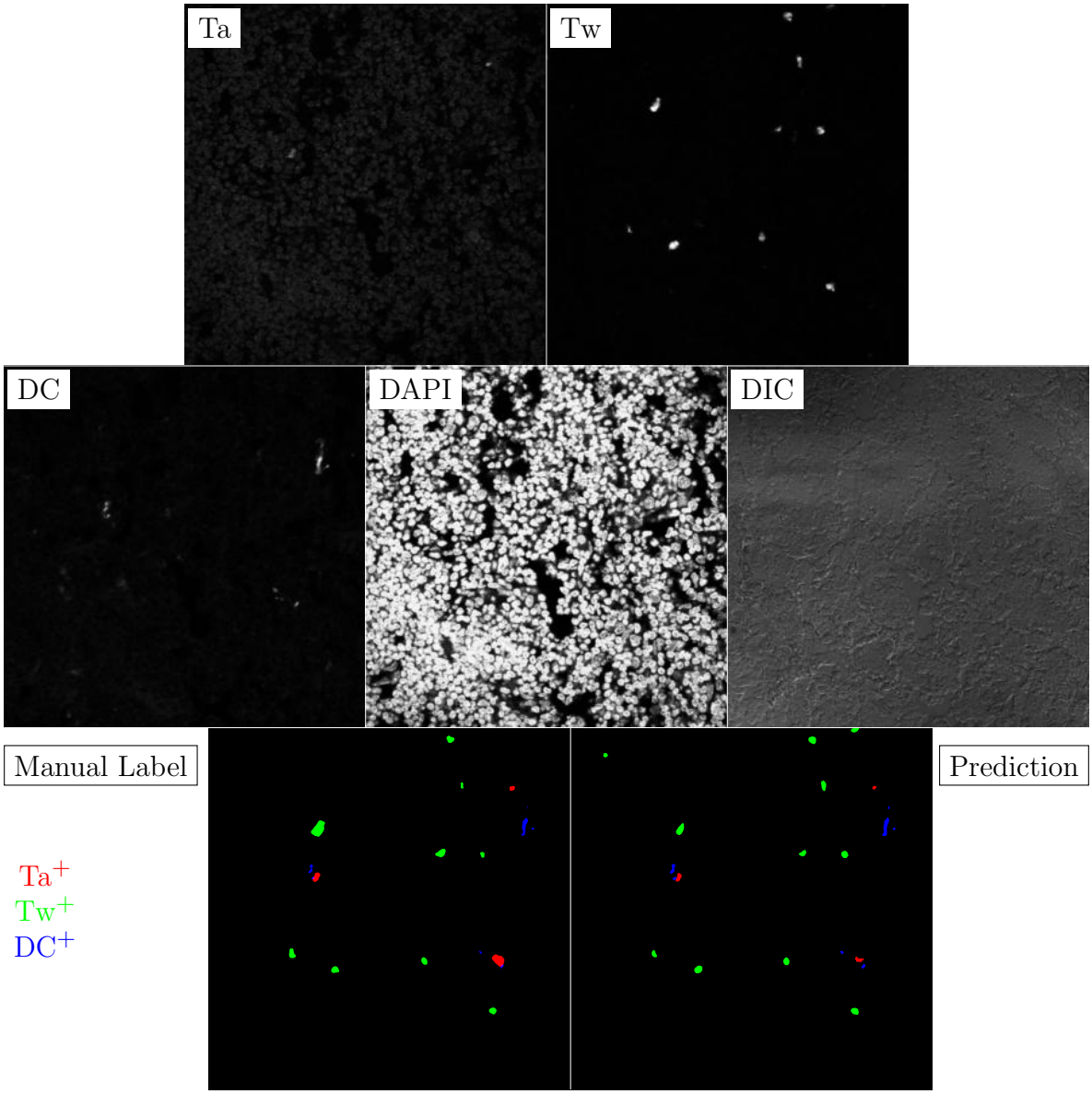

Figure F.10: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

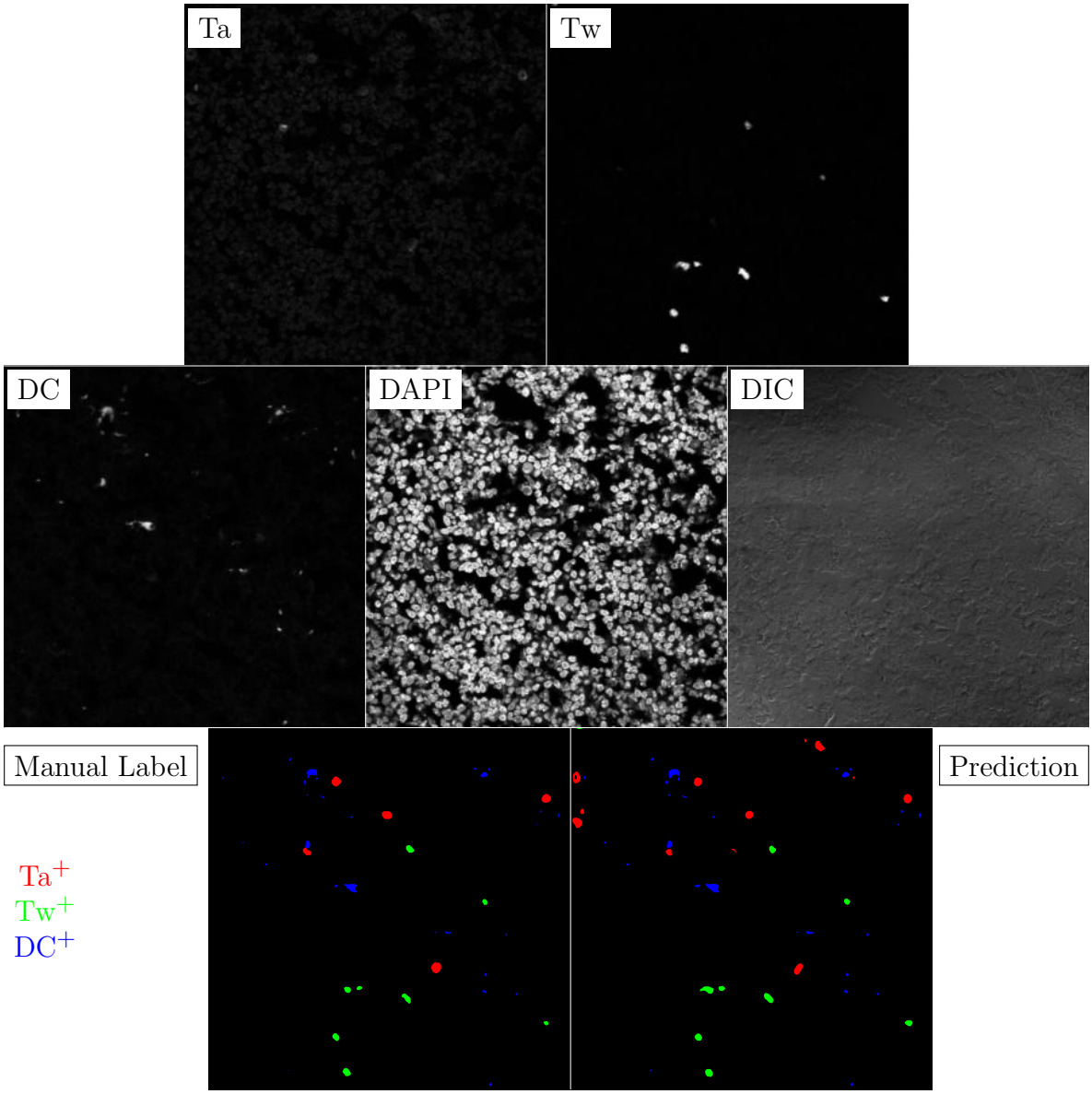

Figure F.11: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

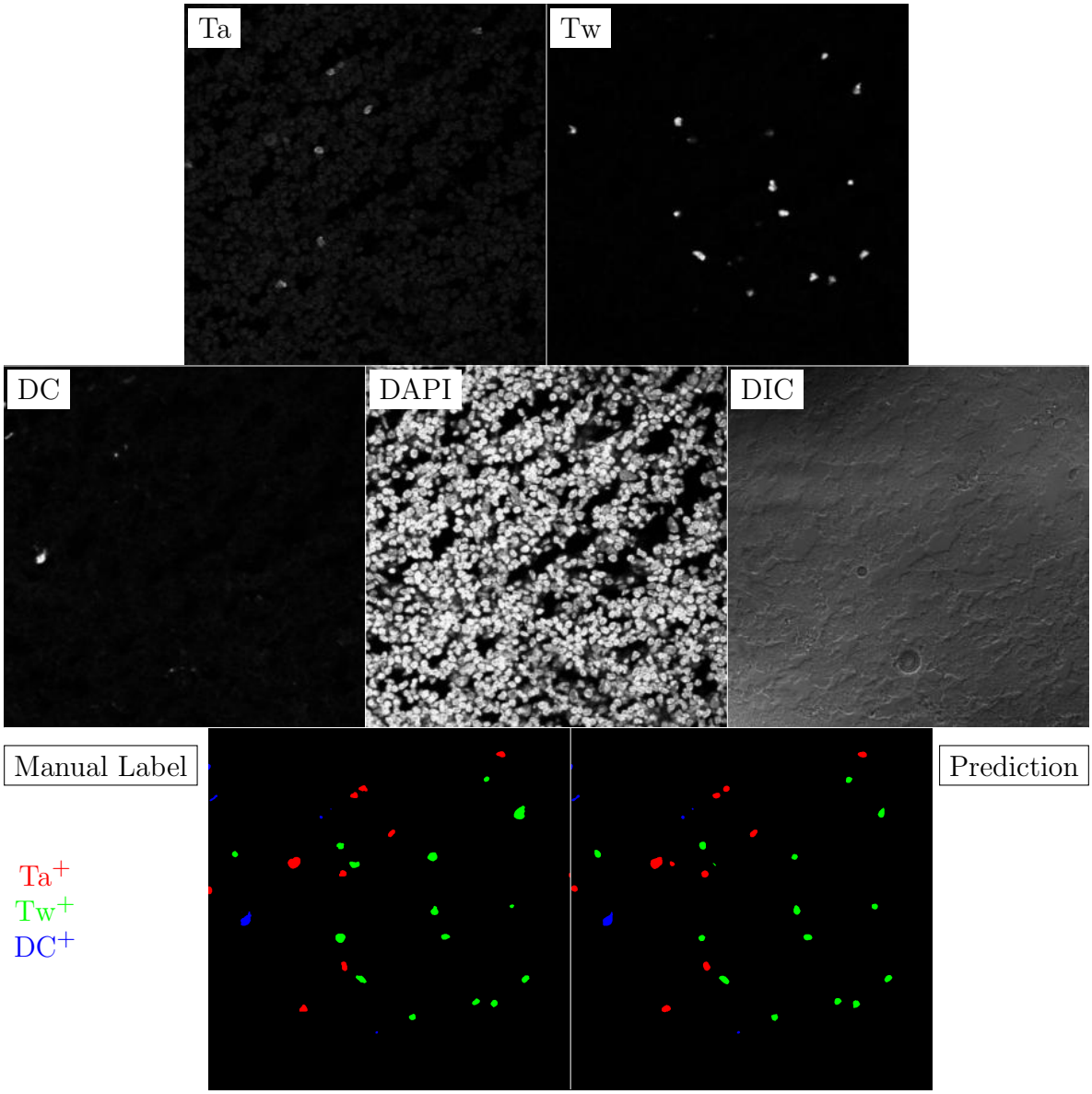

Figure F.12: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

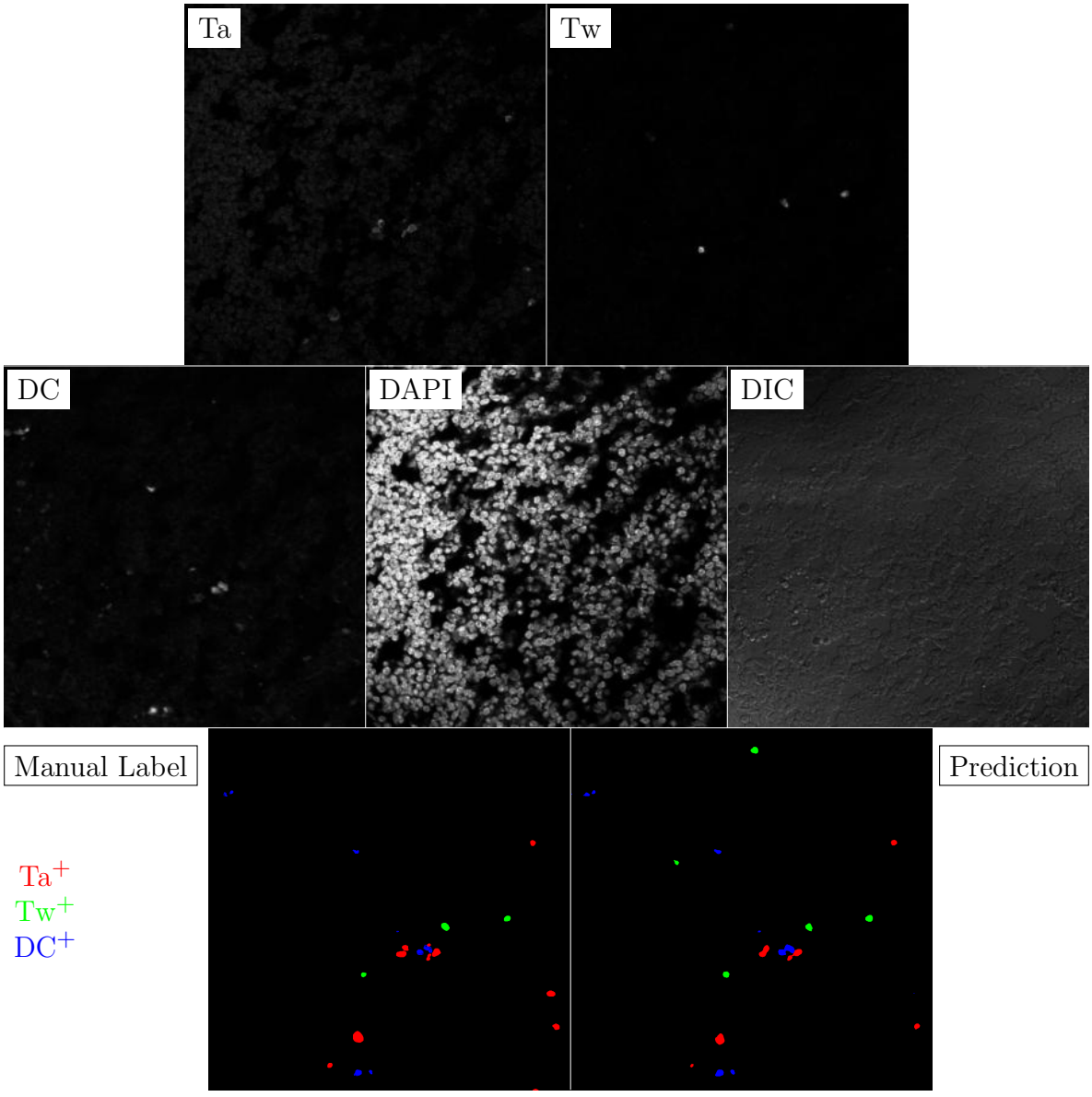

Figure F.13: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

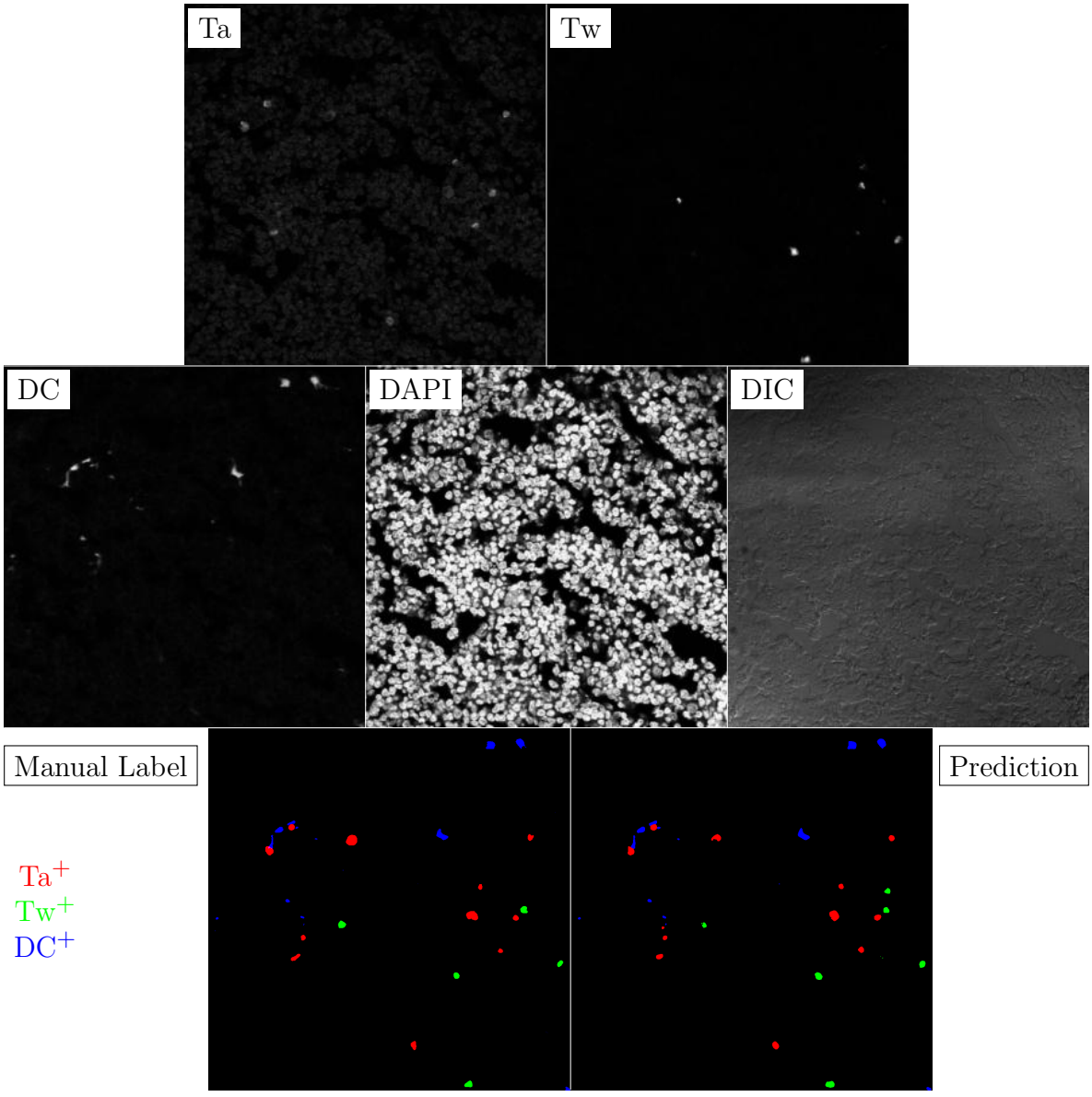

Figure F.14: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

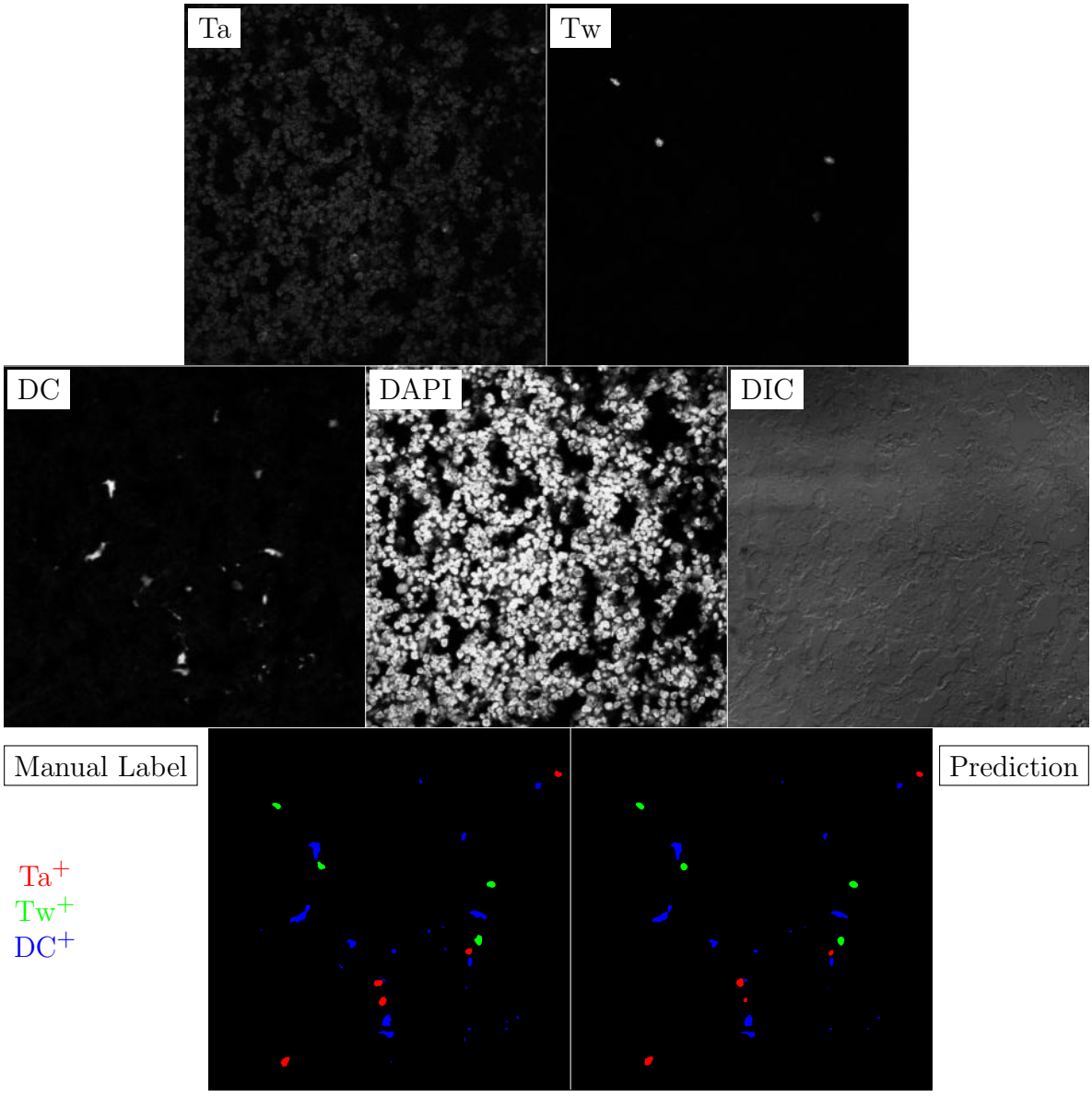

Figure F.15: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

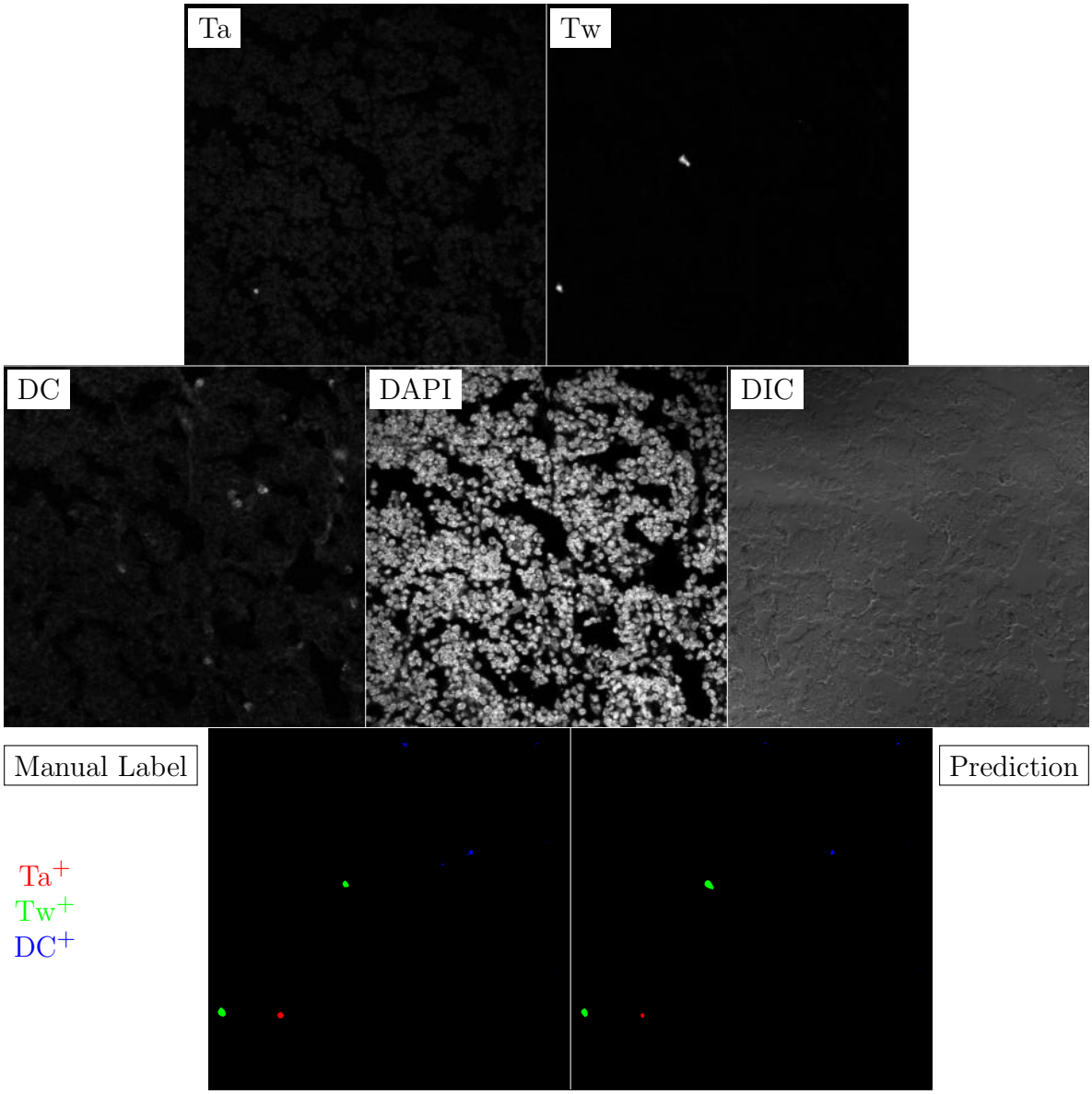

Figure F.16: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

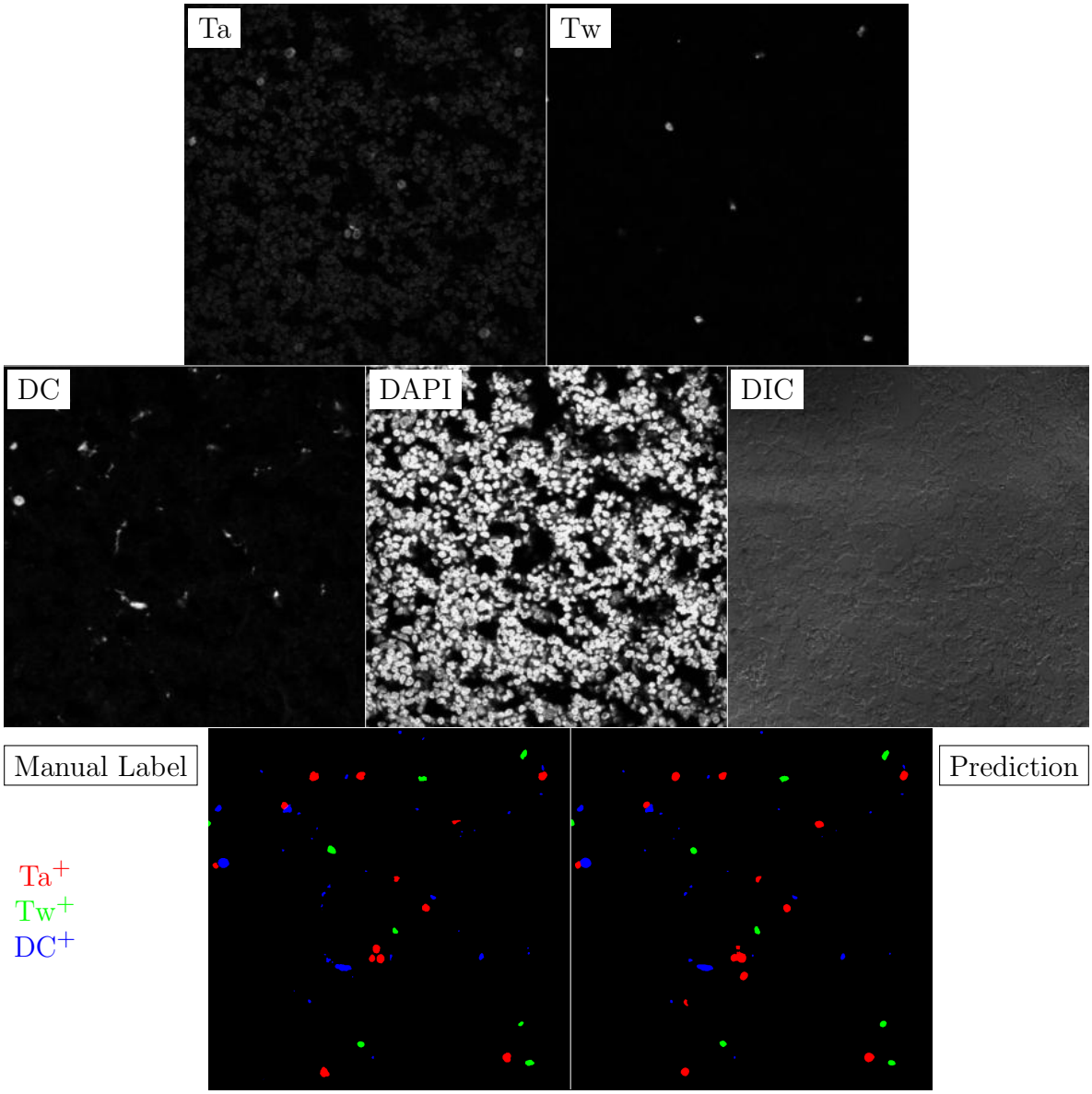

Figure F.17: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

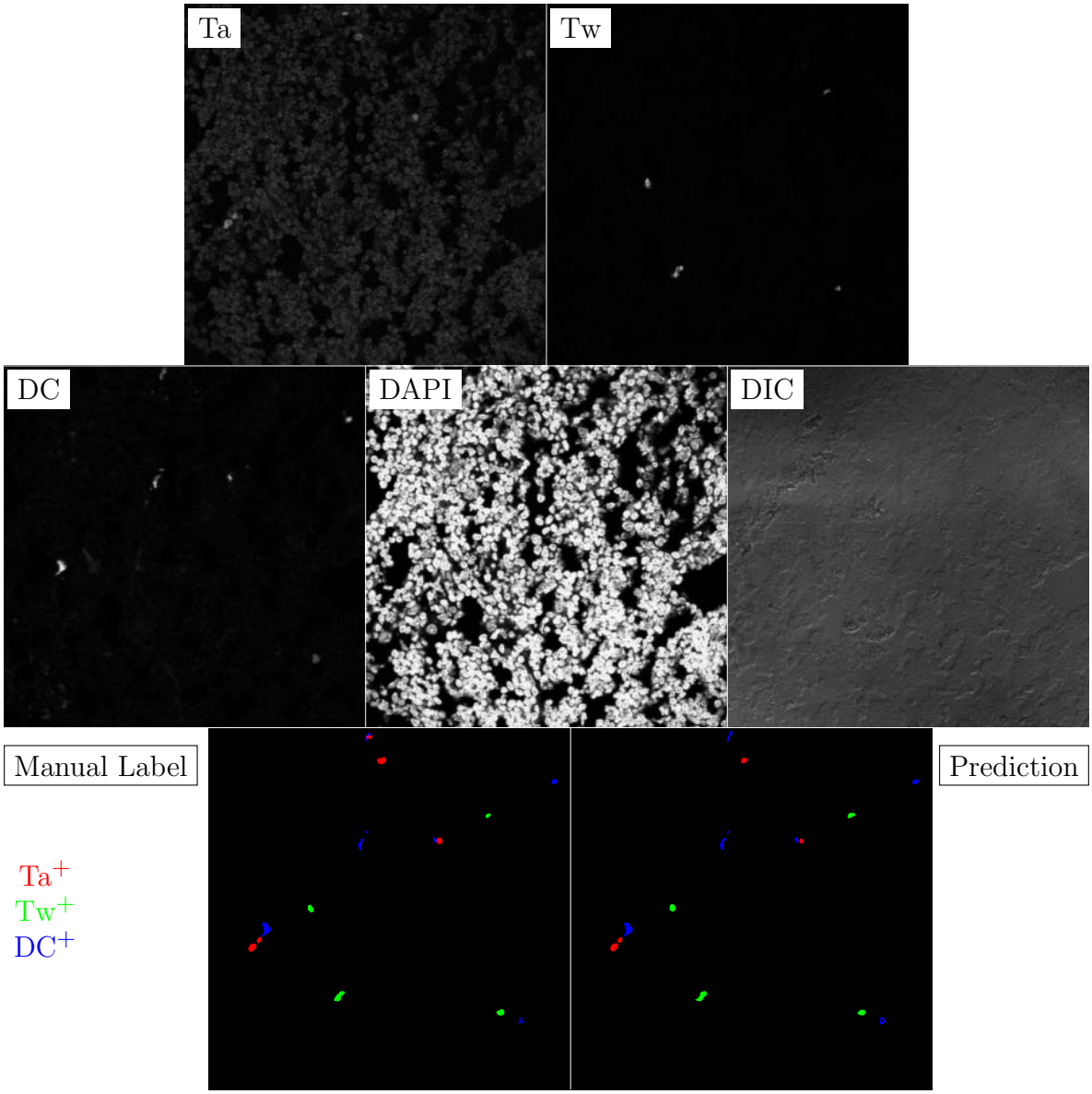

Figure F.18: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

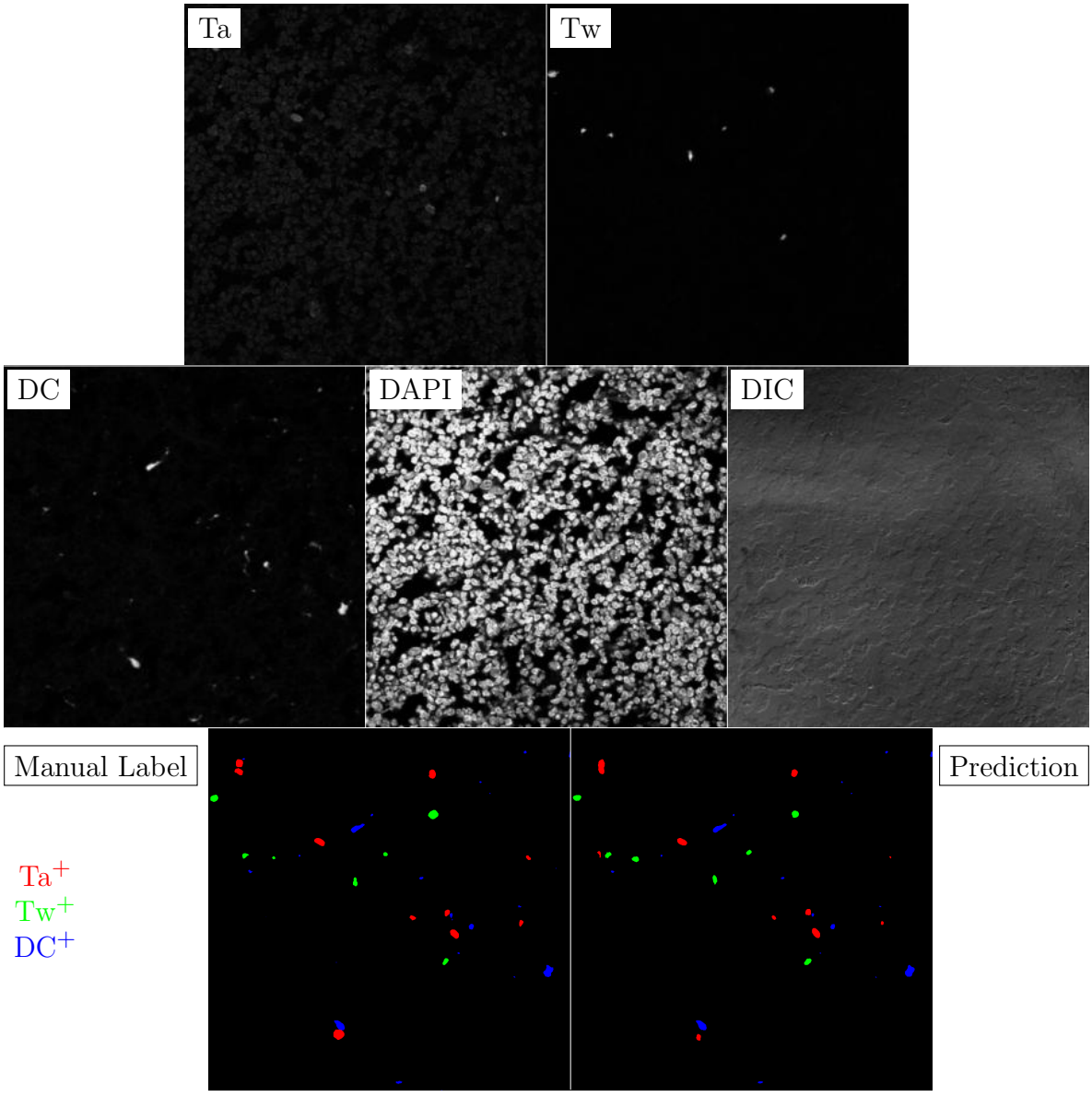

Figure F.19: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

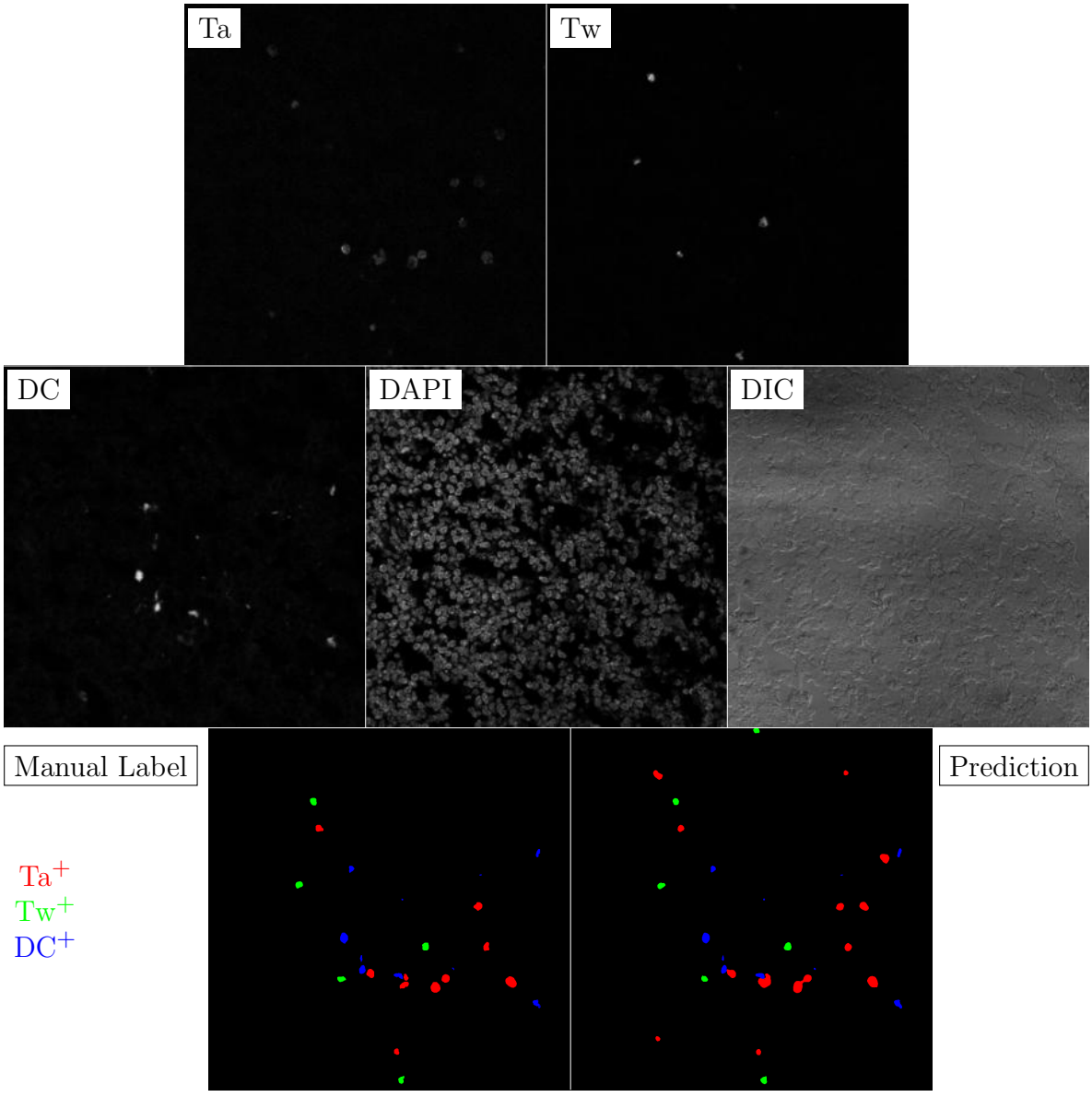

Figure F.20: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

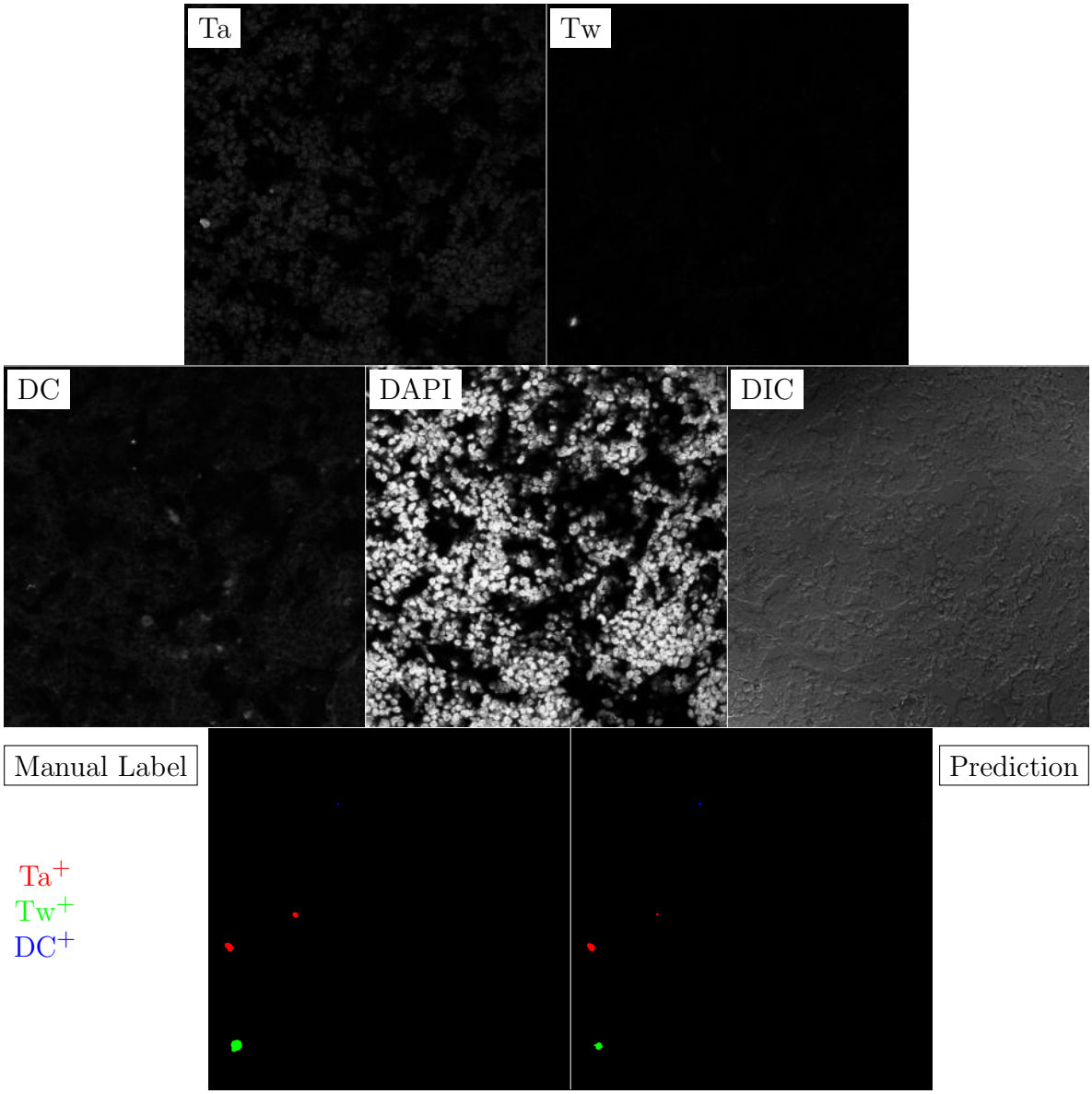

Figure F.21: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

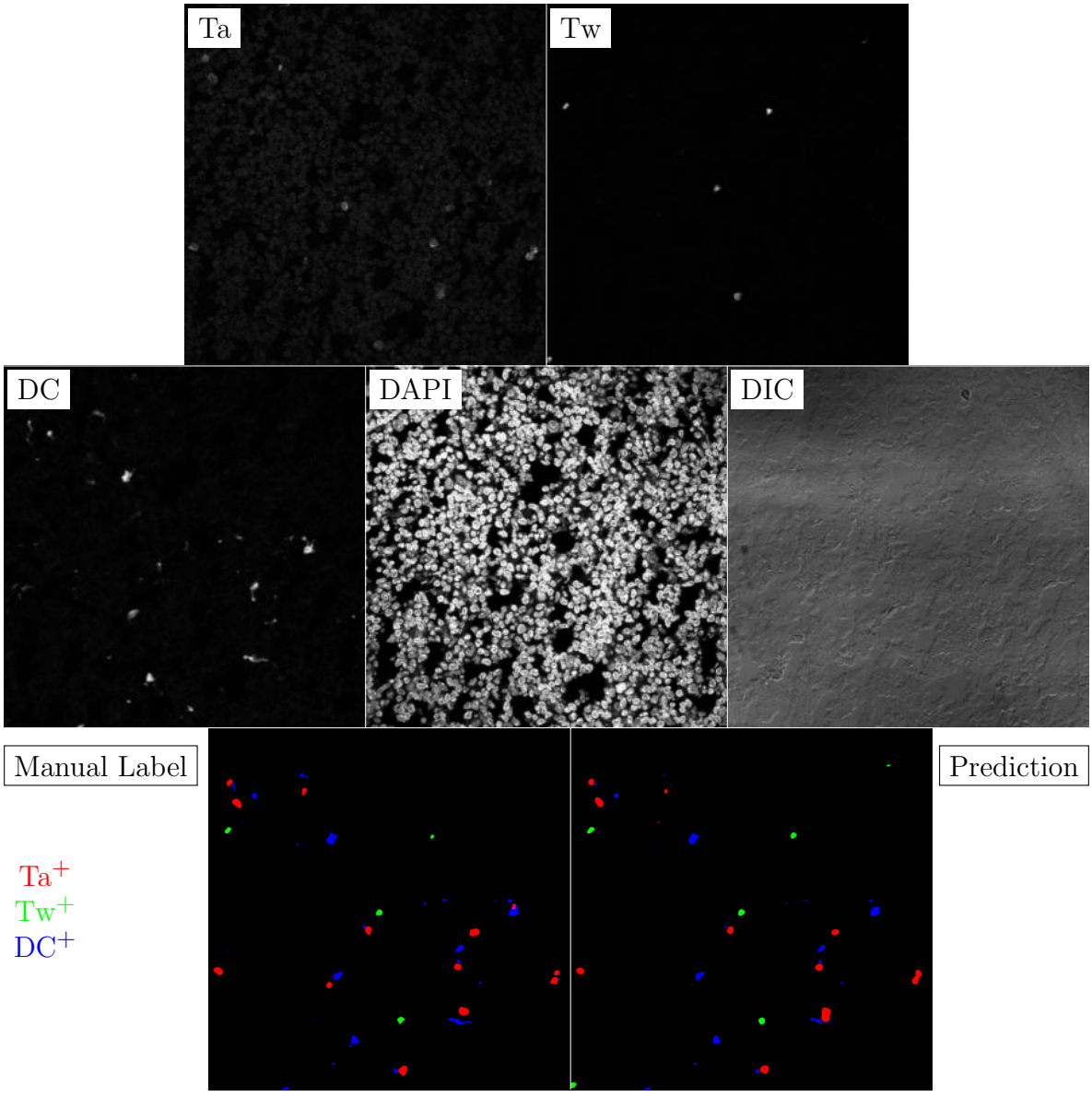

Figure F.22: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

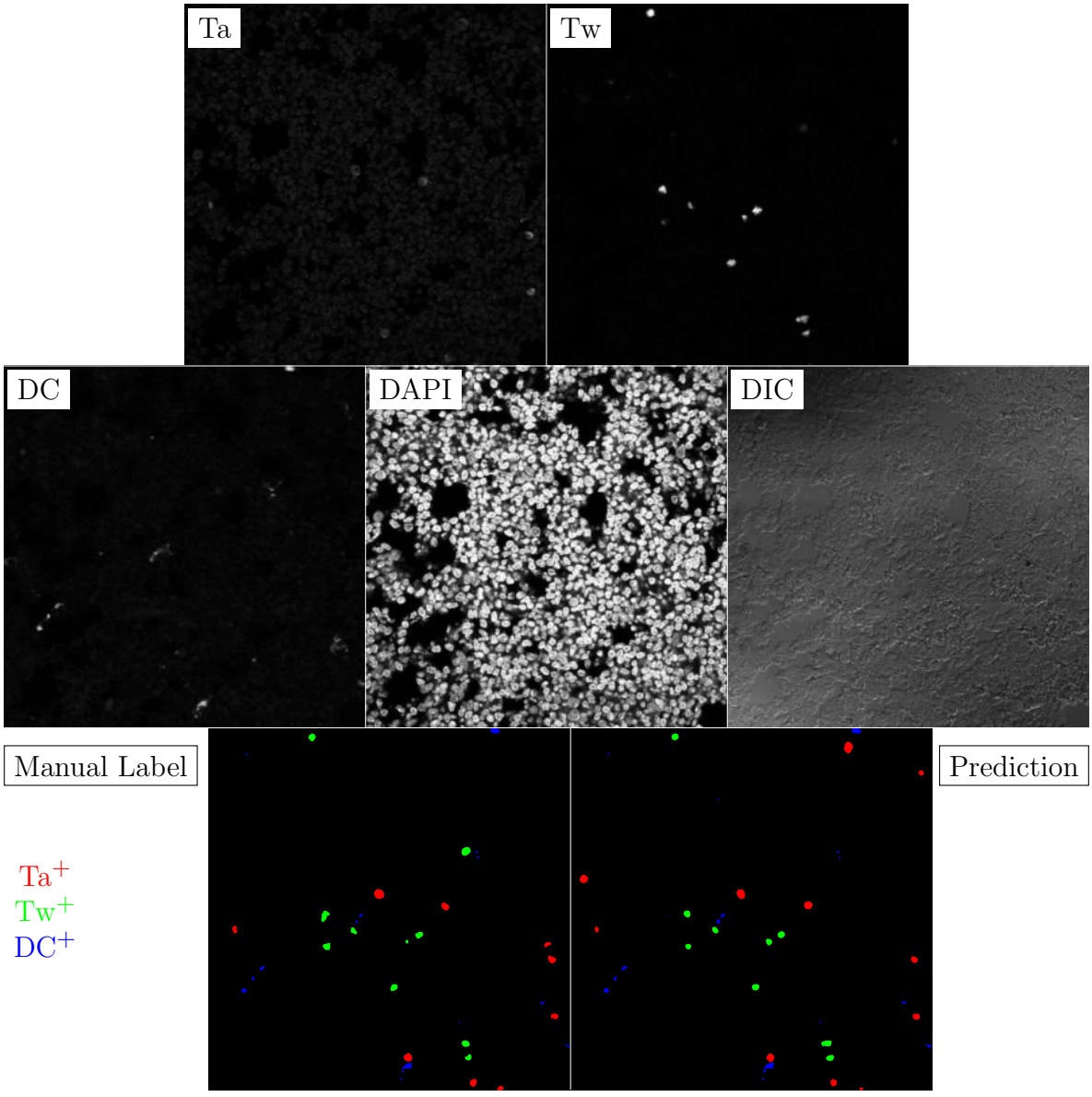

Figure F.23: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

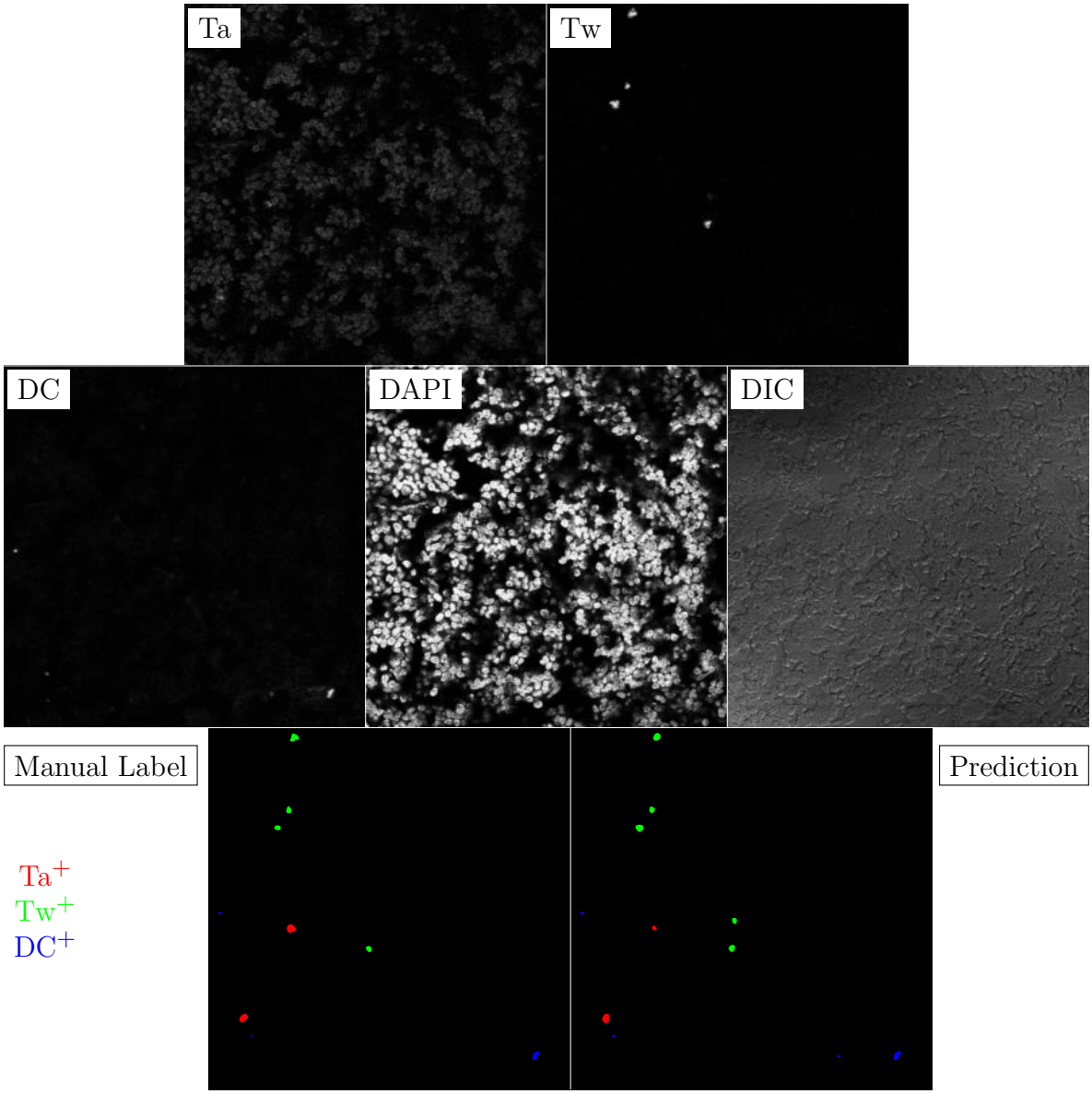

Figure F.24: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

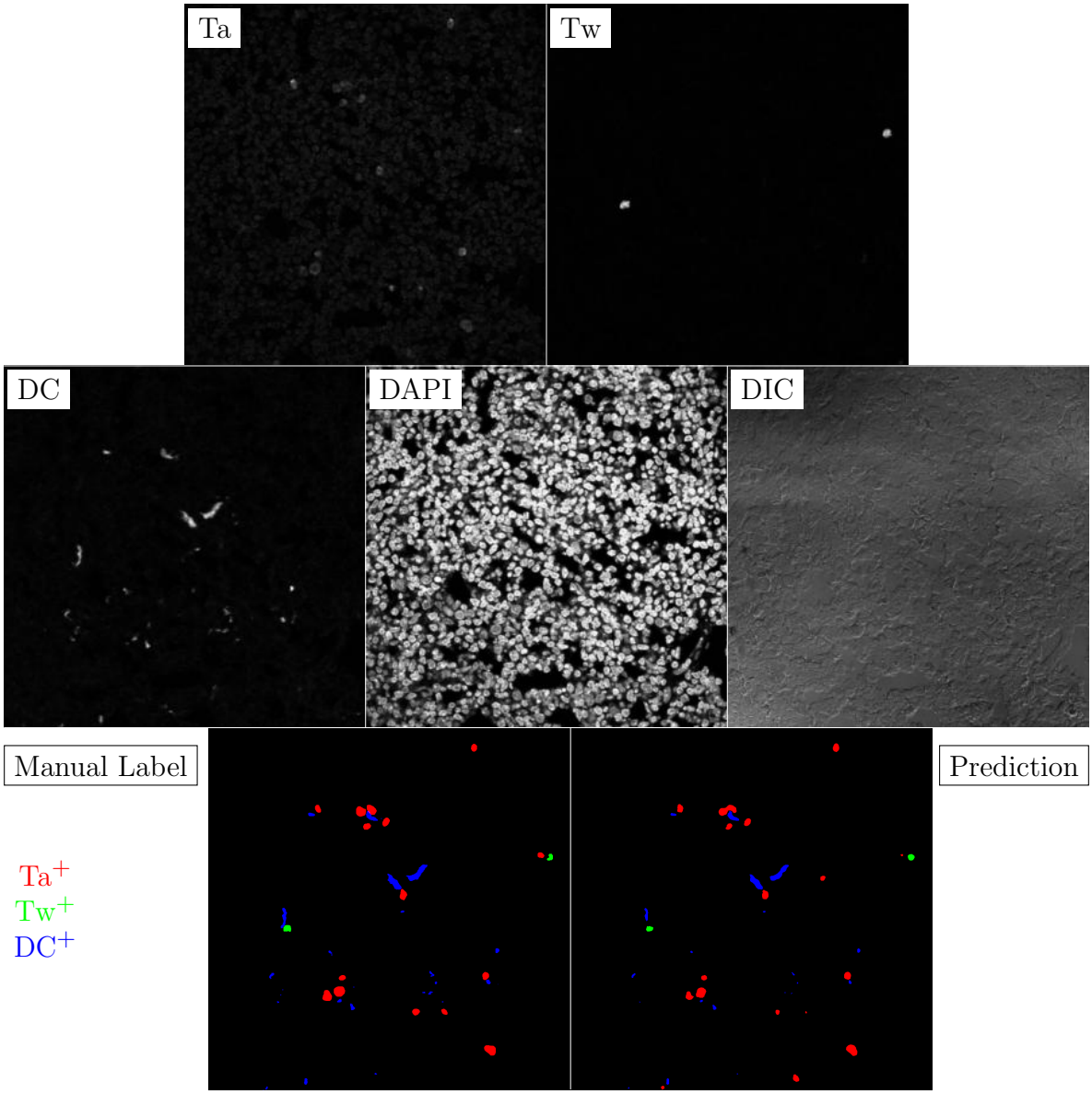

Figure F.25: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

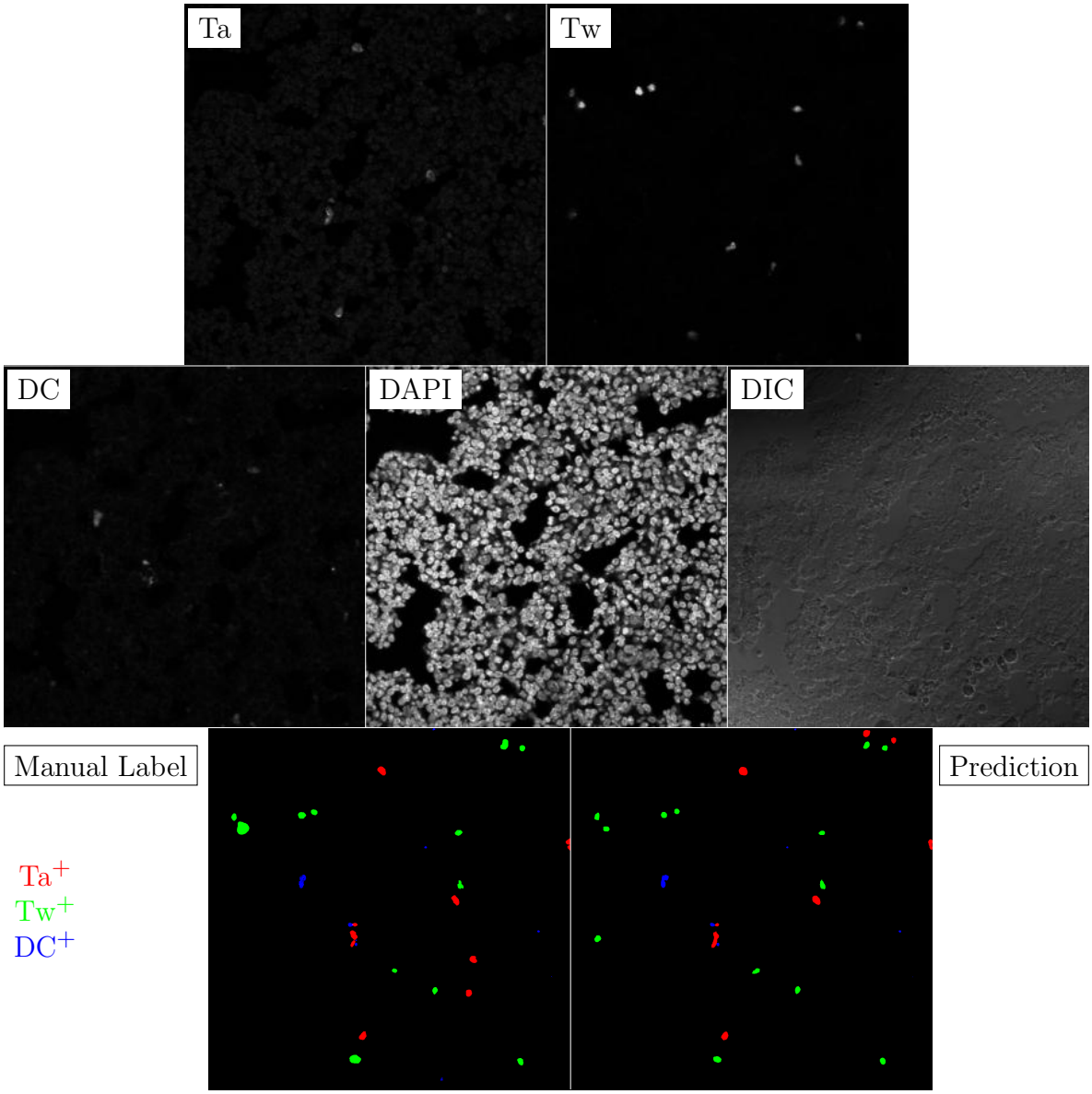

Figure F.26: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

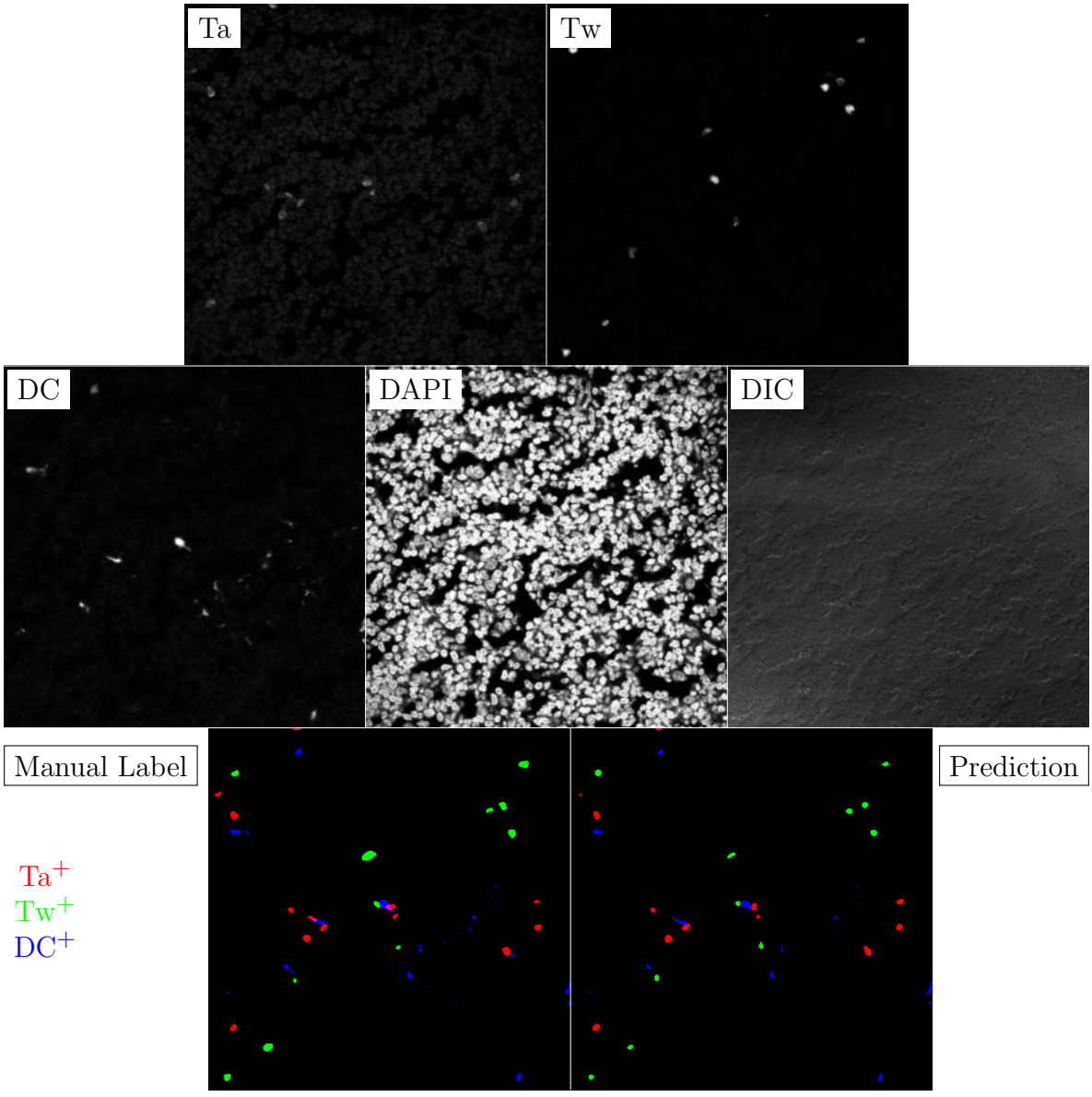

Figure F.27: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

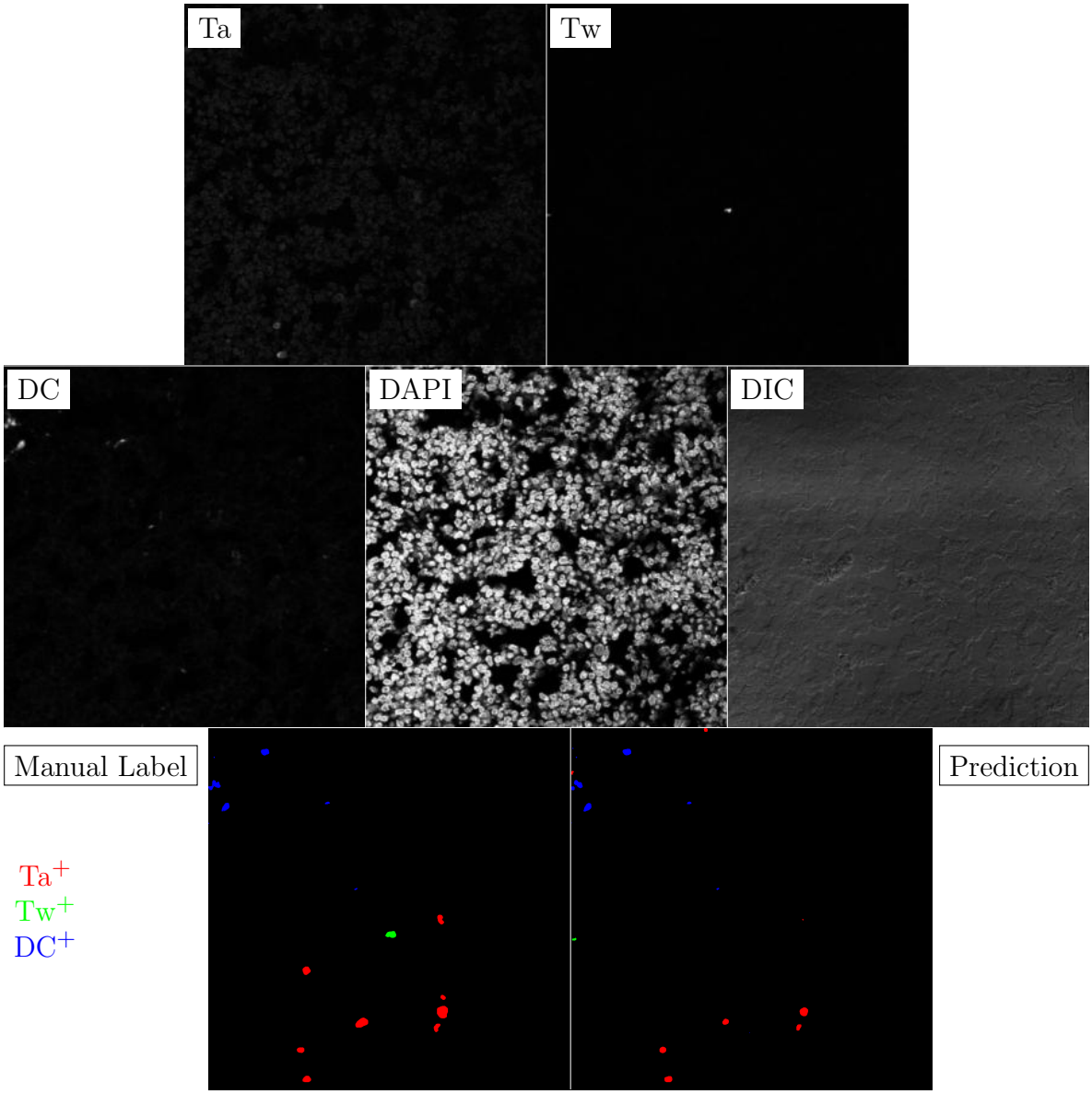

Figure F.28: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

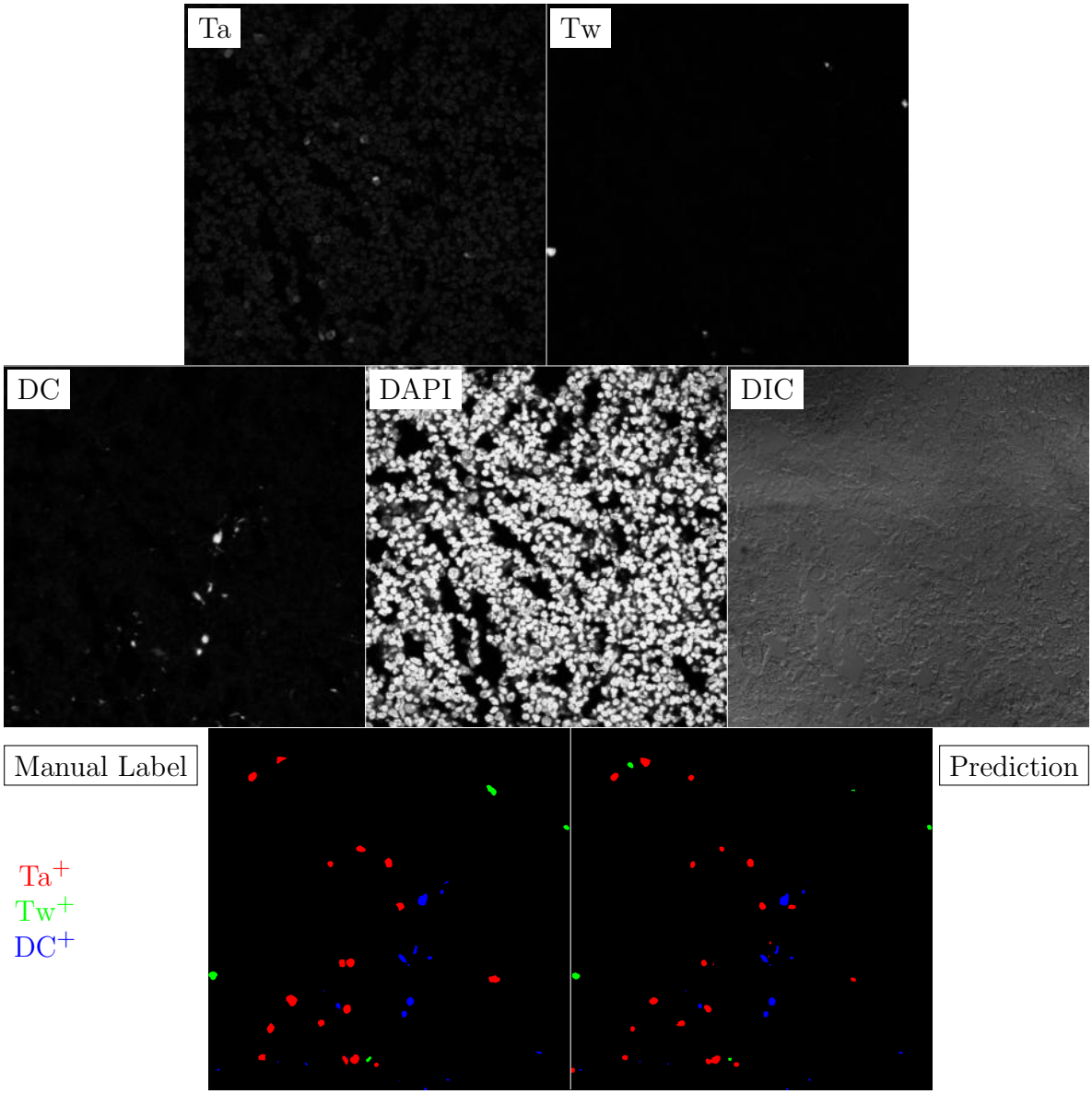

Figure F.29: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

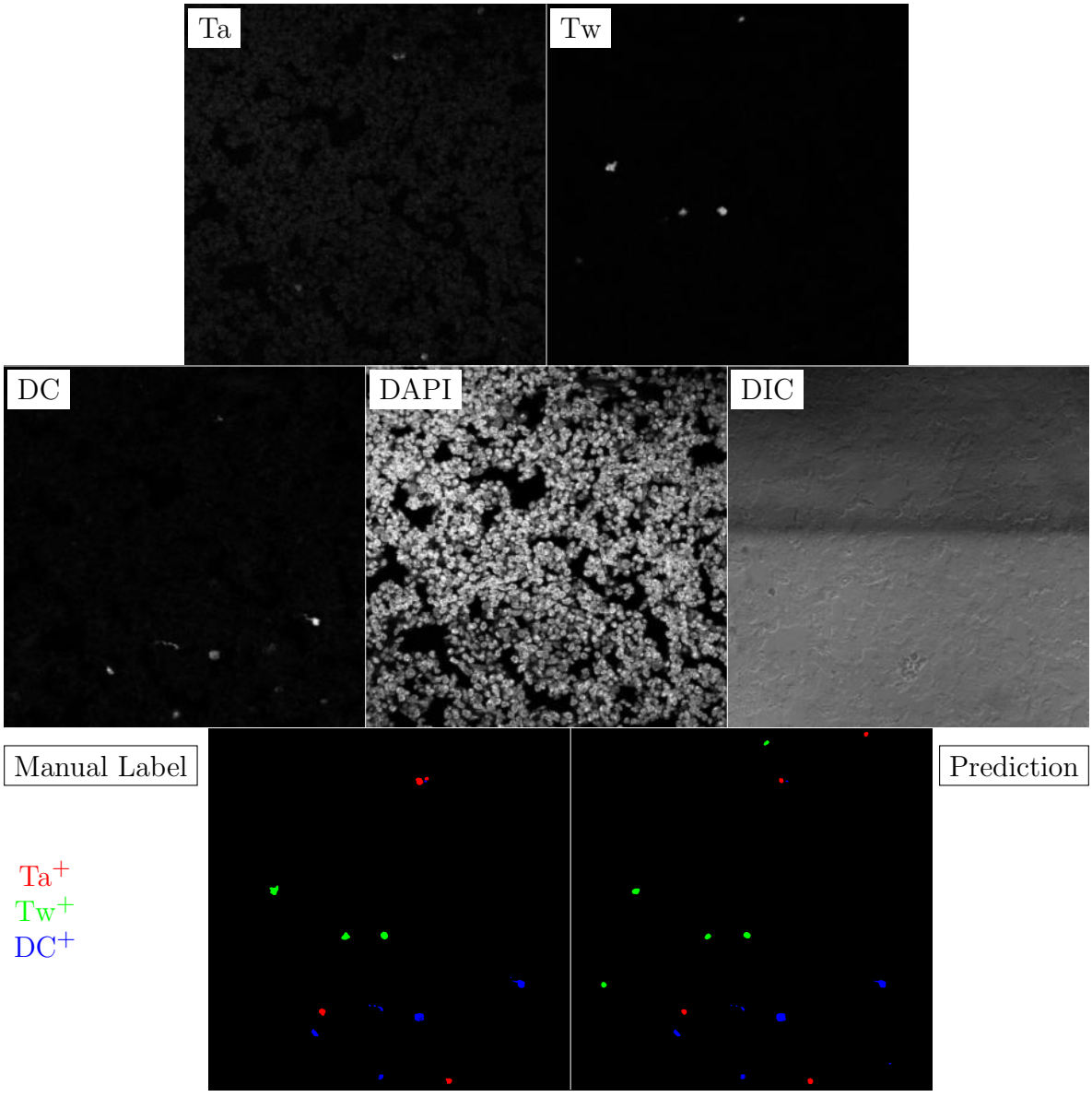

Figure F.30: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

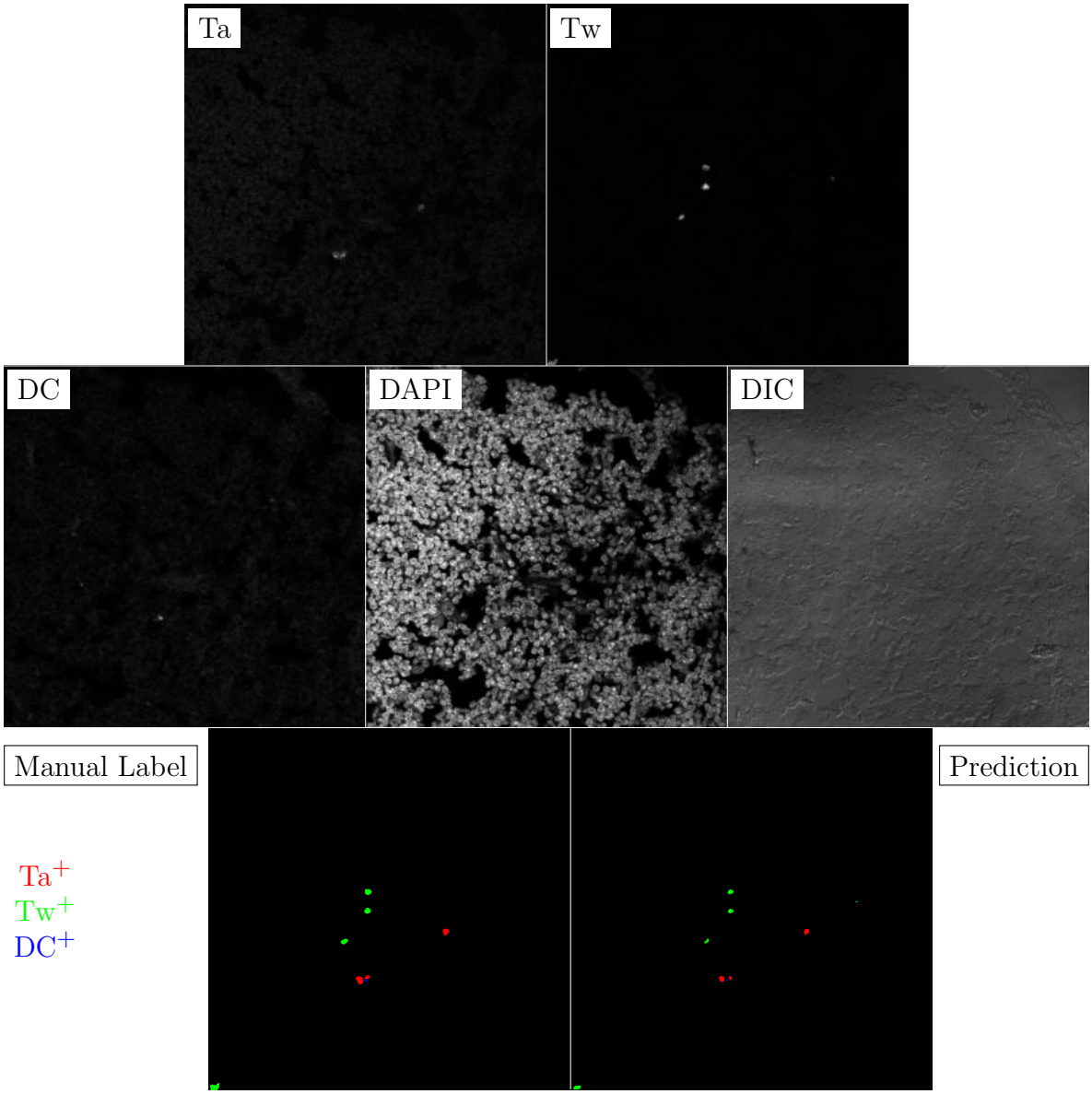

Figure F.31: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

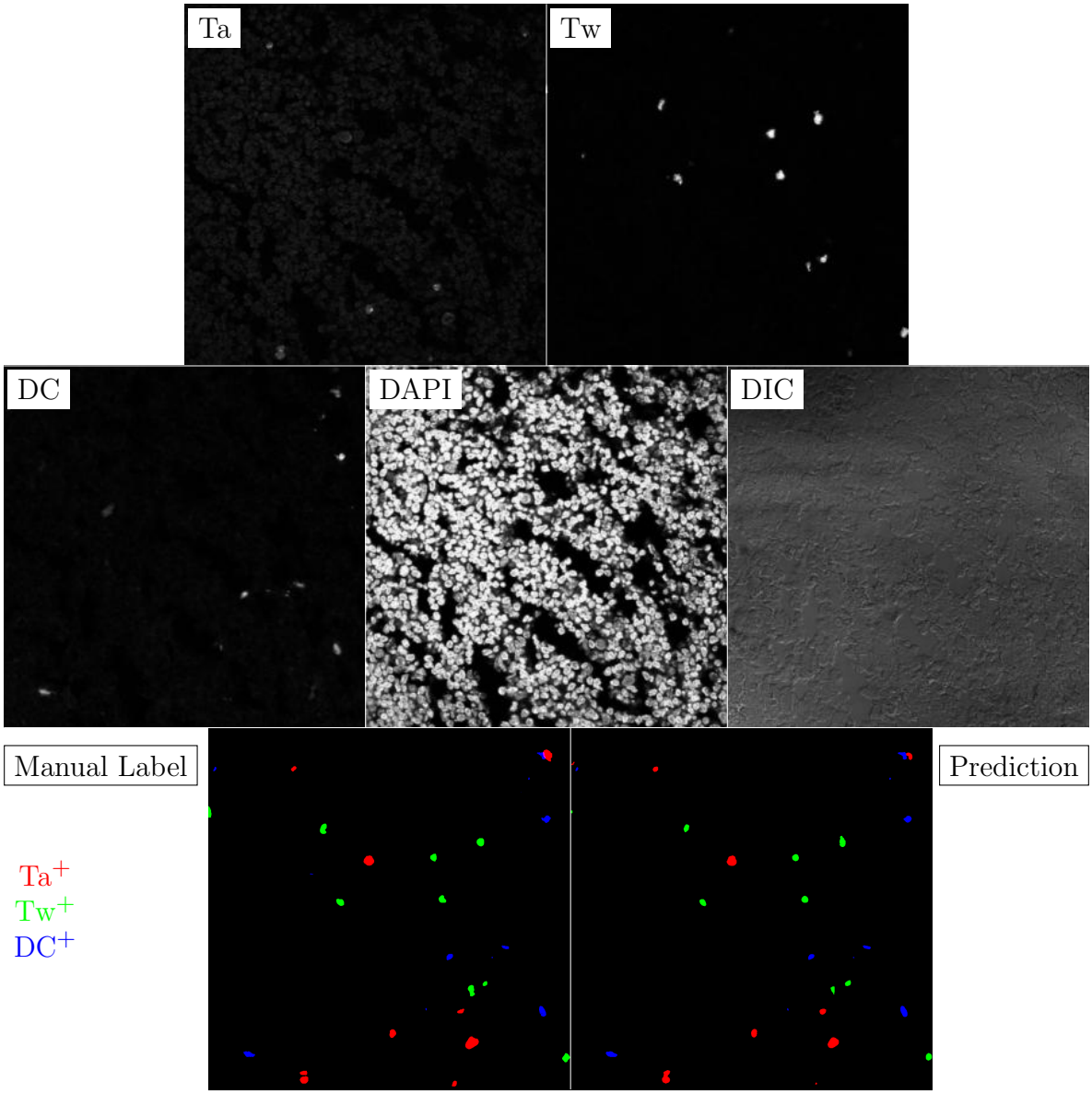

Figure F.32: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

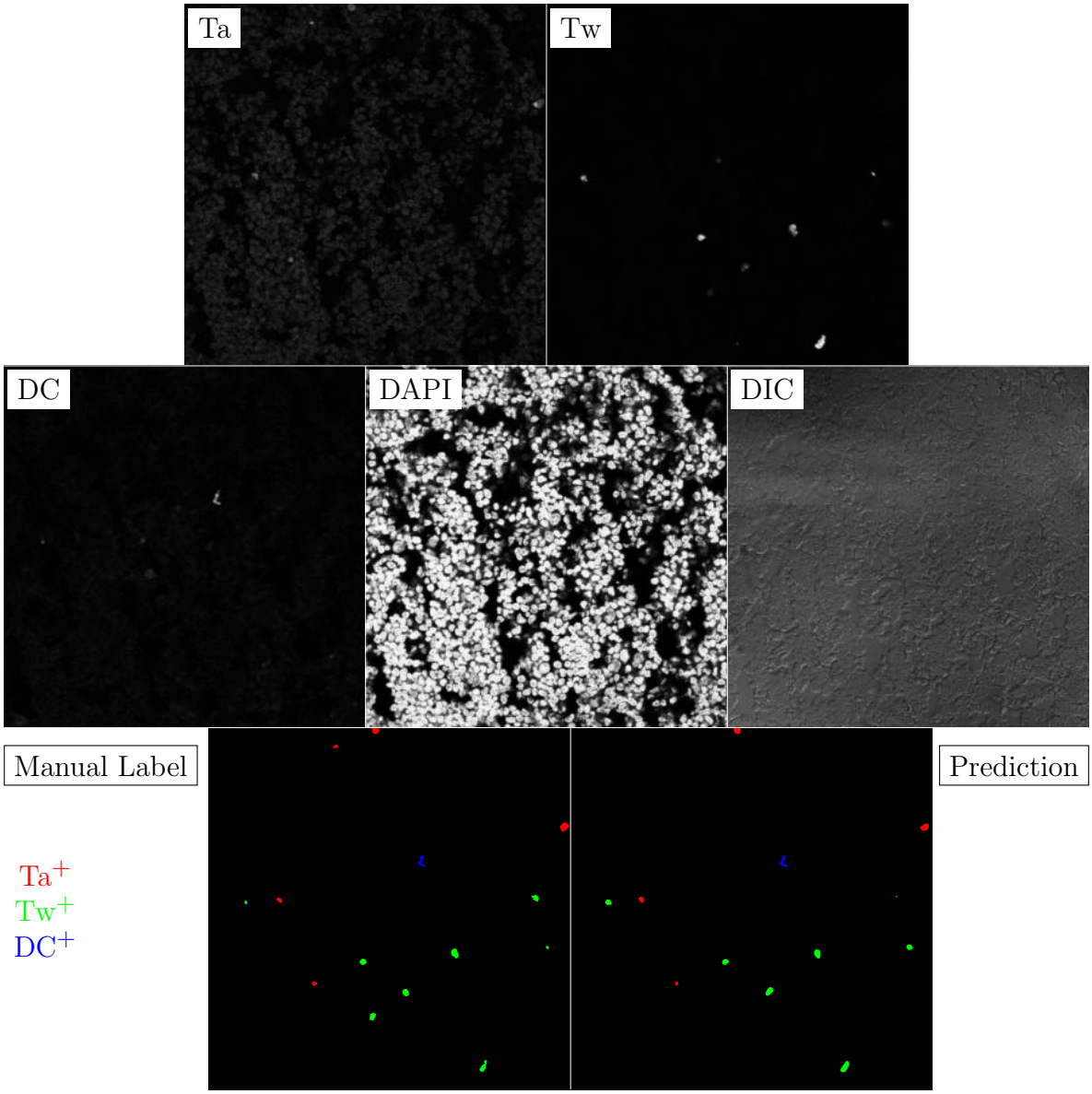

Figure F.33: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

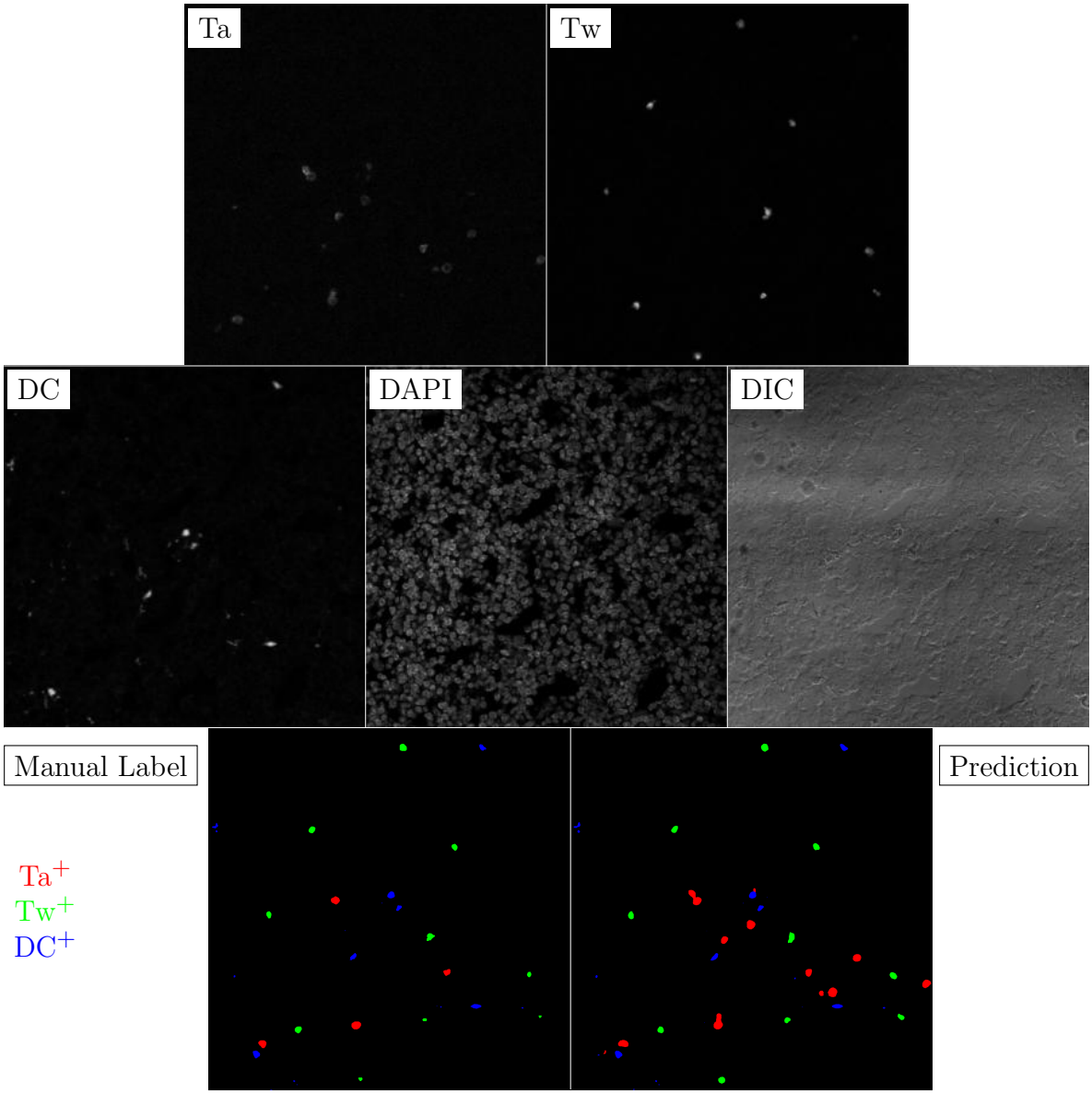

Figure F.34: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

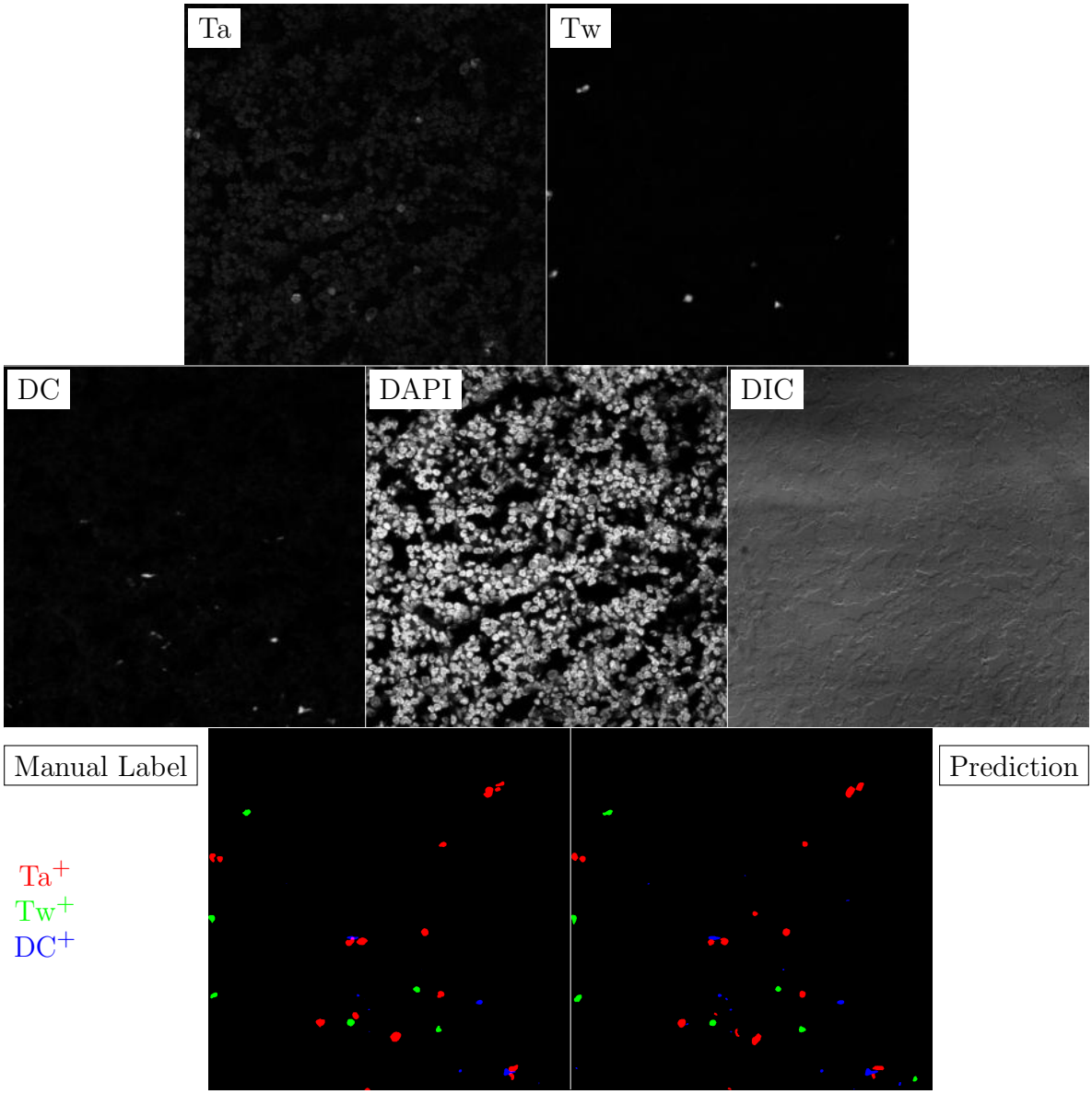

Figure F.35: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

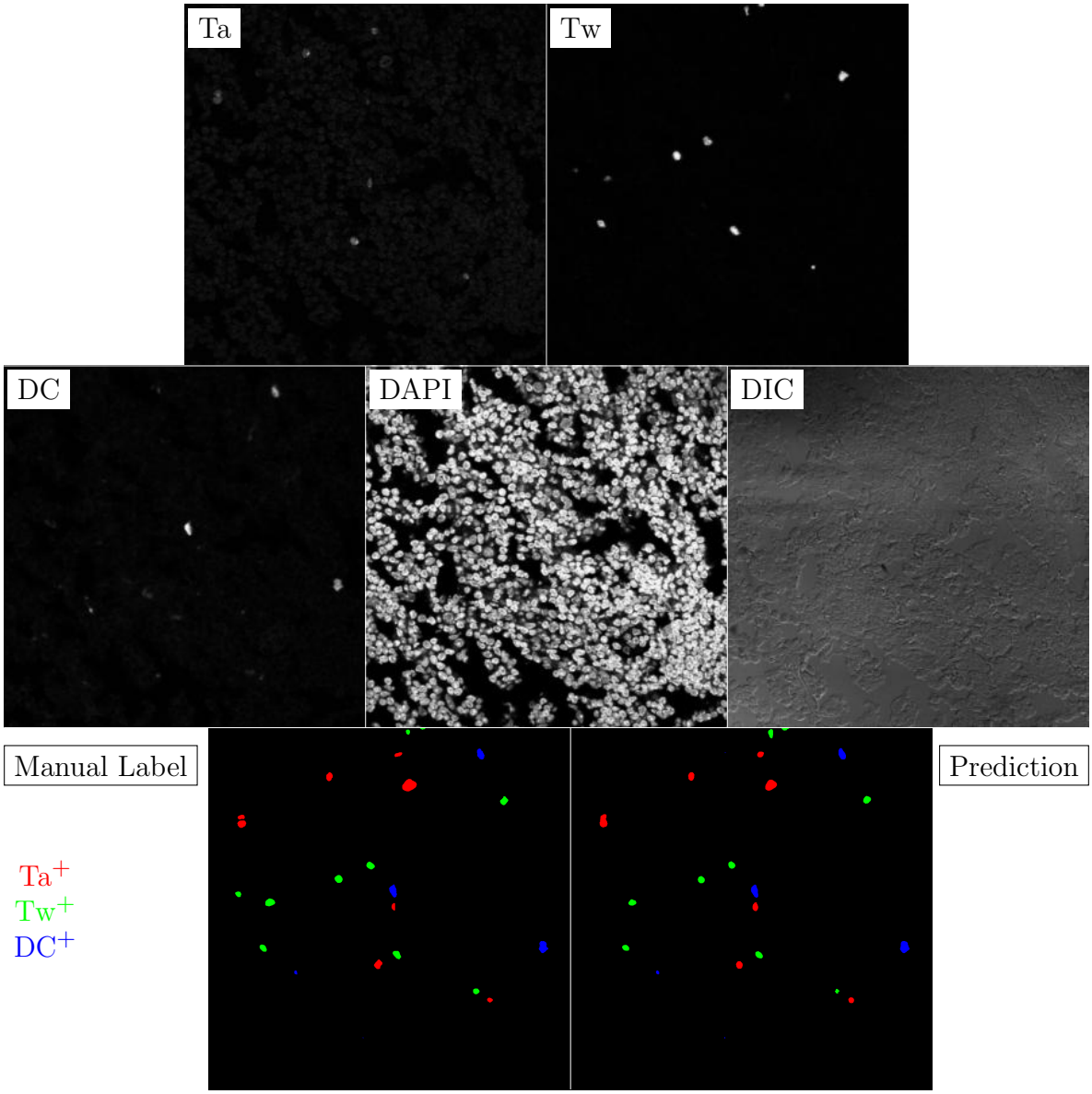

Figure F.36: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

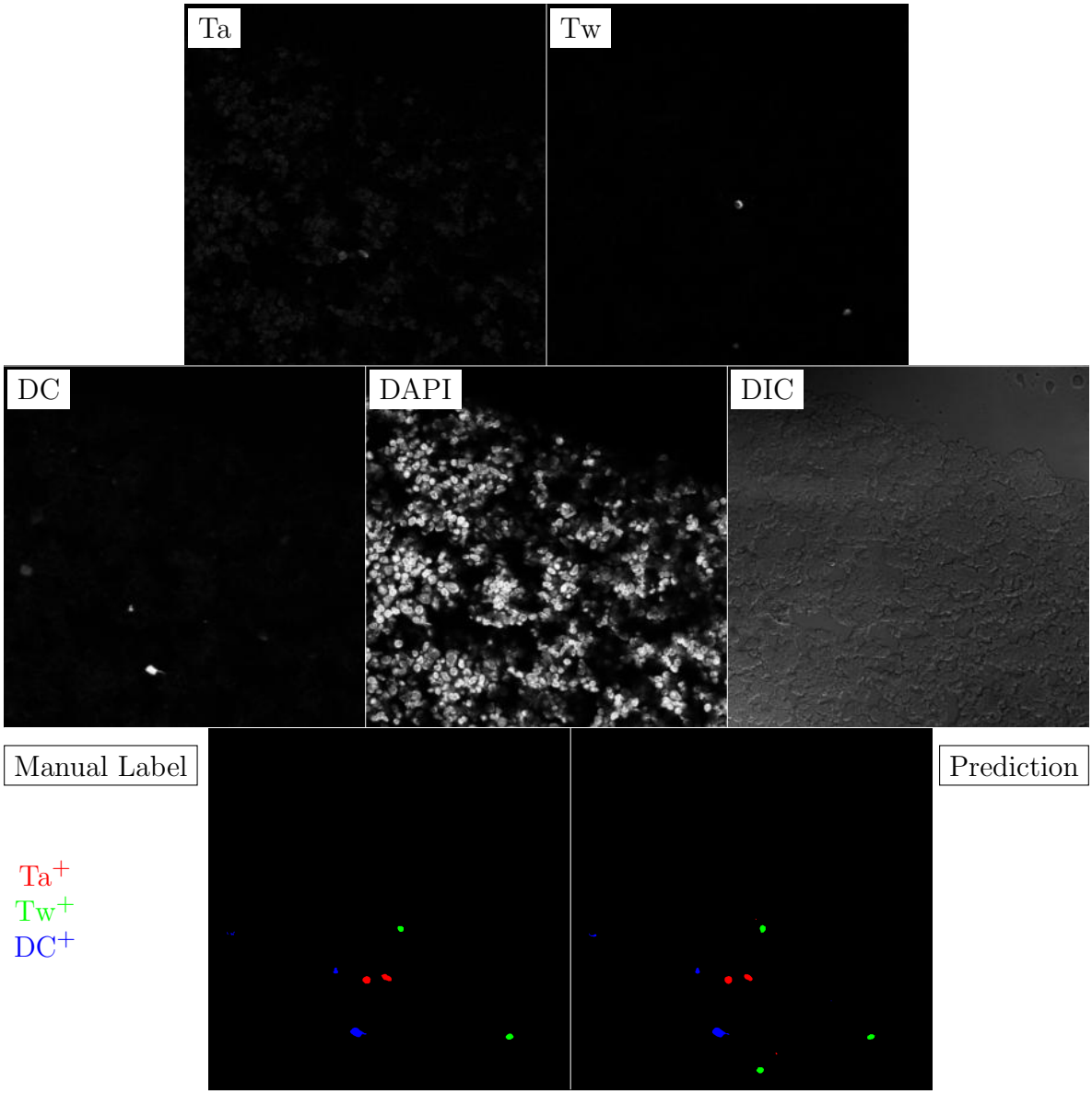

Figure F.37: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

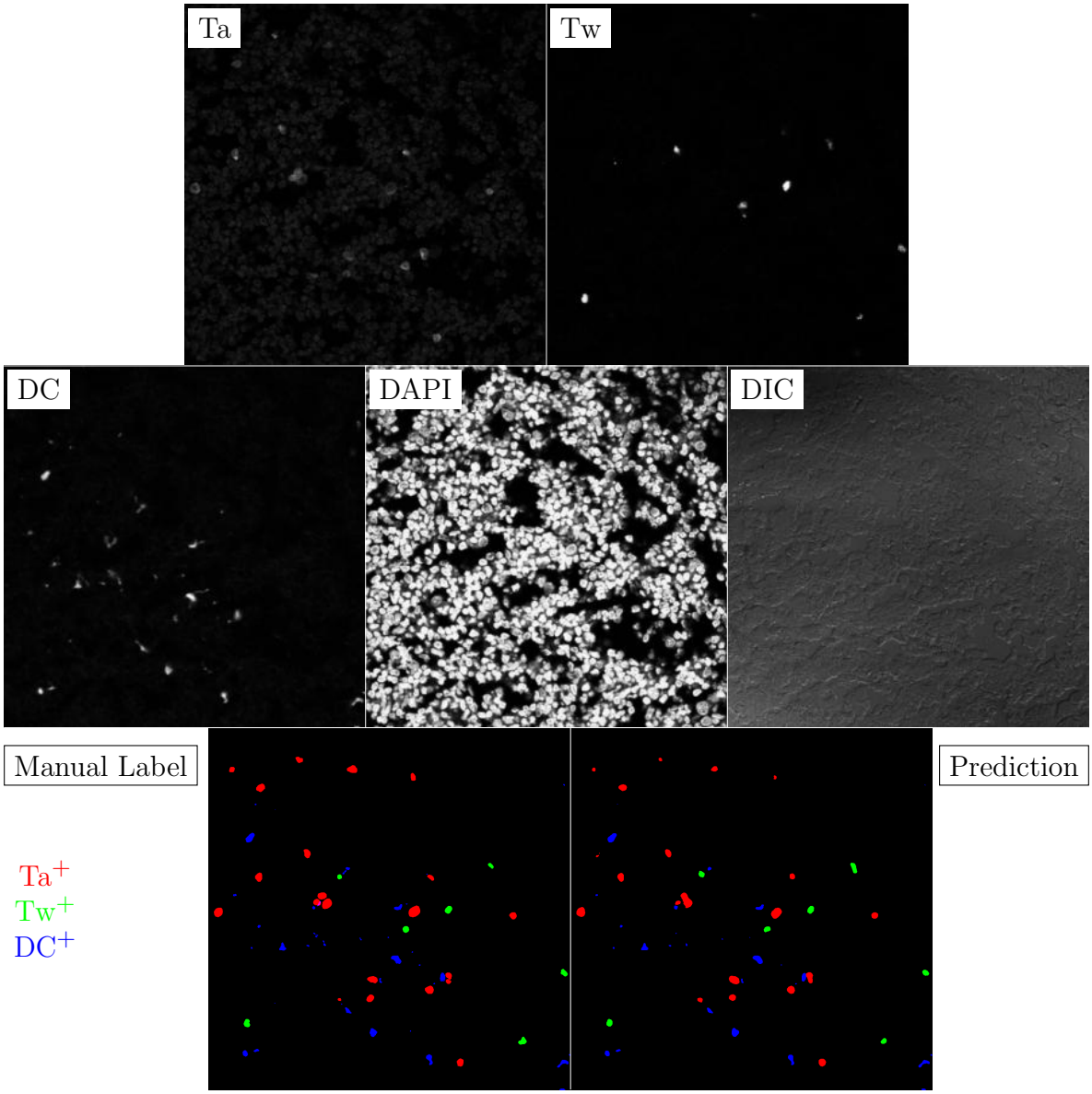

Figure F.38: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

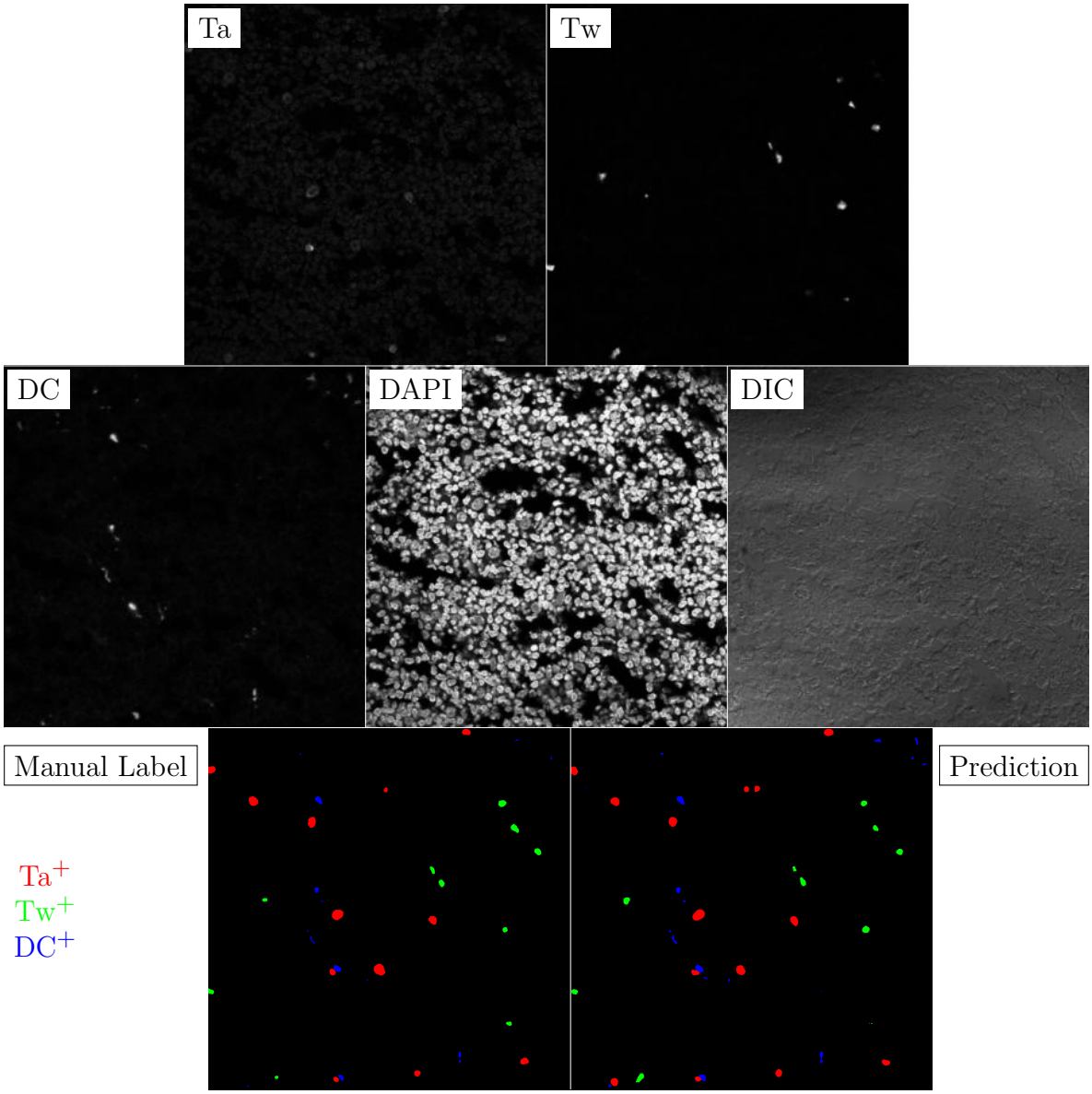

Figure F.39: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

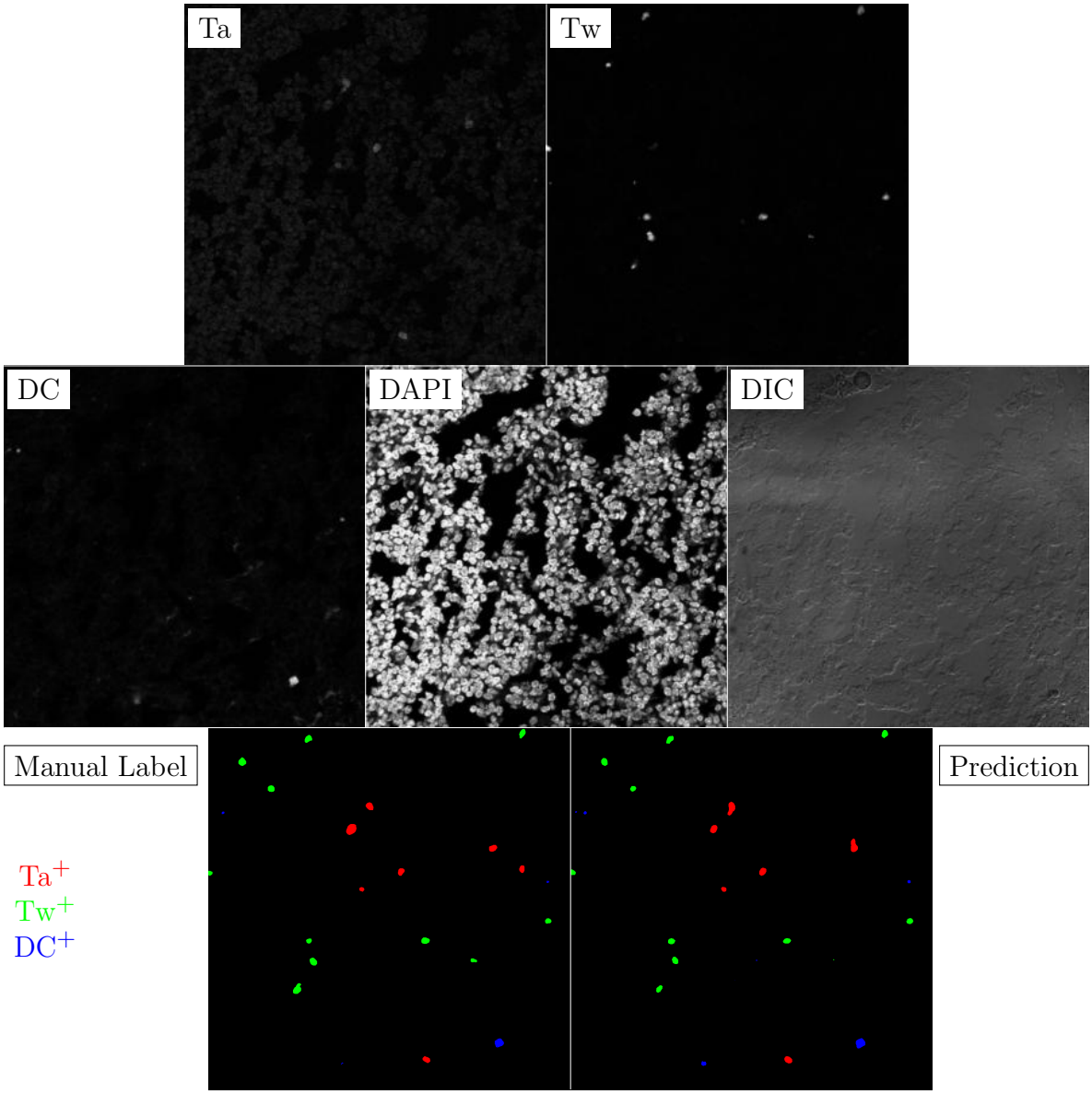

Figure F.40: **Murine FF**. True image channel matrix size is 1024×1024. Depicted image channel matrix size is 300×300.

APPENDIX G

HUMAN CROSS VALIDATION EXAMPLES

This section presents the full dataset from the cross validation run of the human fresh frozen dataset as discussed in Chapter 5.

Label ROIs are embedded in document with DEFLATE png compression at full resolution. All images for the biopsy channel were downsampled with bilinear interpolation such that the maximum x,y dimension of the image was equal to 300 pixels. This was done to keep the memory footprint of this pdf document at a reasonable level while also avoiding the compression artifacts of high level jpeg compression. Images were transformed from native bit depth of 12 by rescaling the image using the standard score (z-score). Z-score images were transformed to an 8-bit range by trucating the distribution of z-scores at 3 standard deviations from the mean and rescaling to the range [0,255]. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$. Bit depth is 8 in embedded images.

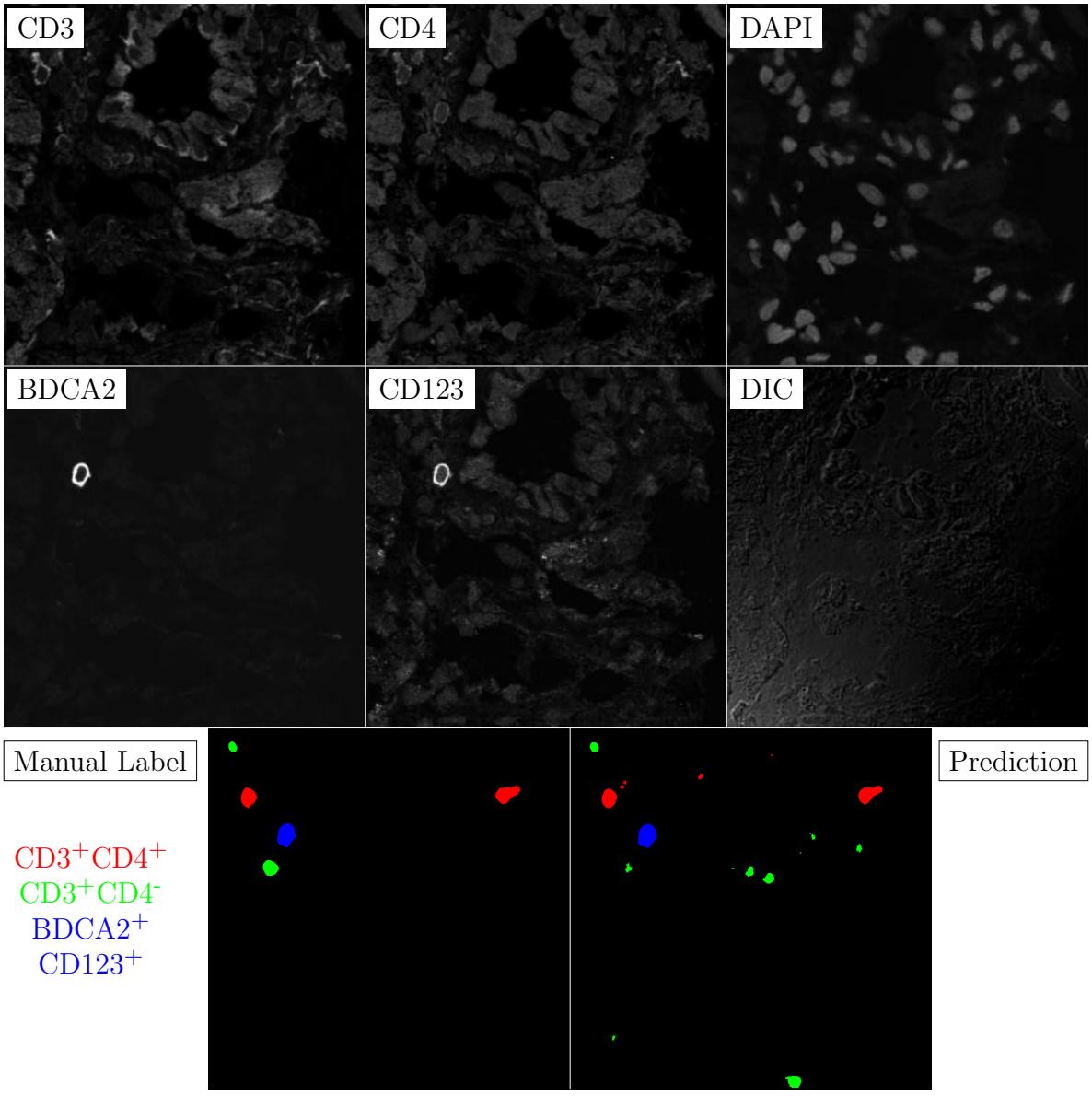

Figure G.1: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

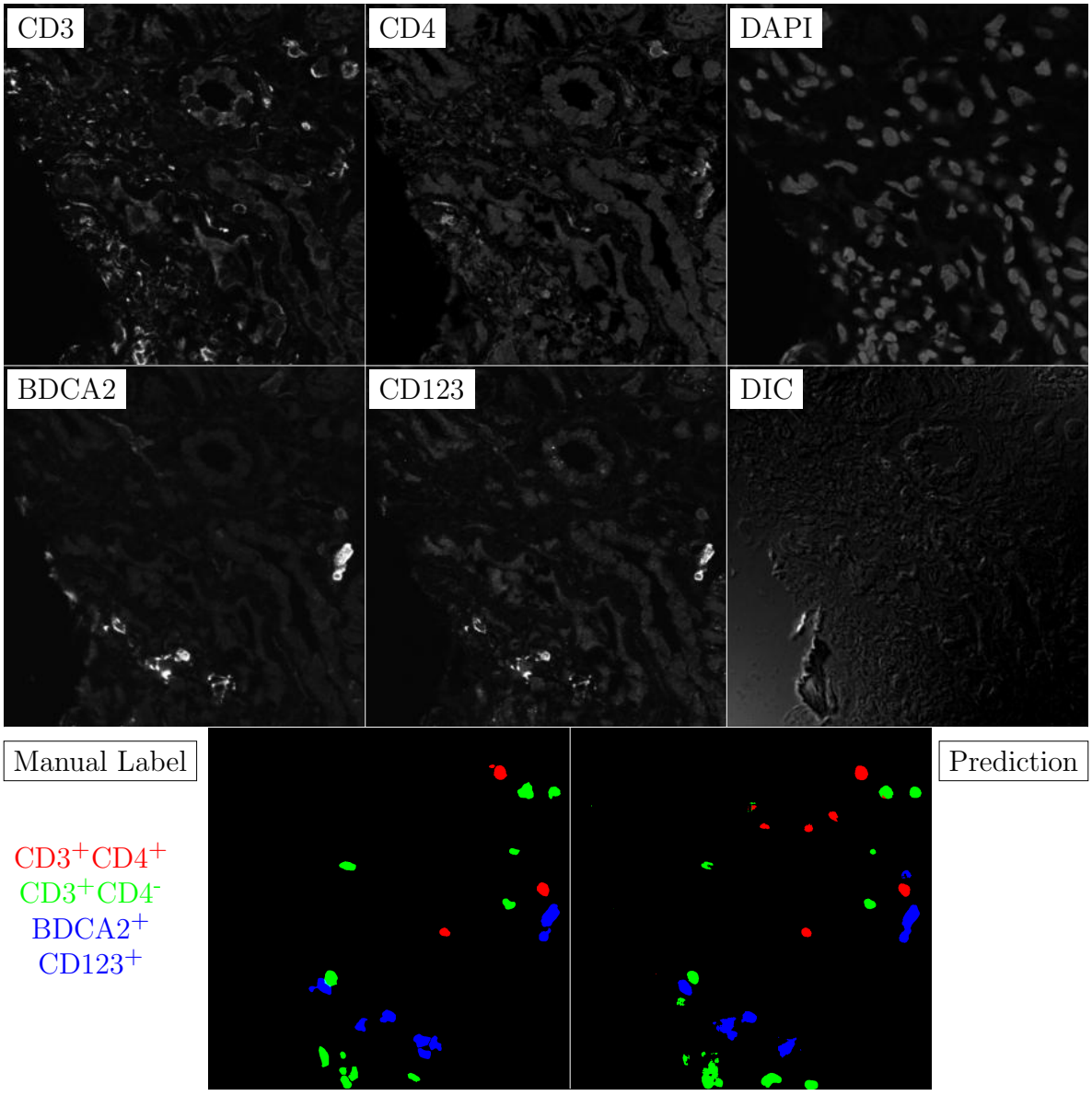

Figure G.2: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

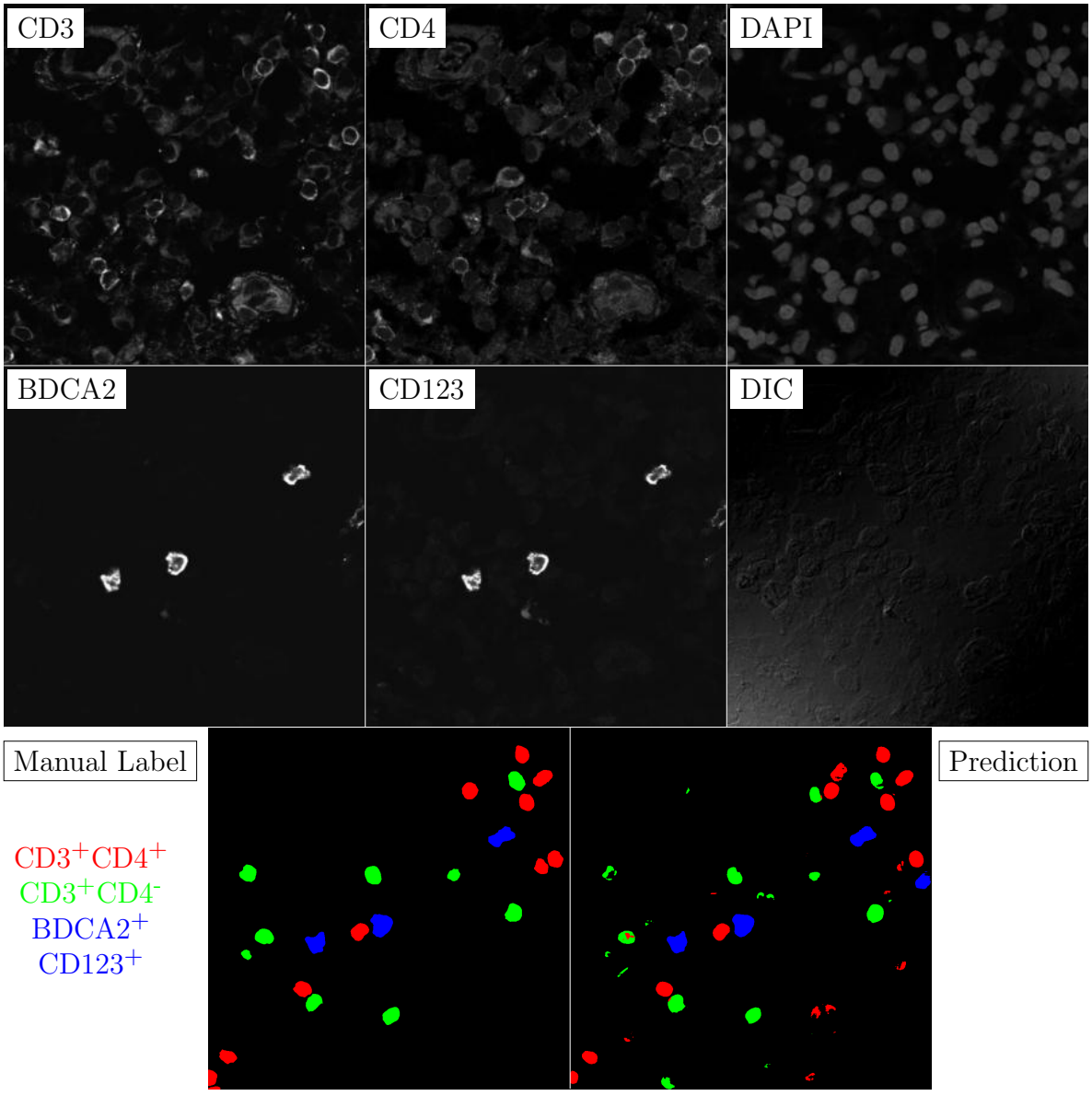

Figure G.3: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

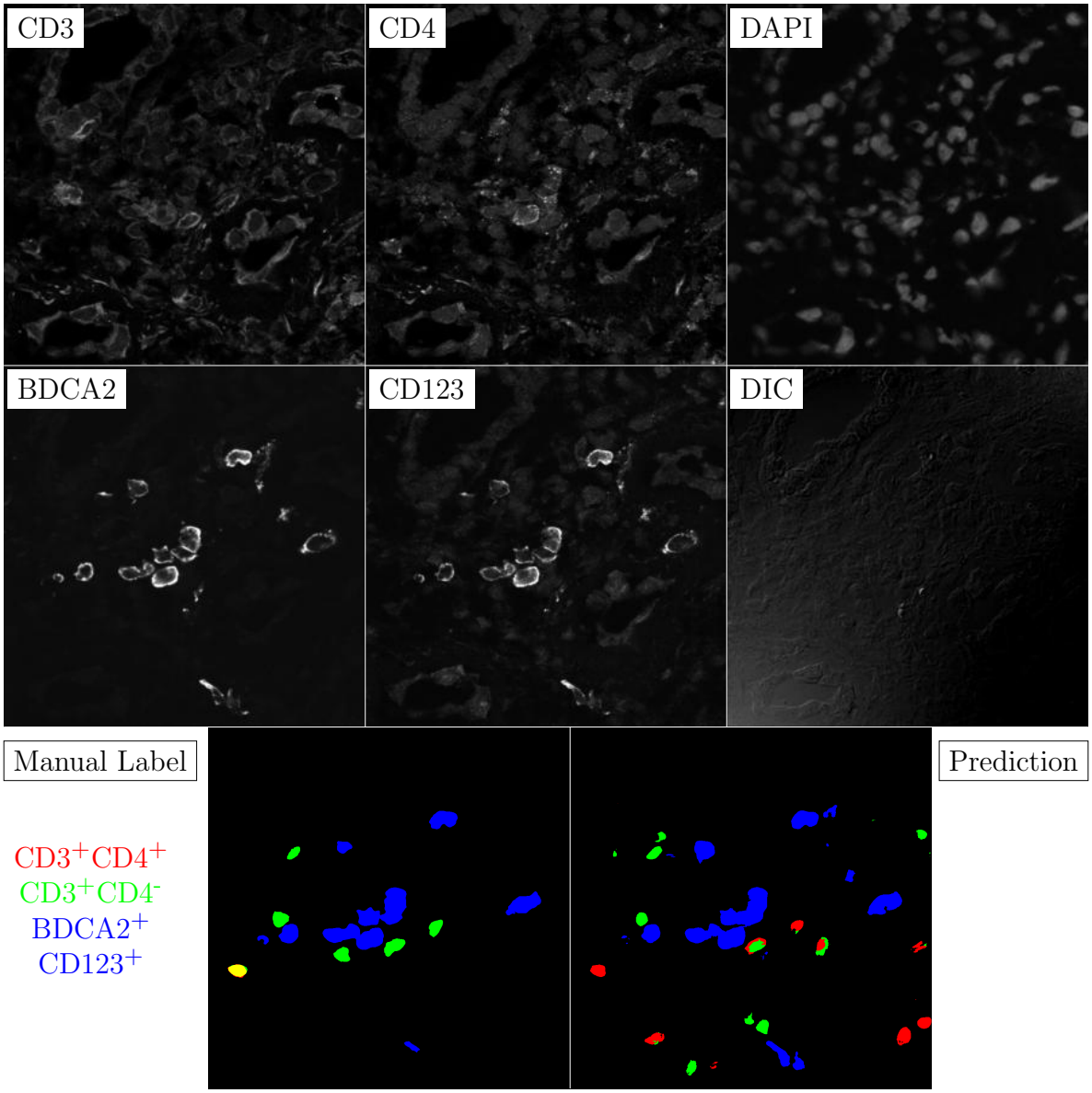

Figure G.4: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

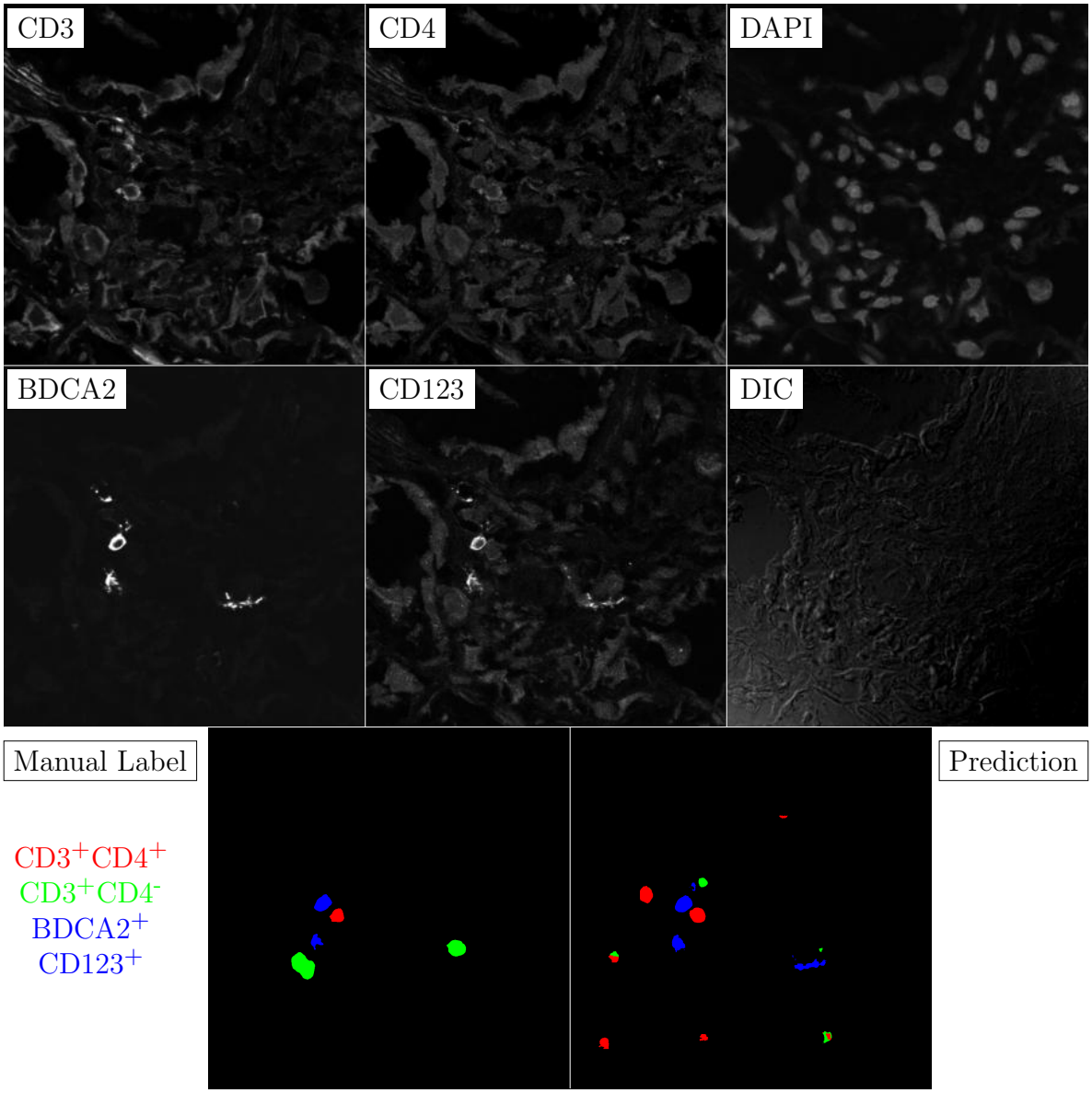

Figure G.5: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

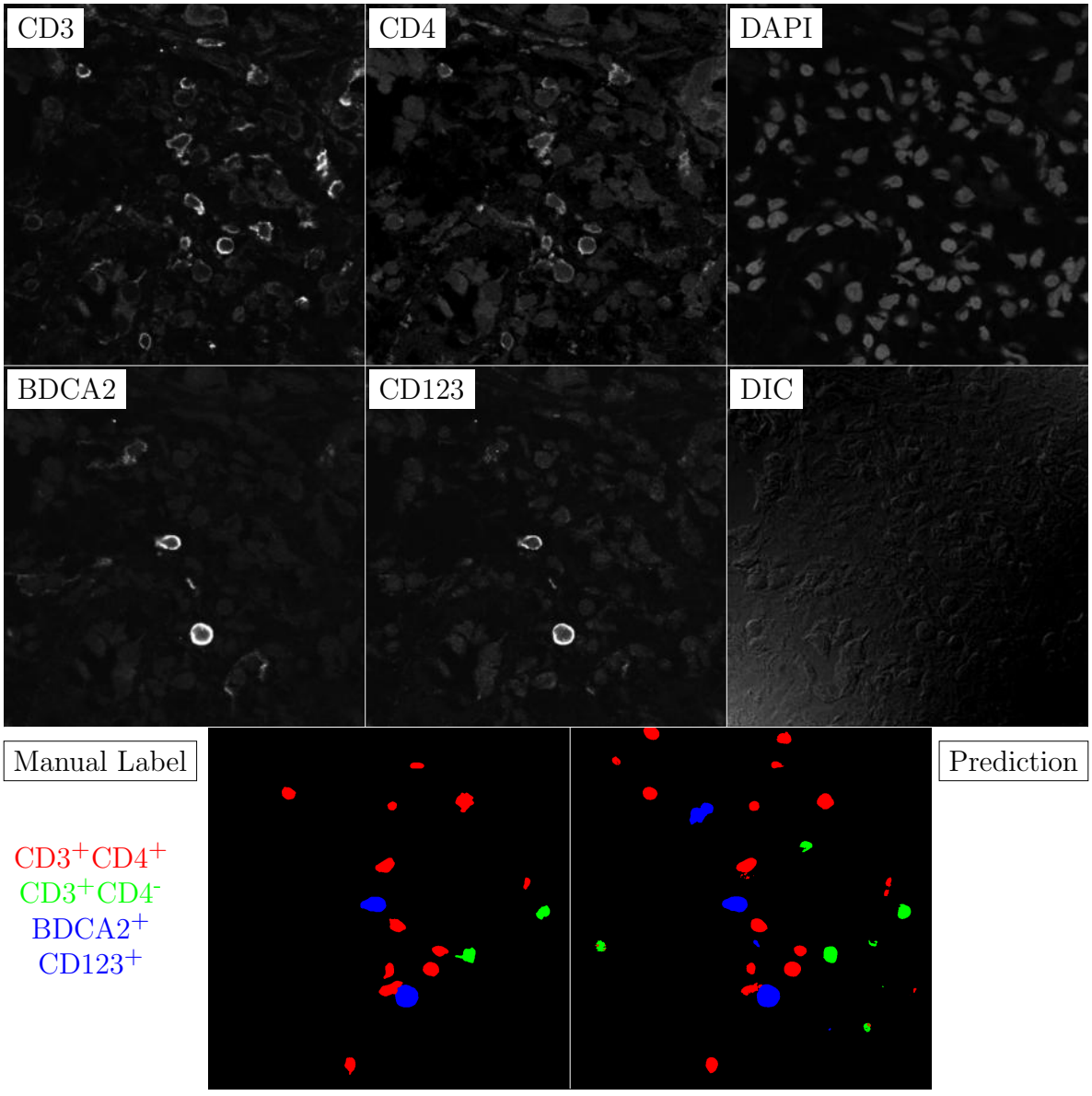

Figure G.6: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

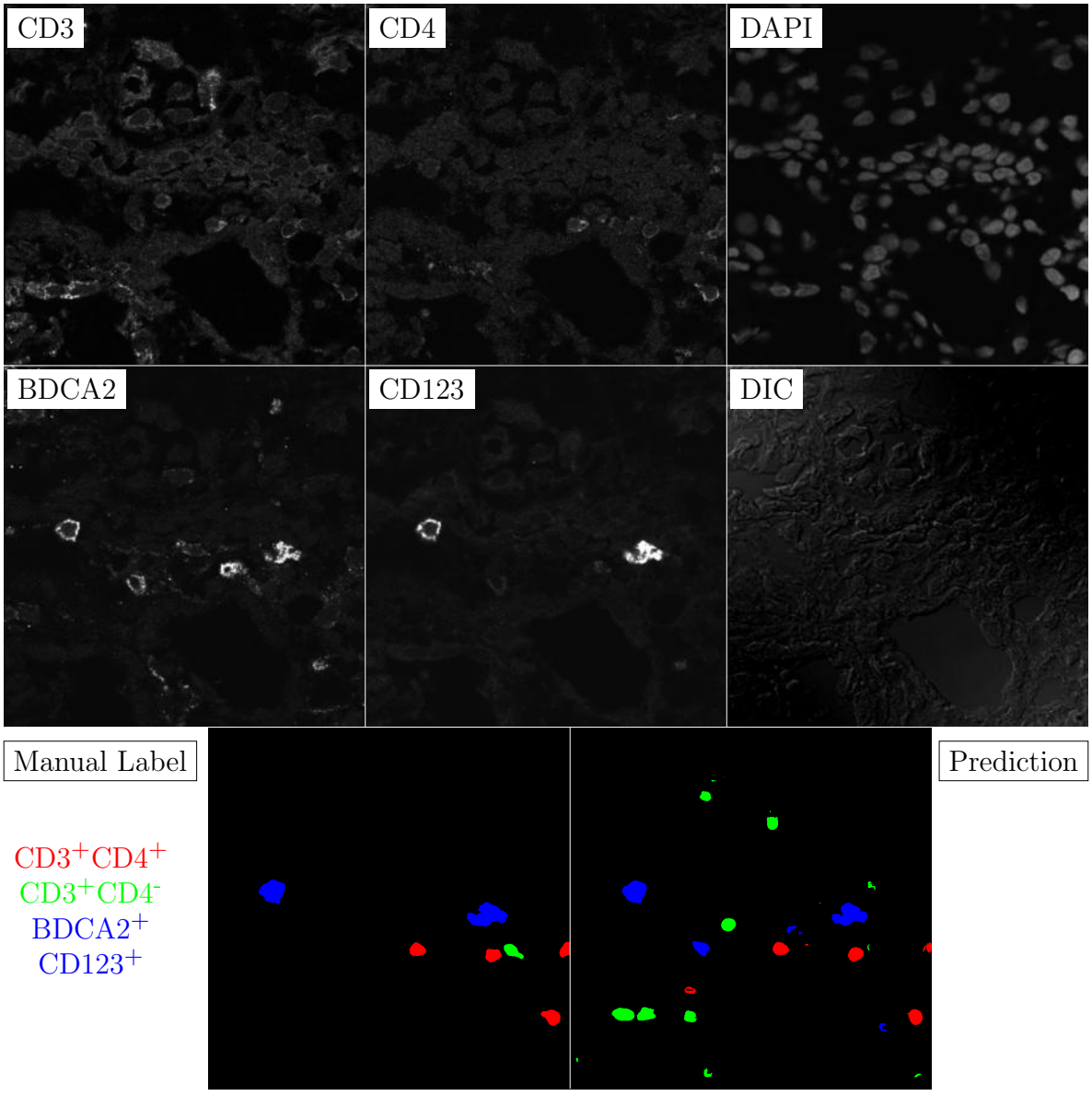

Figure G.7: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

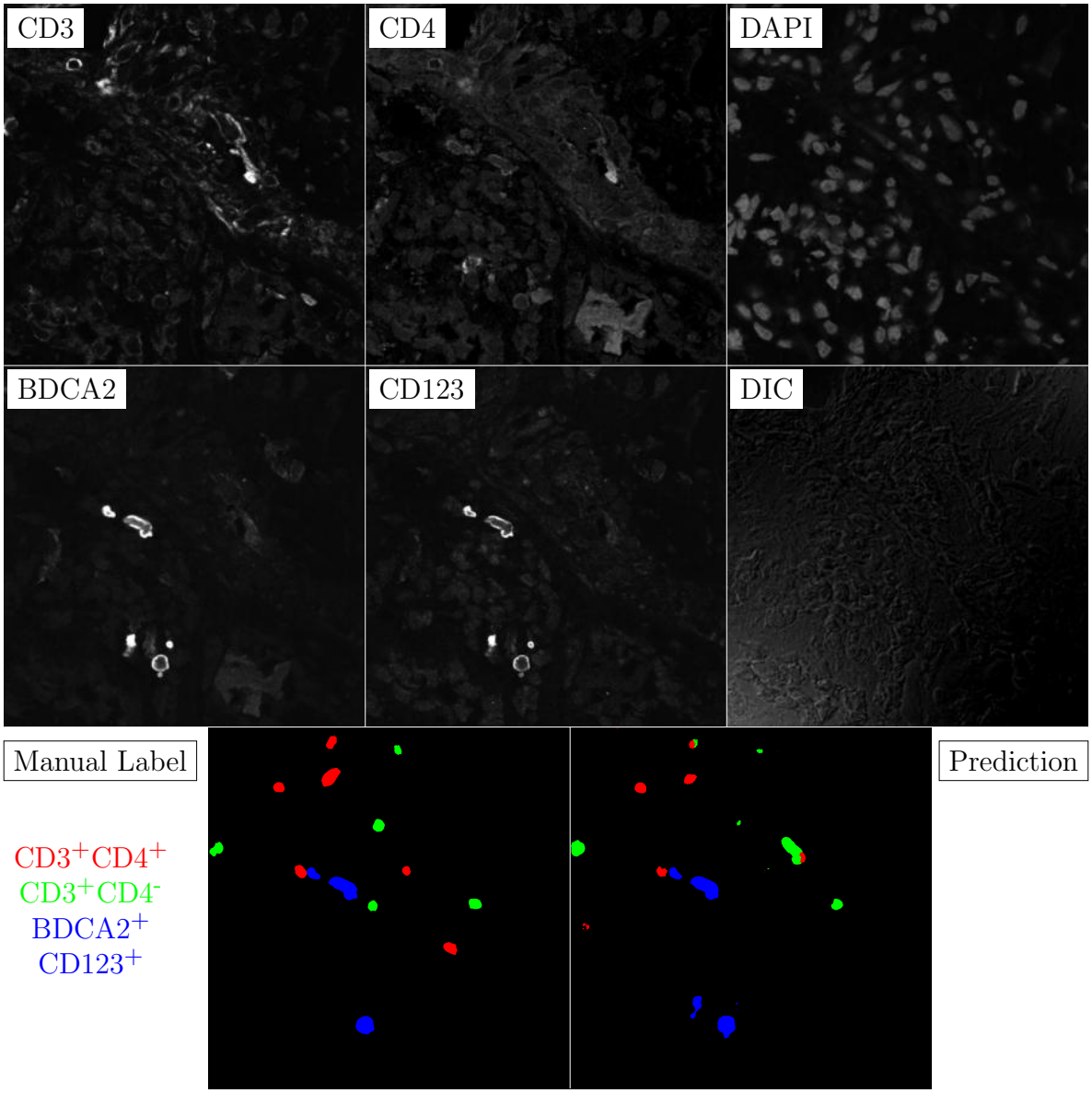

Figure G.8: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

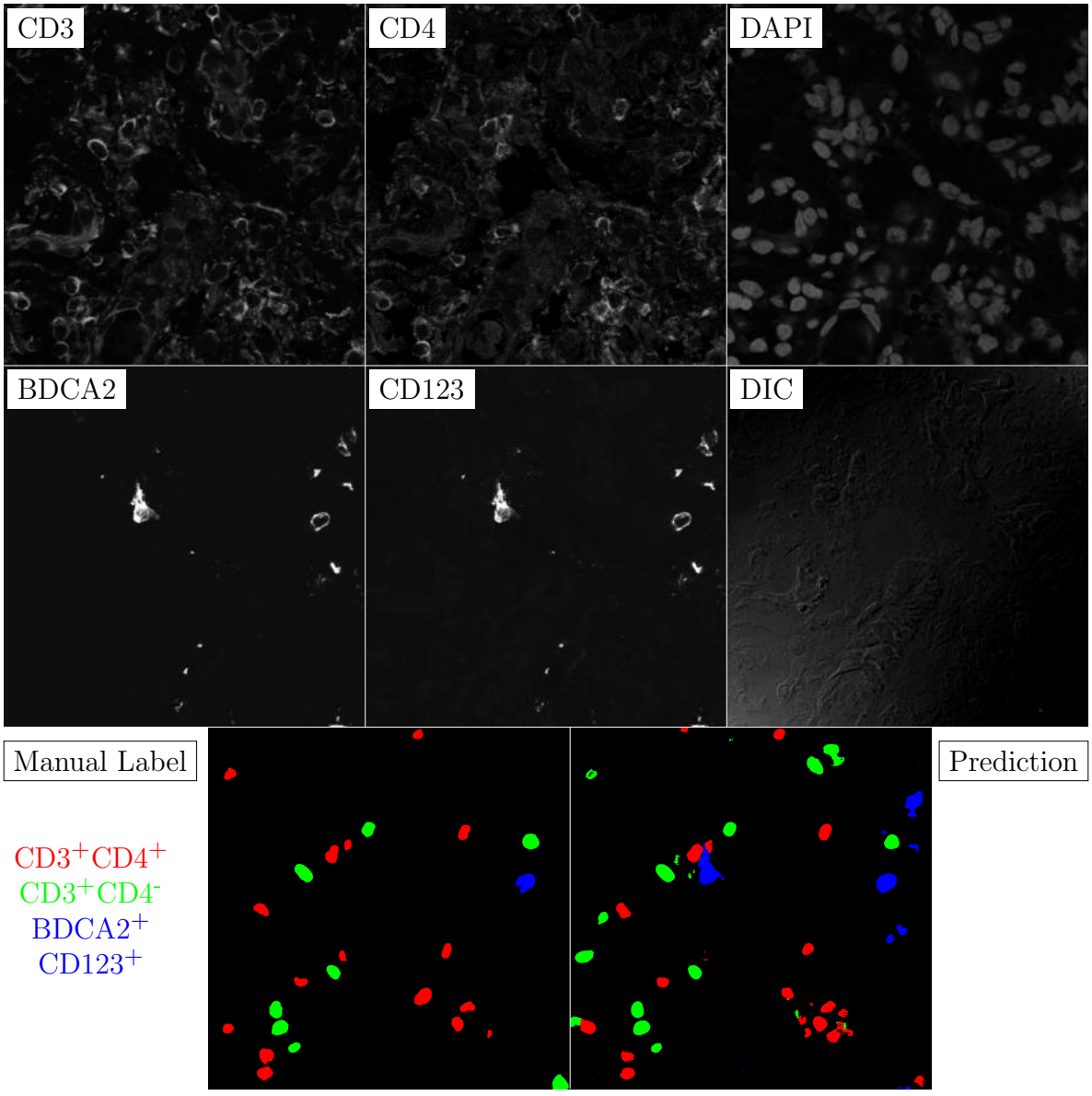

Figure G.9: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

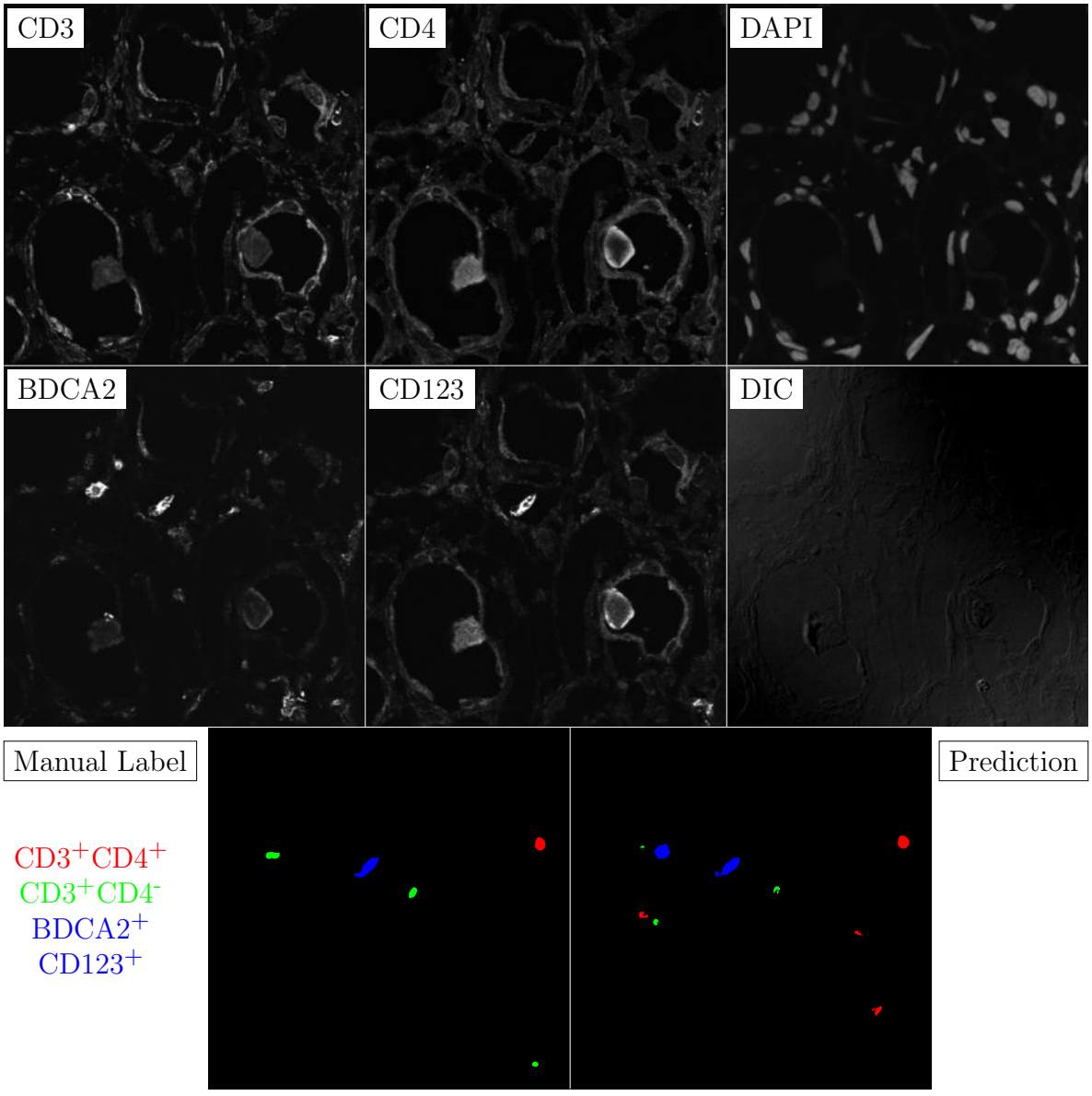

Figure G.10: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

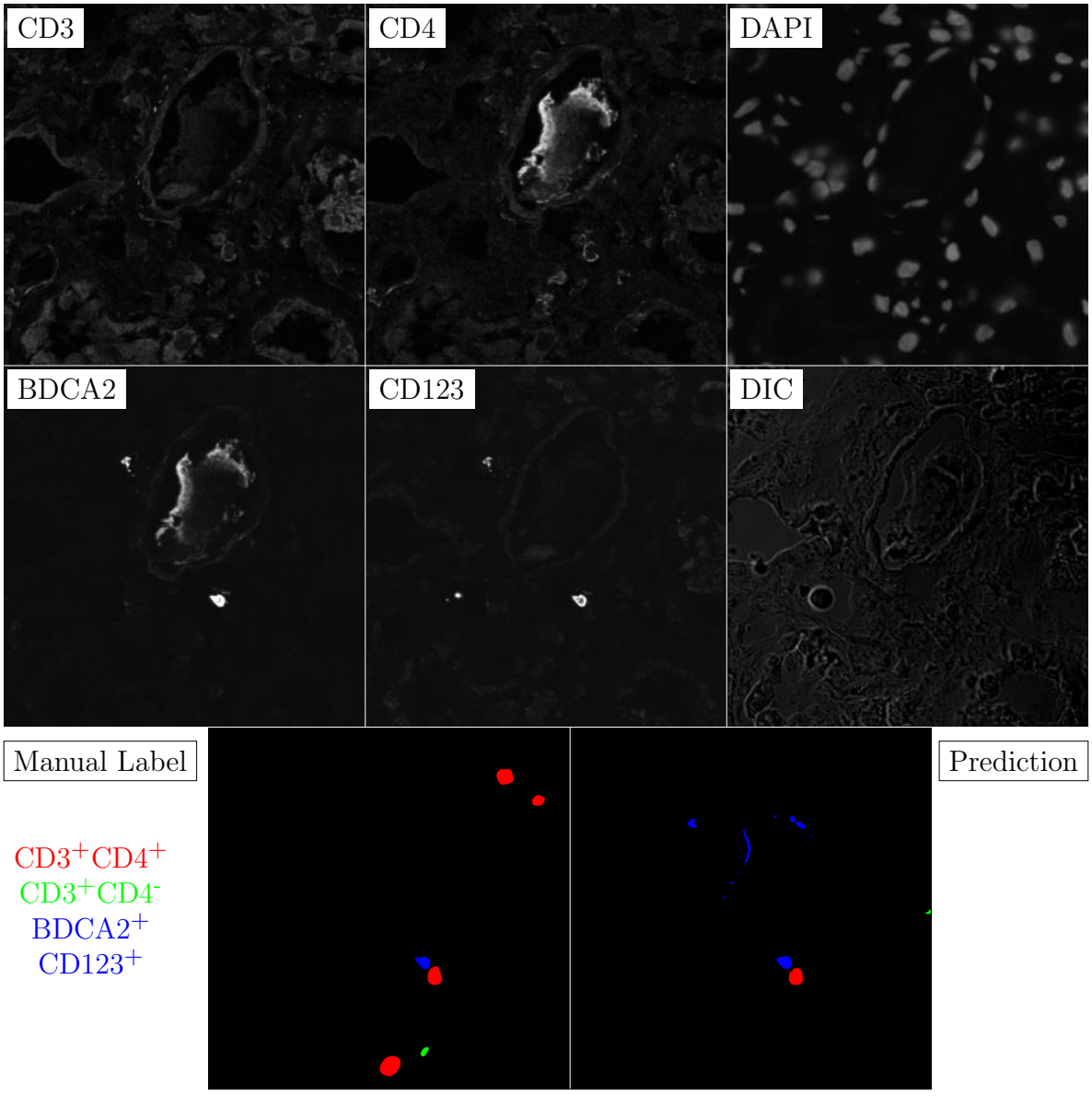

Figure G.11: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

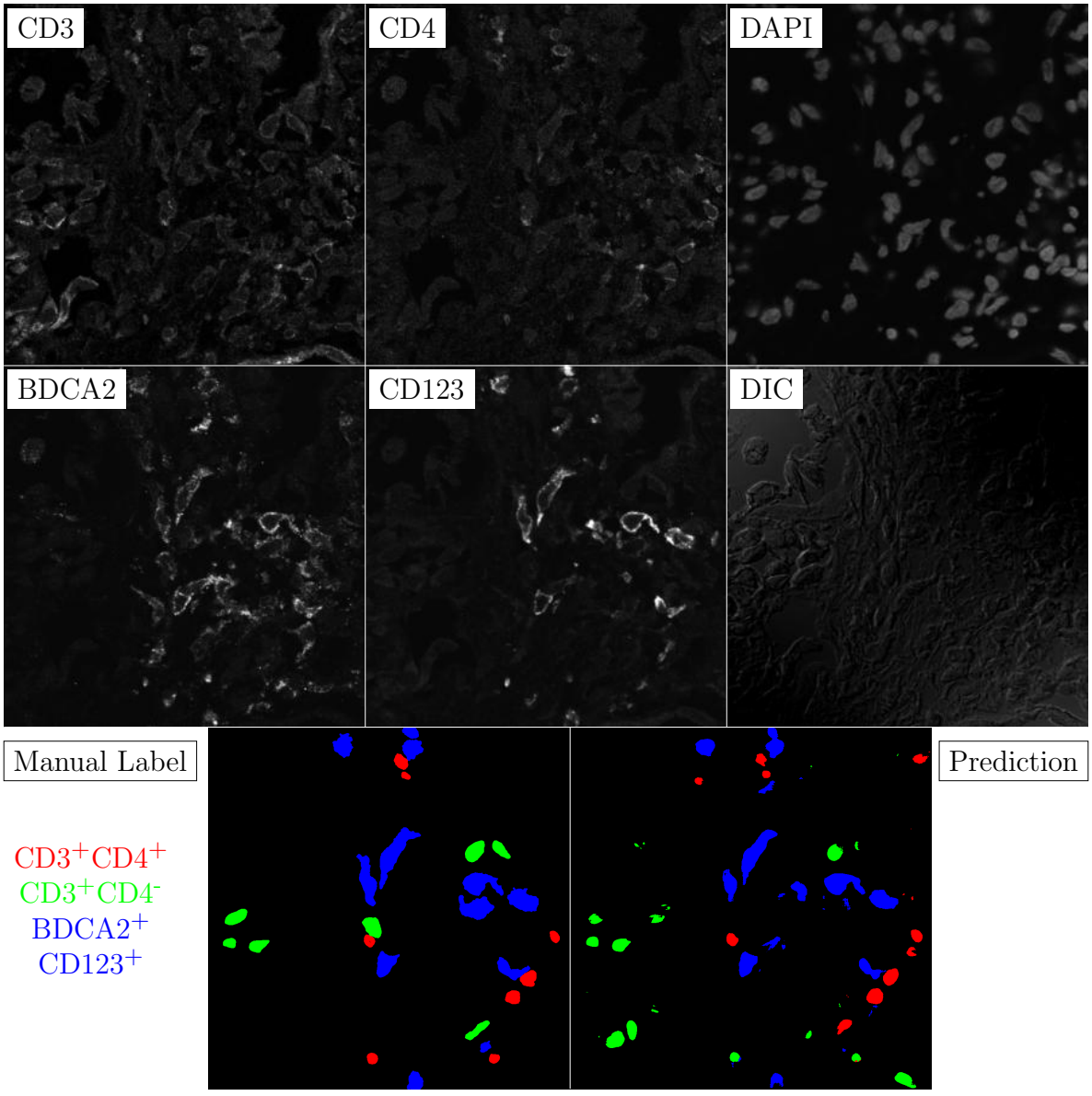

Figure G.12: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

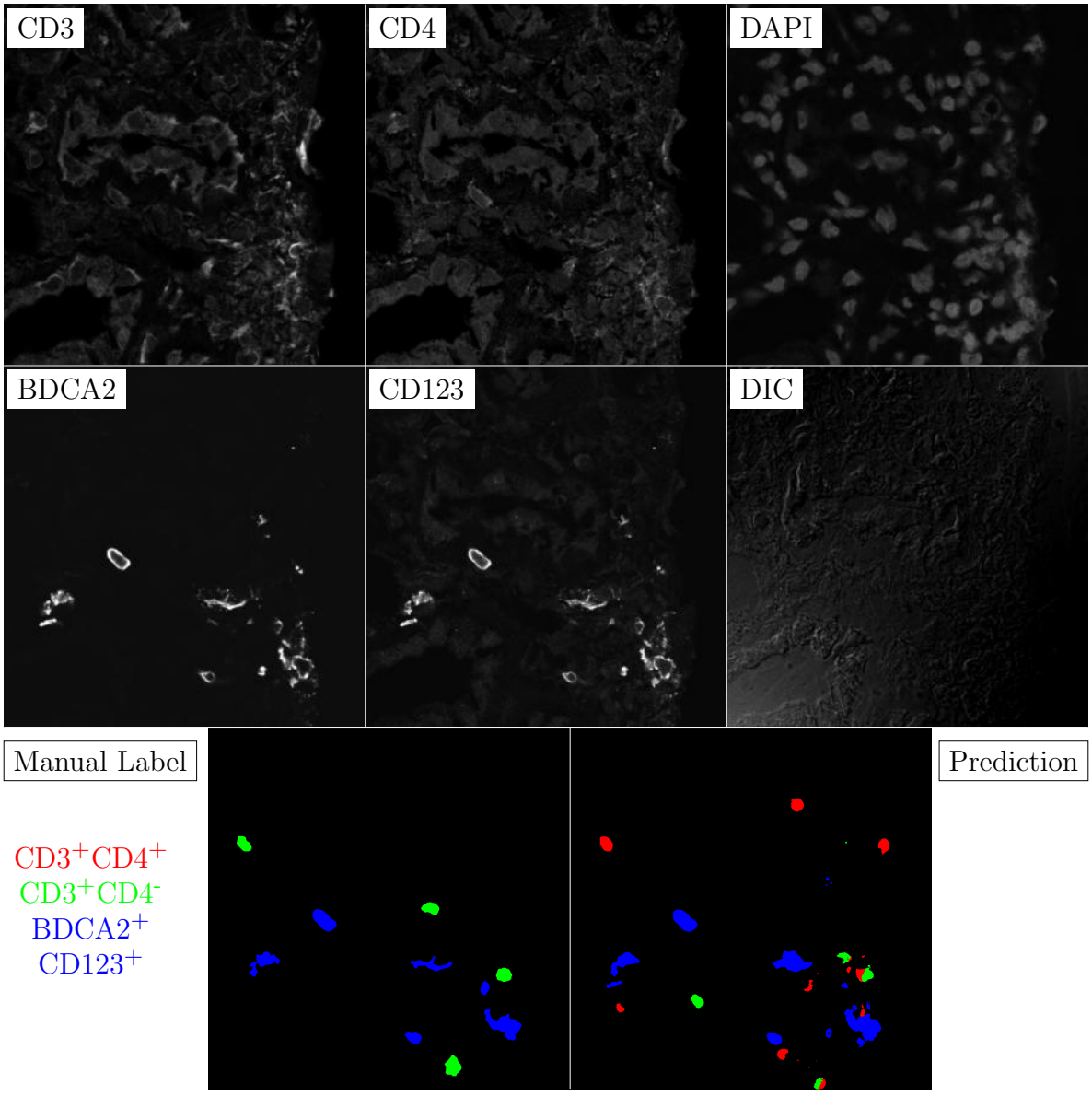

Figure G.13: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

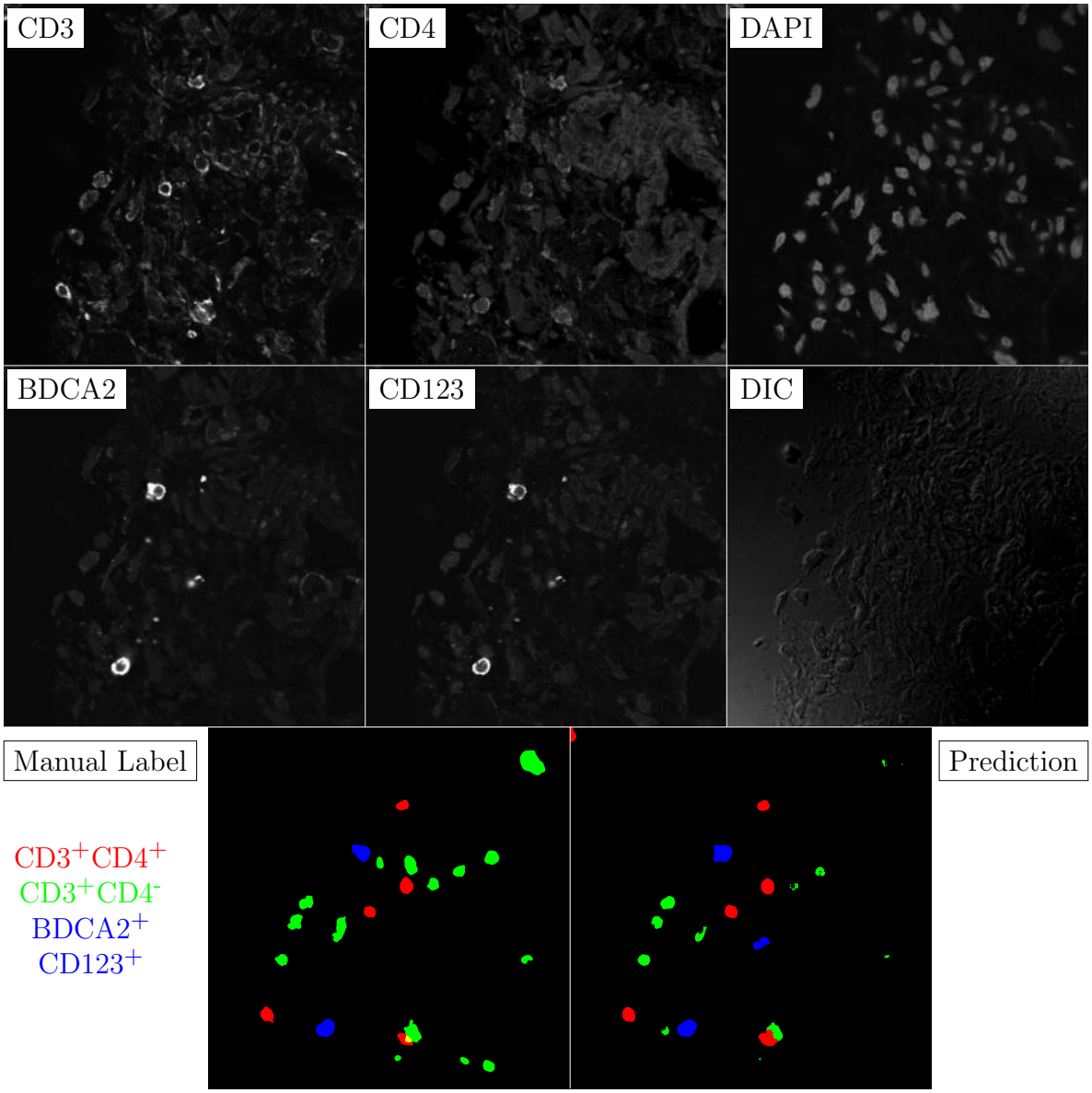

Figure G.14: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

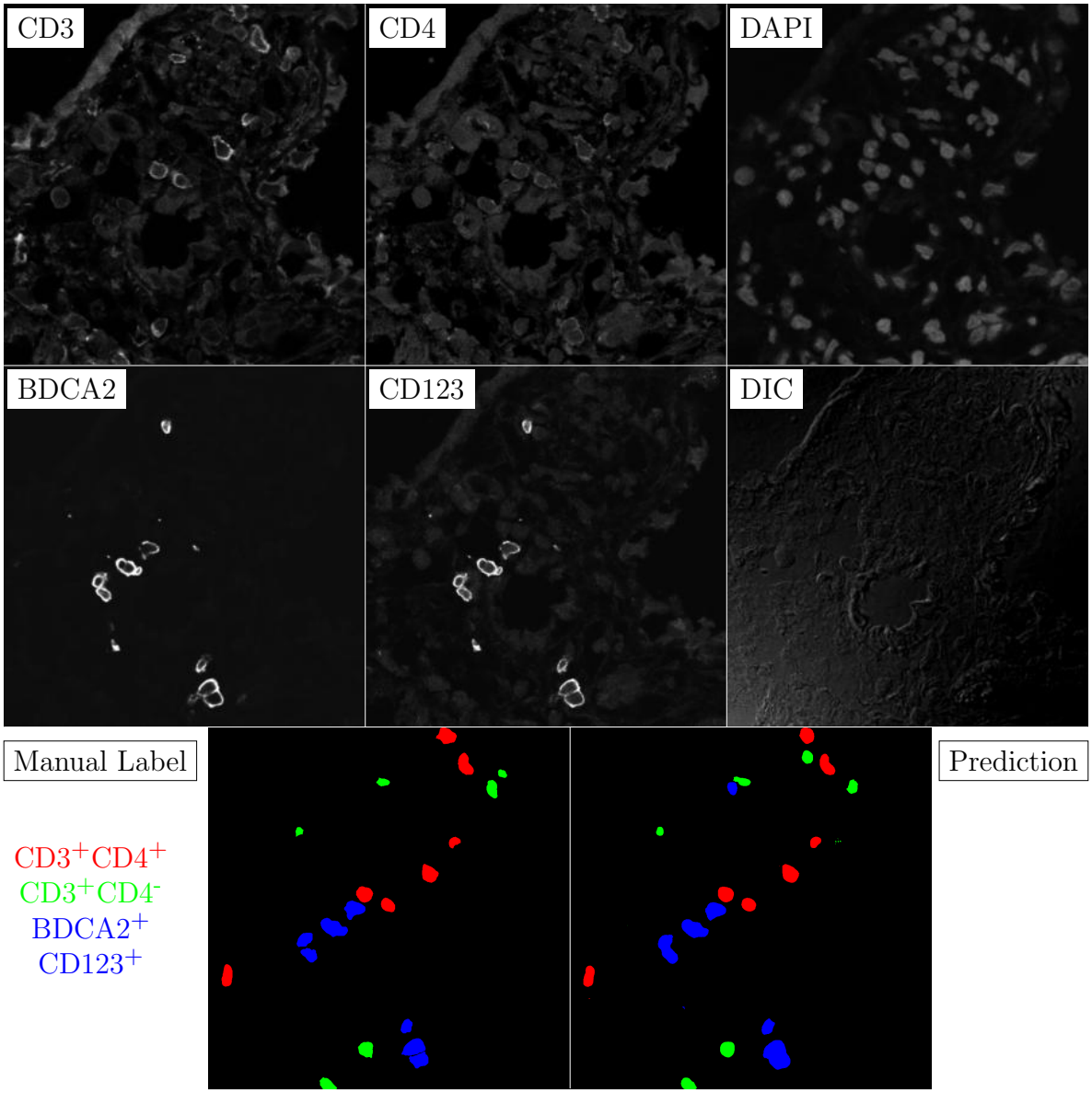

Figure G.15: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

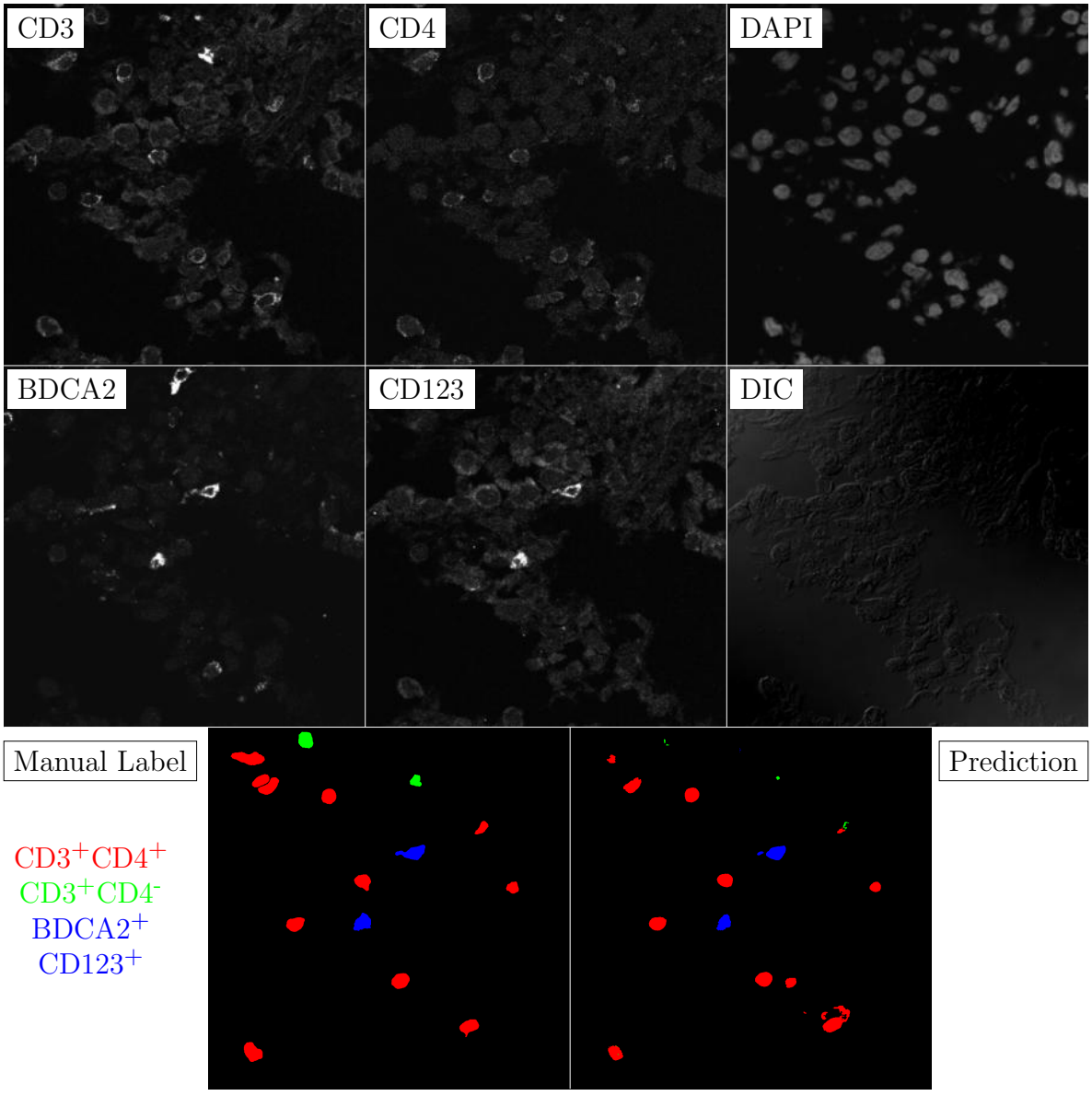

Figure G.16: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

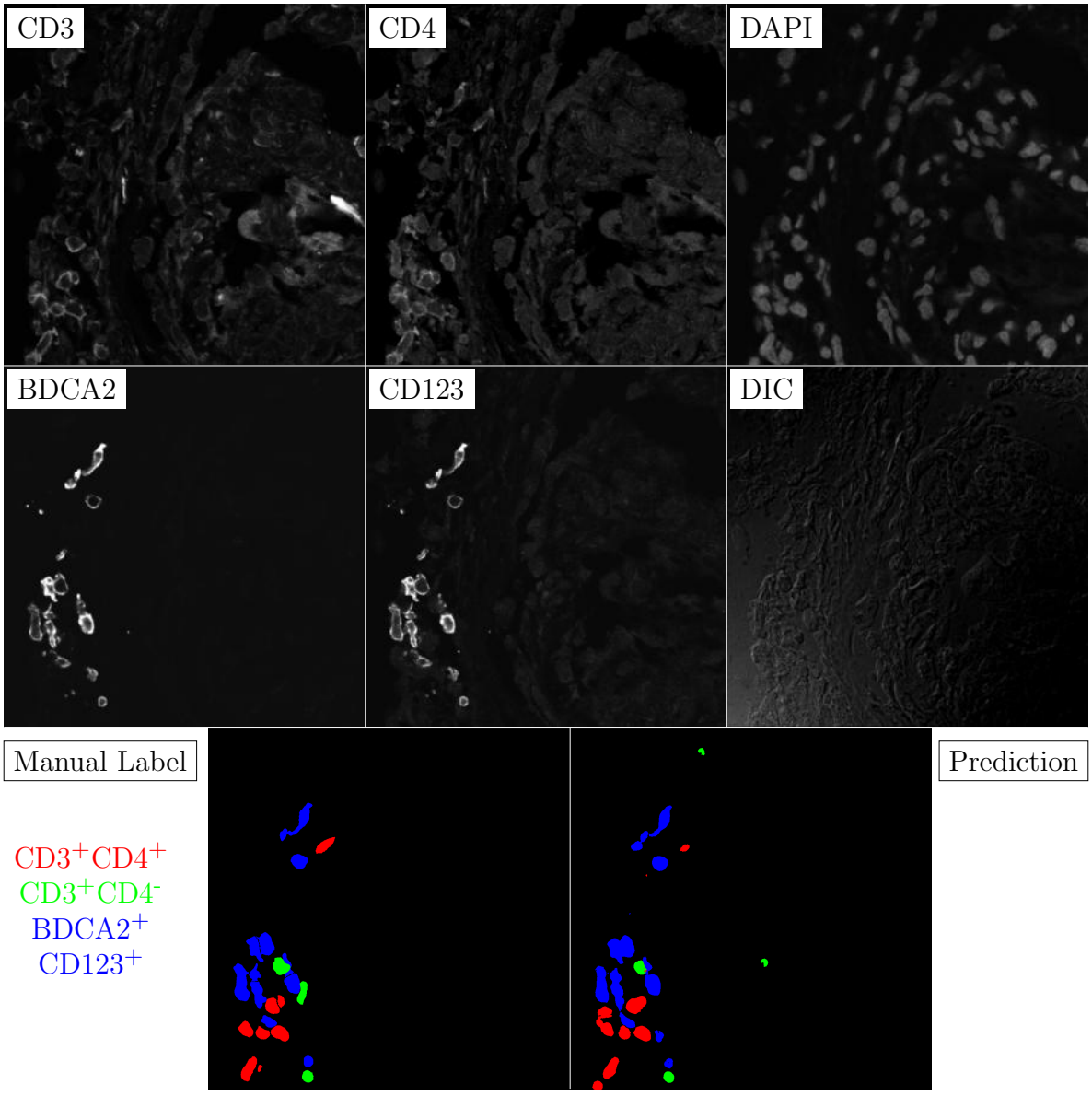

Figure G.17: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

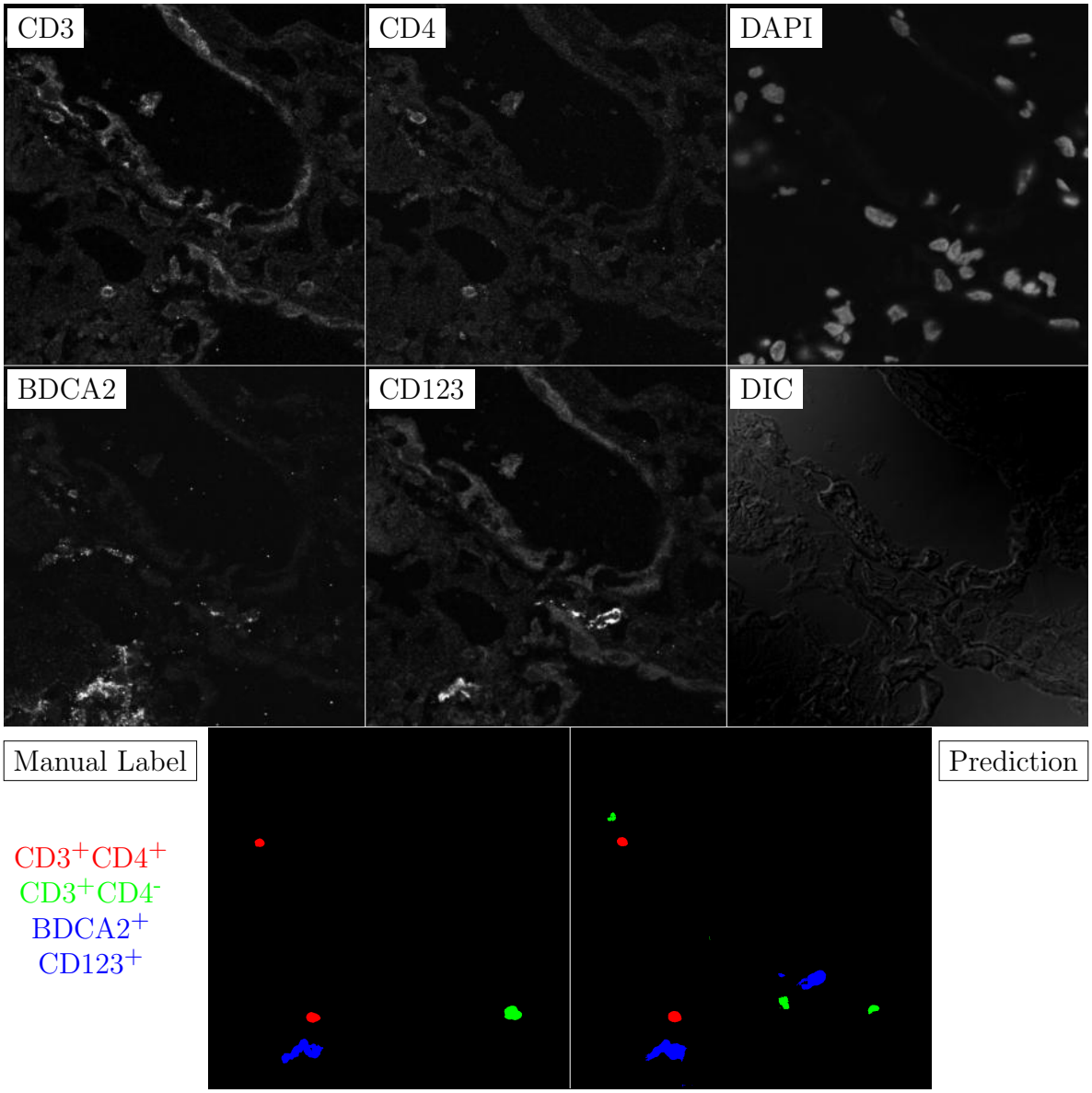

Figure G.18: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

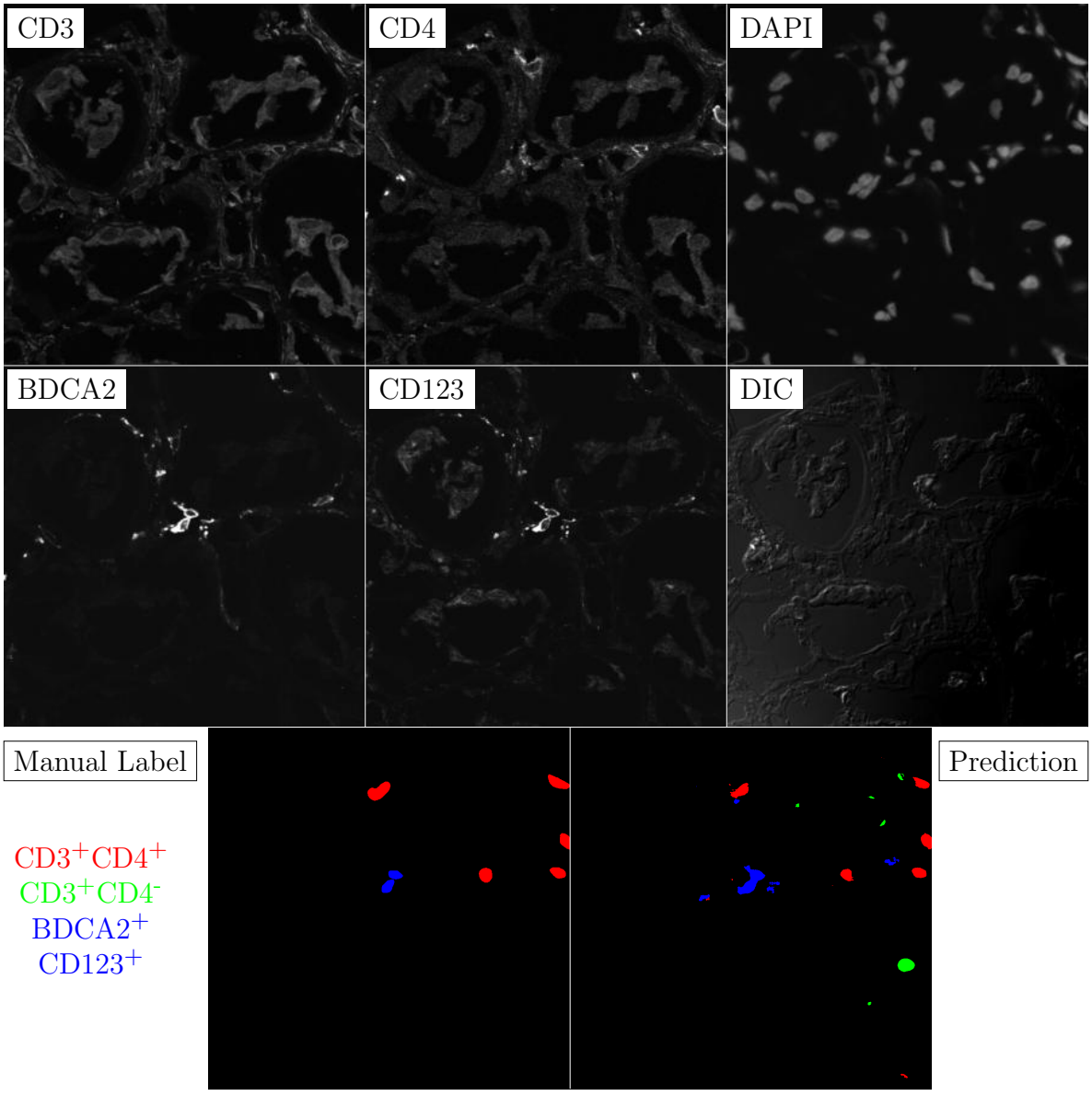

Figure G.19: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

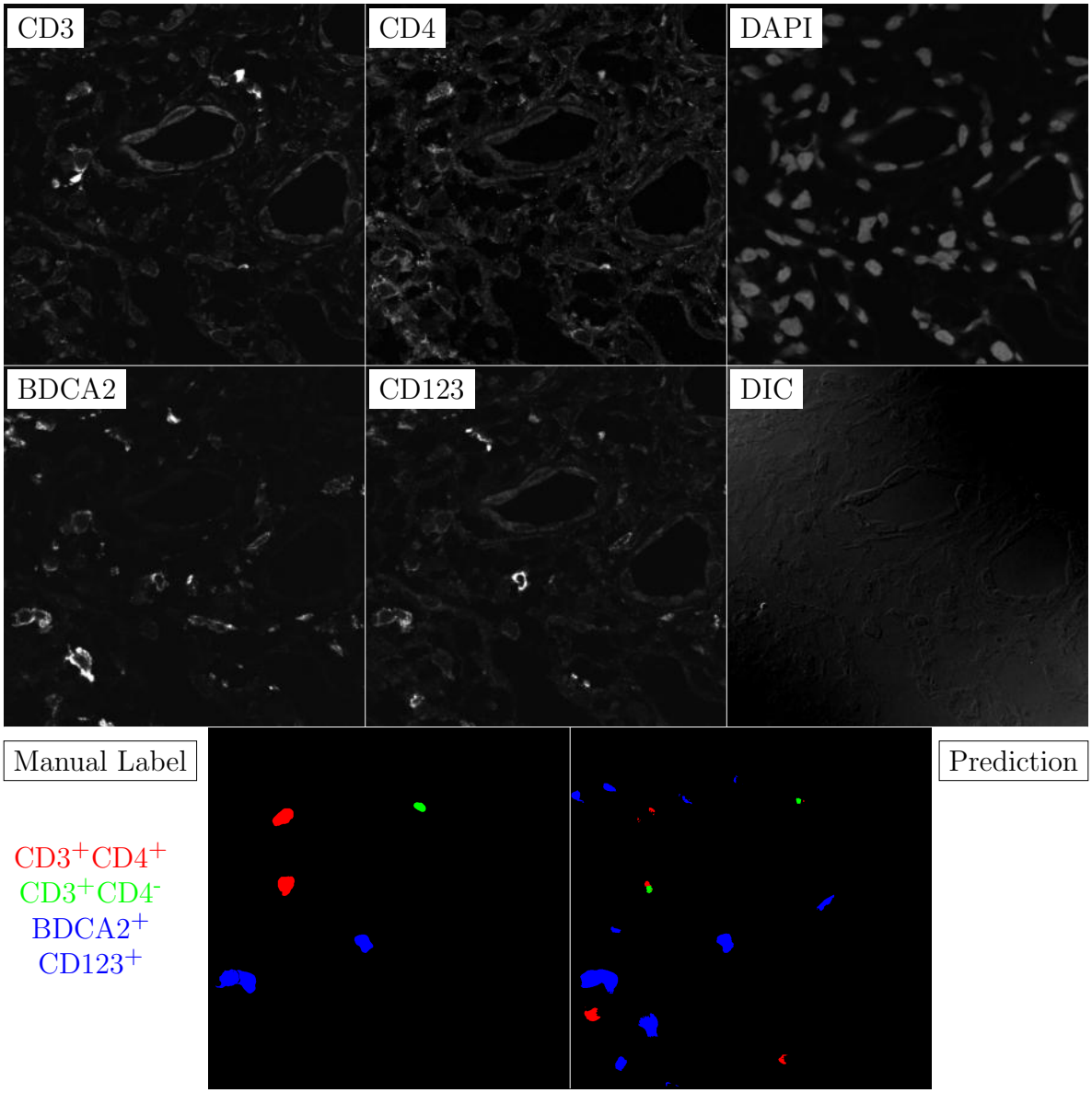

Figure G.20: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

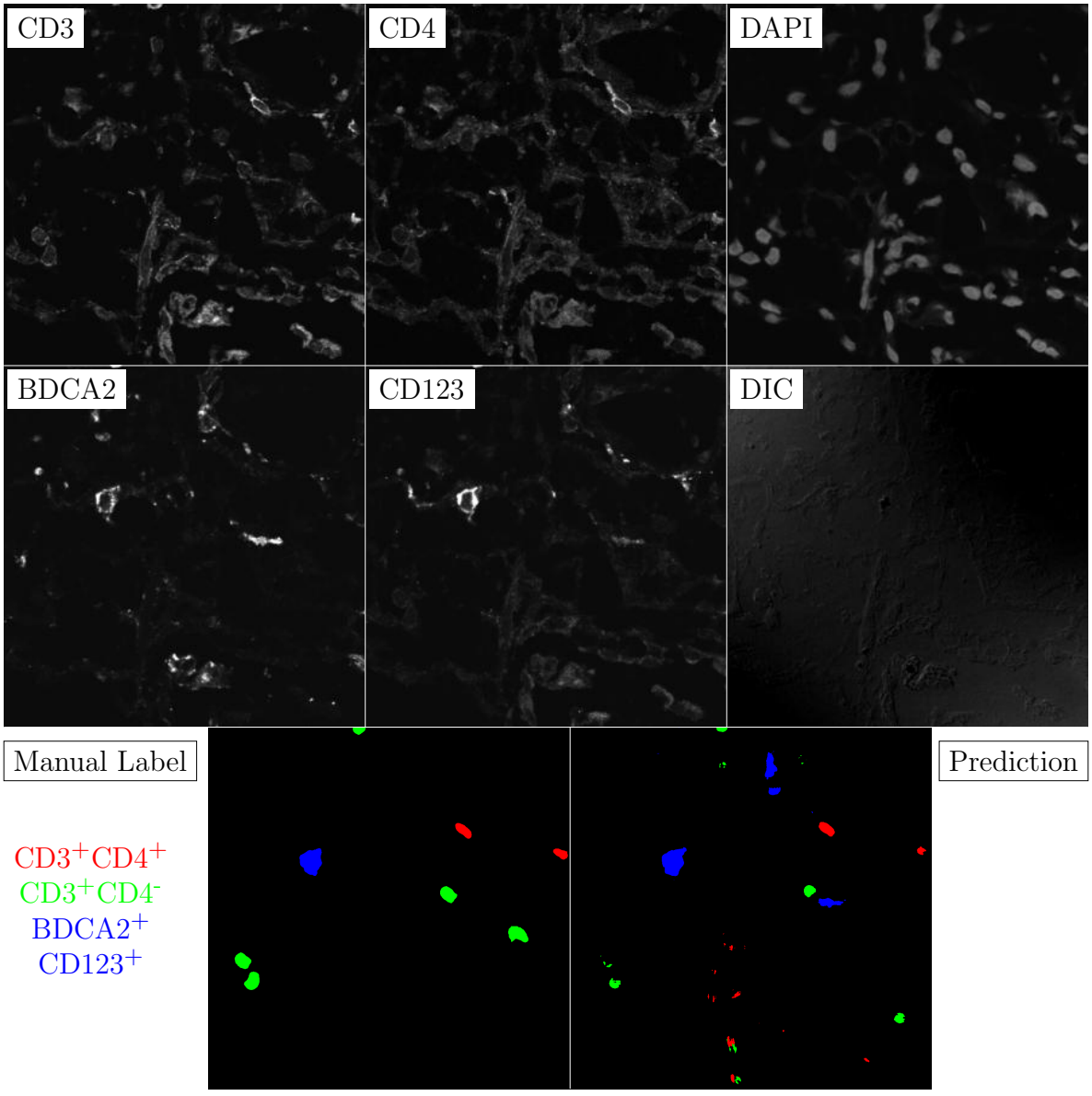

Figure G.21: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

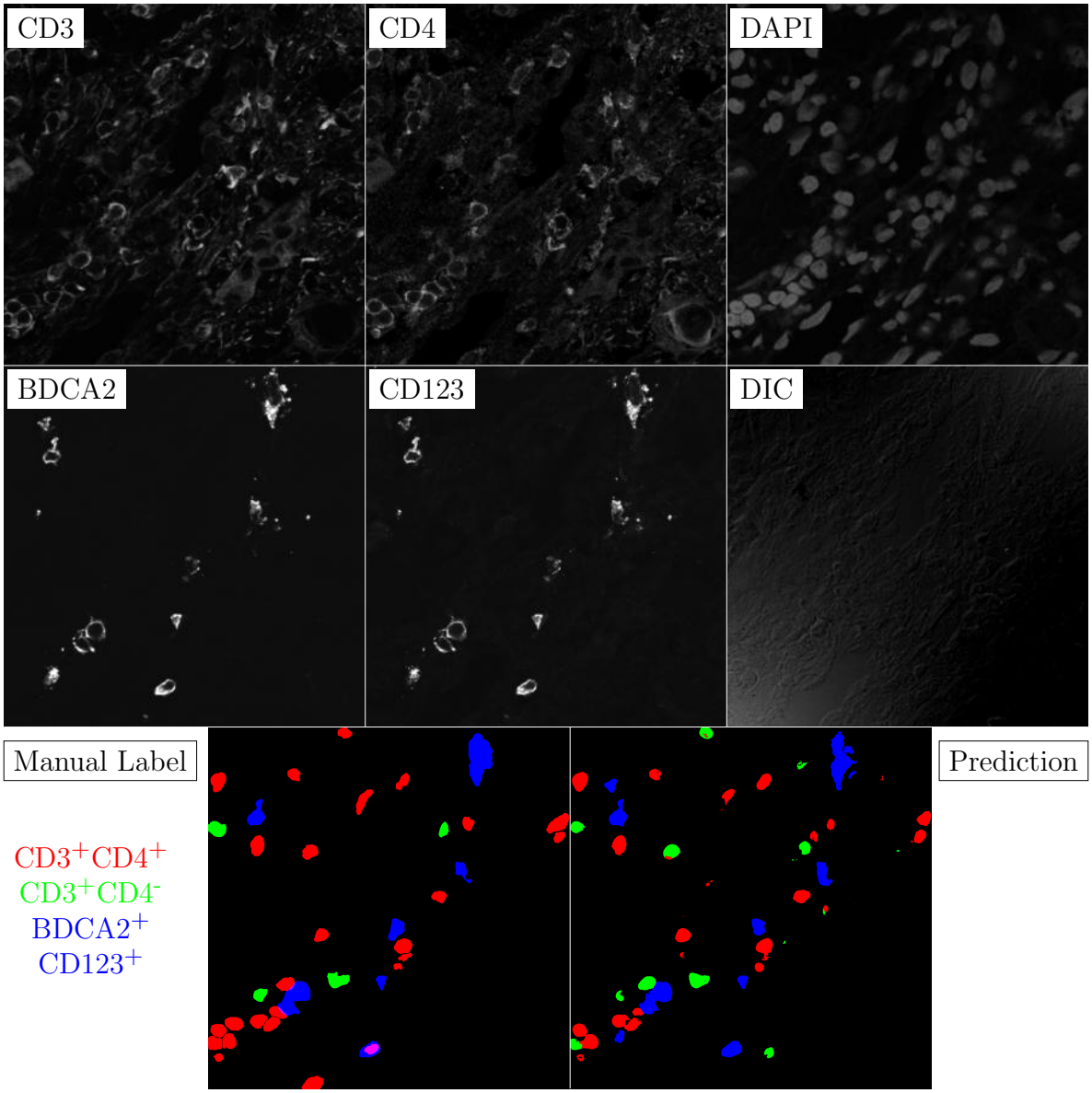

Figure G.22: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

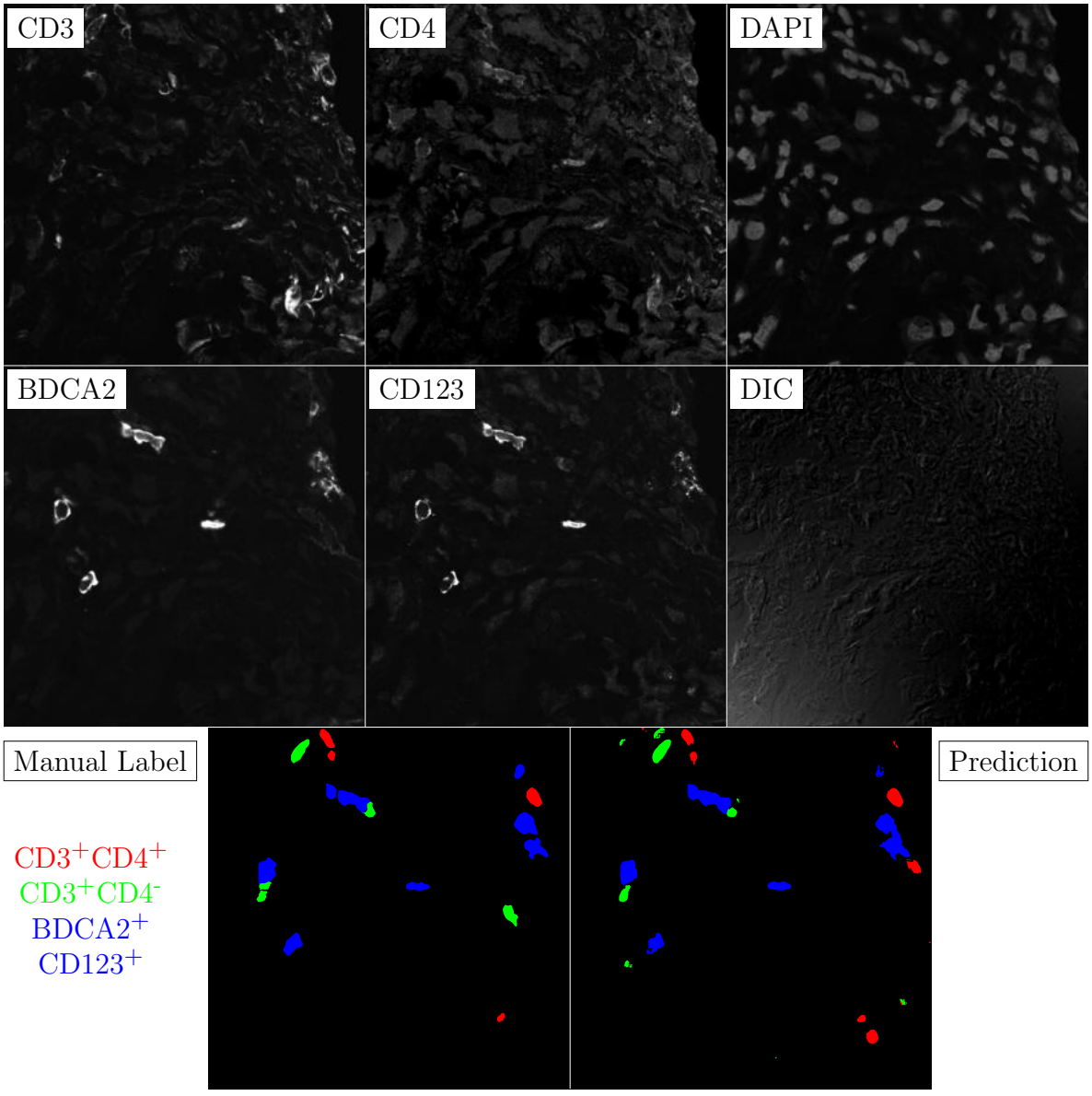

Figure G.23: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

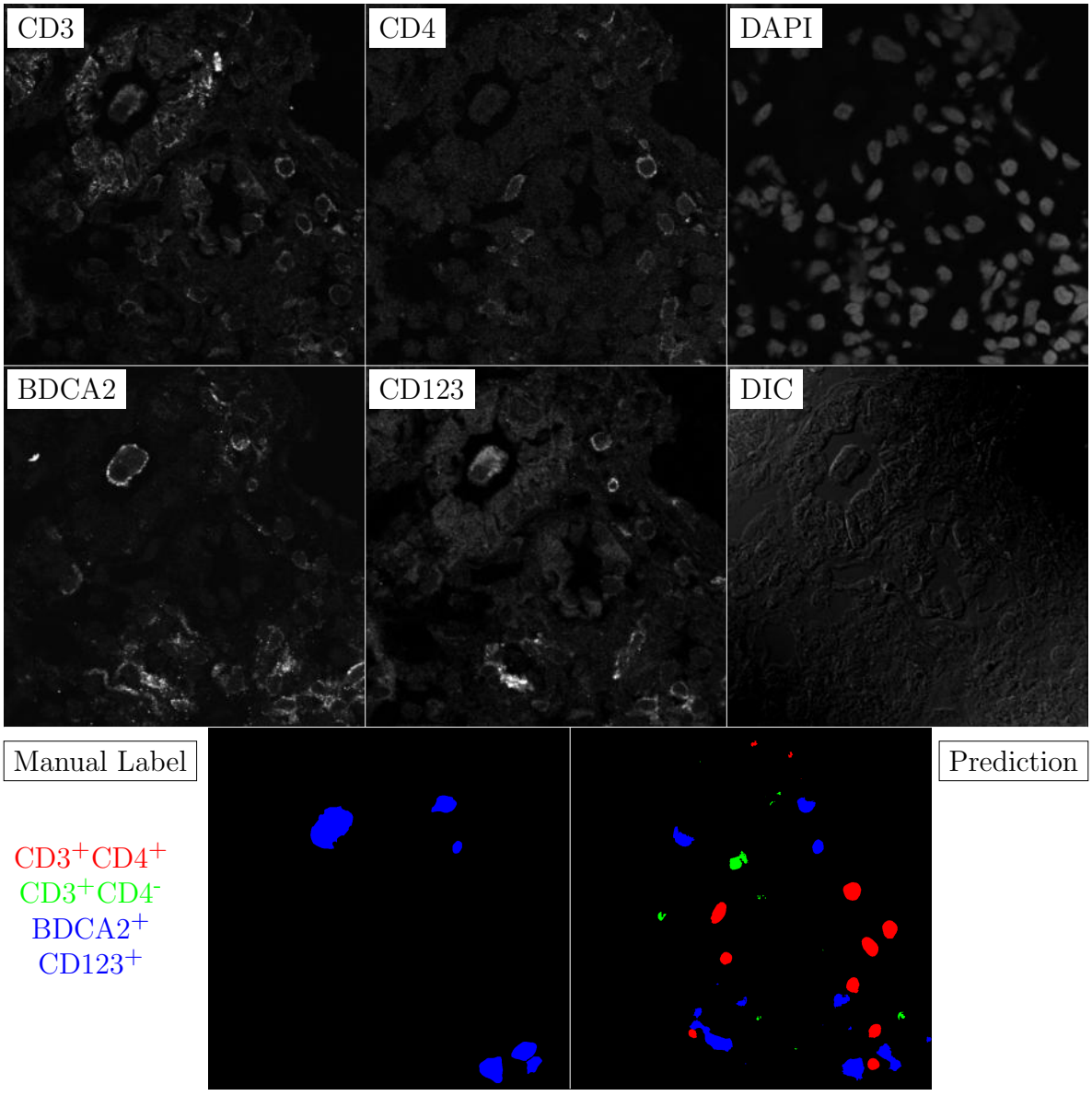

Figure G.24: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

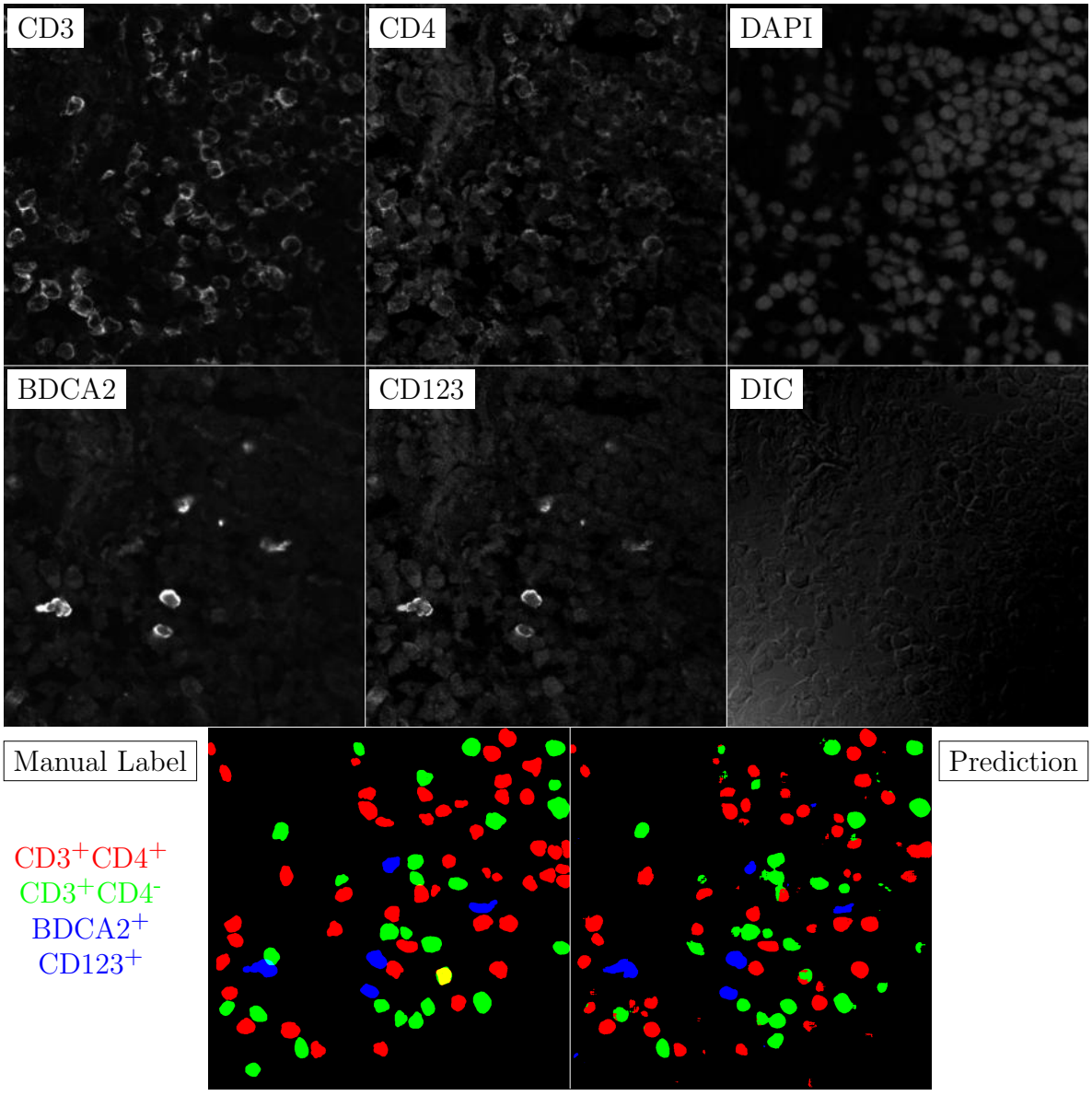

Figure G.25: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

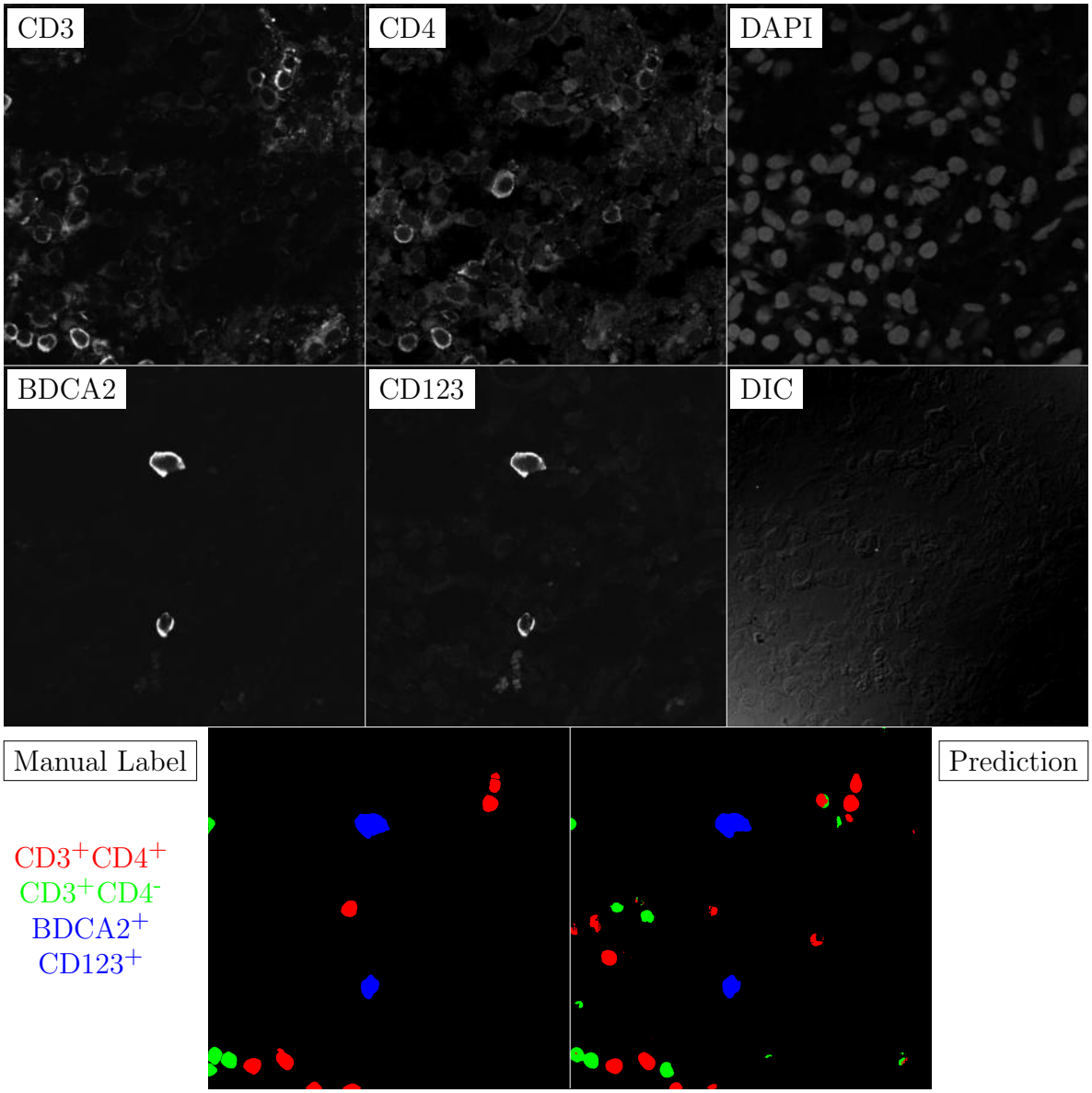

Figure G.26: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

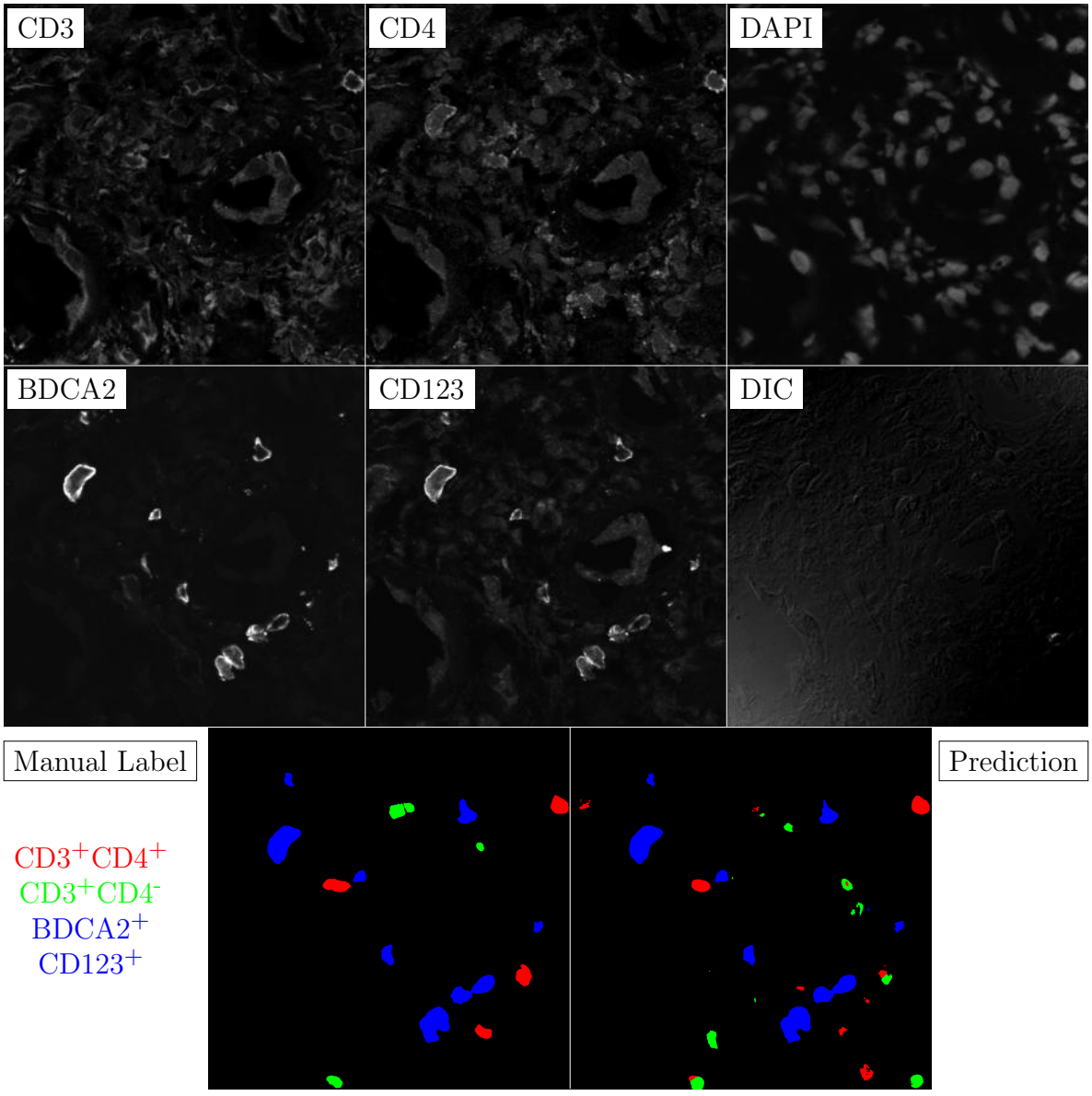

Figure G.27: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

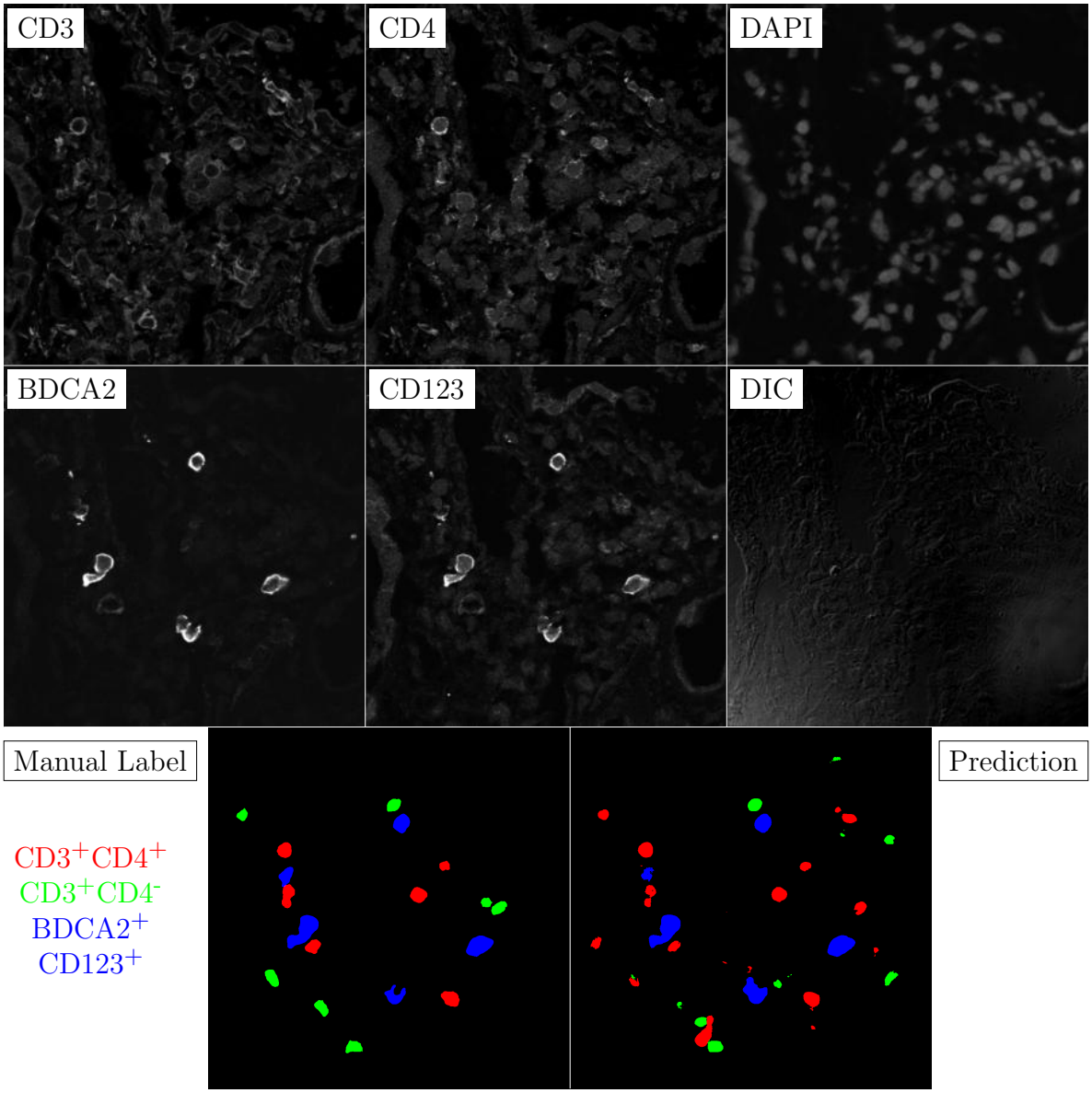

Figure G.28: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

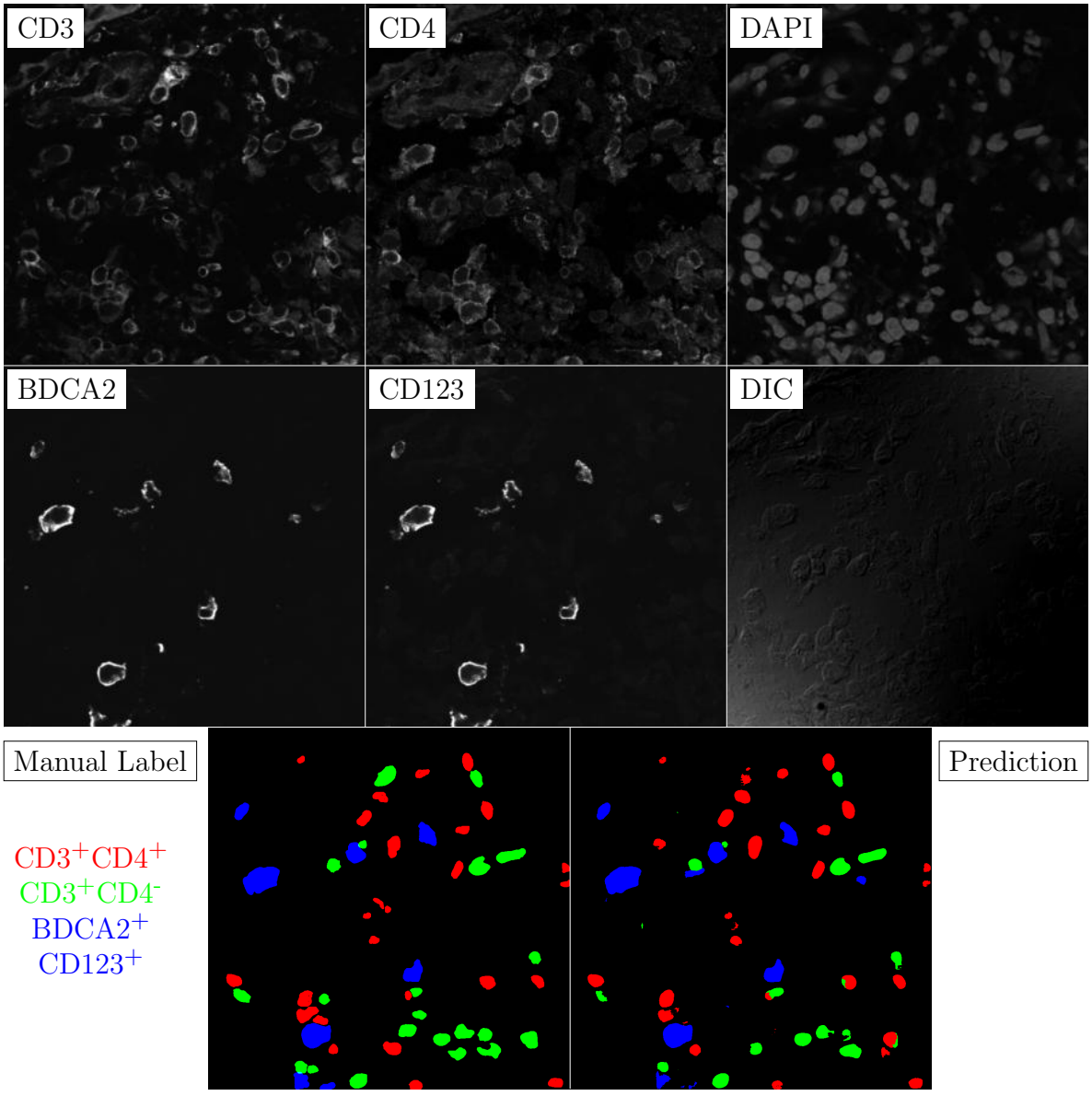

Figure G.29: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

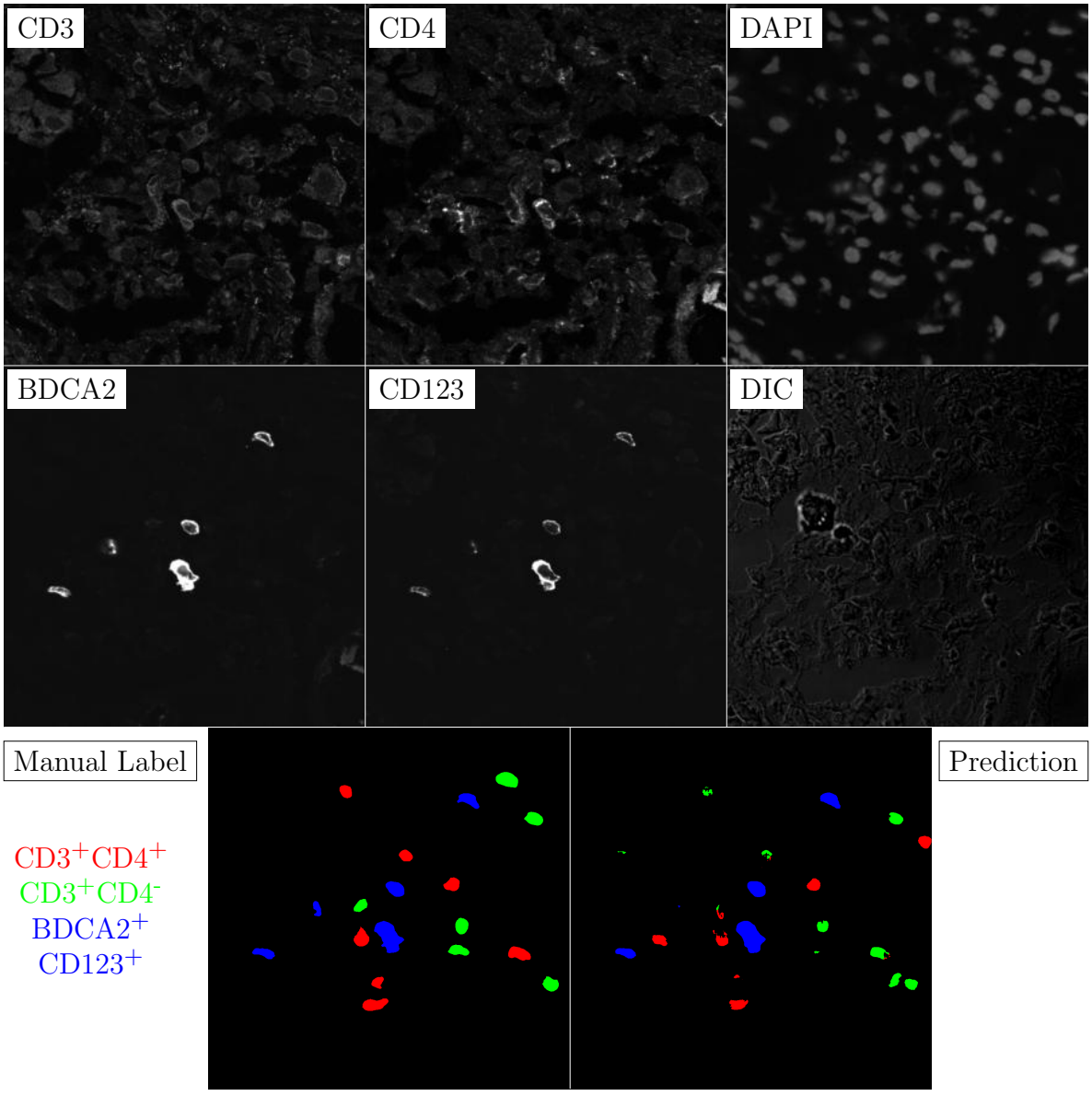

Figure G.30: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

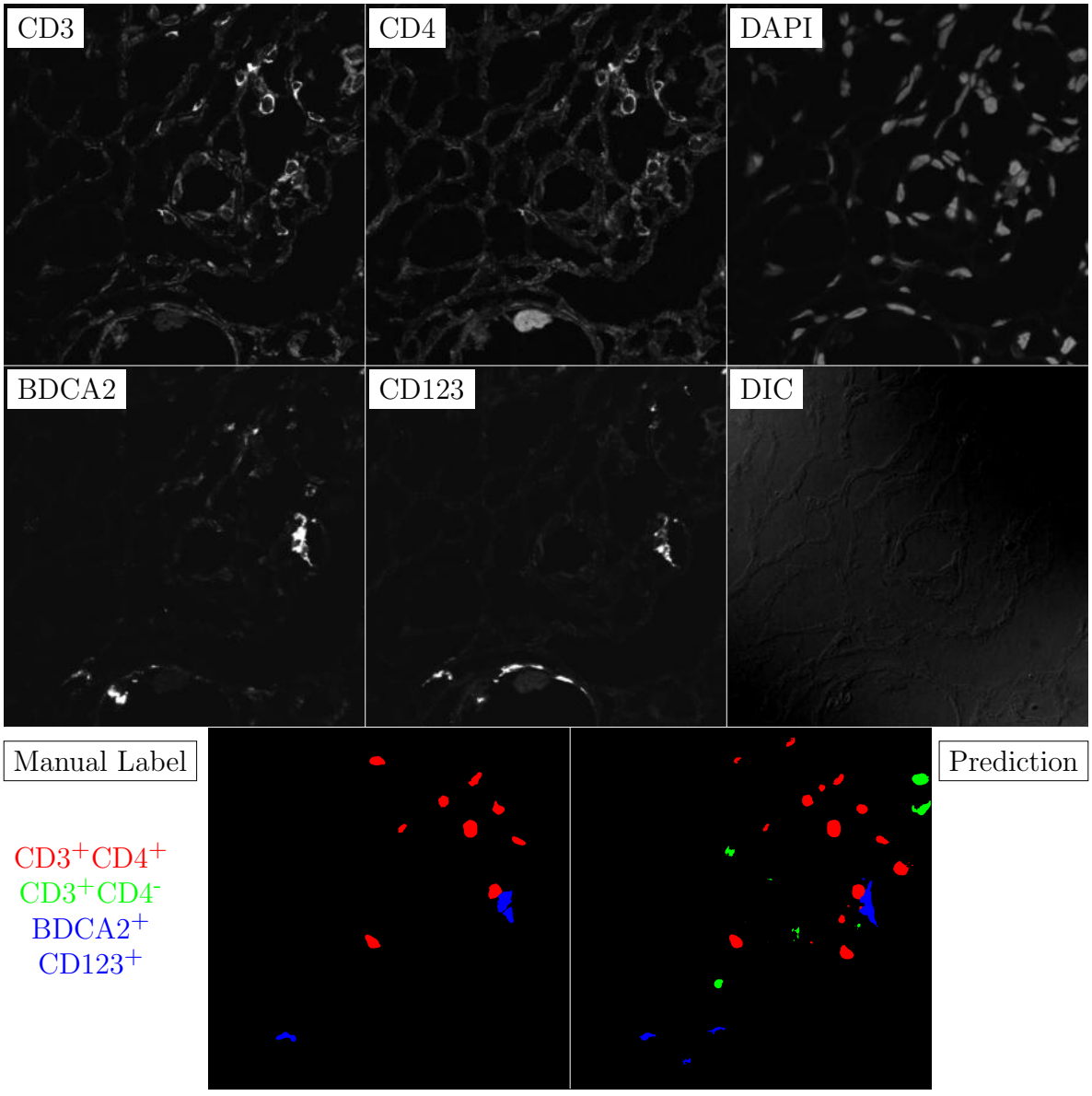

Figure G.31: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

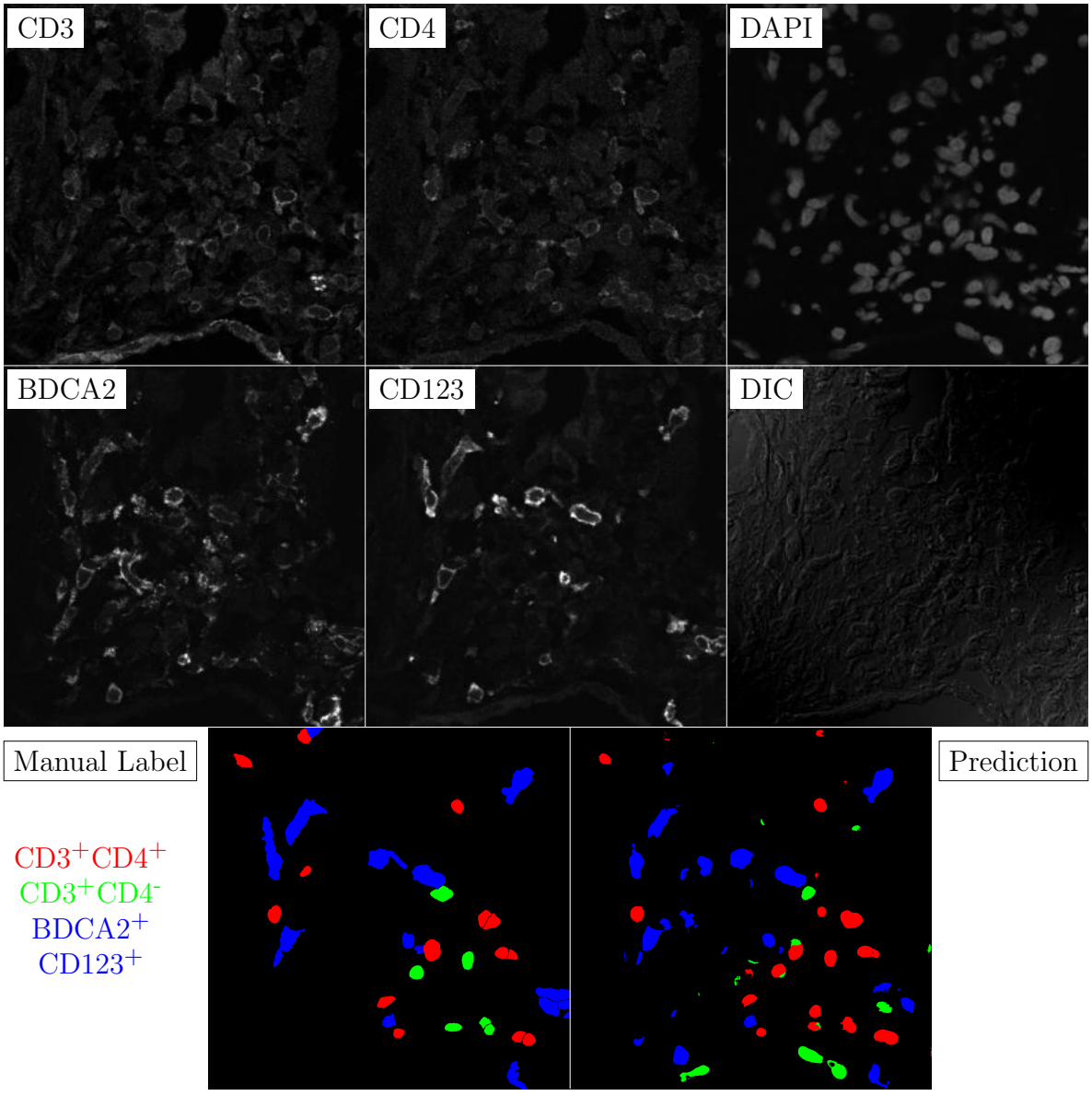

Figure G.32: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

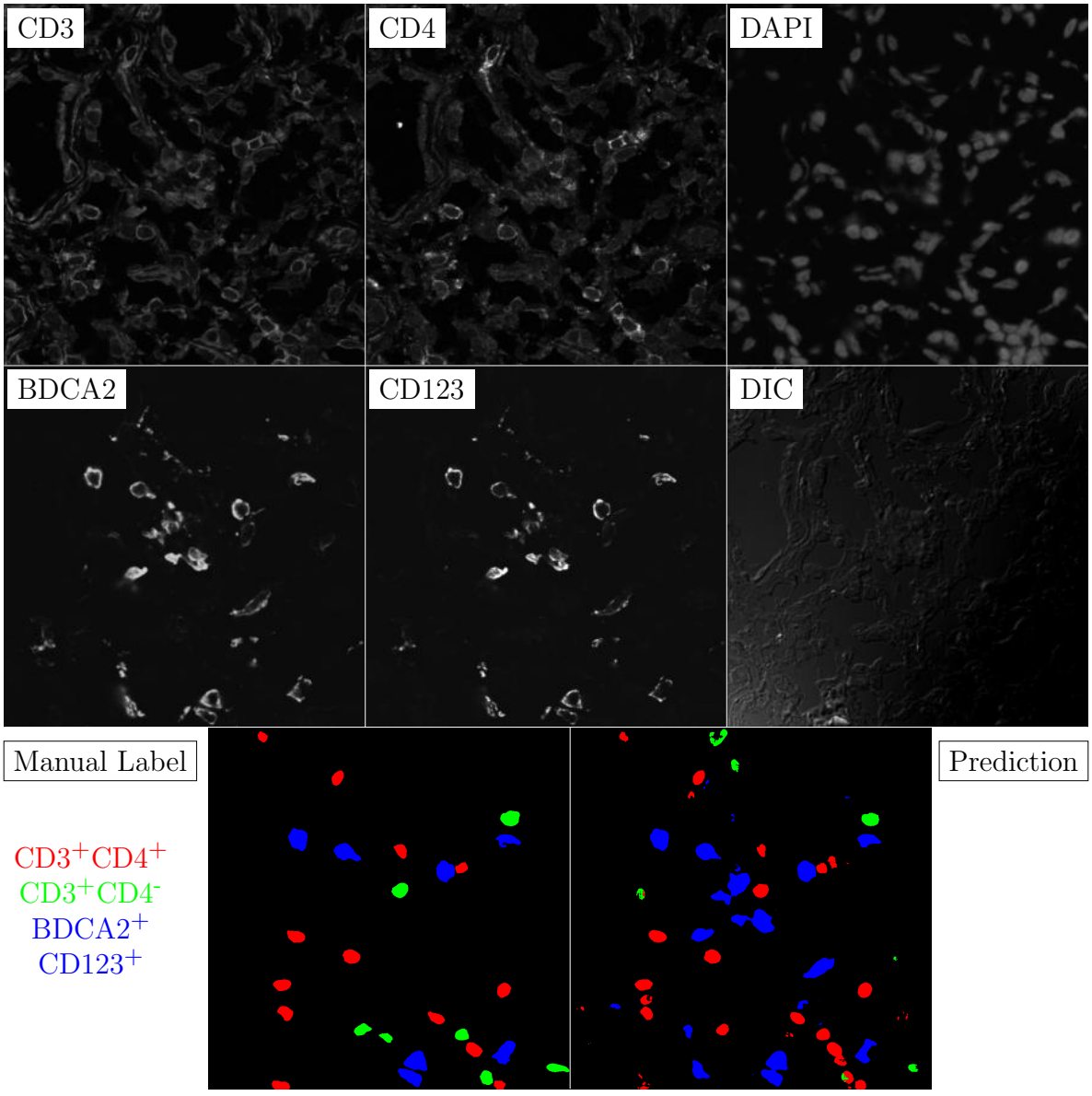

Figure G.33: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

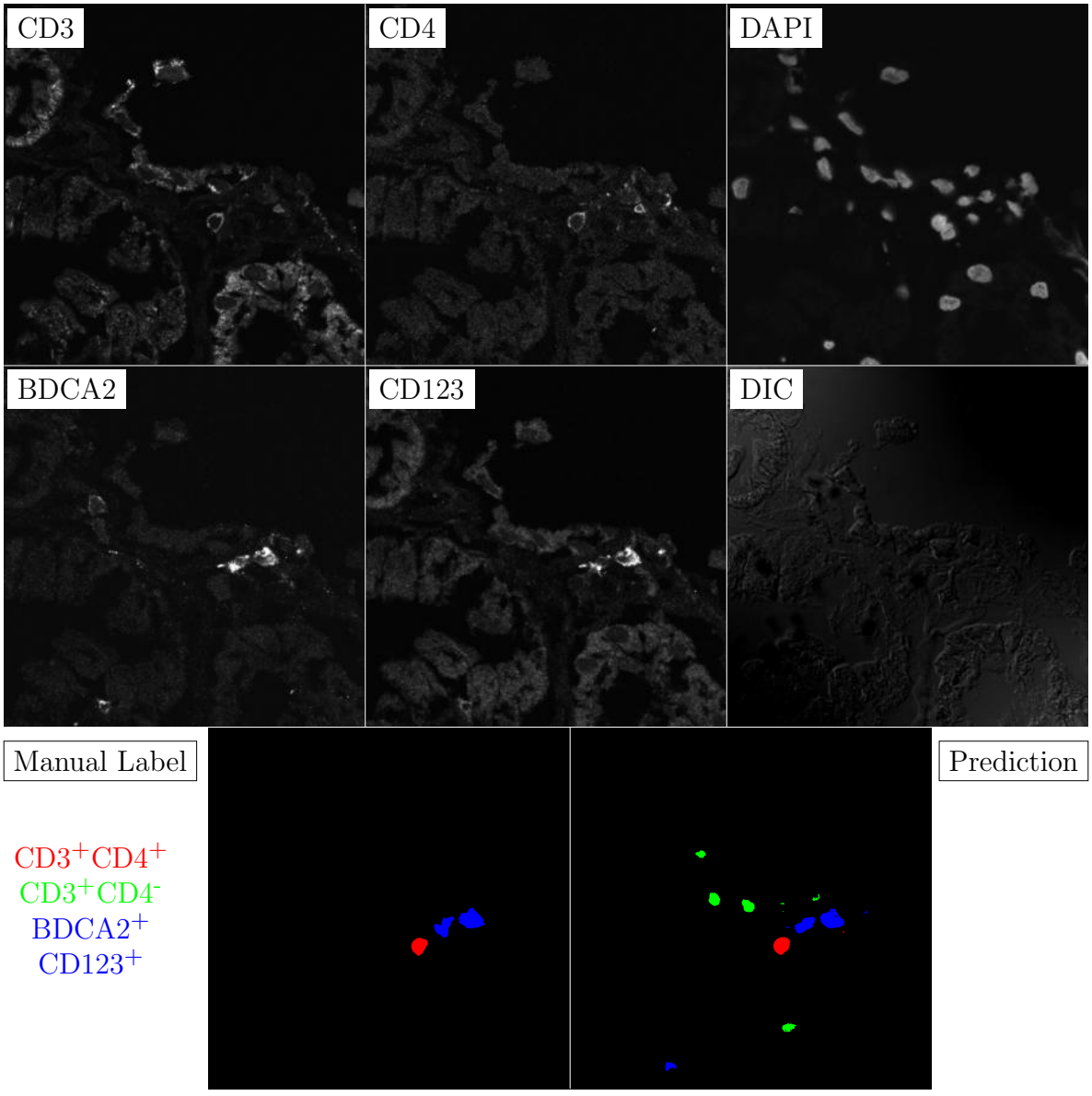

Figure G.34: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

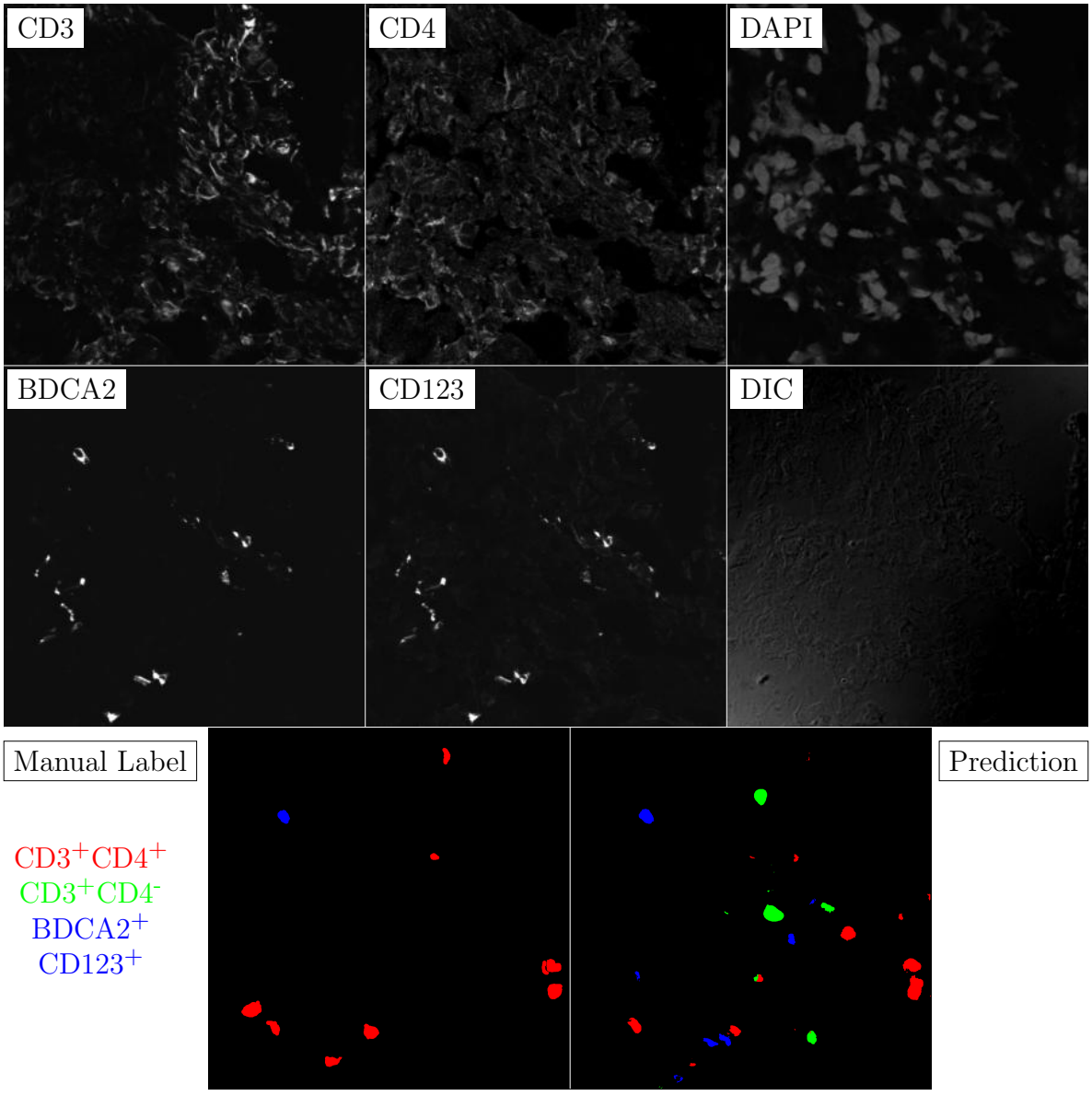

Figure G.35: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

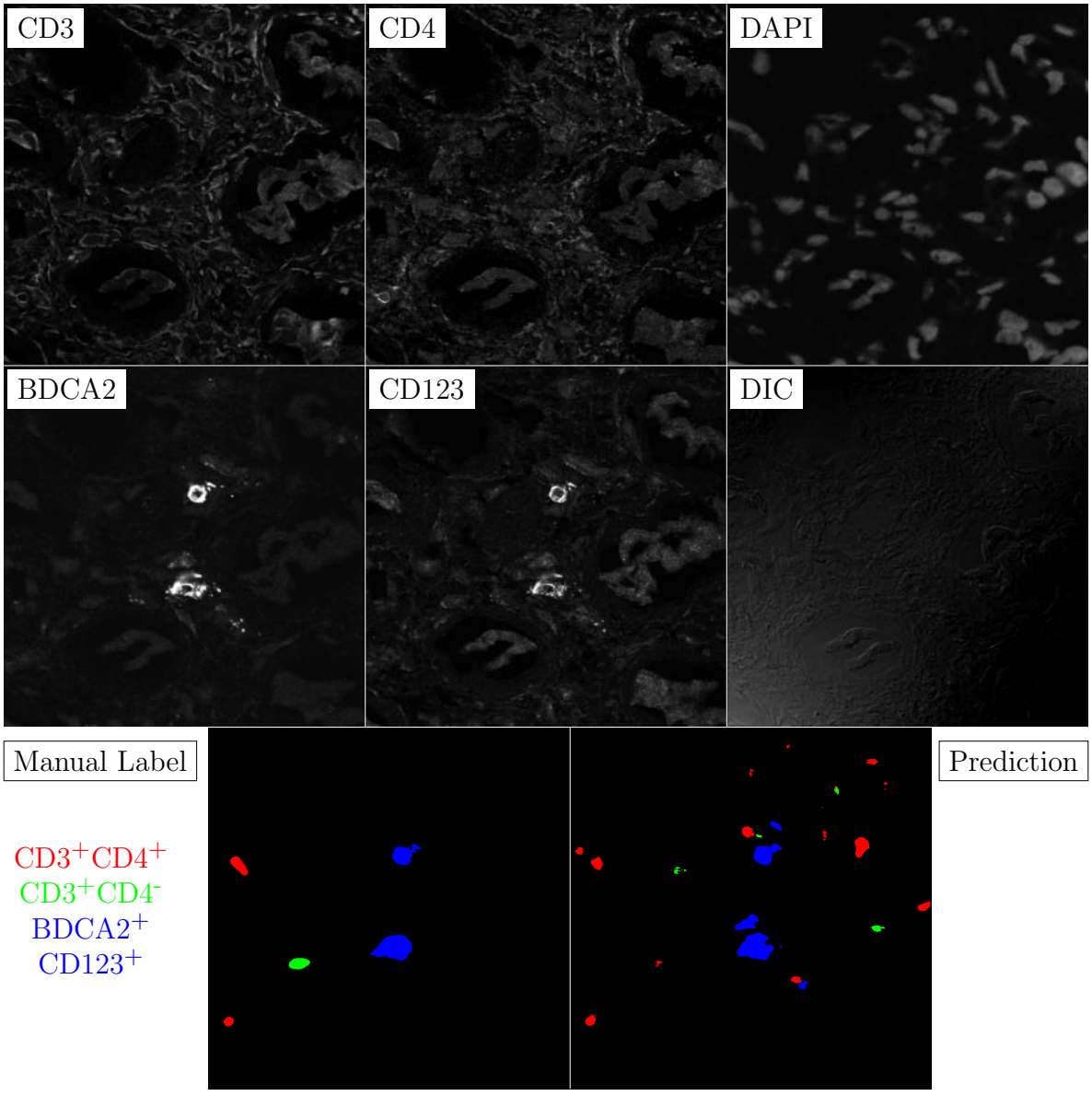

Figure G.36: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

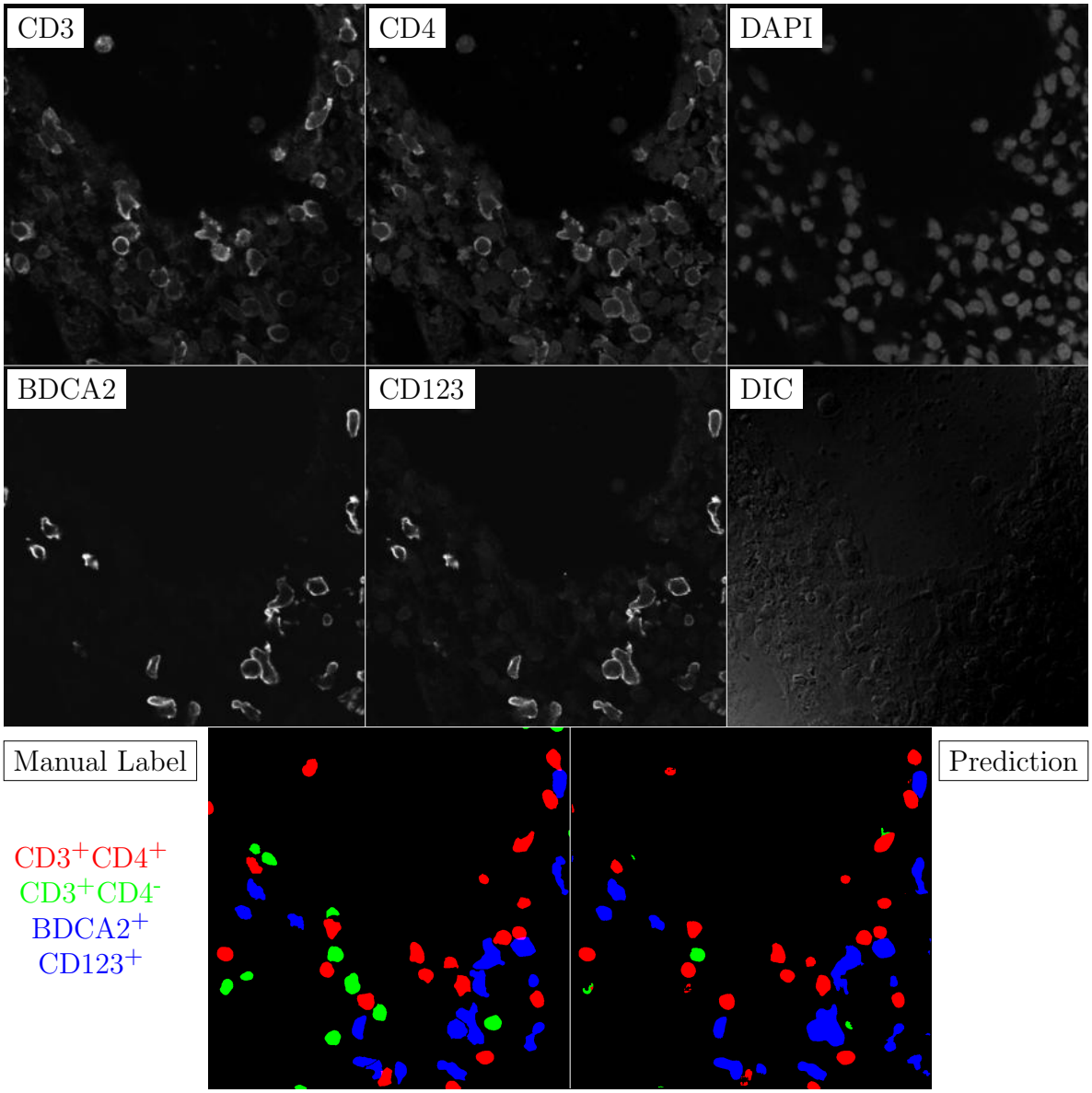

Figure G.37: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

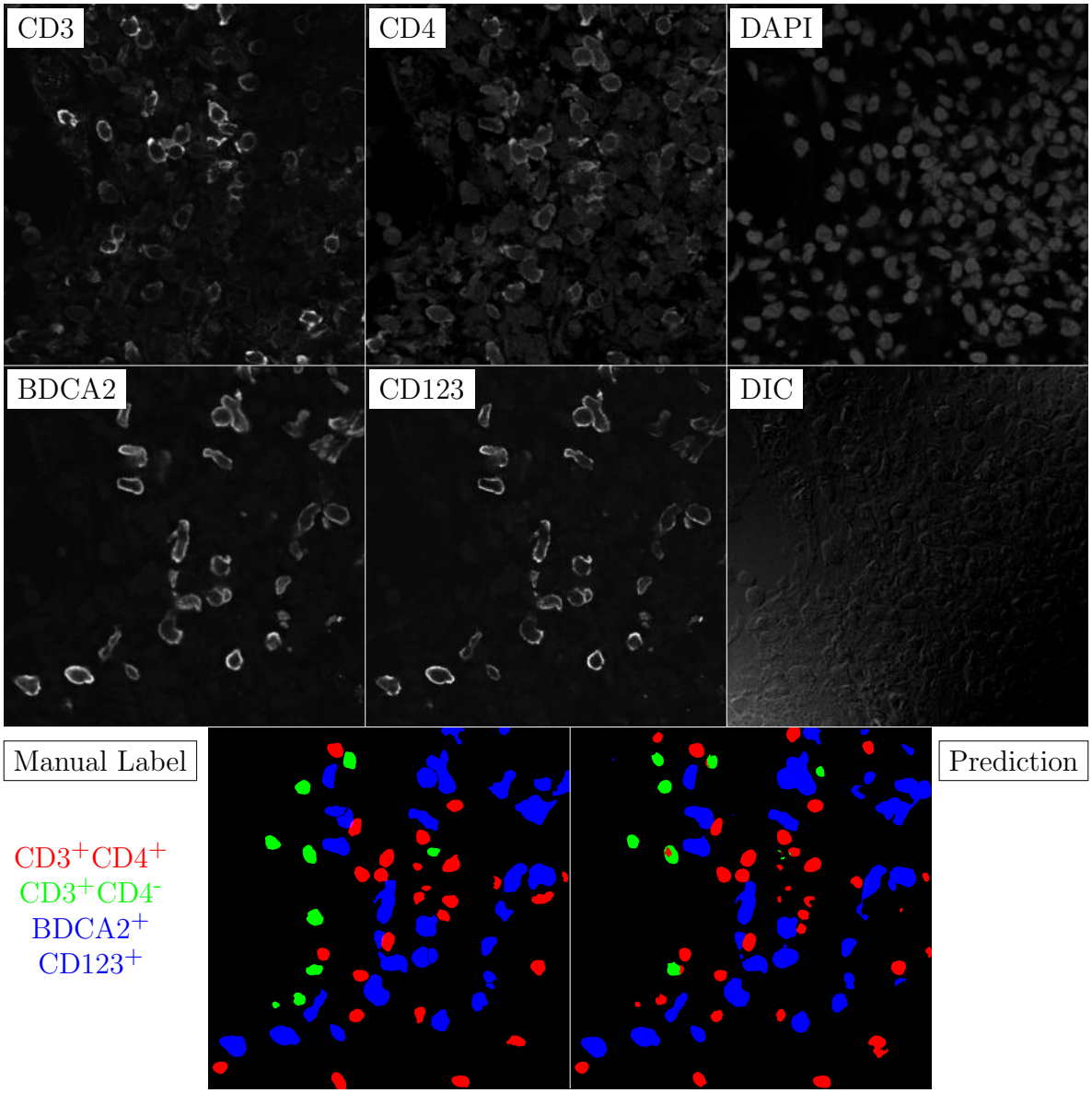

Figure G.38: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

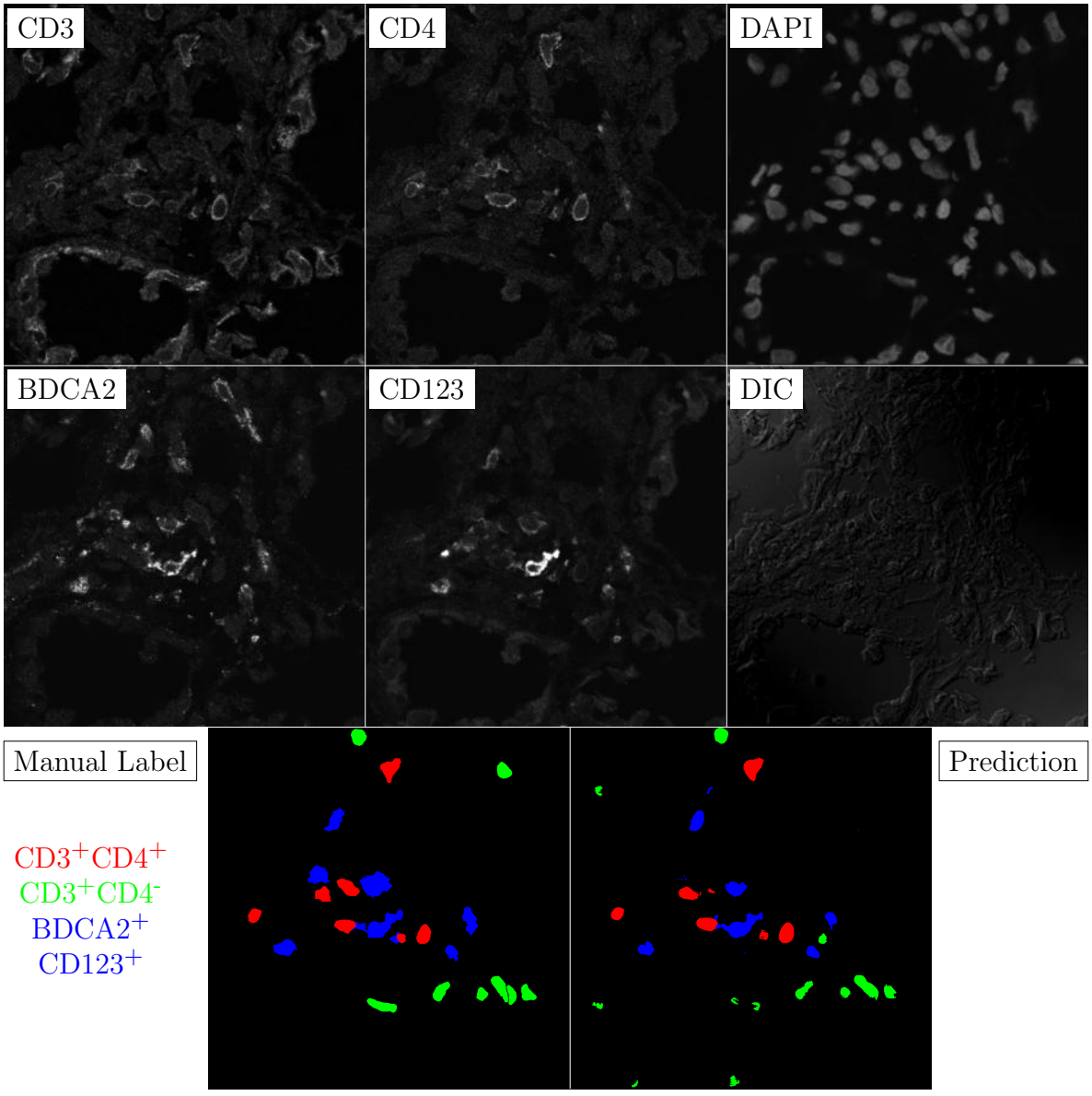

Figure G.39: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

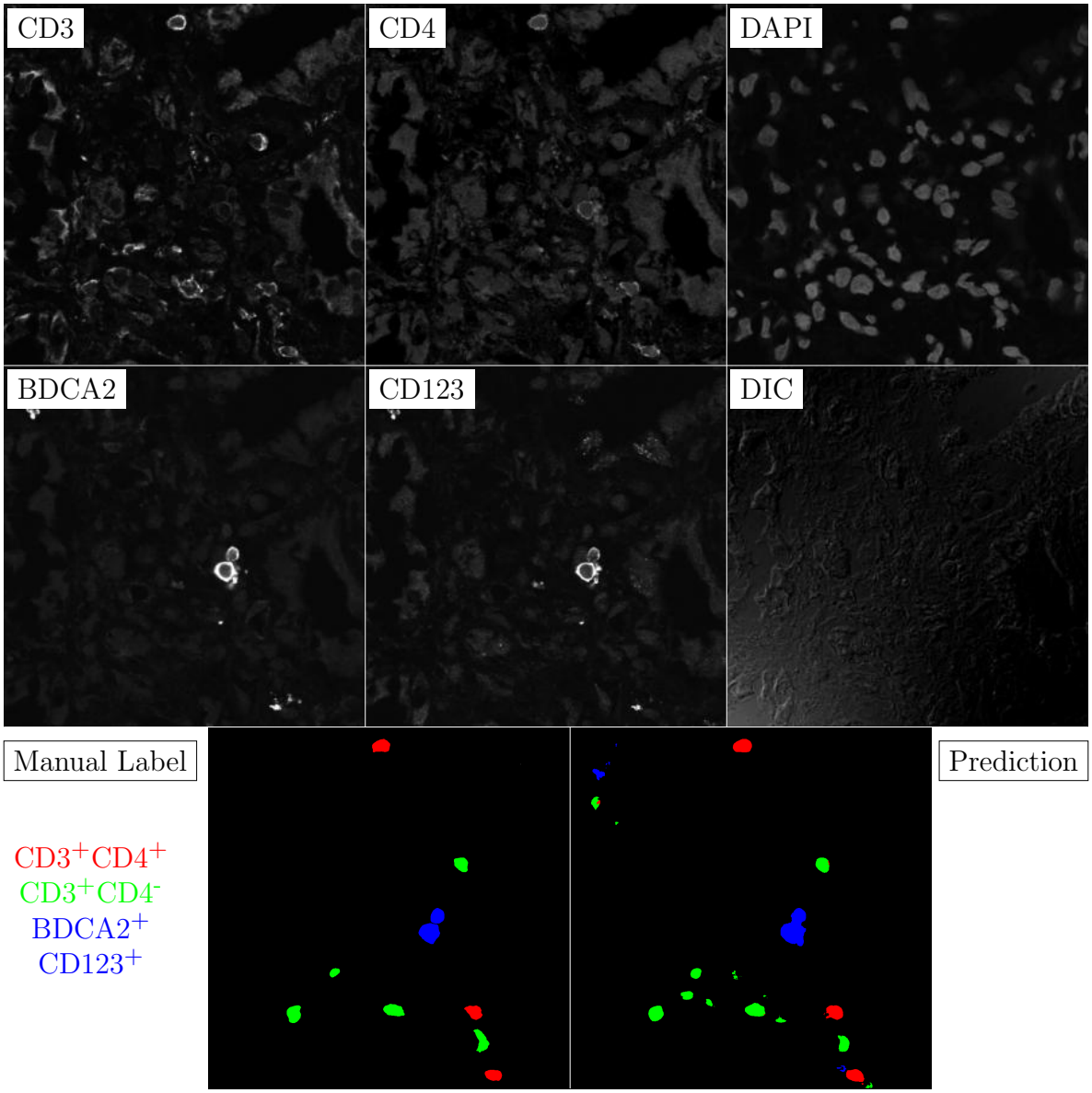

Figure G.40: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

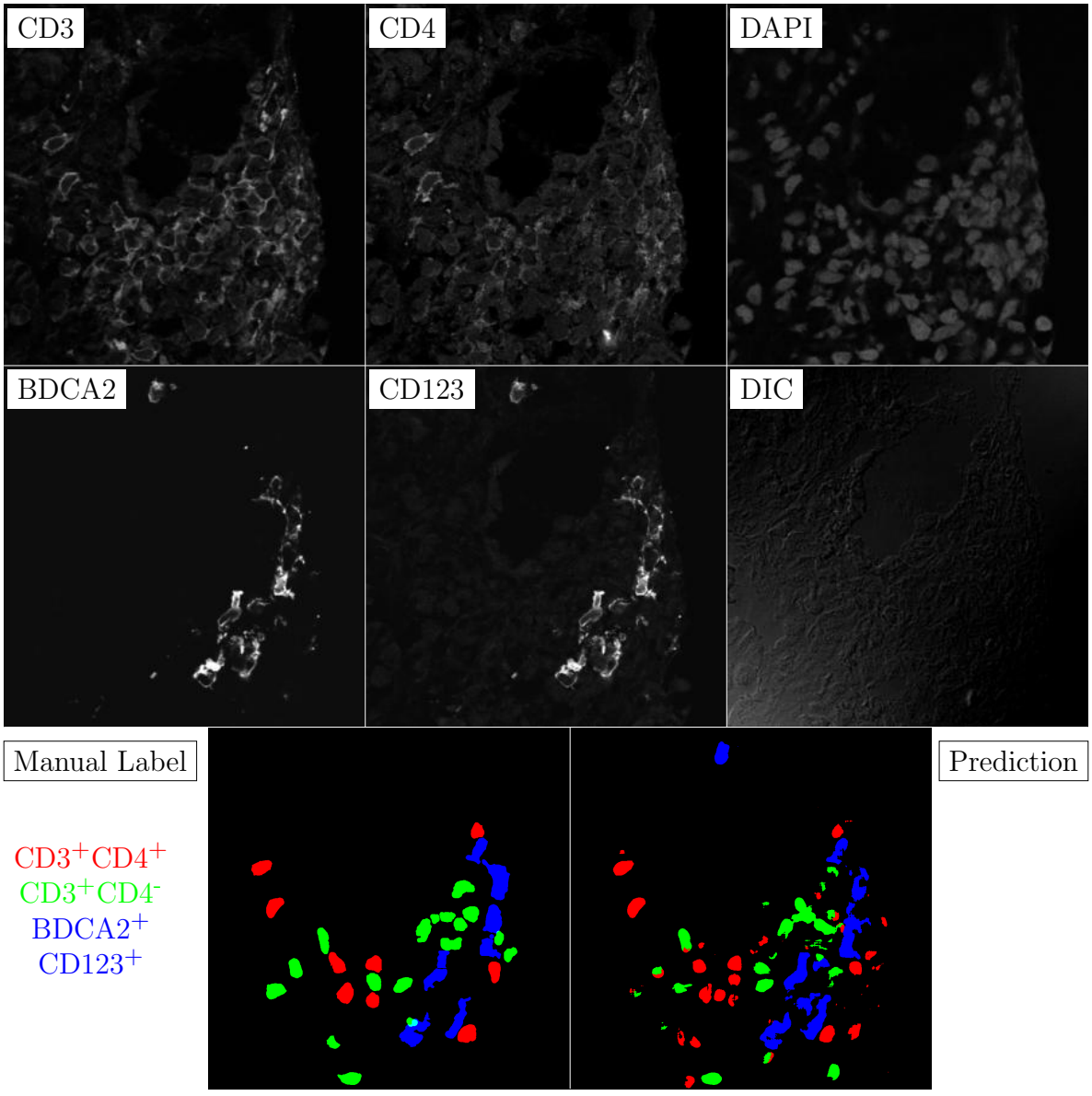

Figure G.41: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

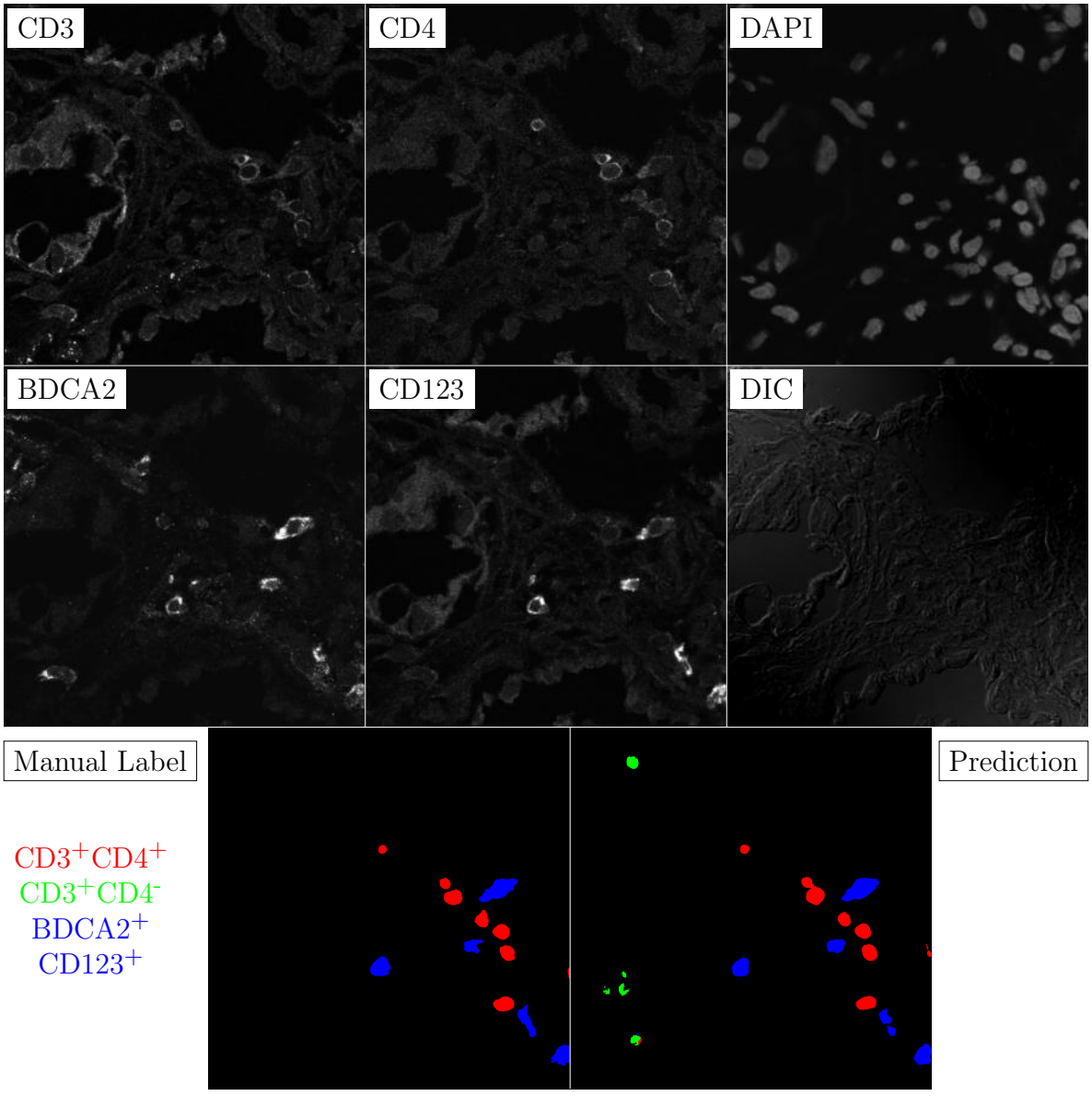

Figure G.42: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

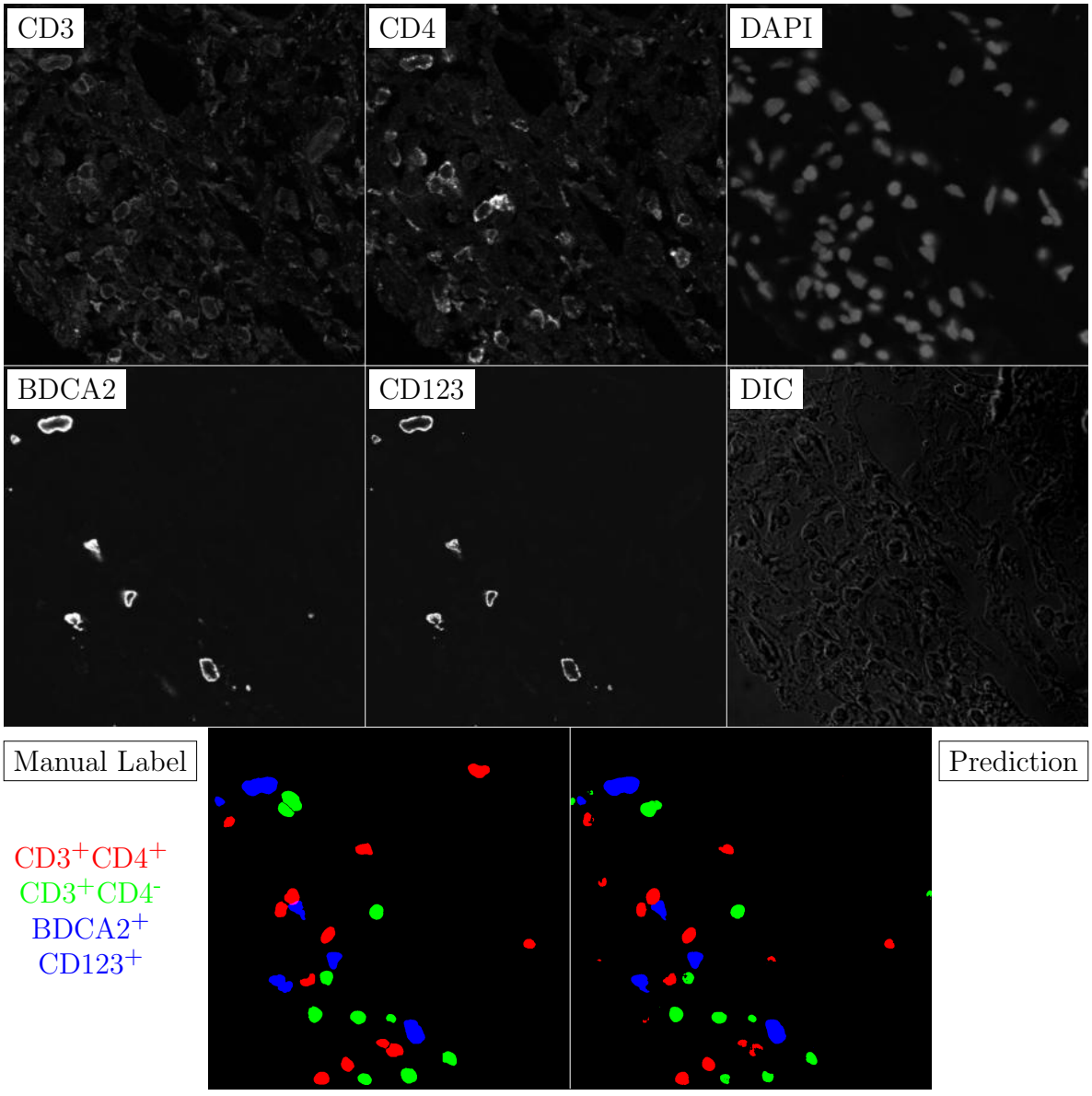

Figure G.43: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

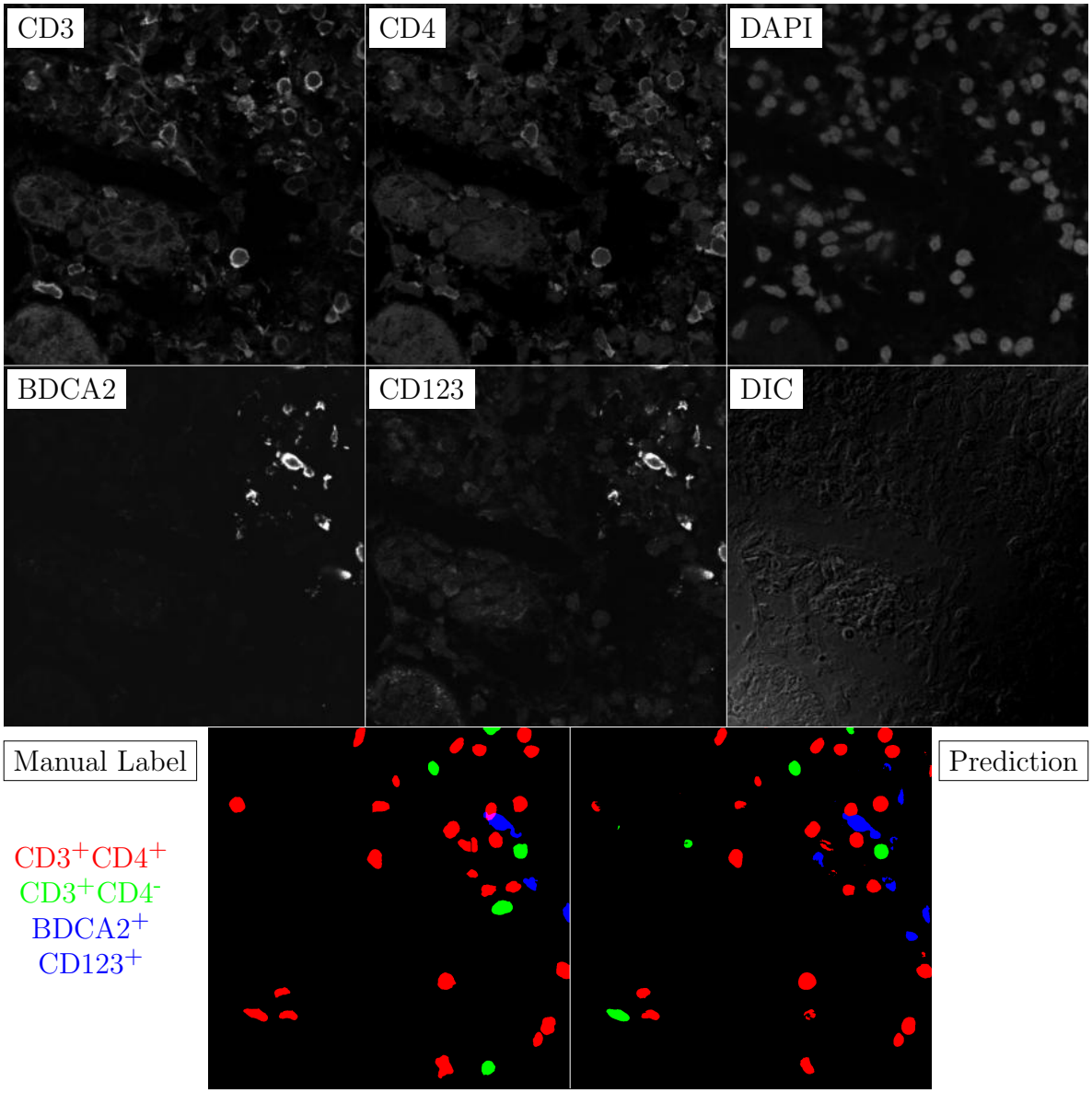

Figure G.44: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

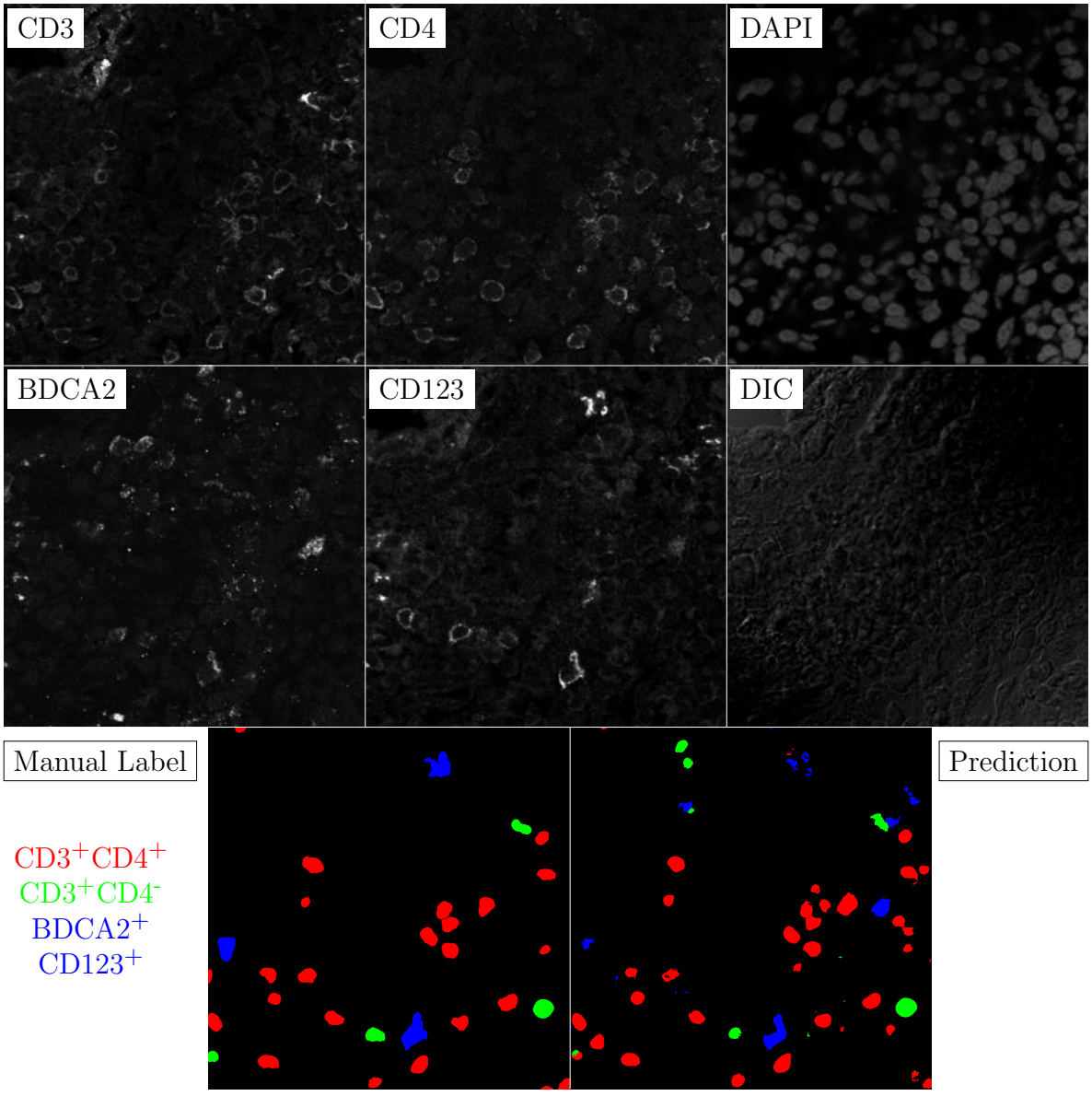

Figure G.45: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

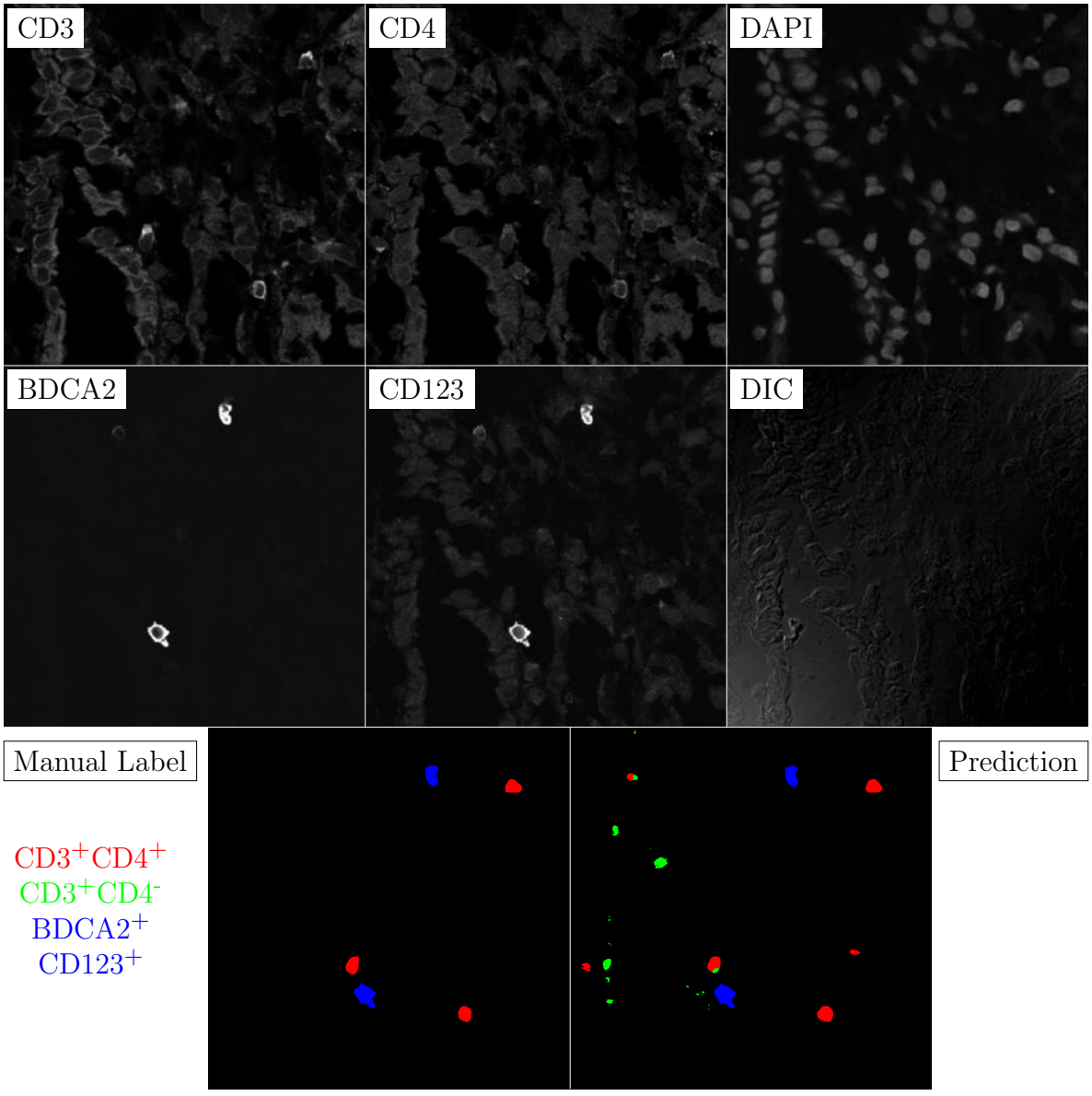

Figure G.46: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

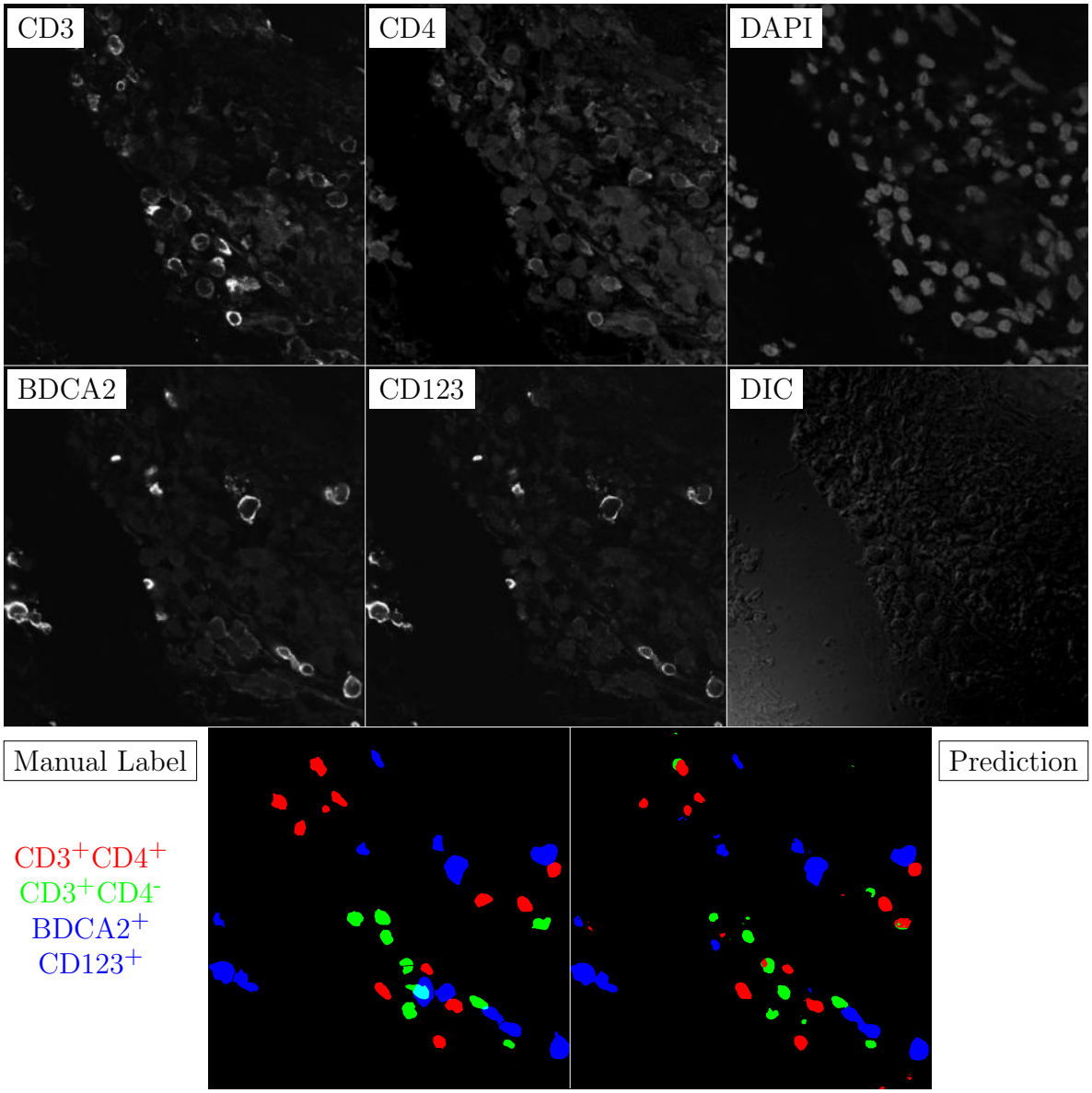

Figure G.47: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

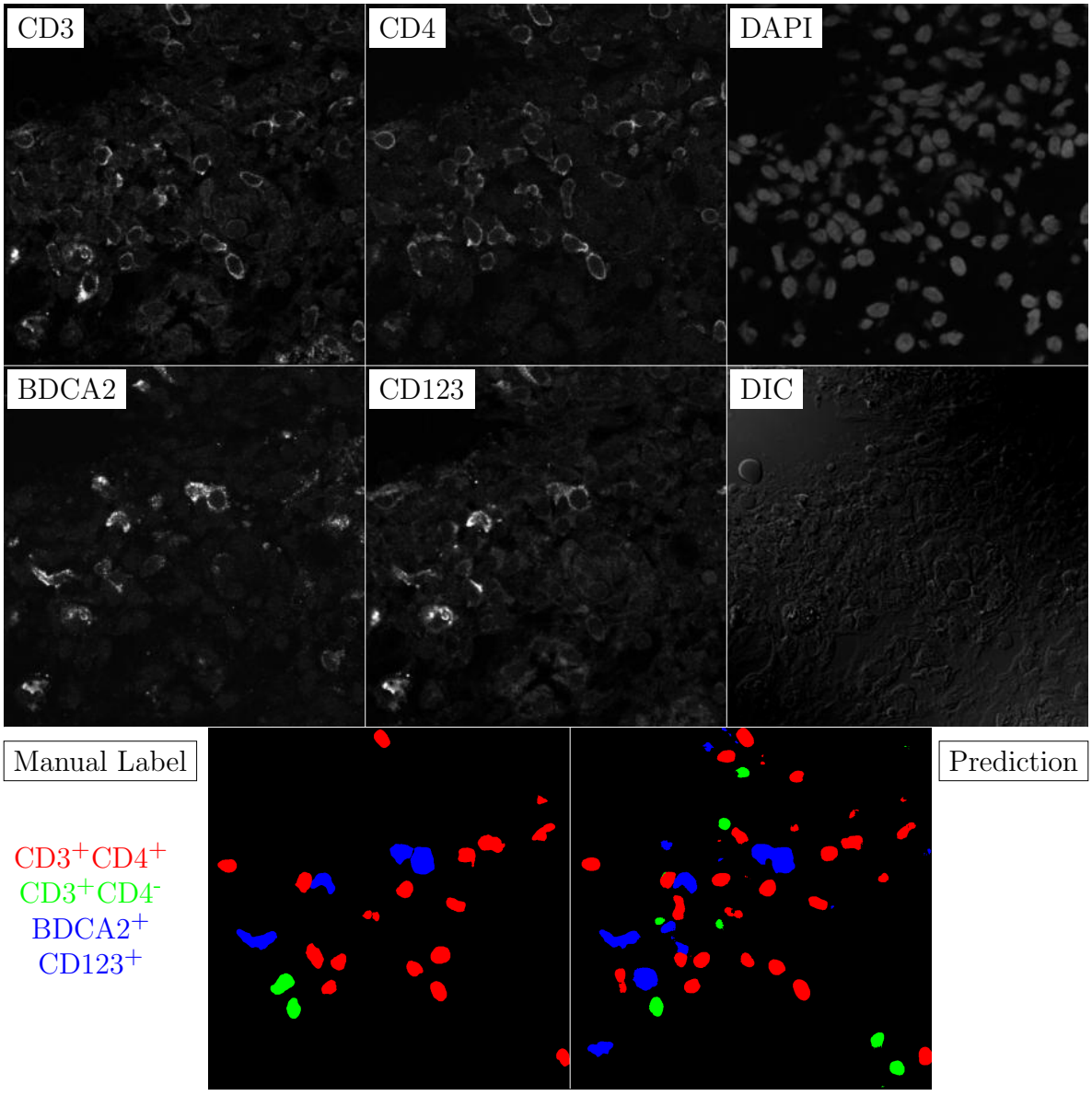

Figure G.48: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

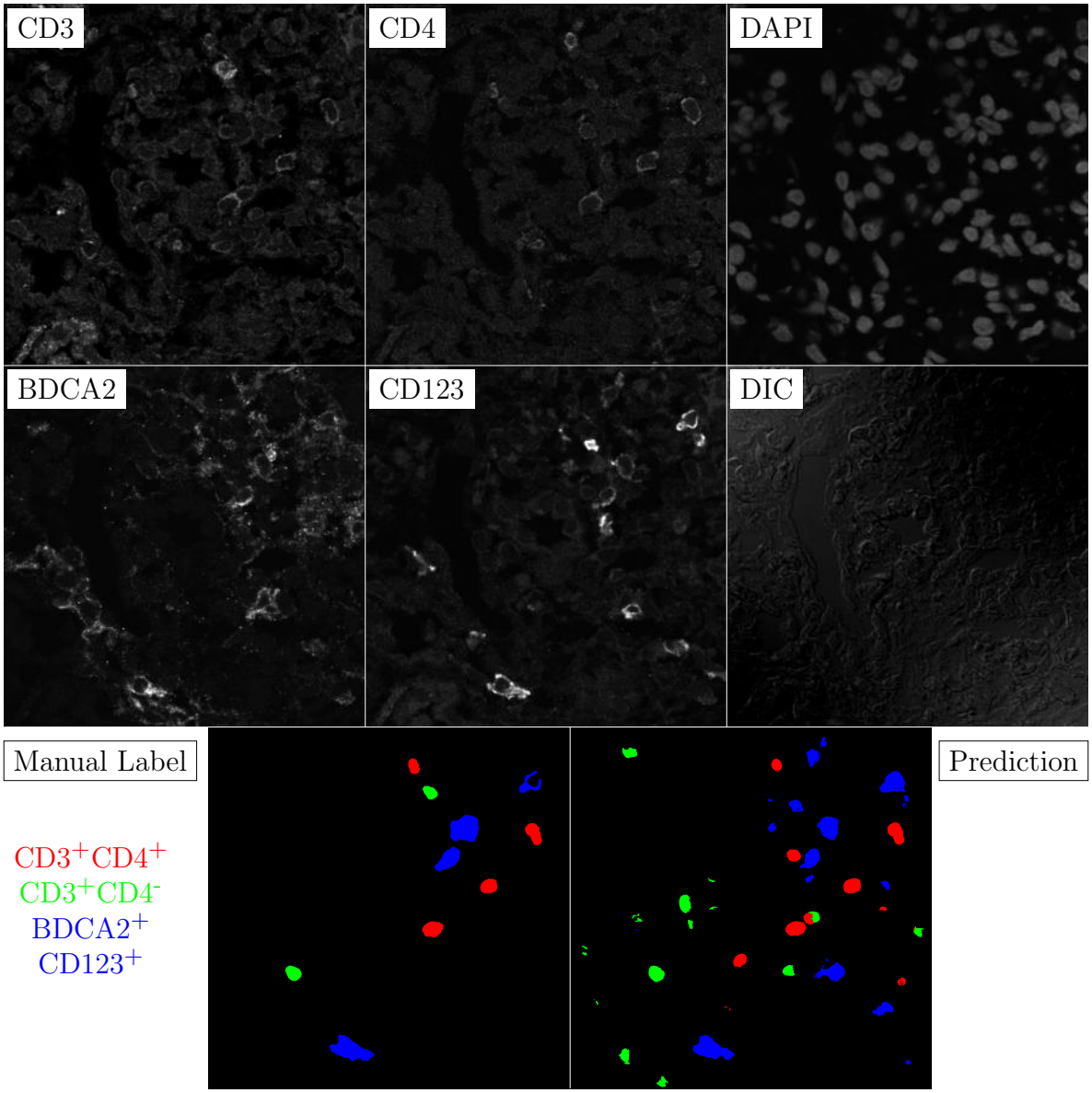

Figure G.49: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

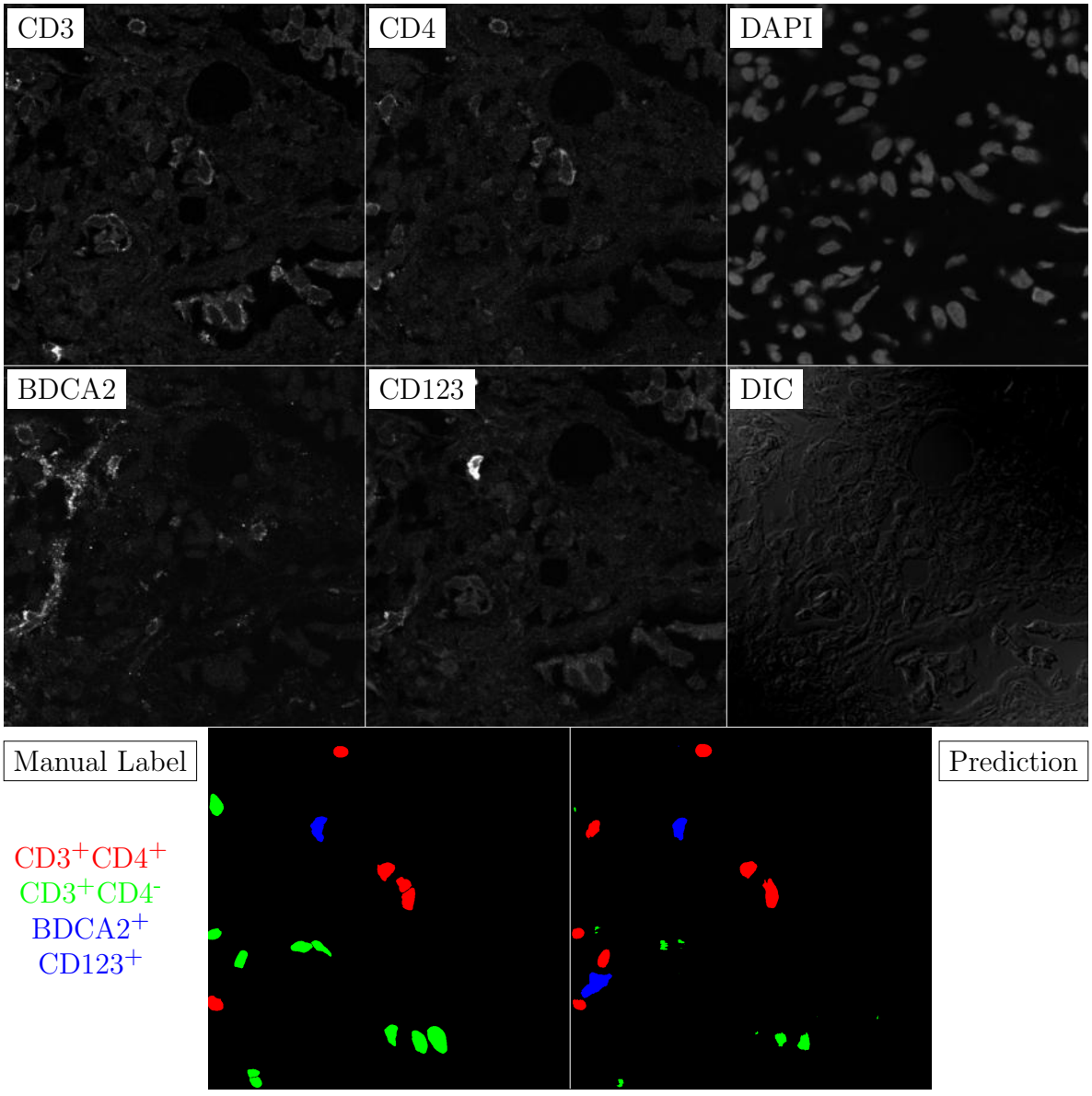

Figure G.50: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

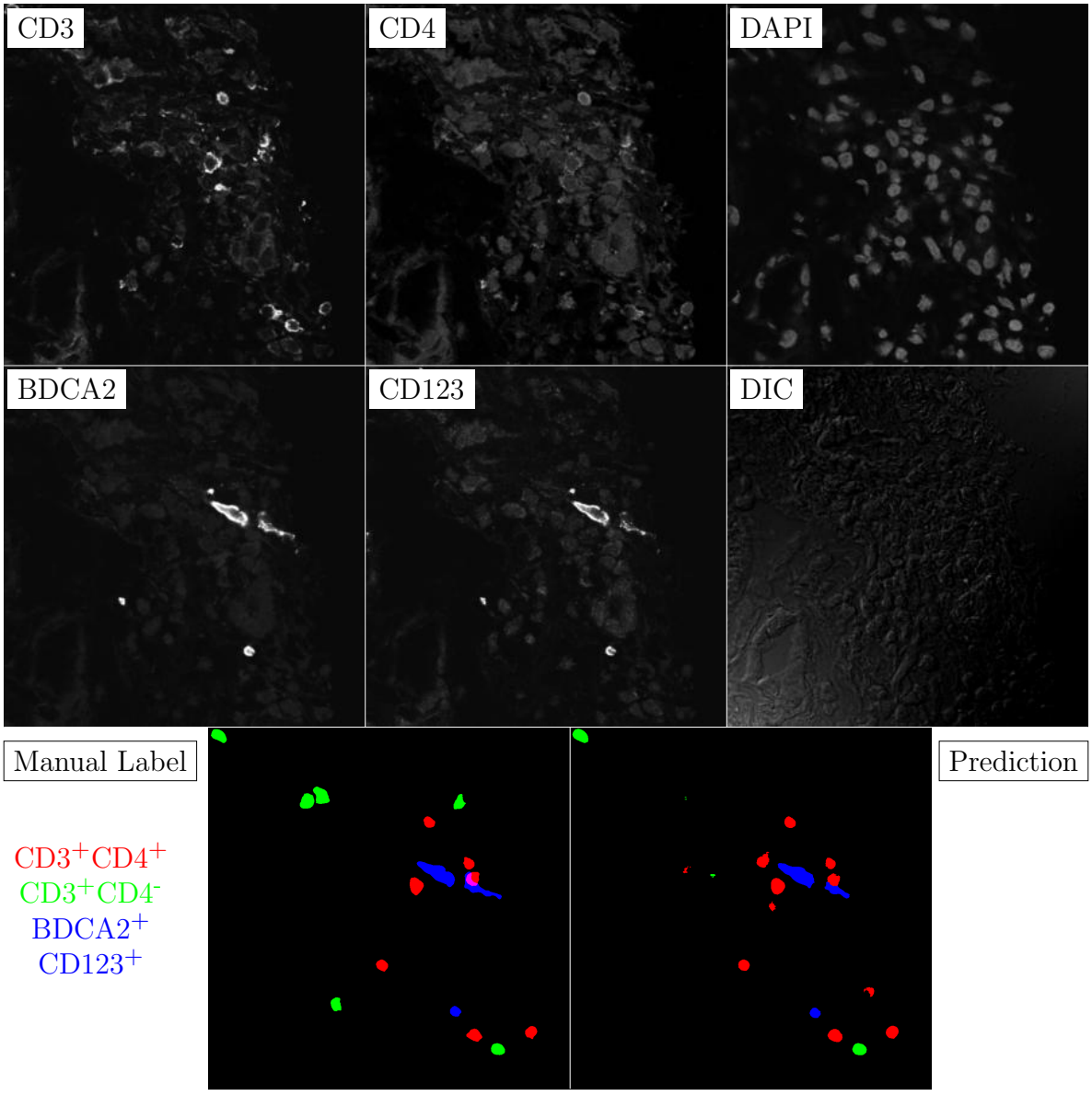

Figure G.51: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

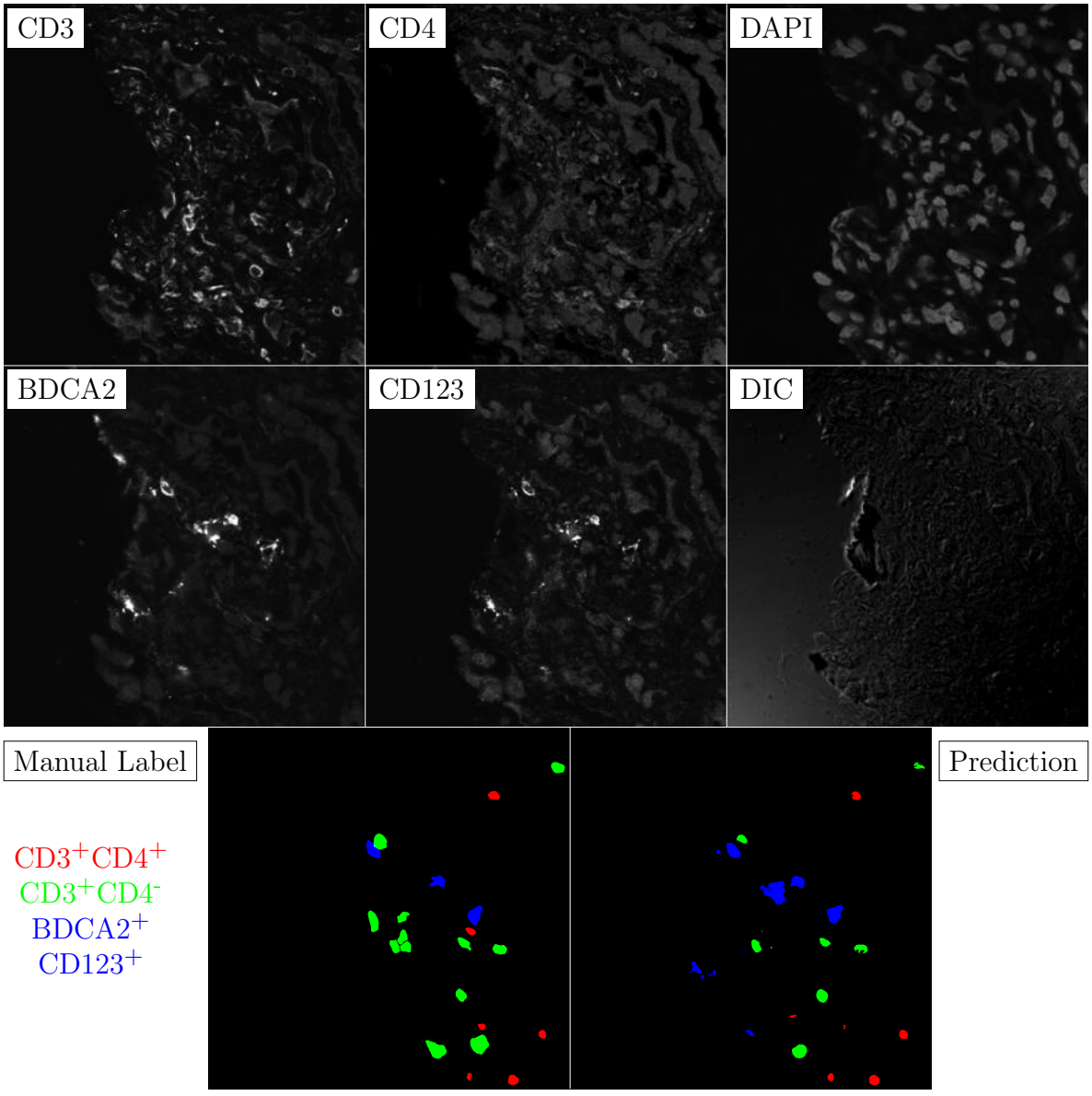

Figure G.52: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

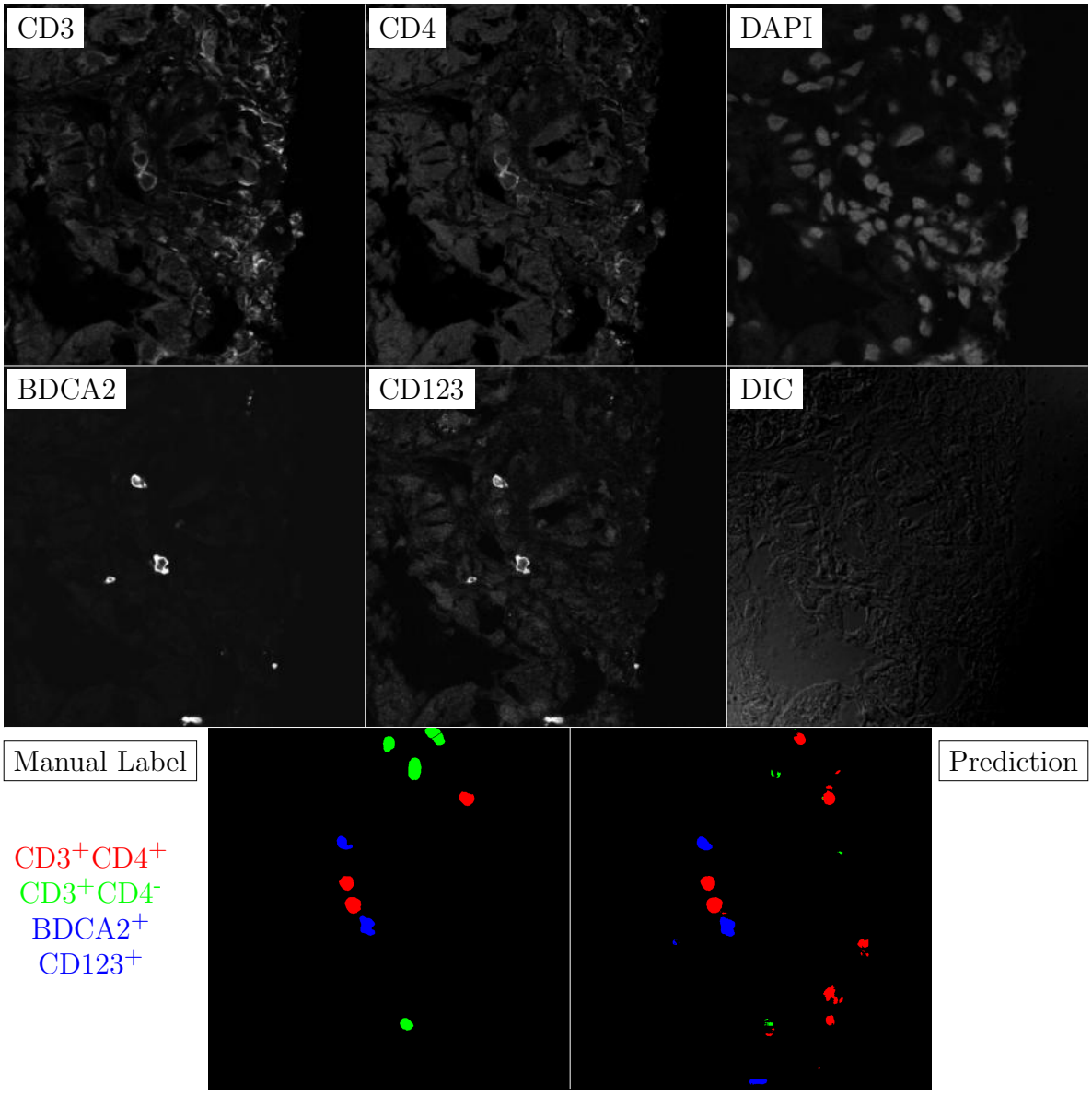

Figure G.53: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

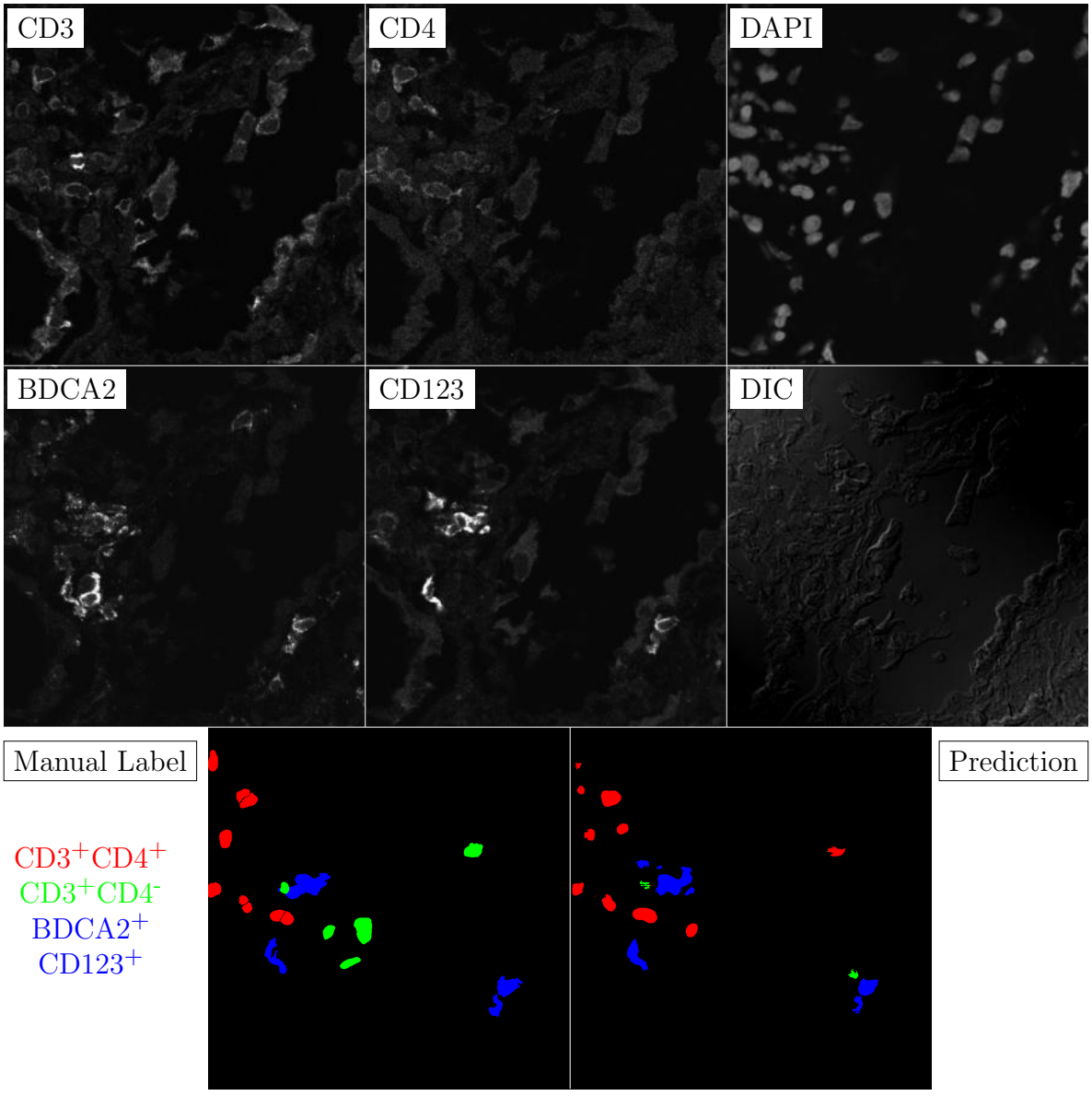

Figure G.54: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

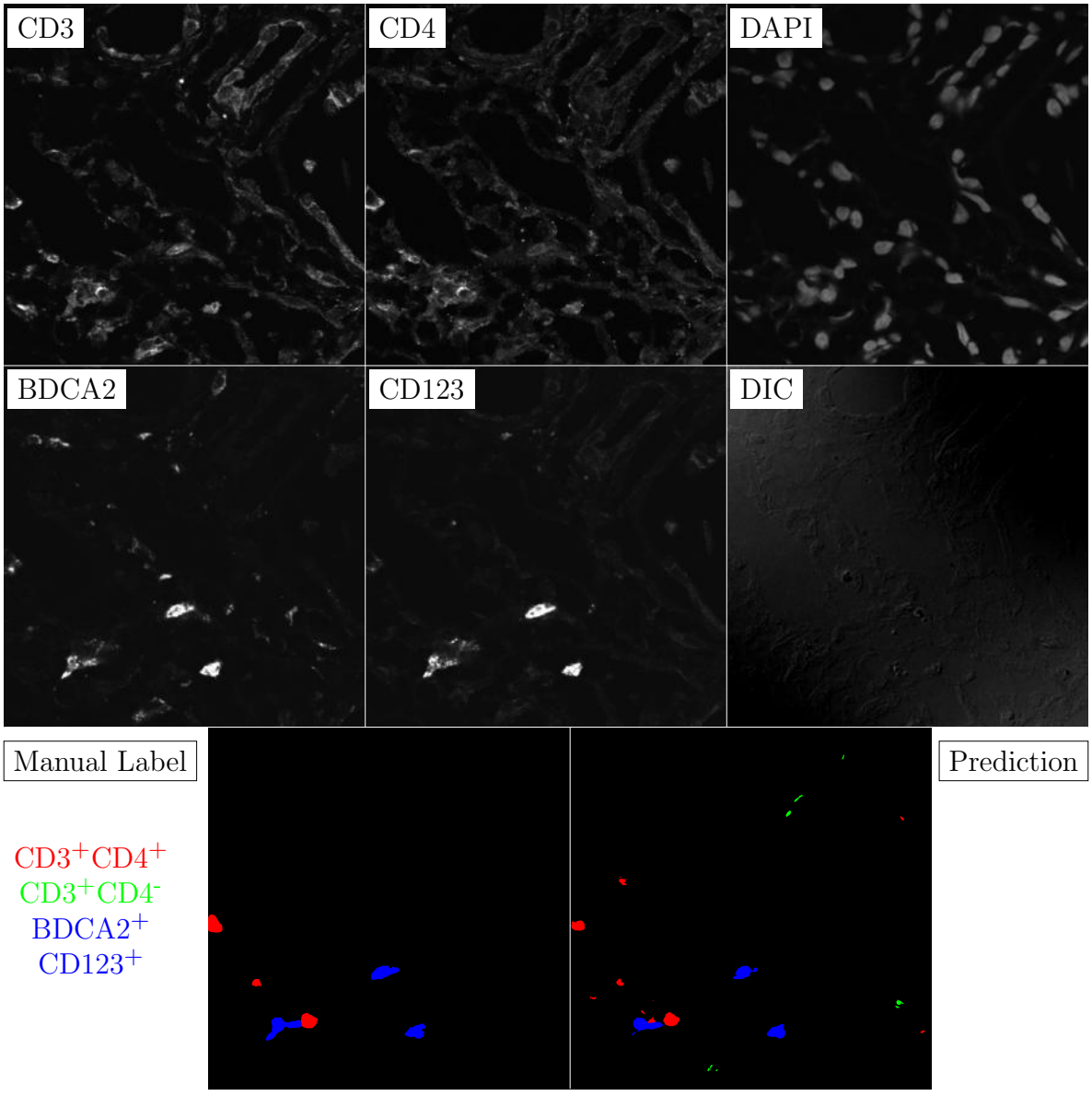

Figure G.55: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

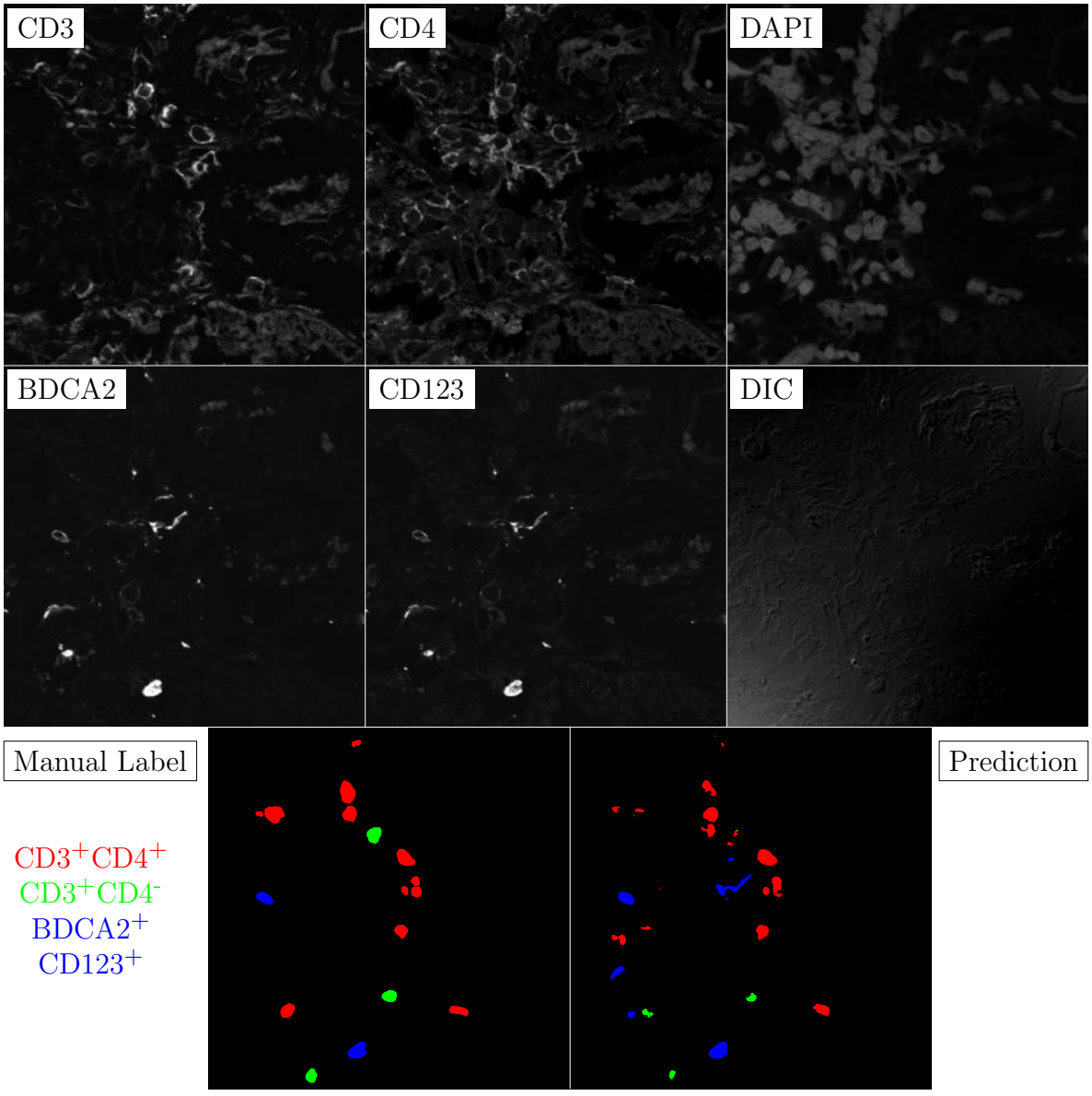

Figure G.56: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

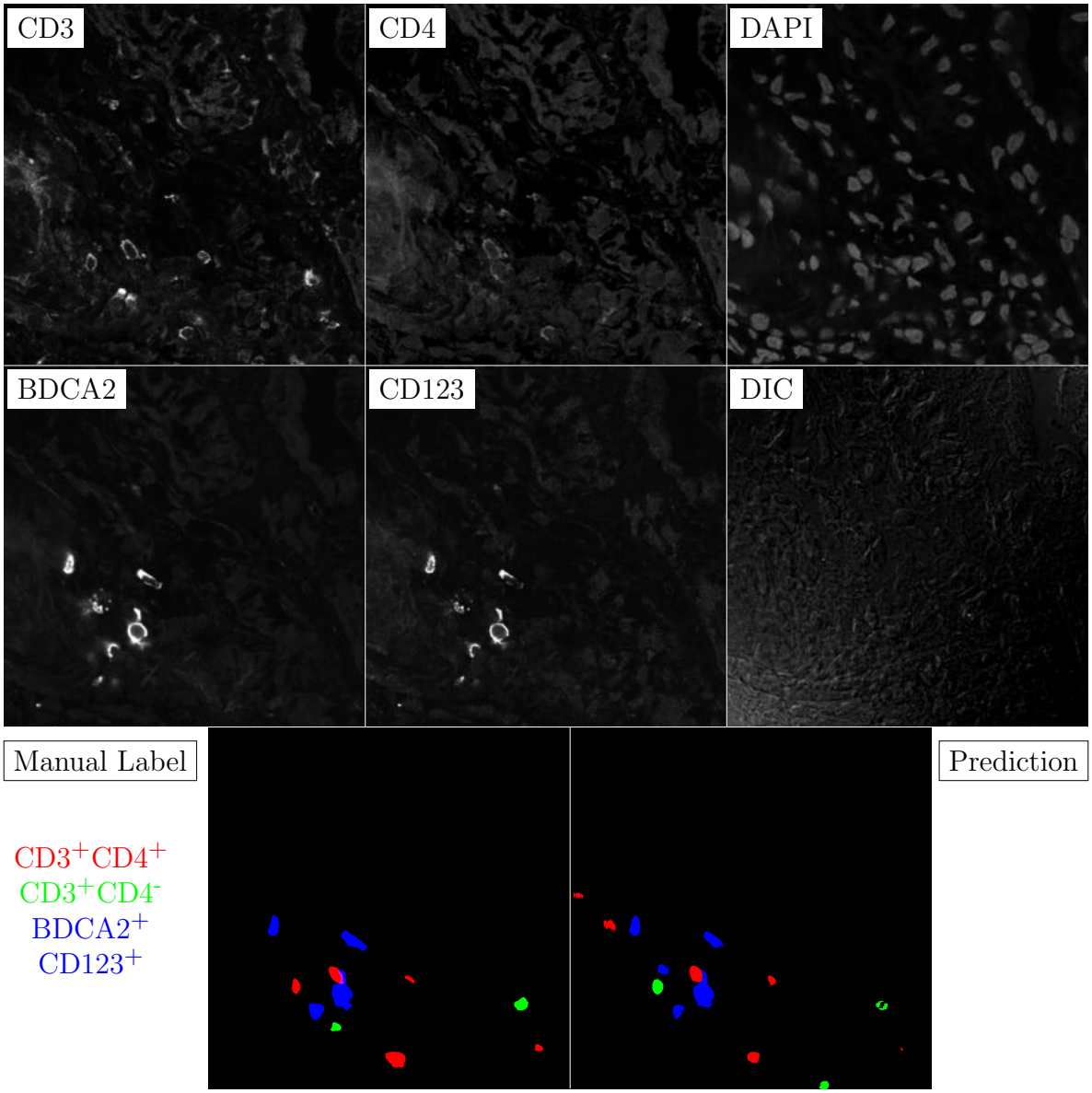

Figure G.57: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

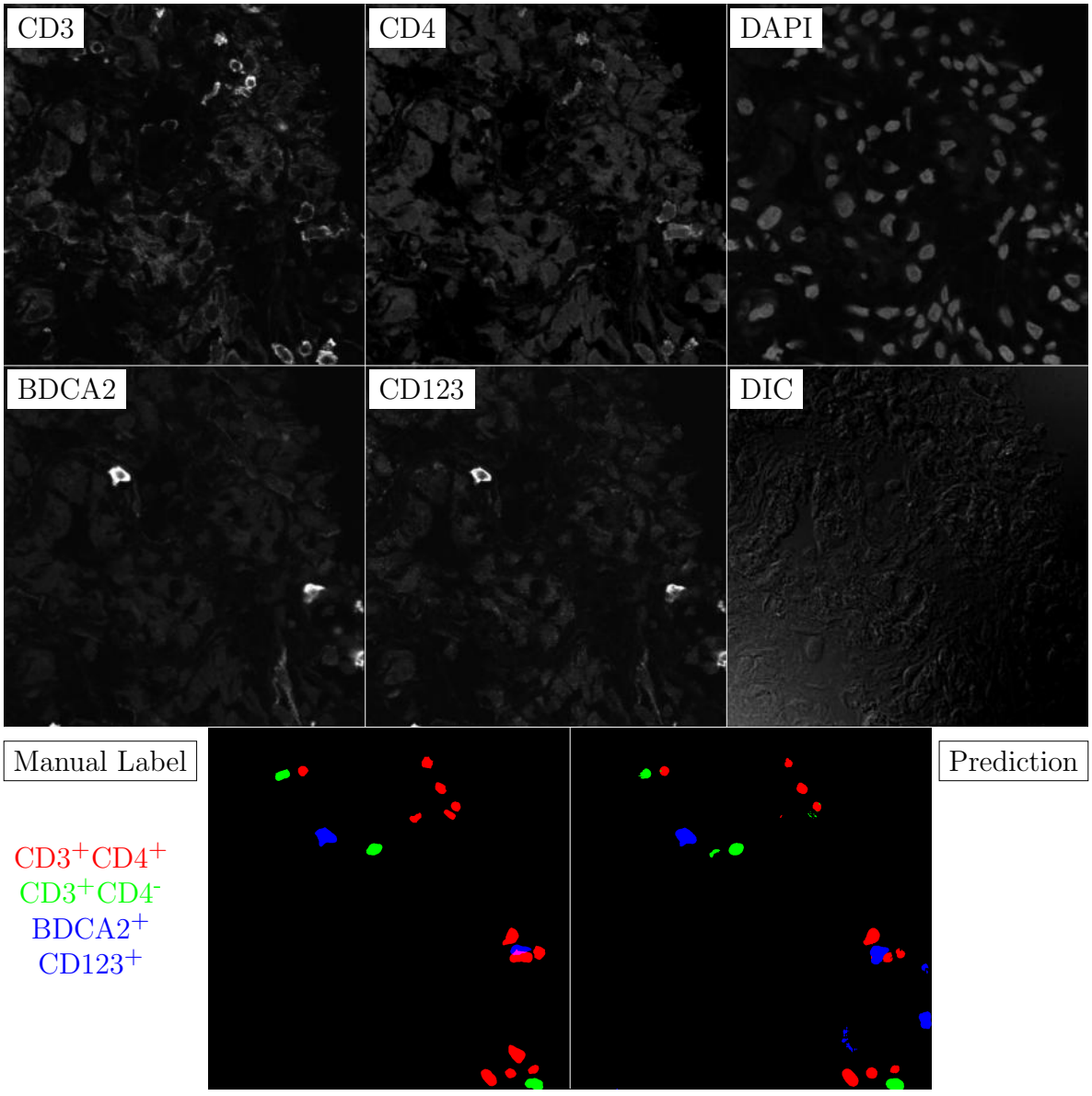

Figure G.58: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

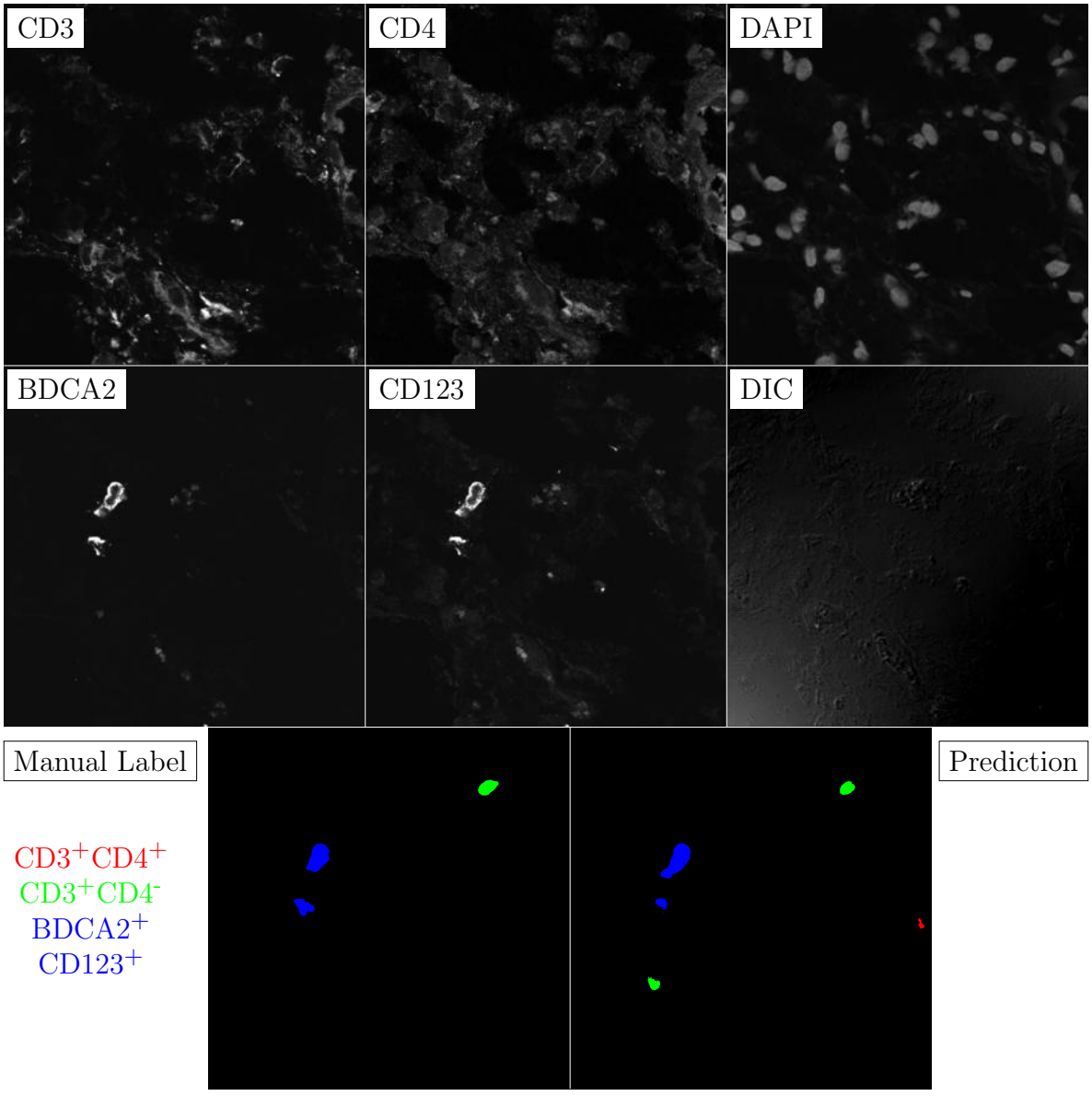

Figure G.59: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

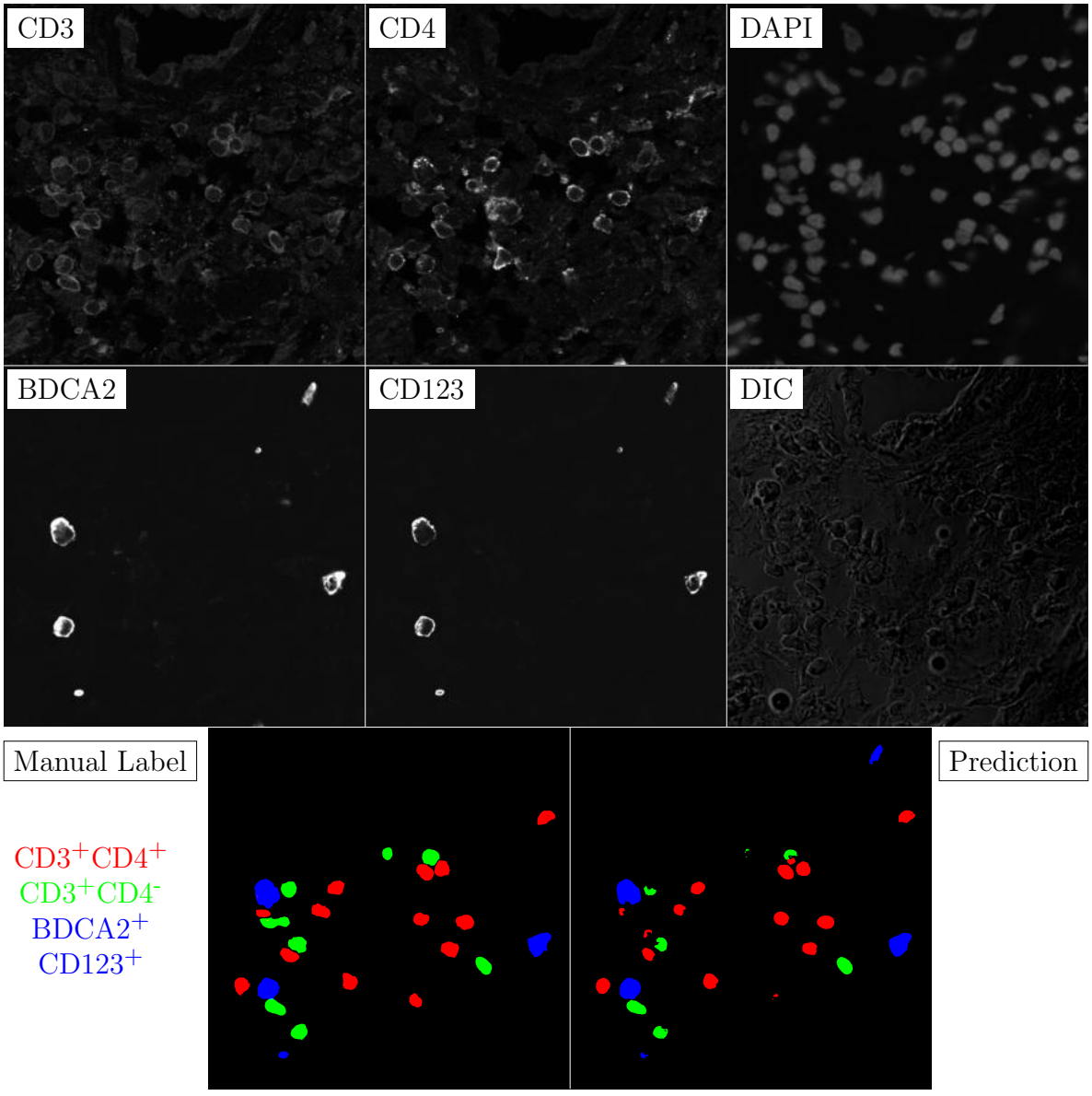

Figure G.60: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

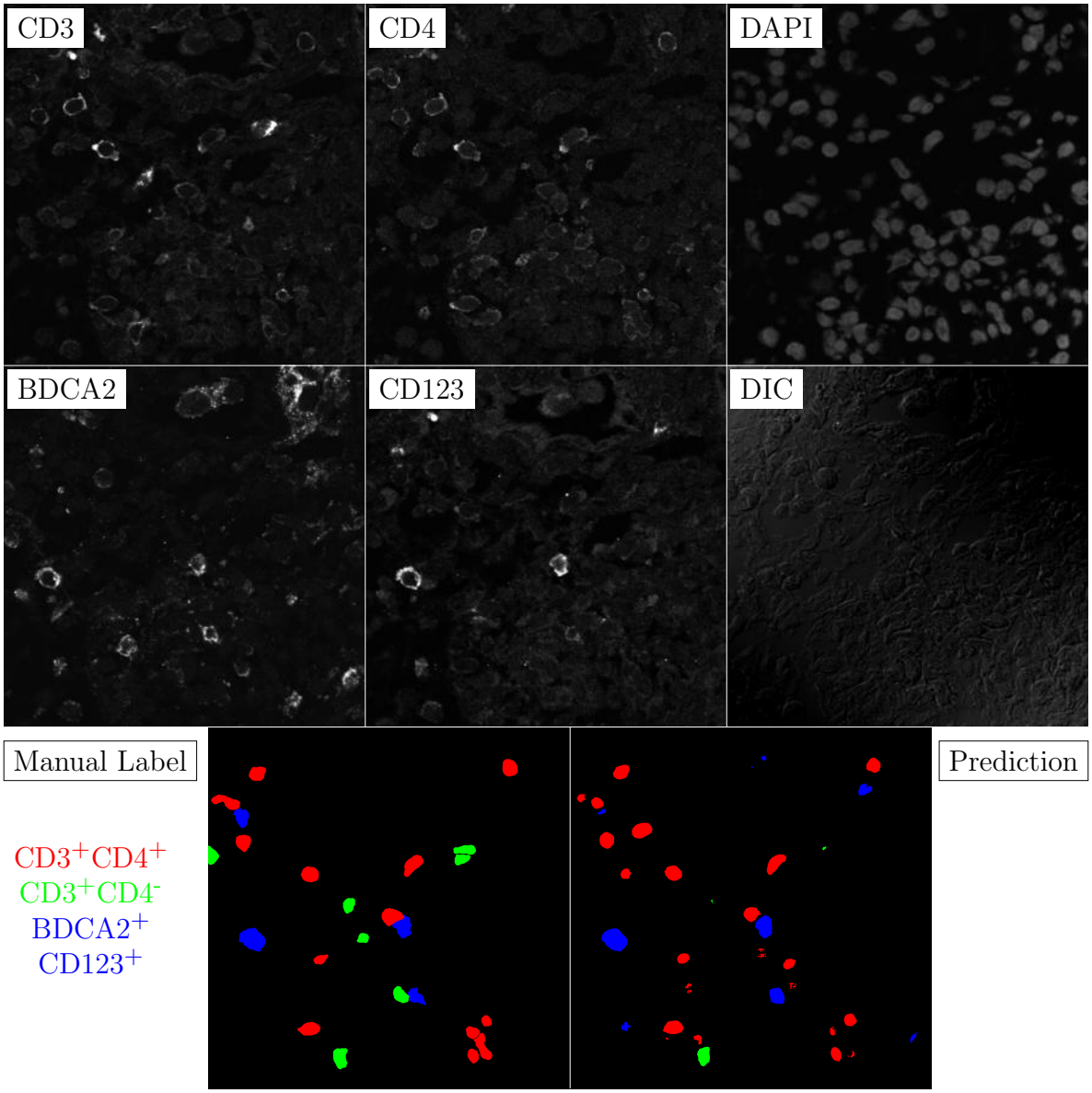

Figure G.61: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

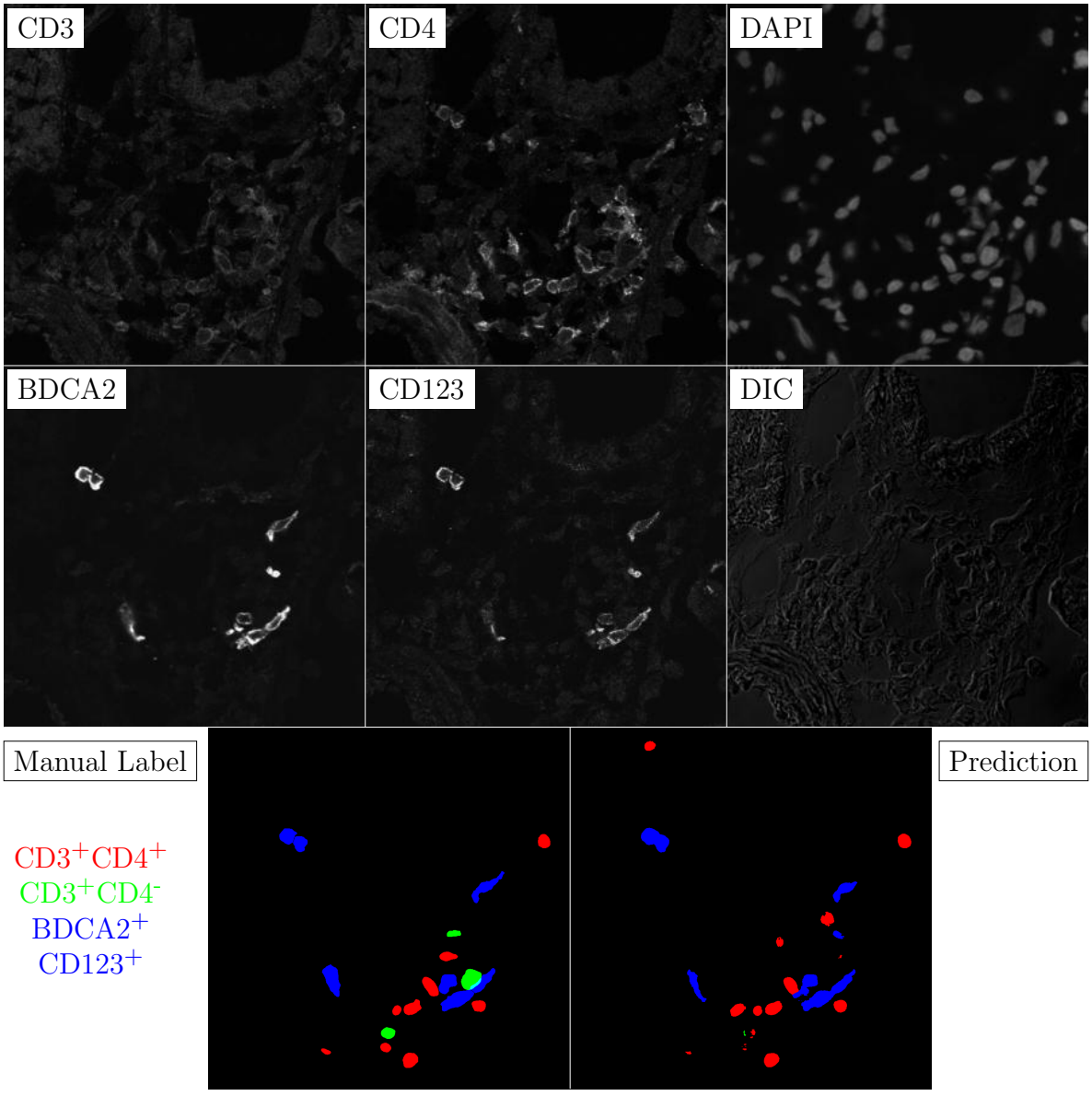

Figure G.62: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

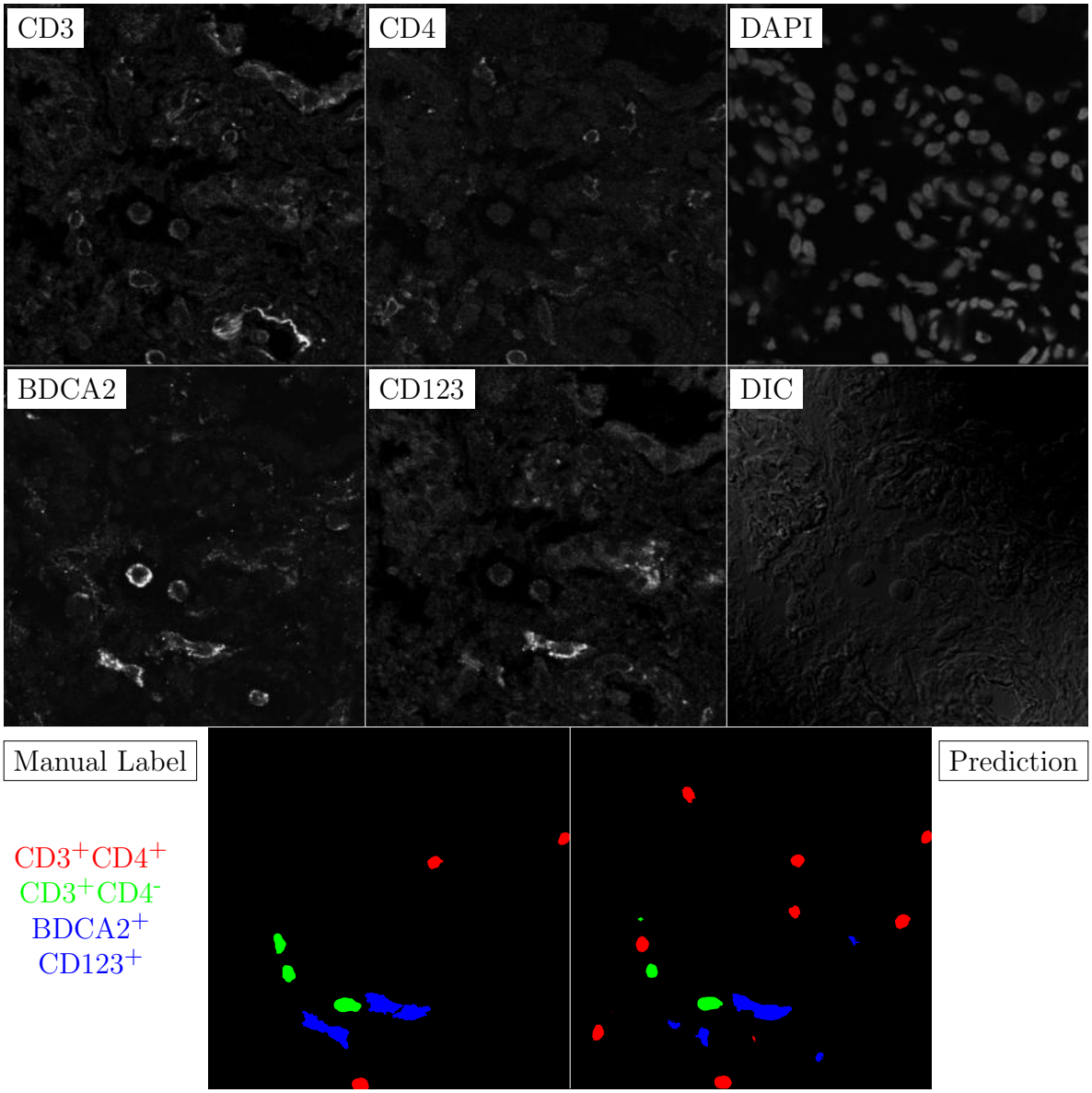

Figure G.63: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

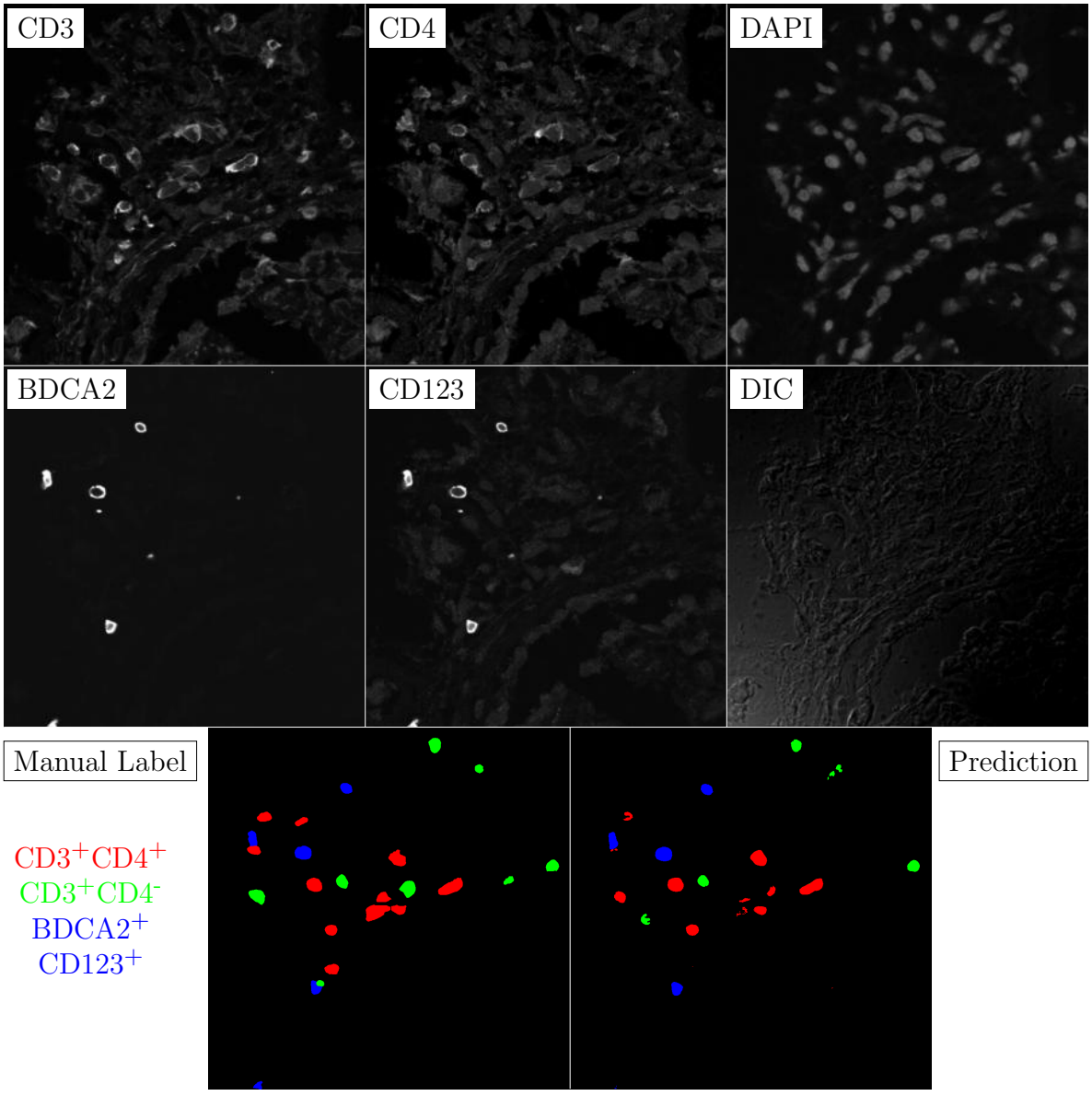

Figure G.64: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

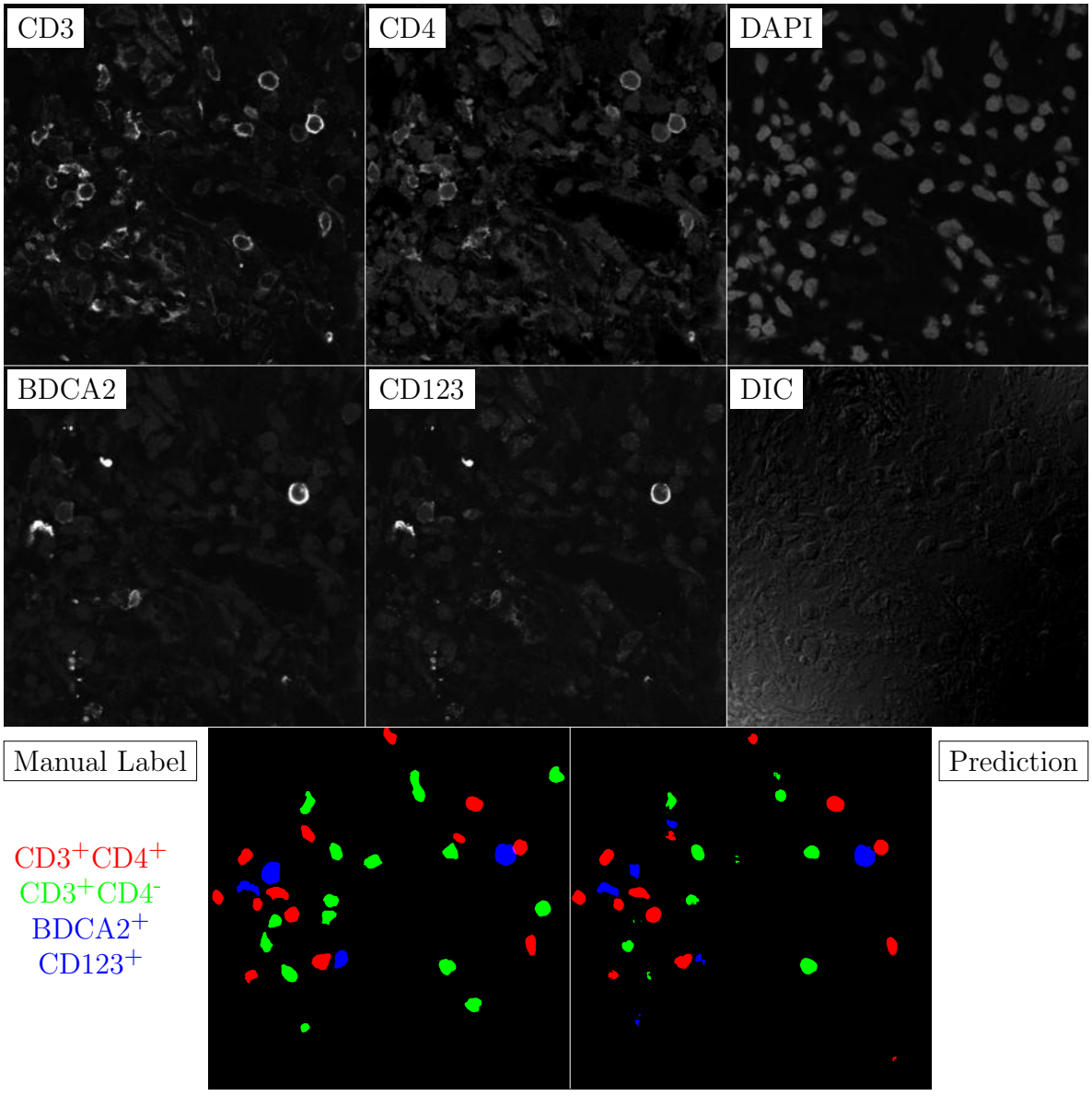

Figure G.65: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

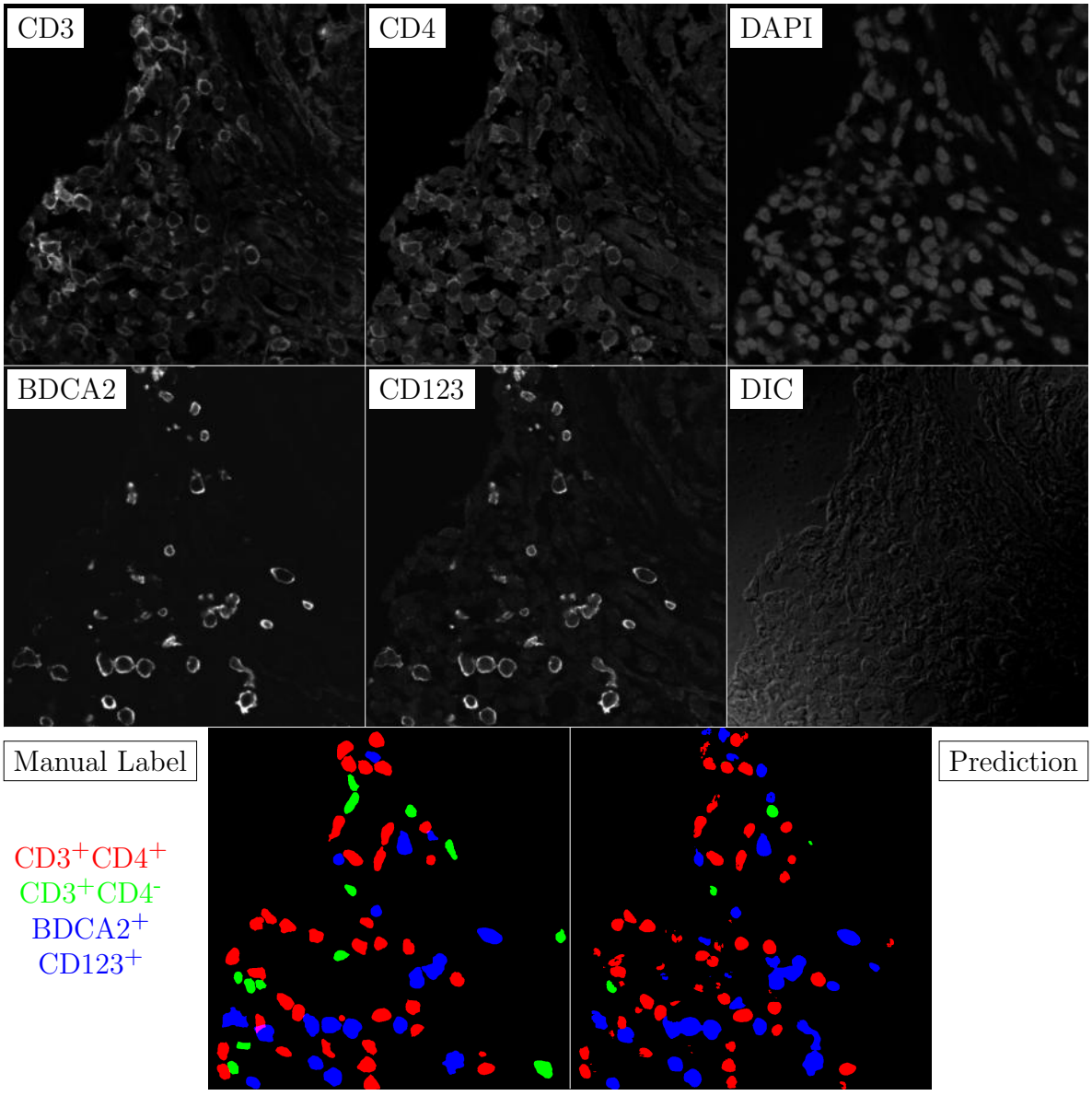

Figure G.66: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

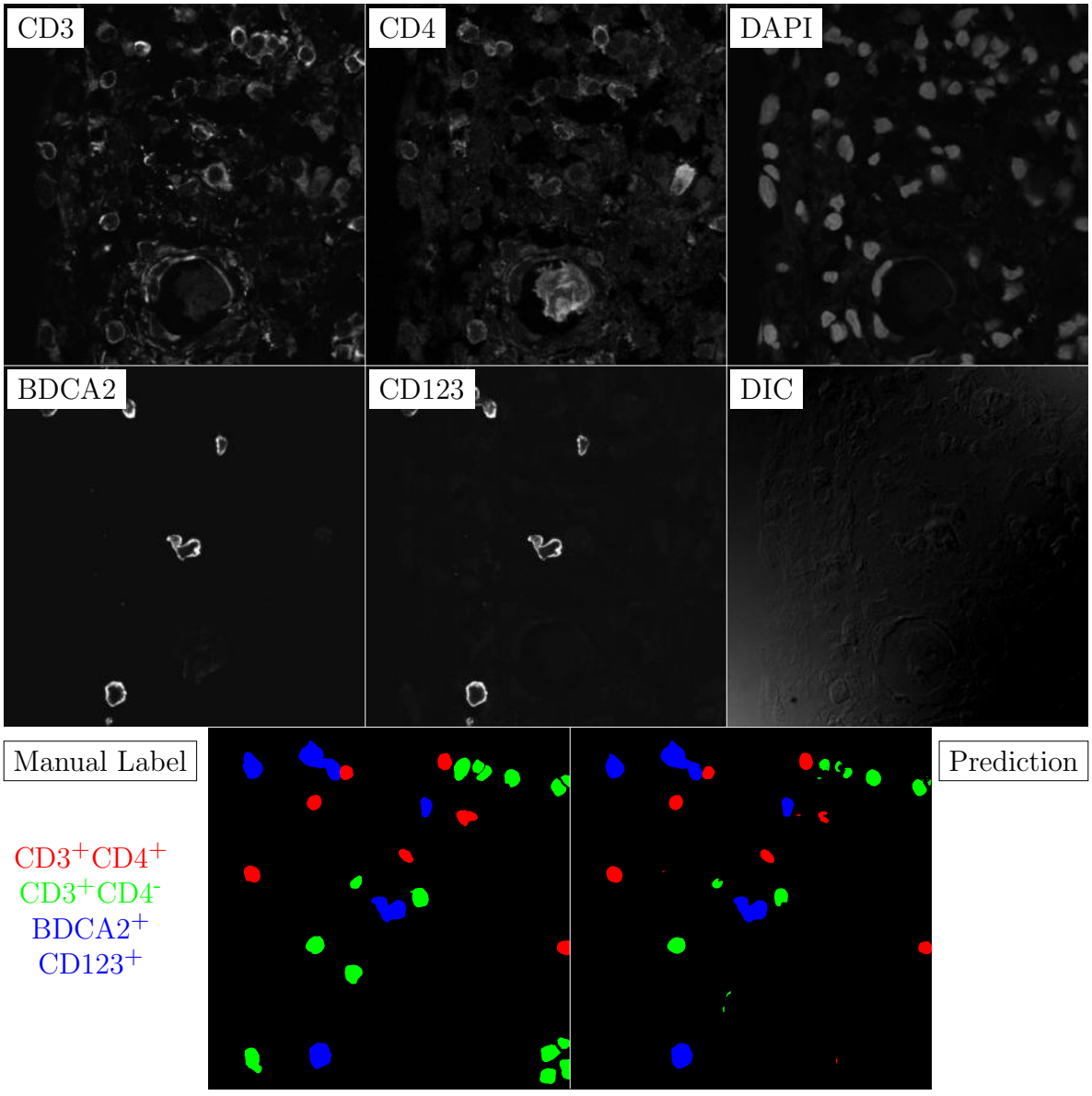

Figure G.67: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

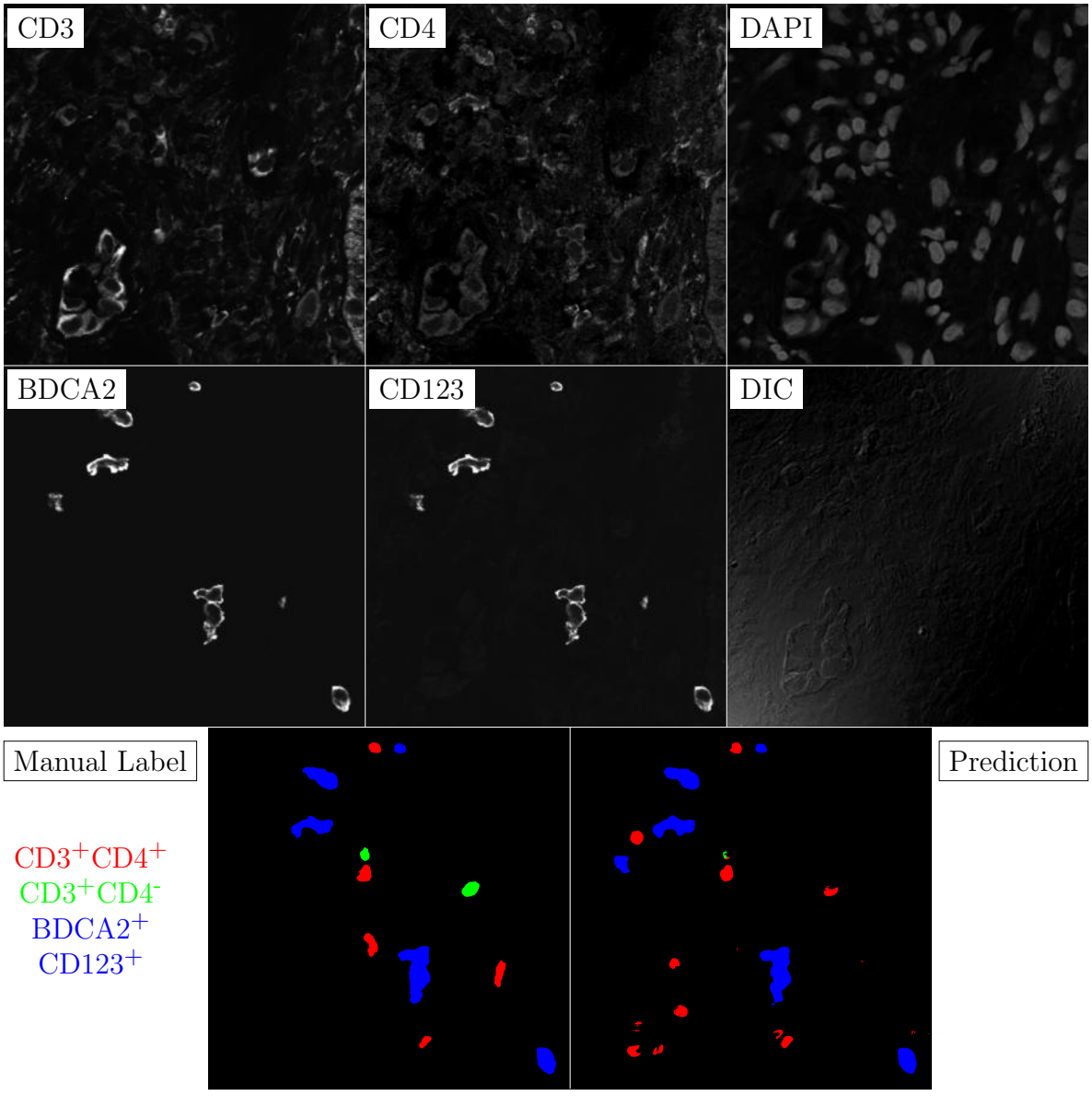

Figure G.68: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

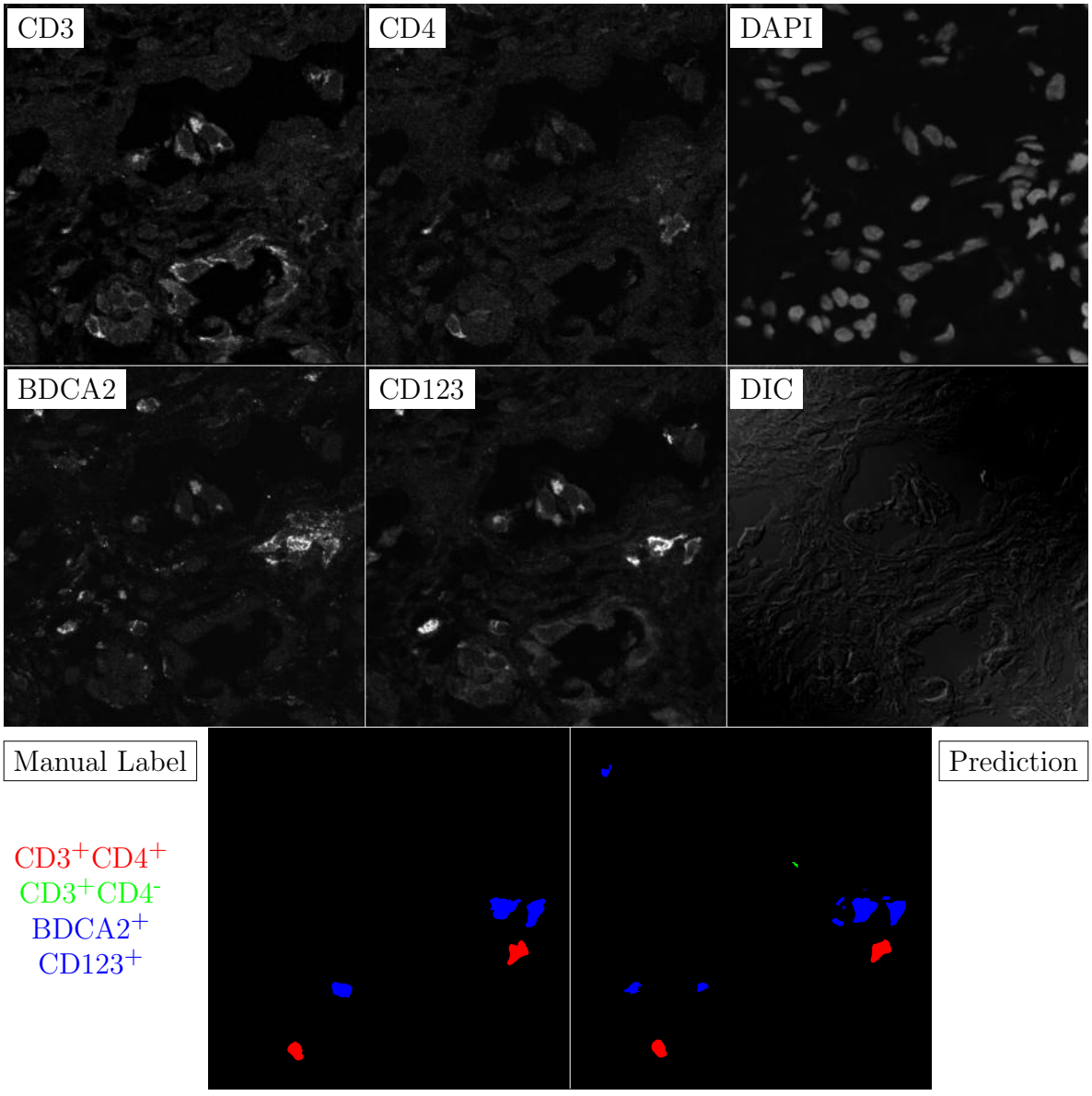

Figure G.69: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

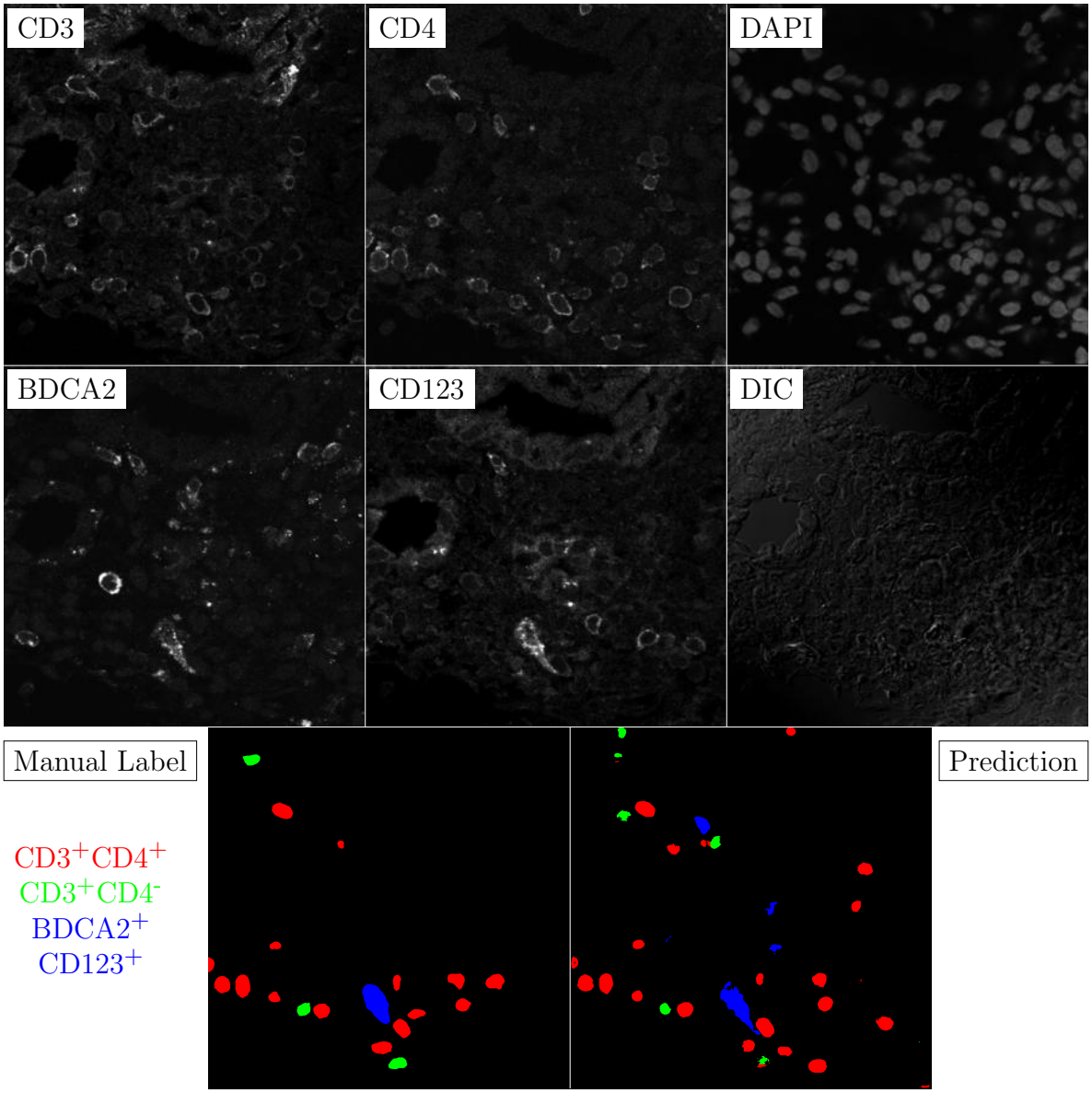

Figure G.70: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

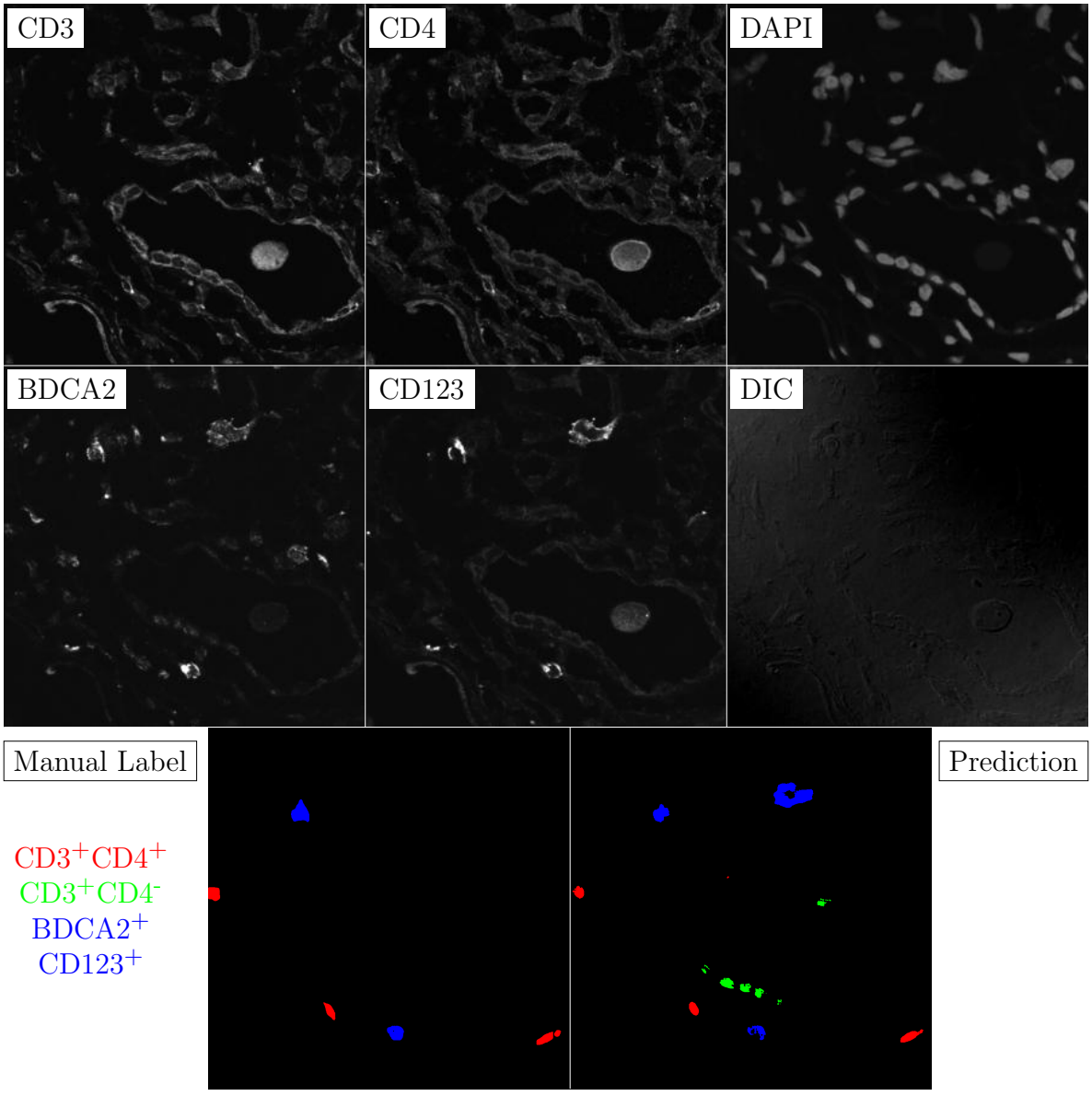

Figure G.71: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

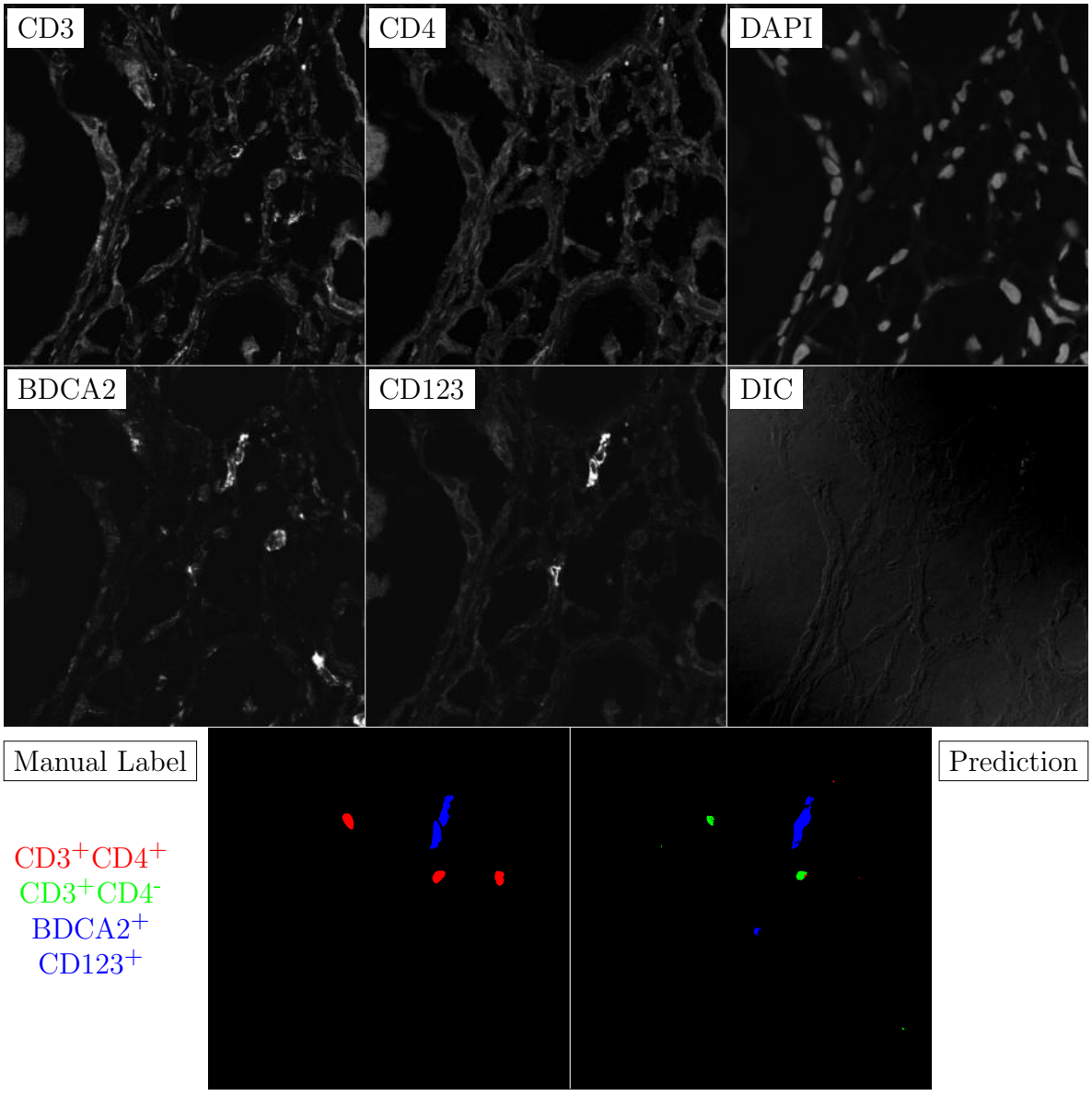

Figure G.72: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

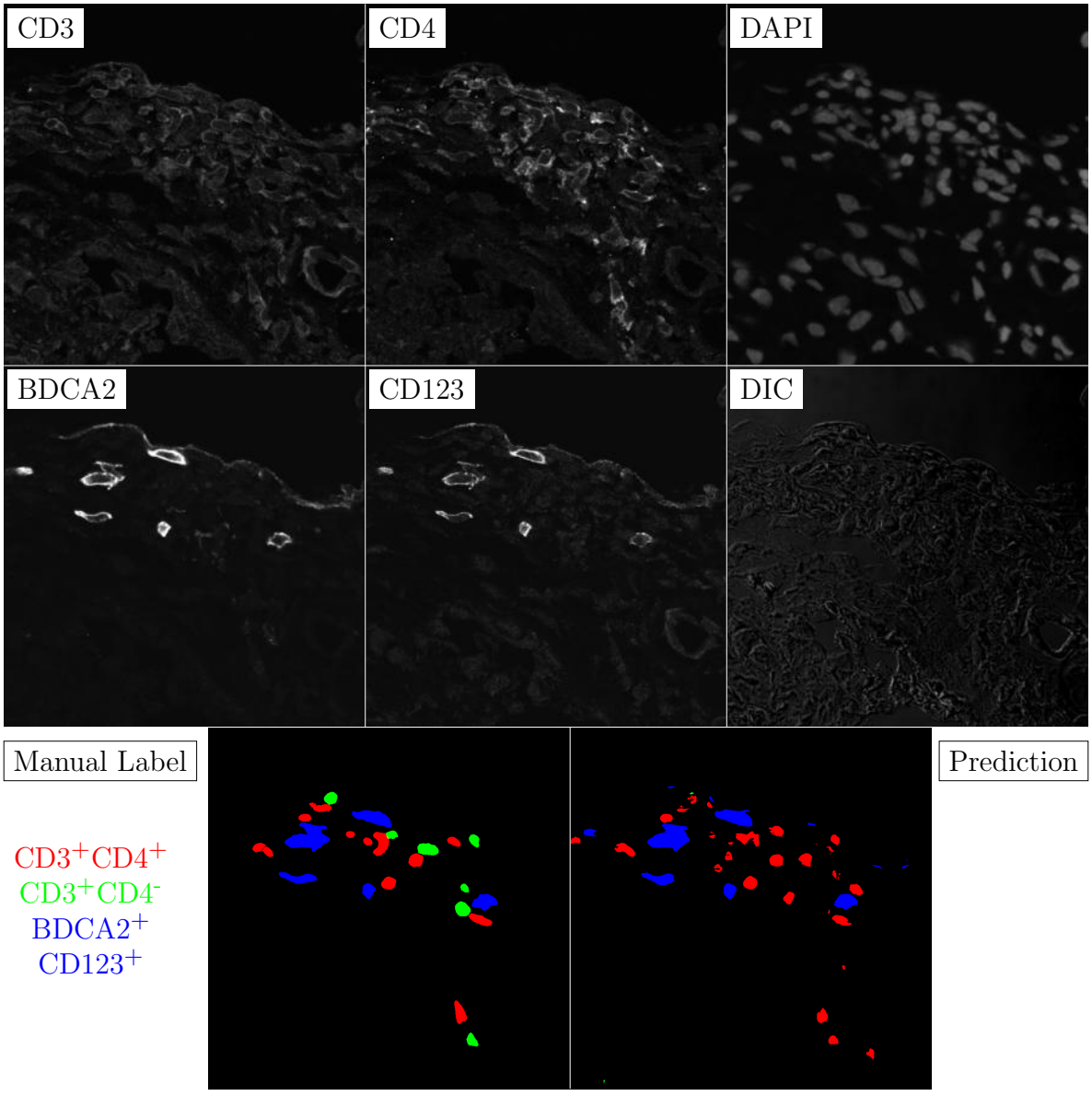

Figure G.73: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

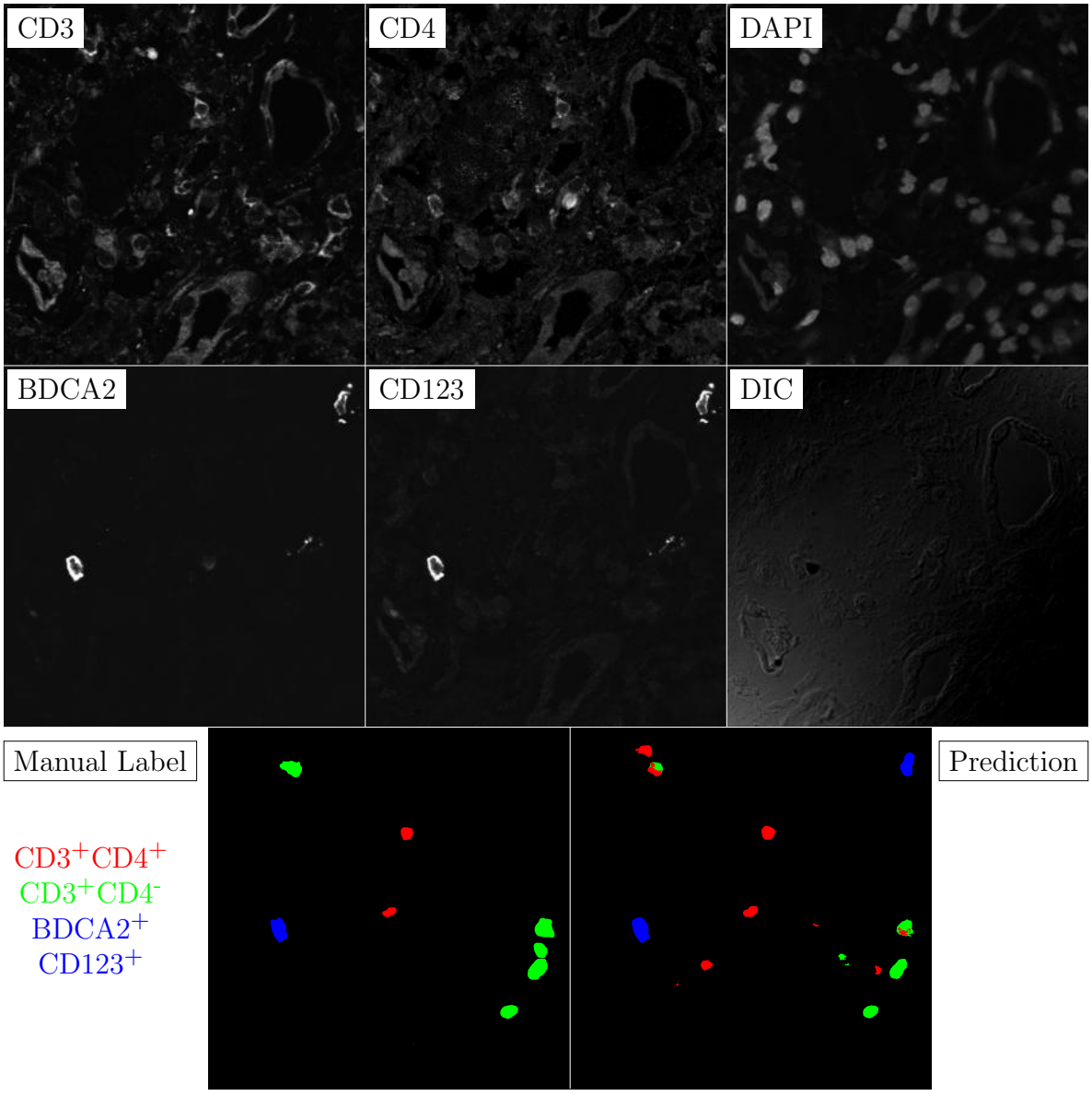

Figure G.74: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

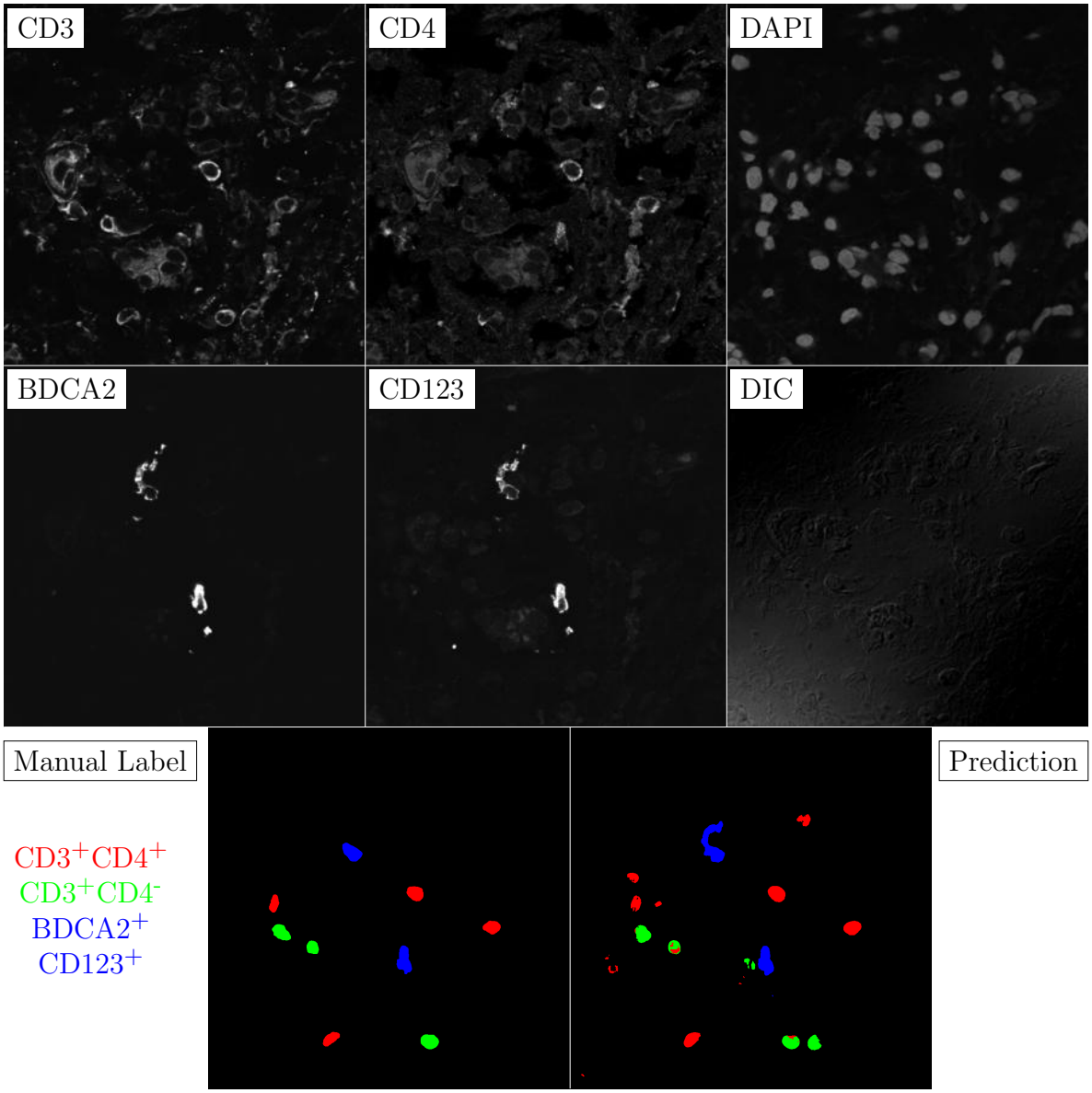

Figure G.75: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

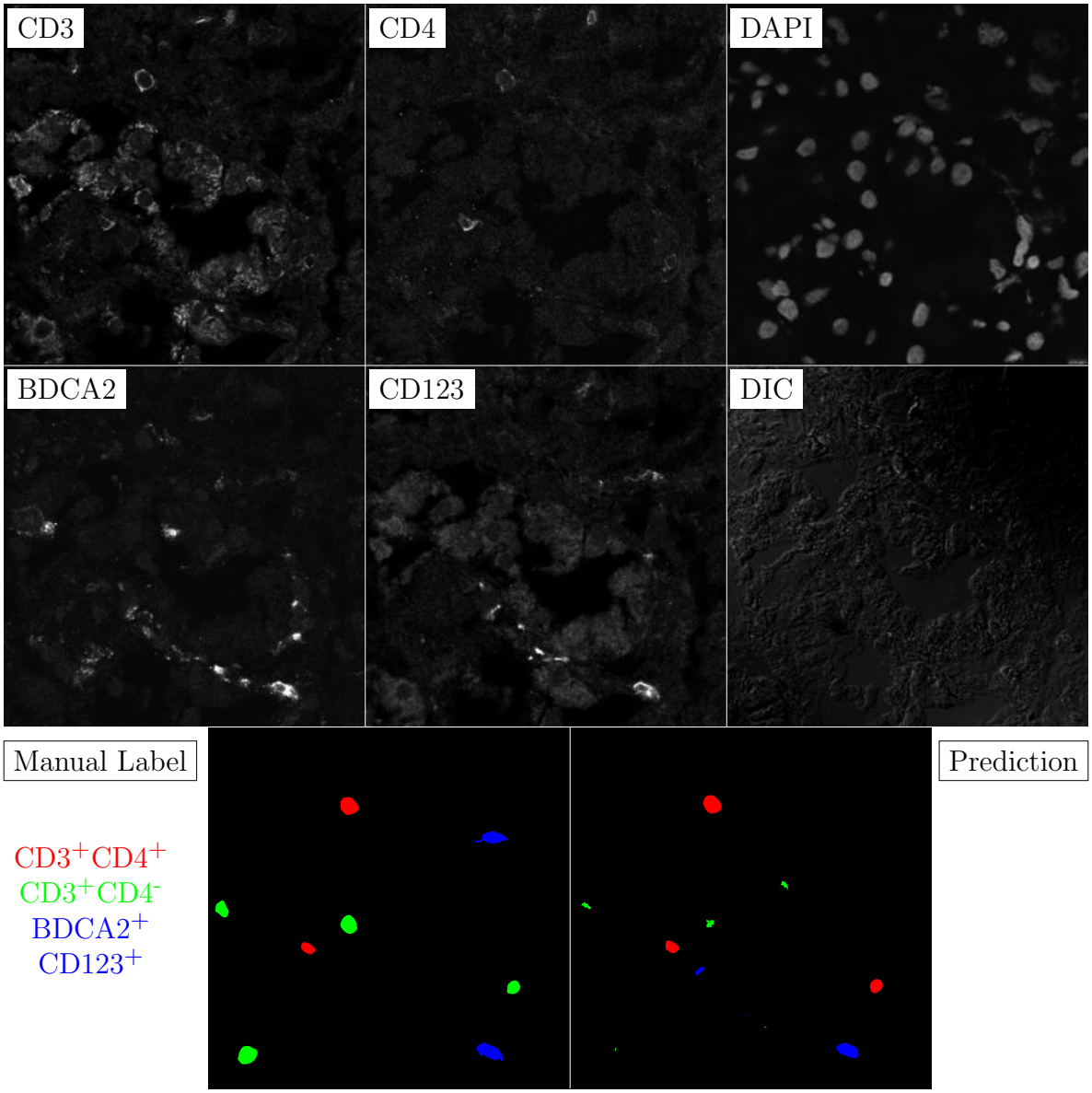

Figure G.76: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

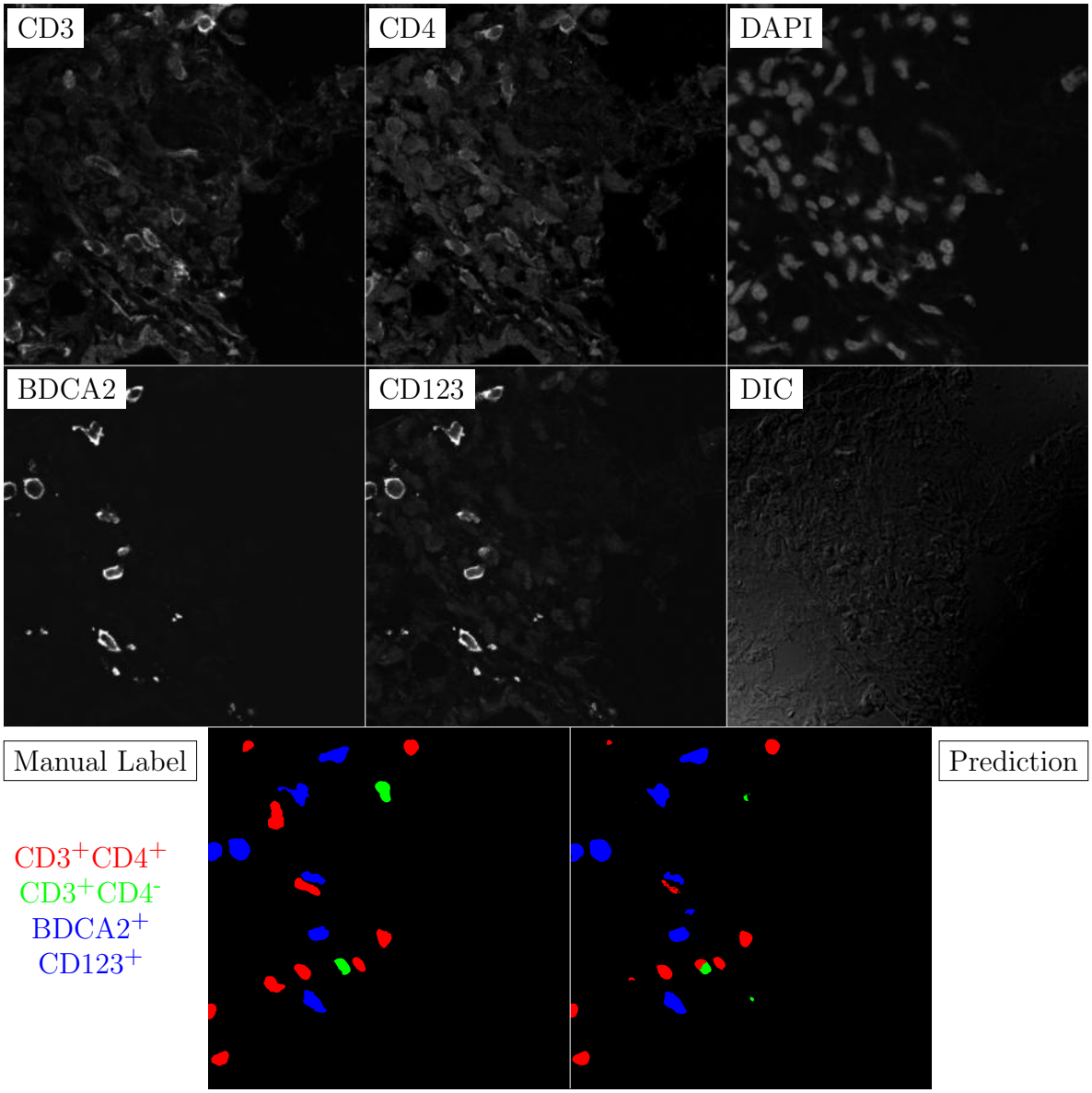

Figure G.77: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

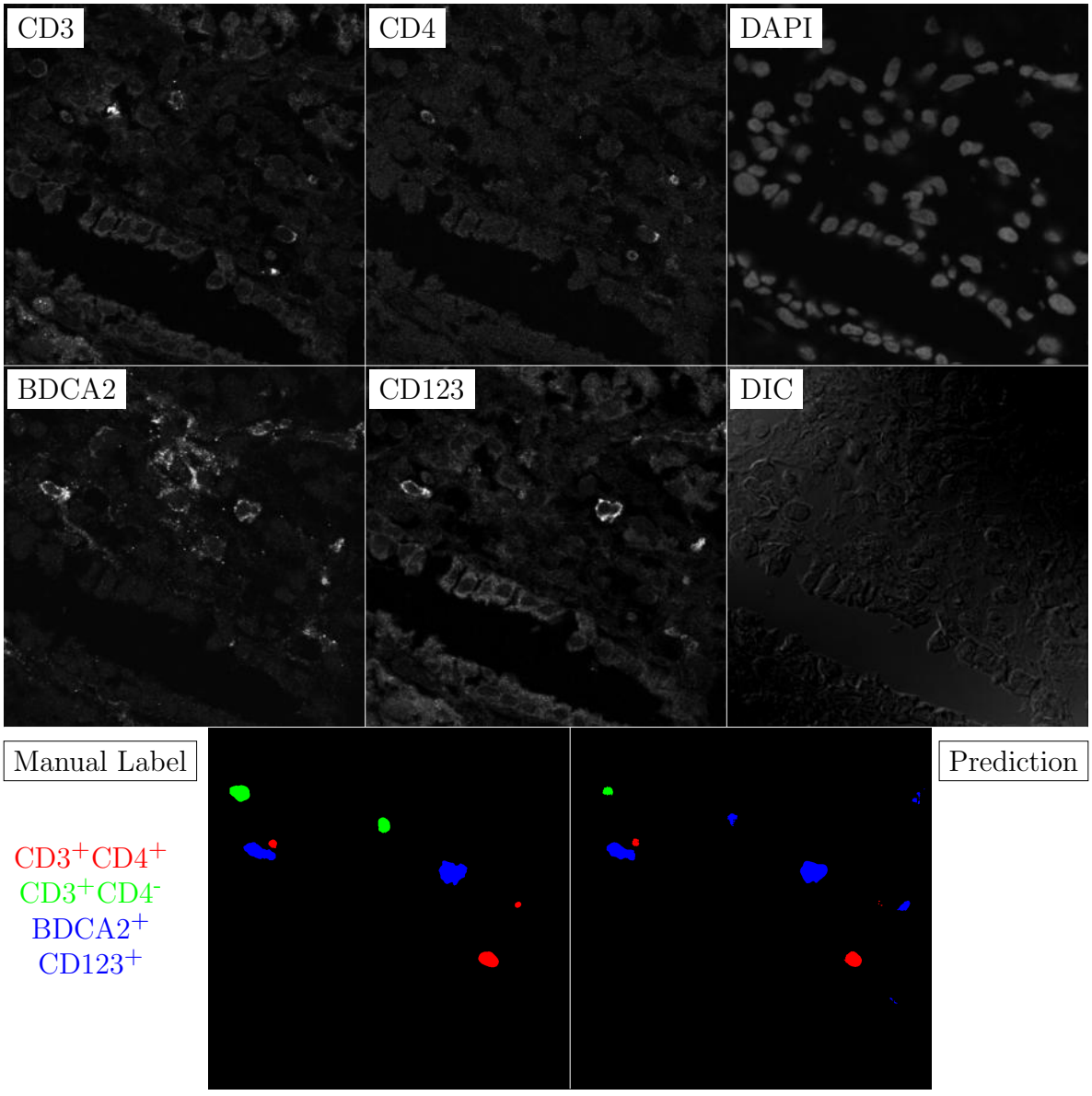

Figure G.78: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

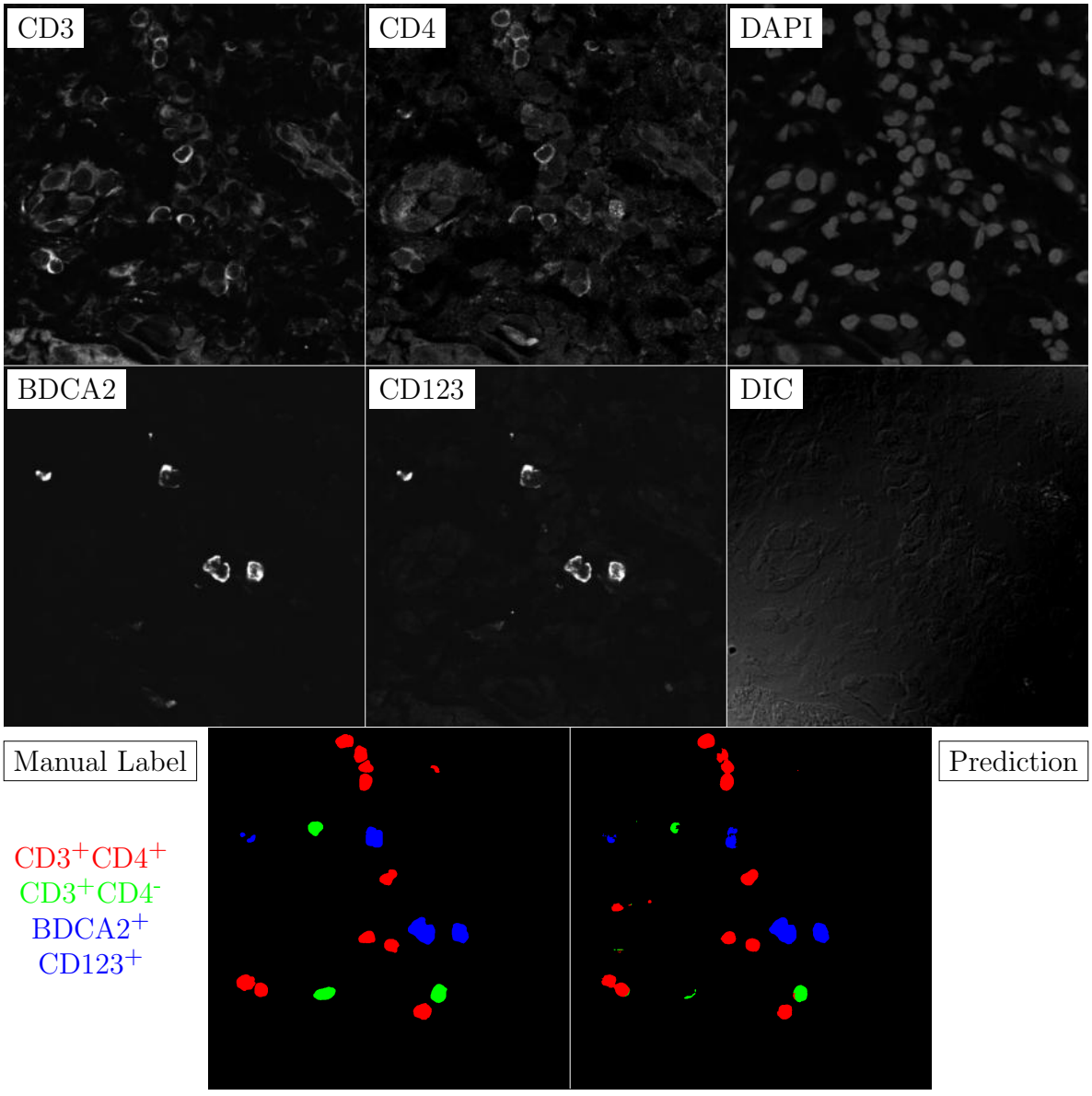

Figure G.79: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

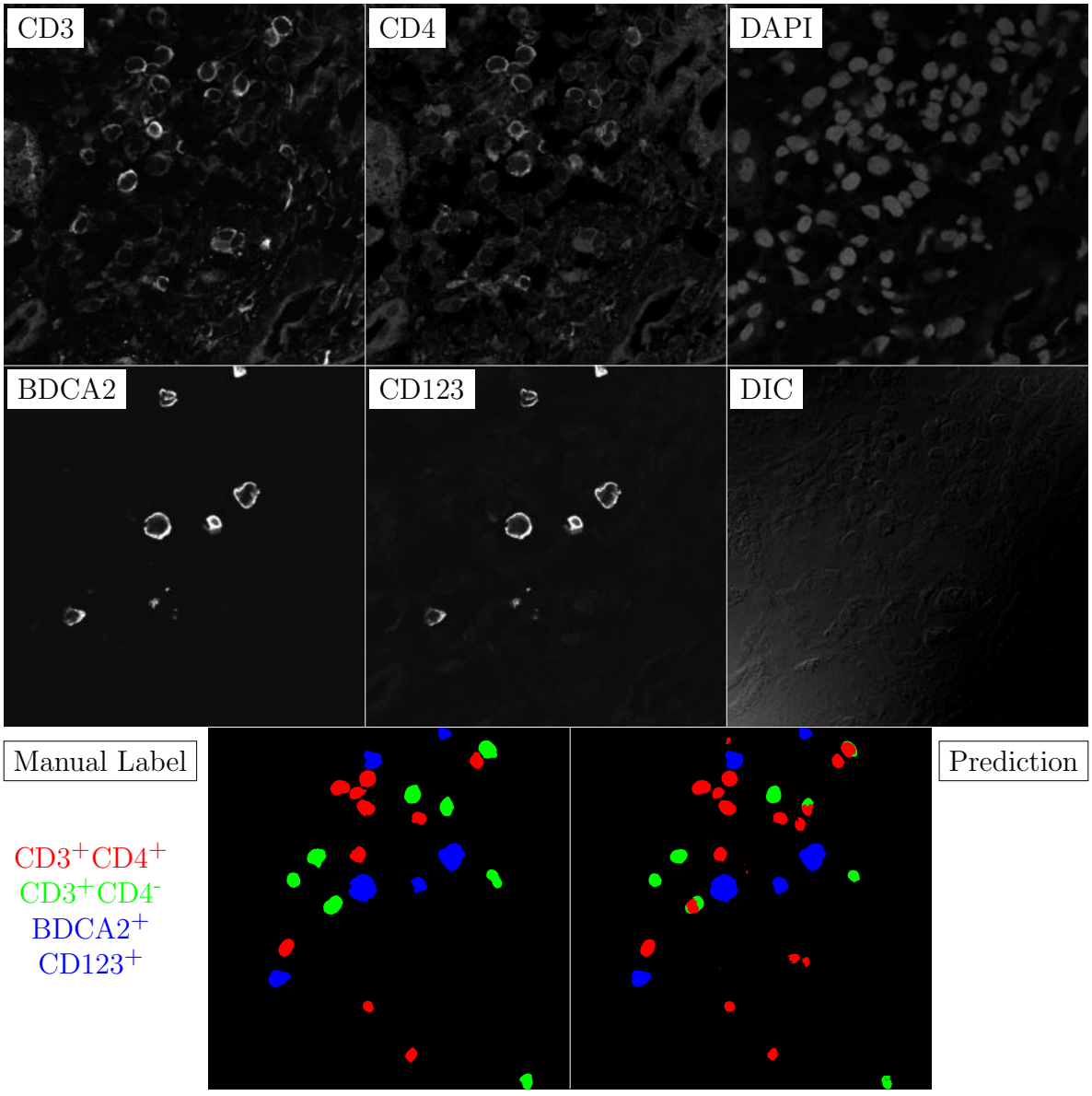

Figure G.80: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

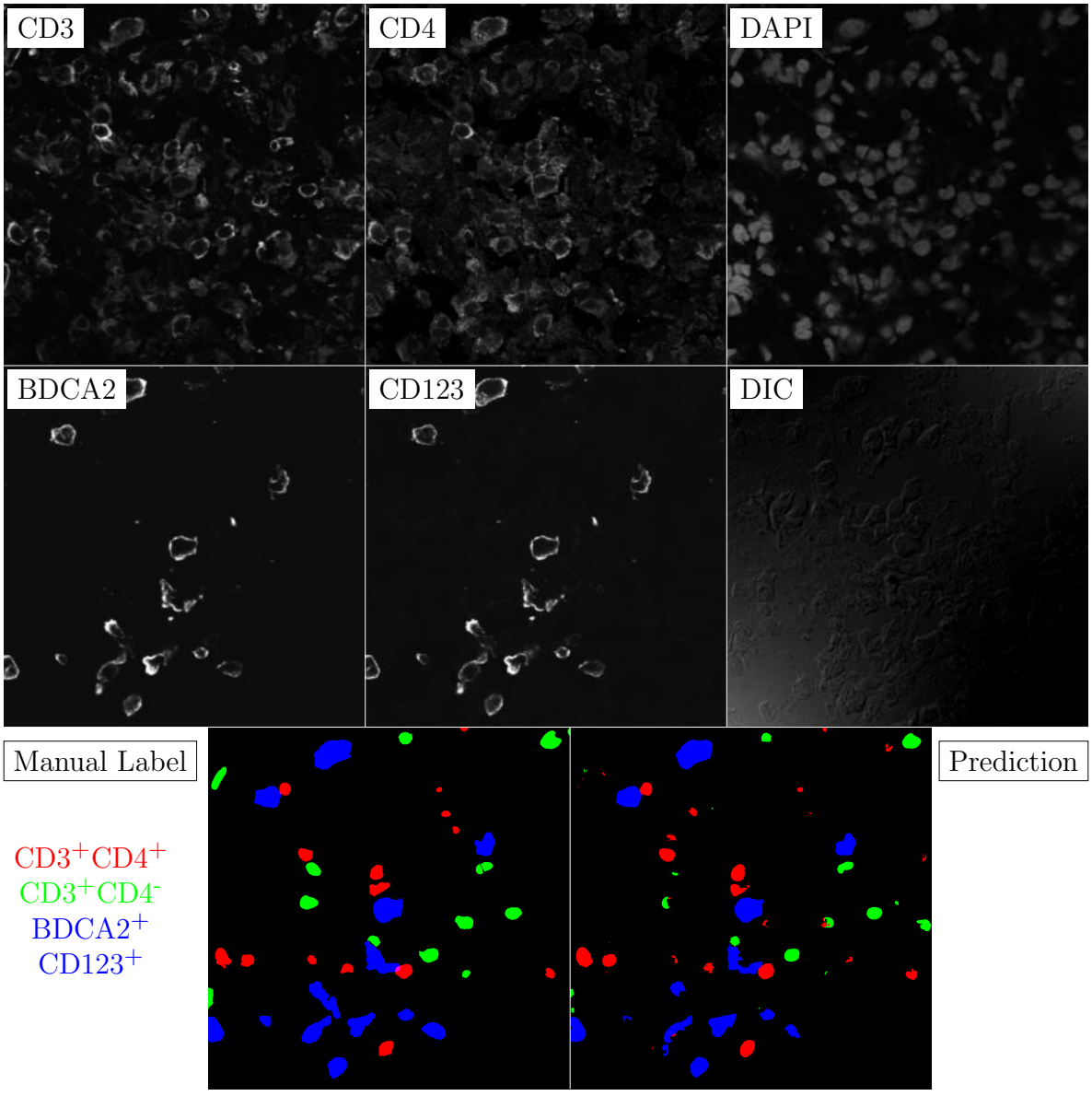

Figure G.81: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

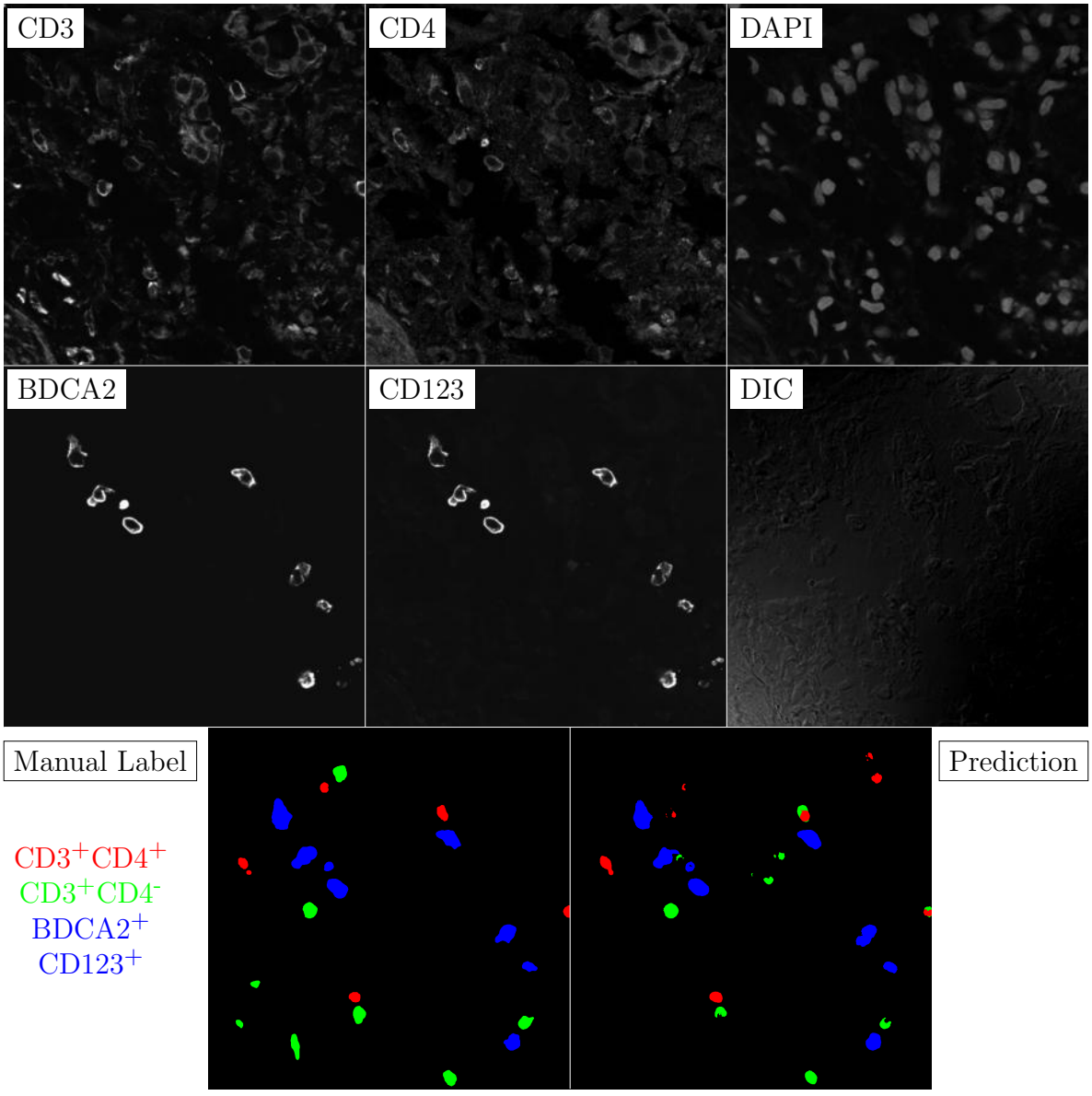

Figure G.82: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

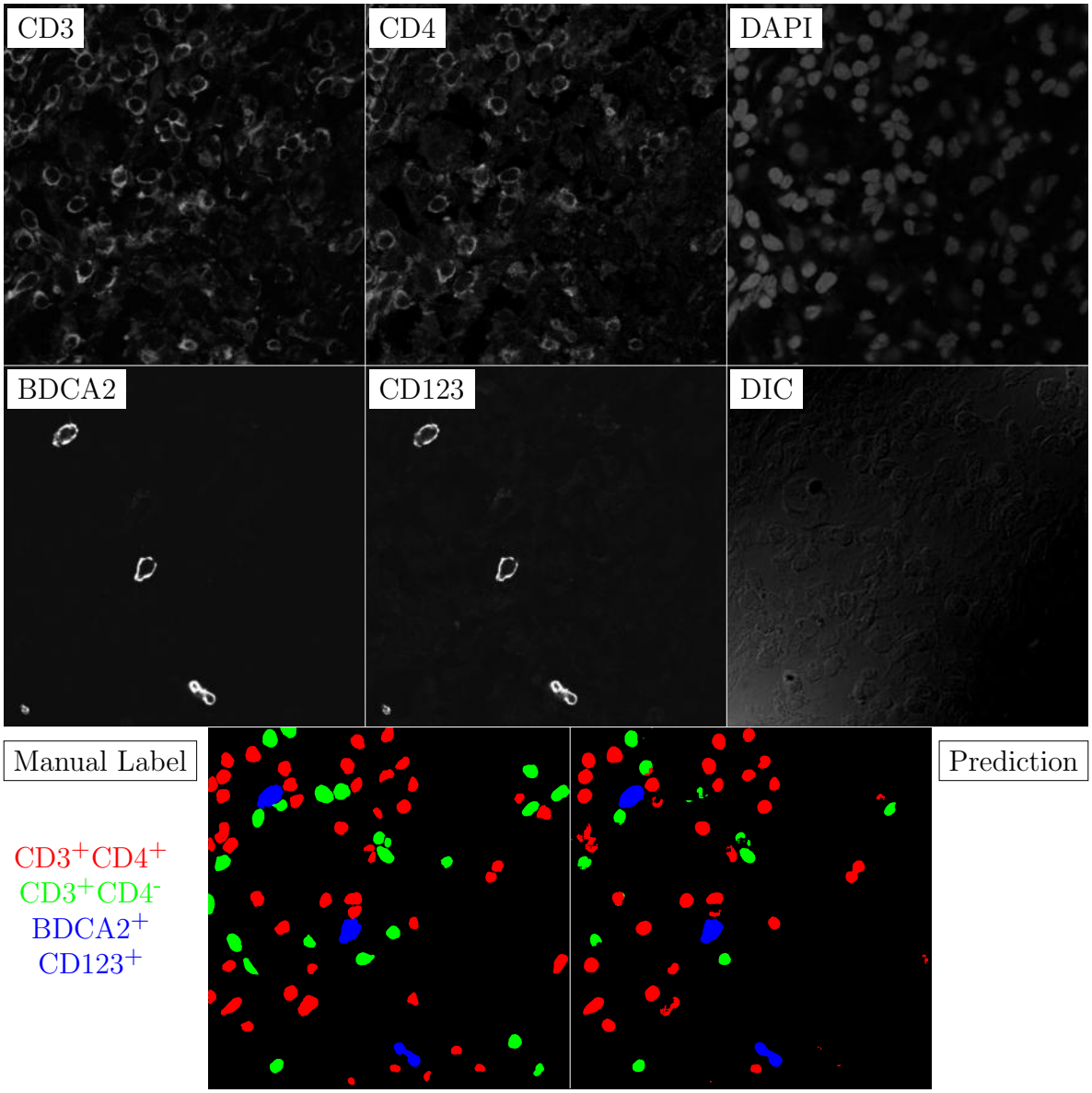

Figure G.83: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

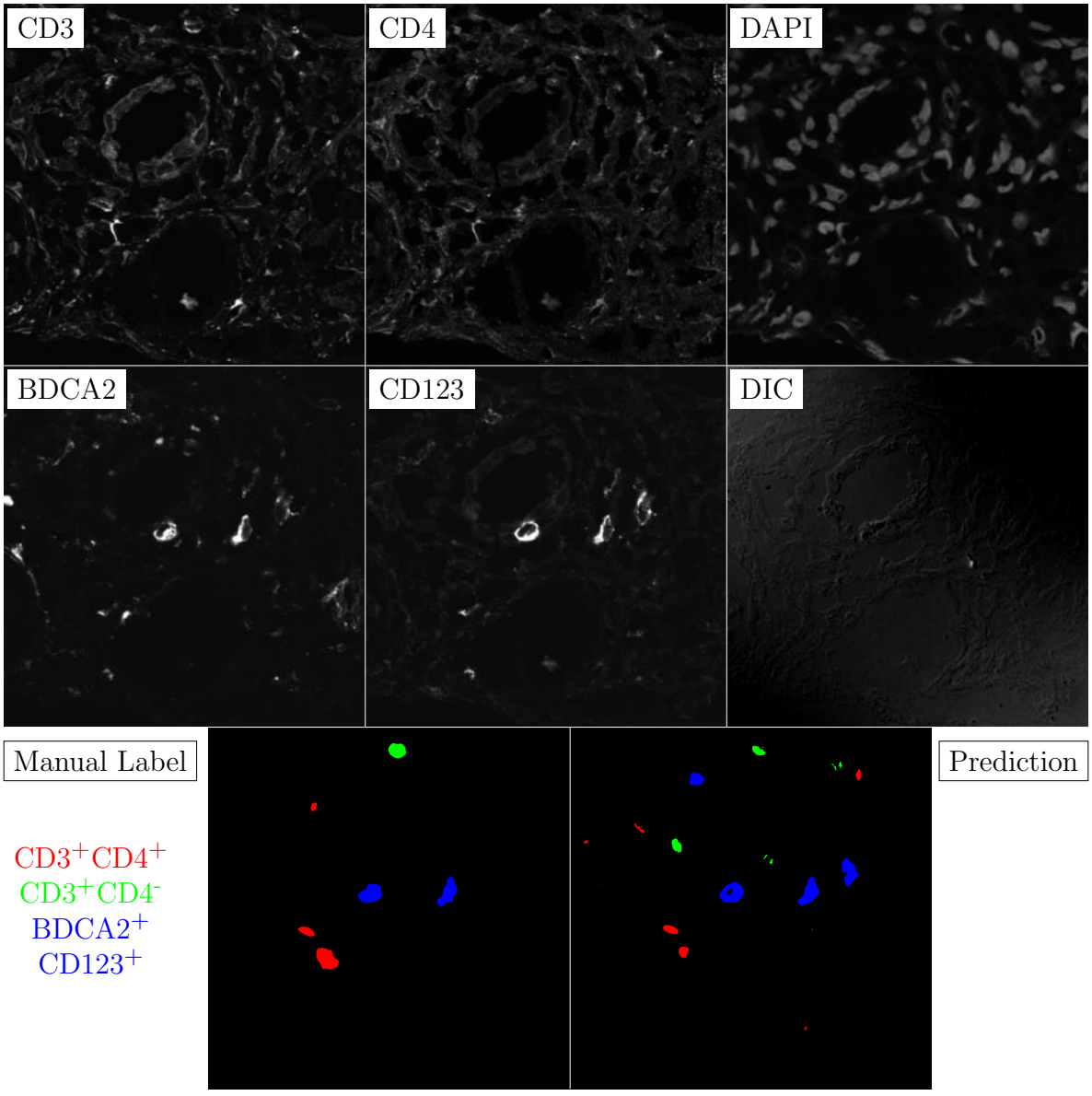

Figure G.84: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

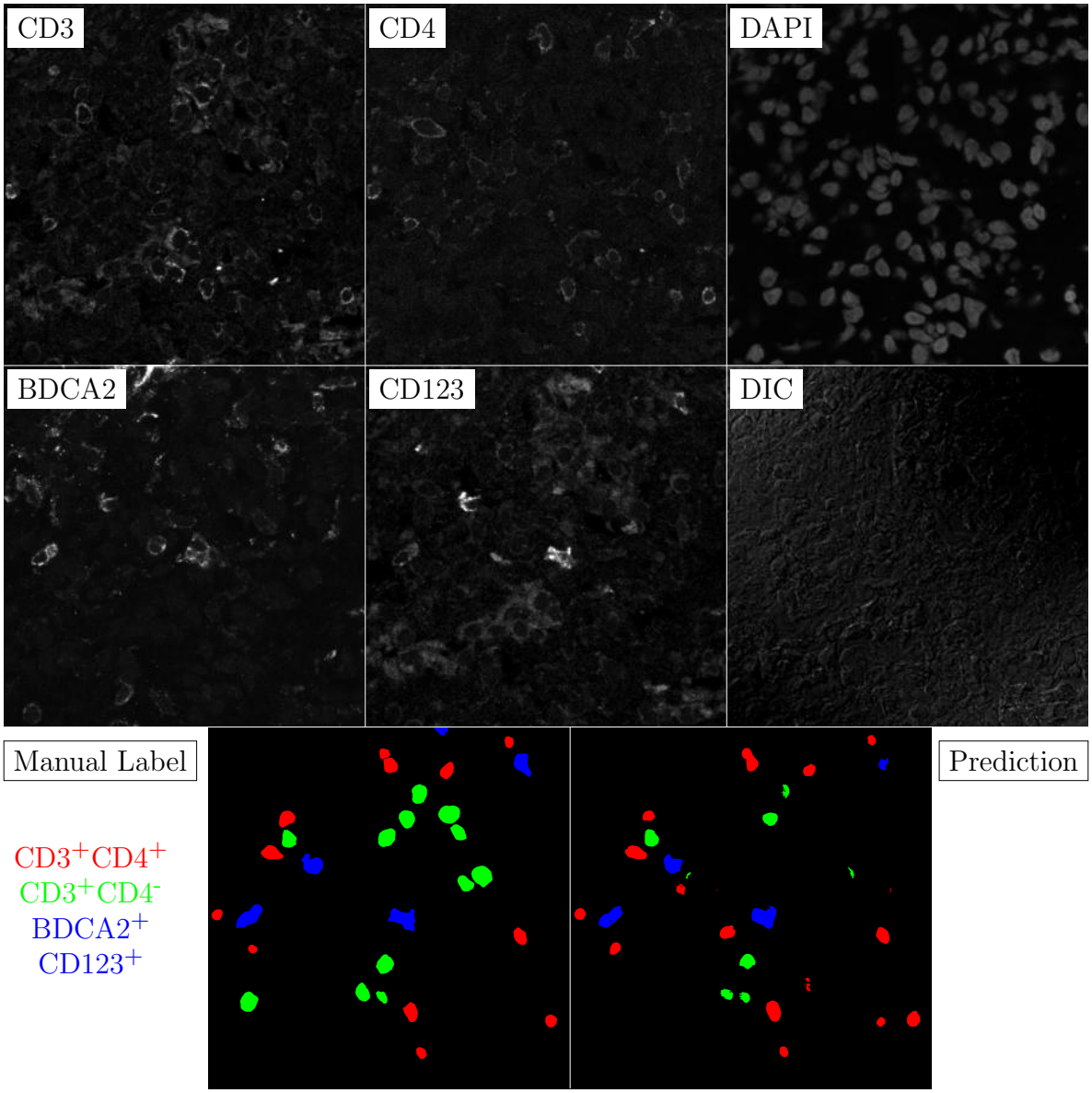

Figure G.85: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

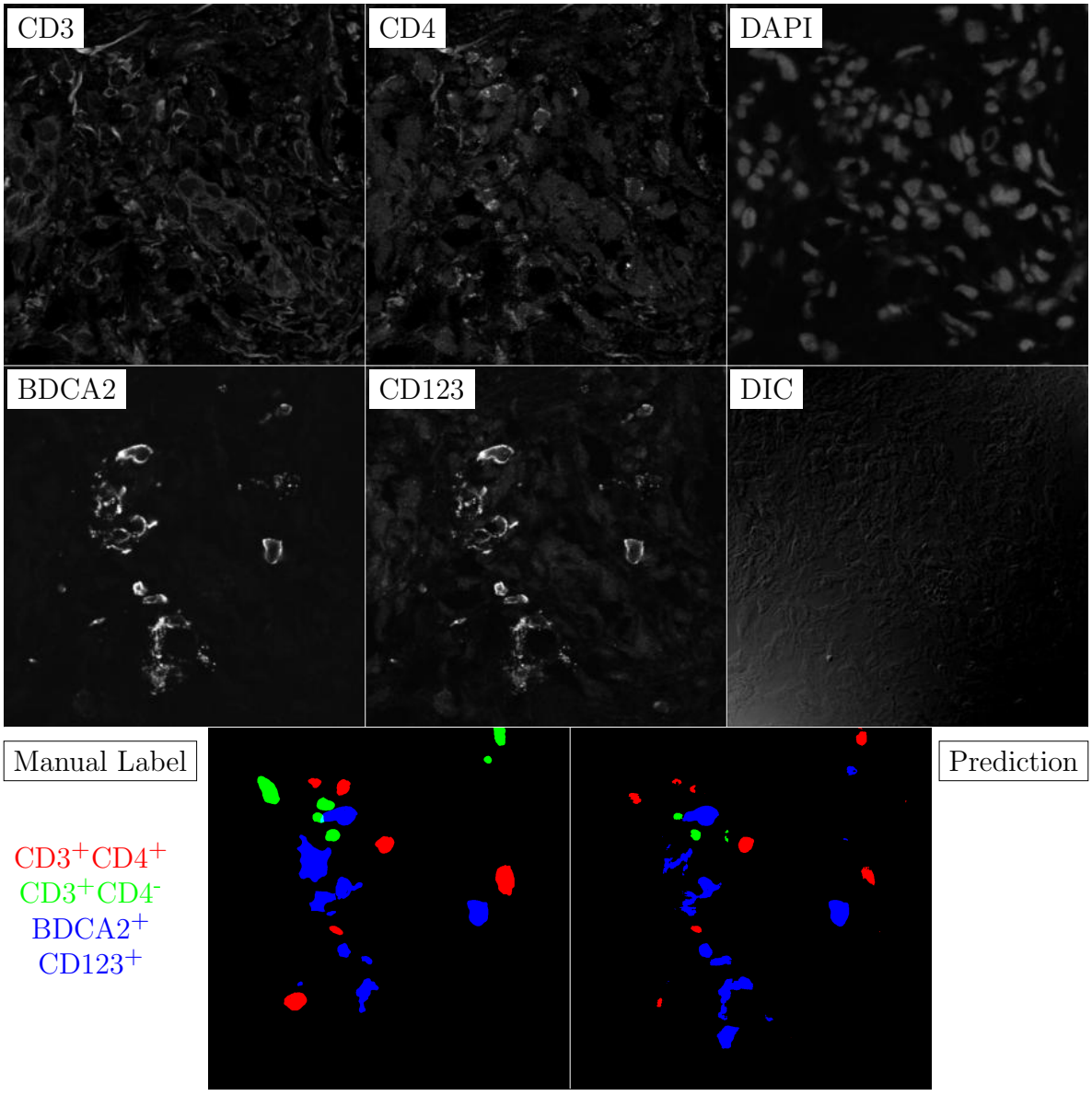

Figure G.86: **Human FF**. True ROI matrix size is 1024 × 1024. Depicted ROI matrix size is 300 × 300. for images and 1024 × 1024 for labels.

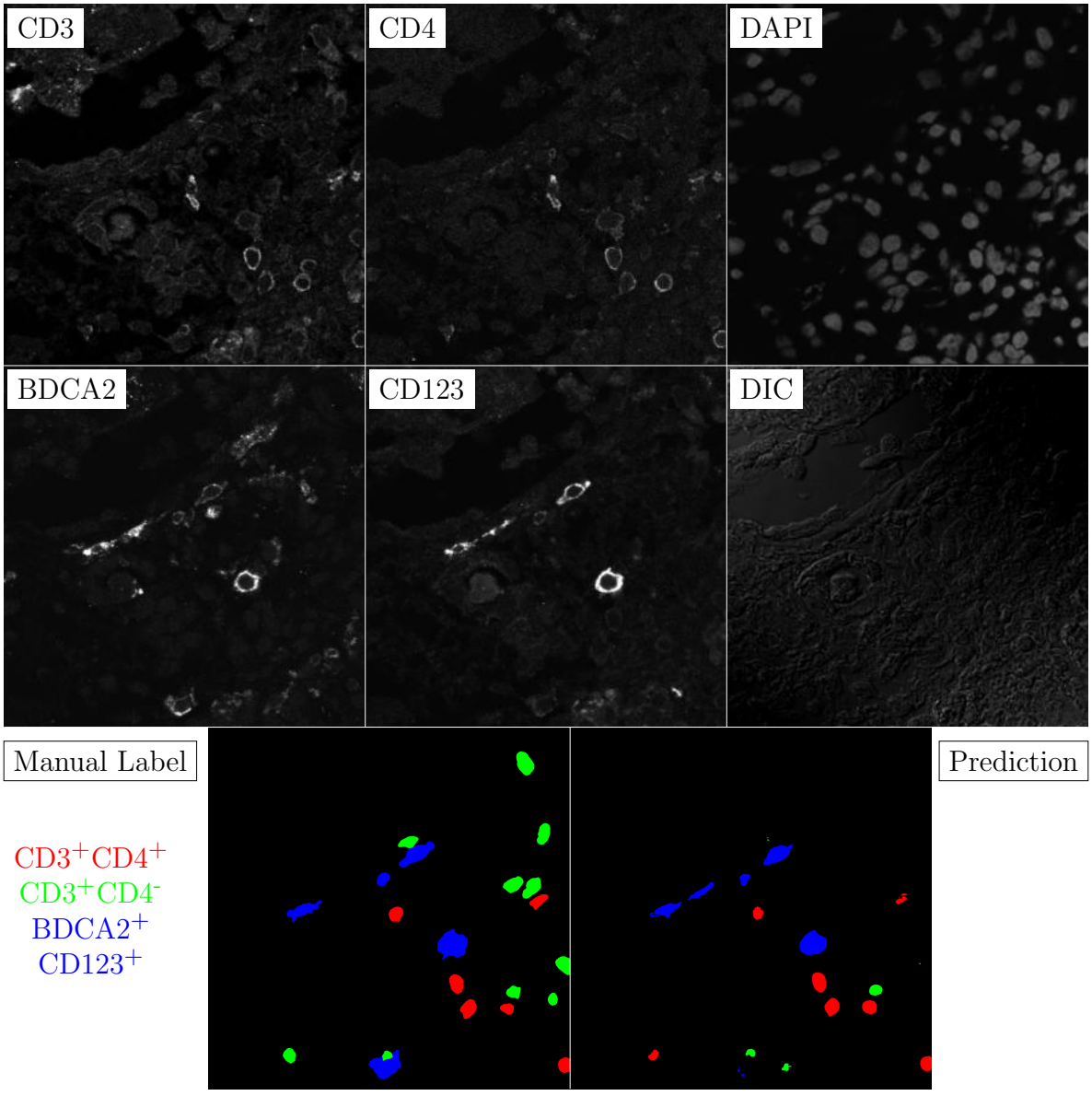

Figure G.87: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

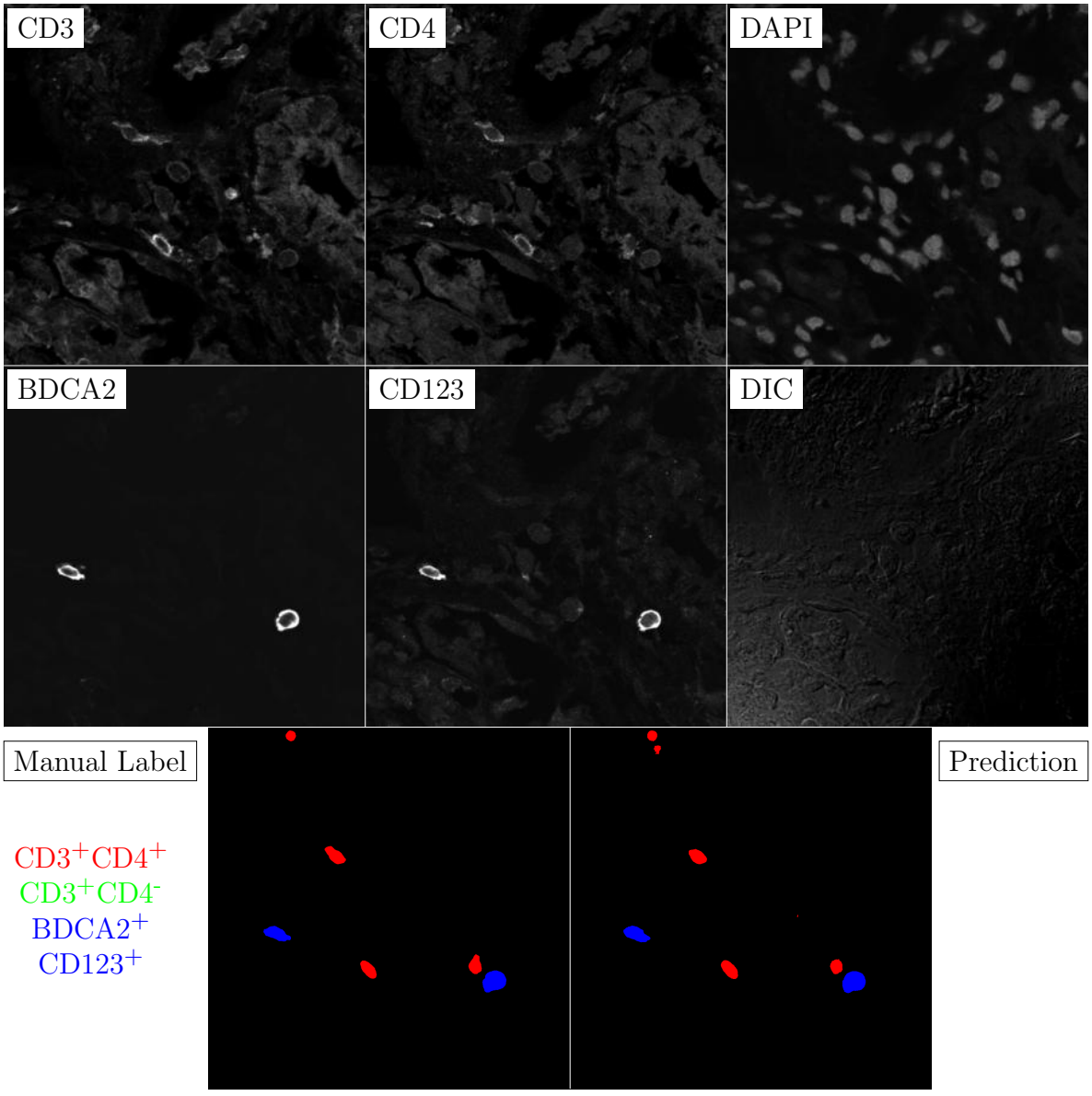

Figure G.88: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

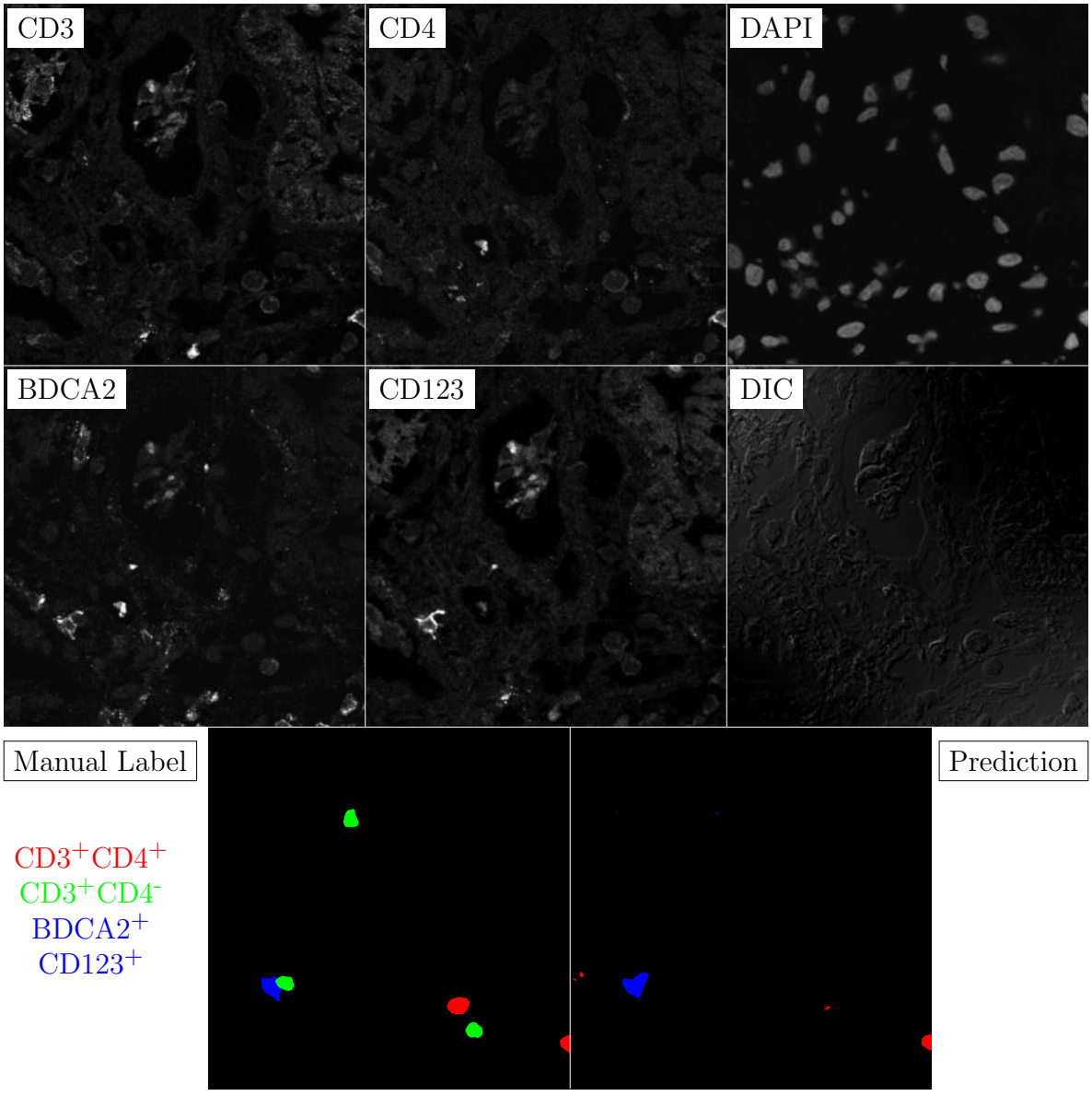

Figure G.89: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

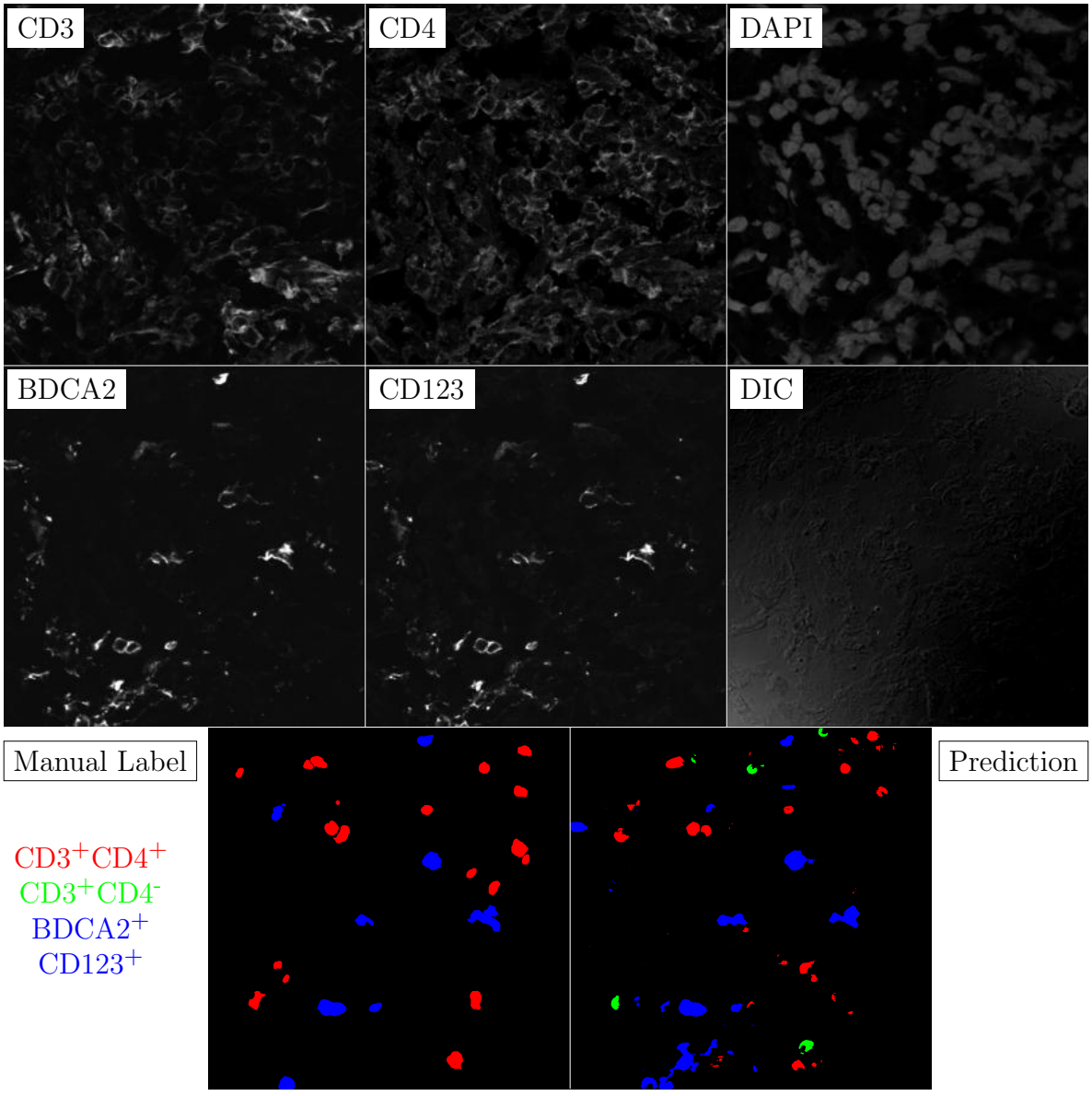

Figure G.90: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

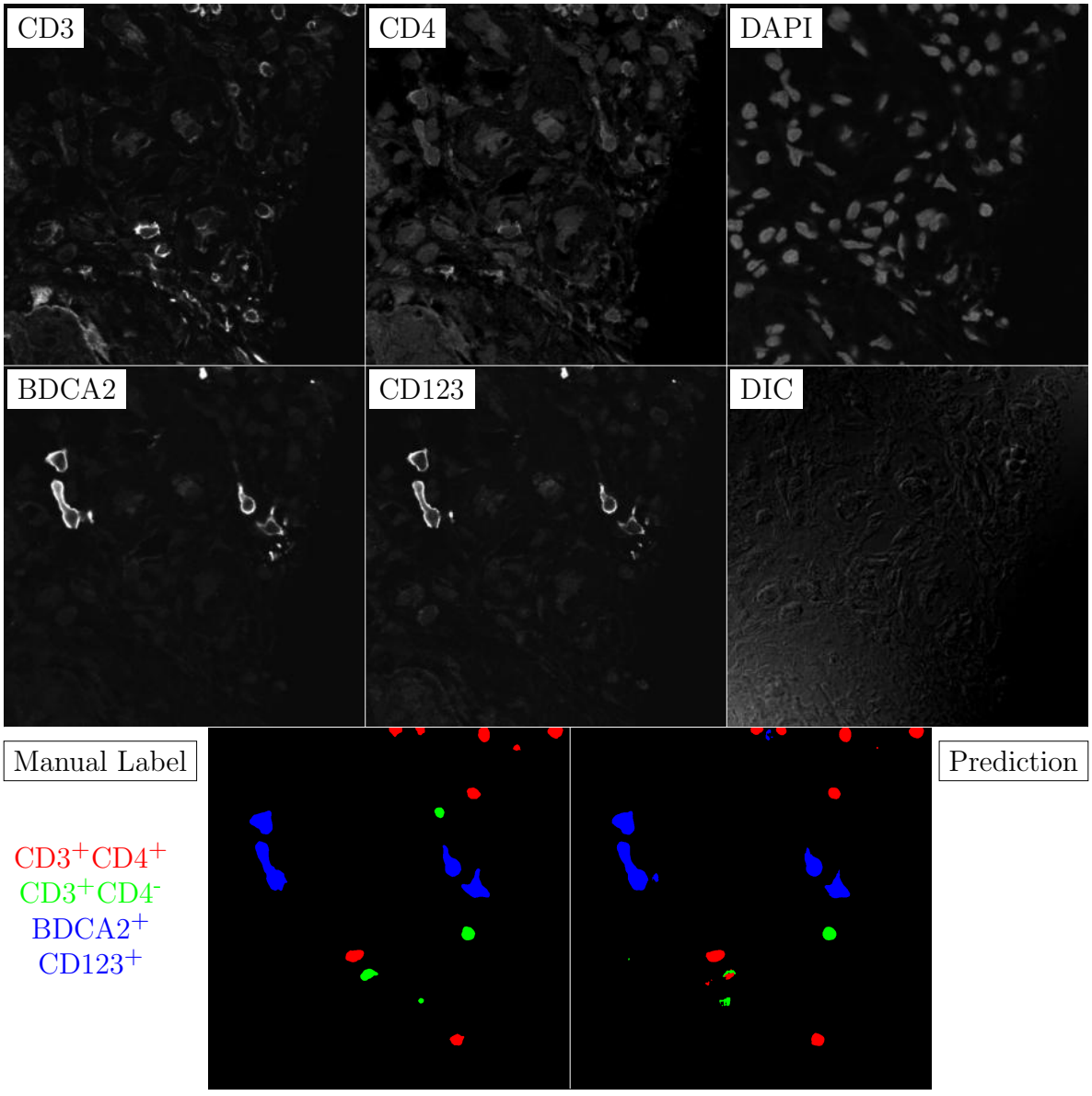

Figure G.91: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

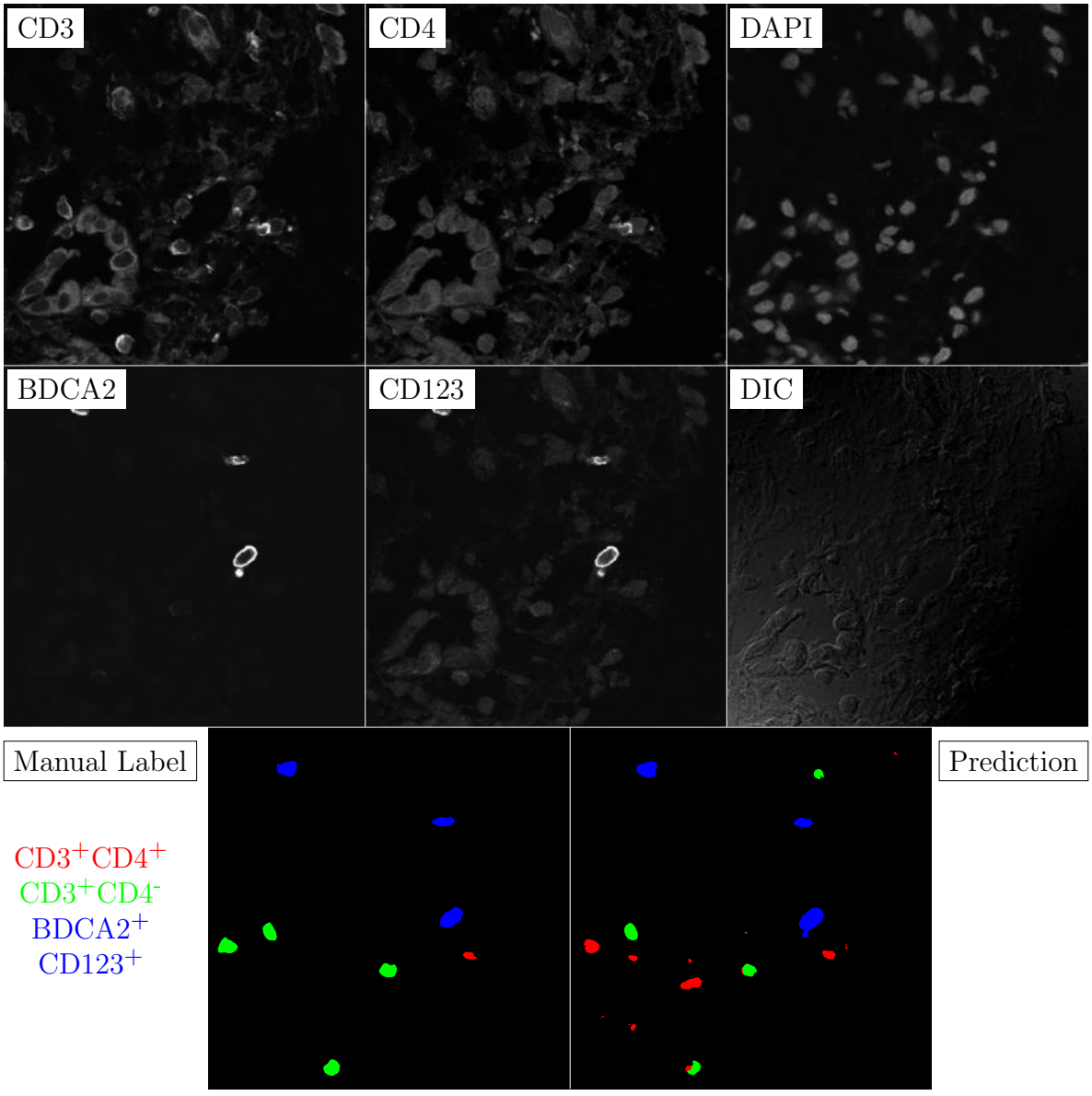

Figure G.92: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

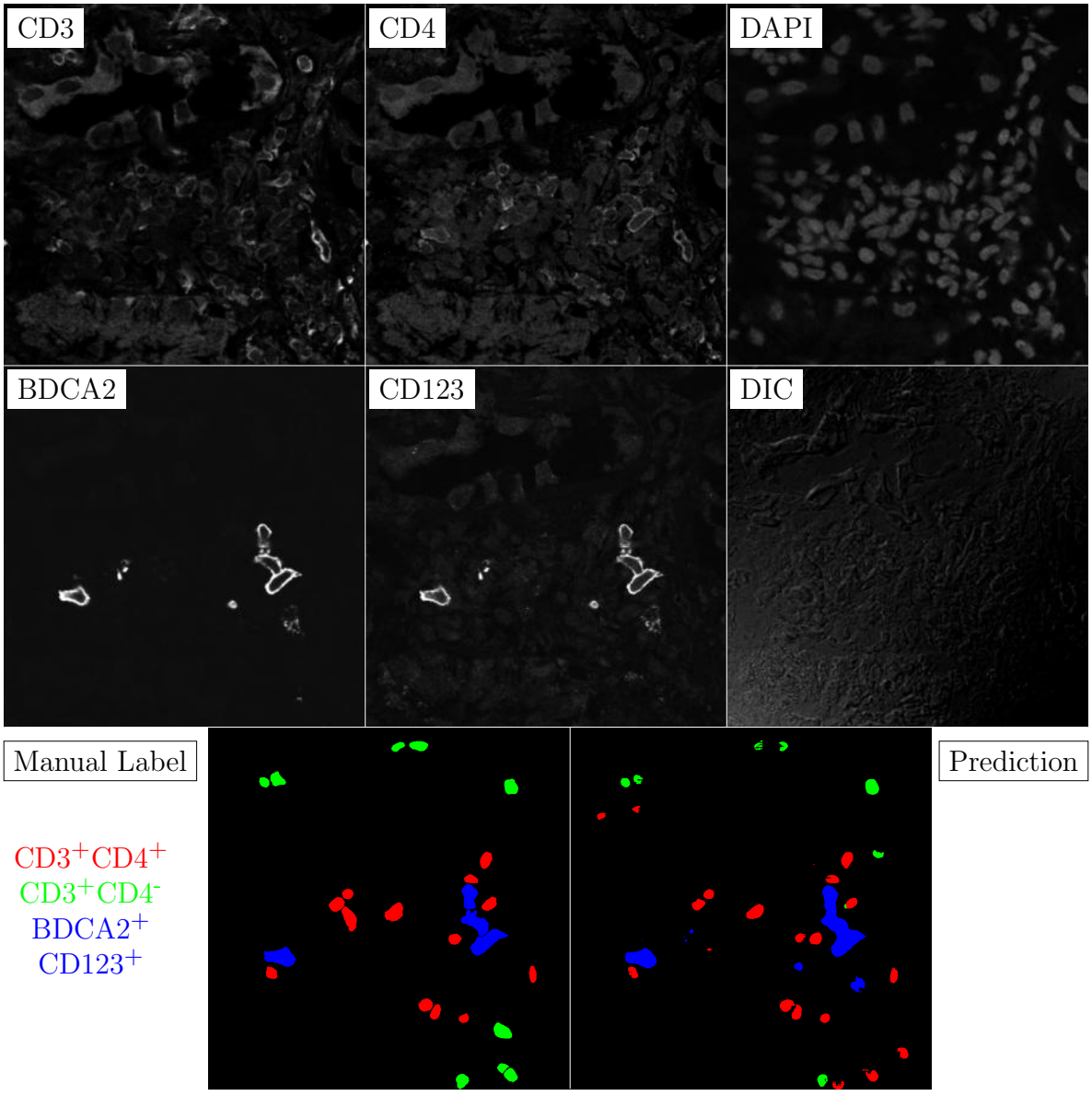

Figure G.93: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

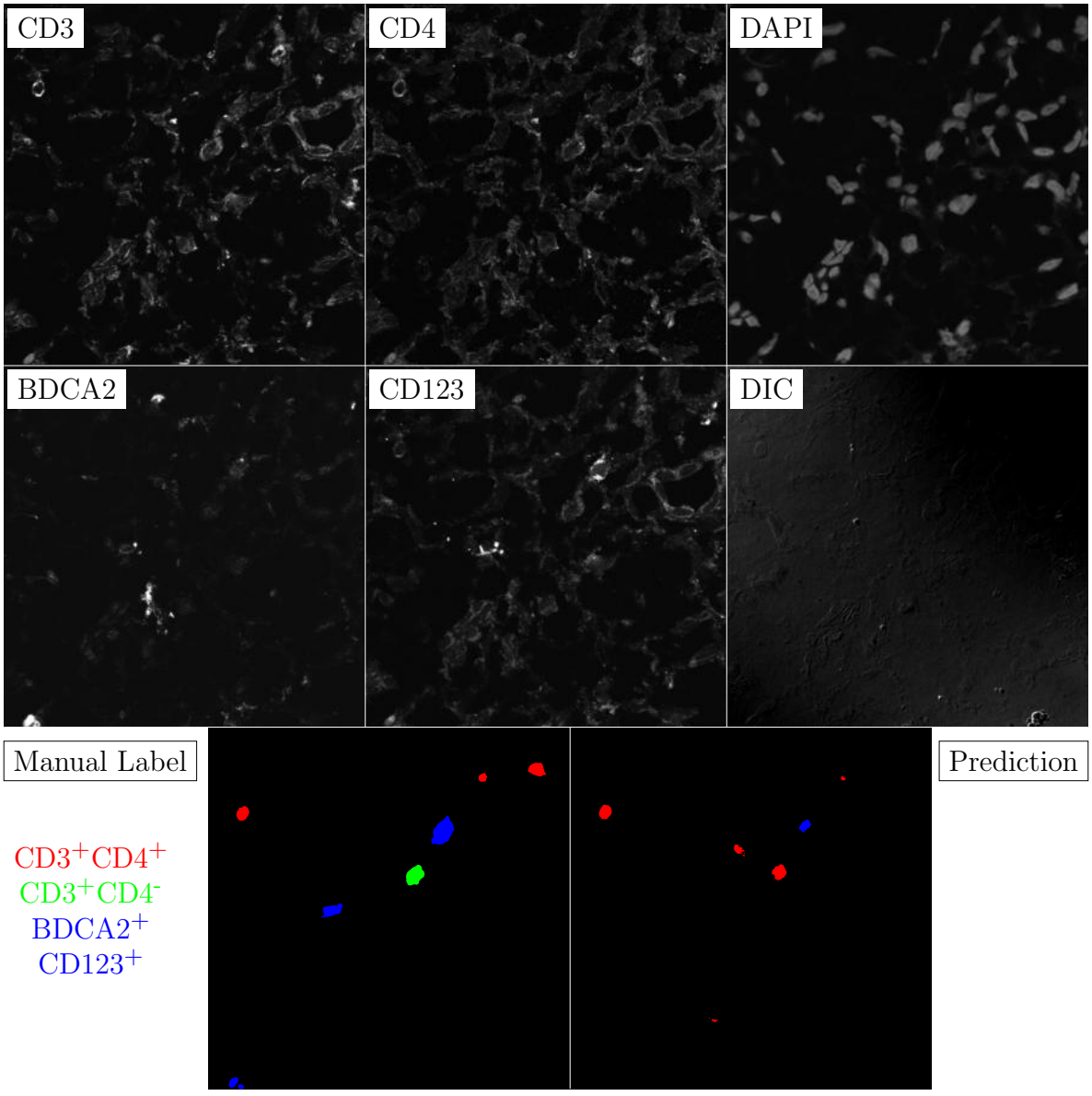

Figure G.94: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

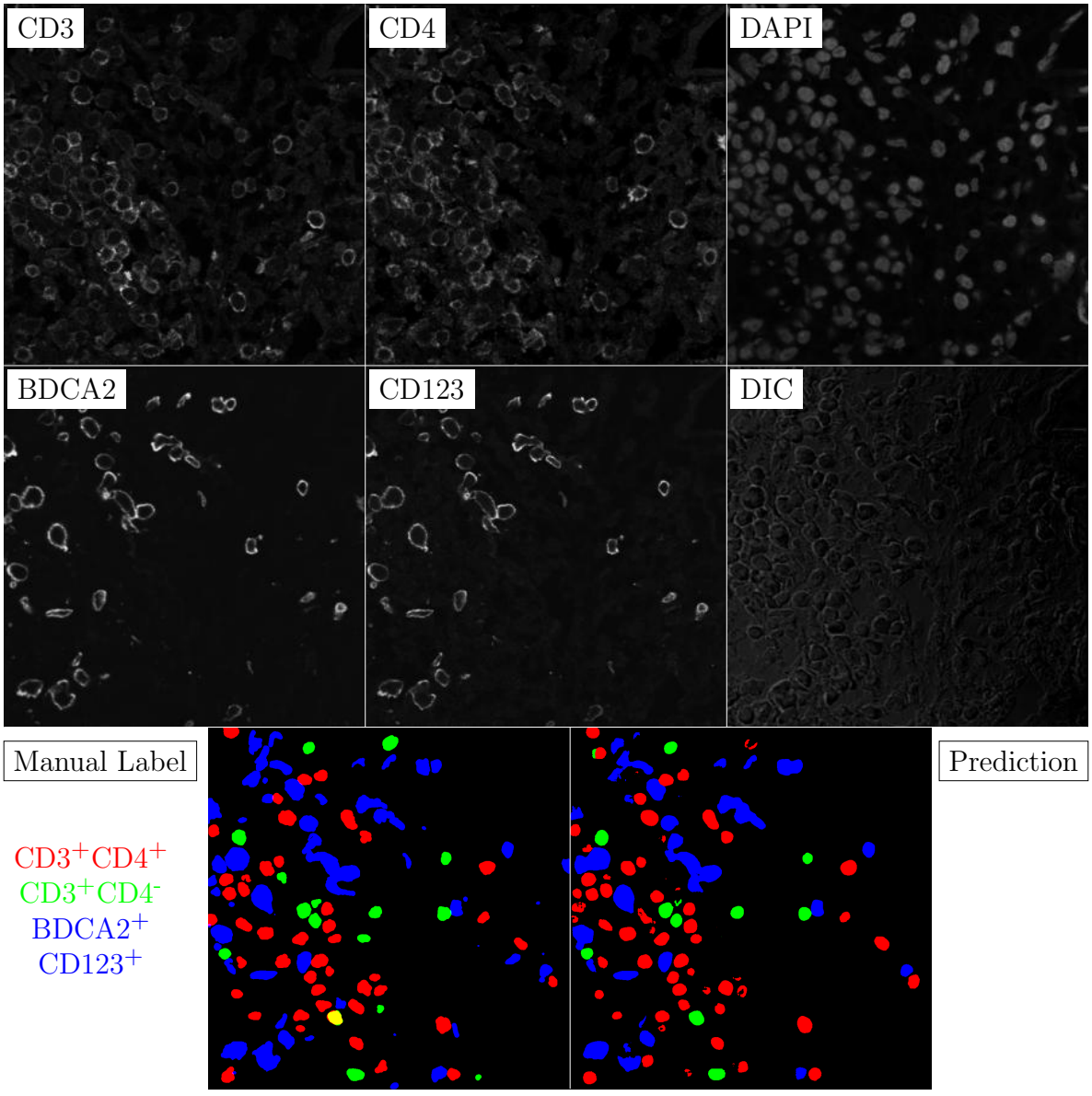

Figure G.95: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

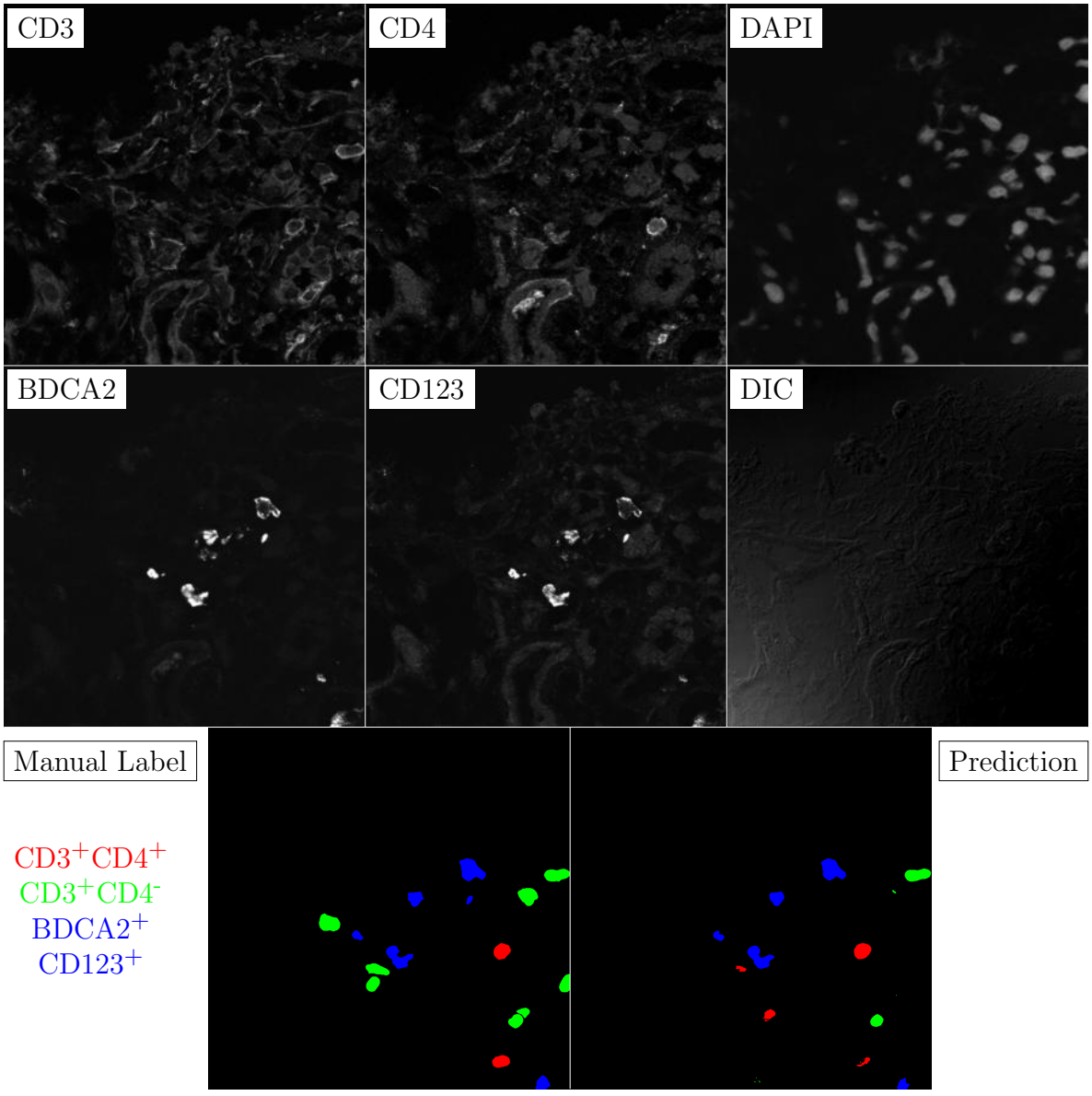

Figure G.96: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

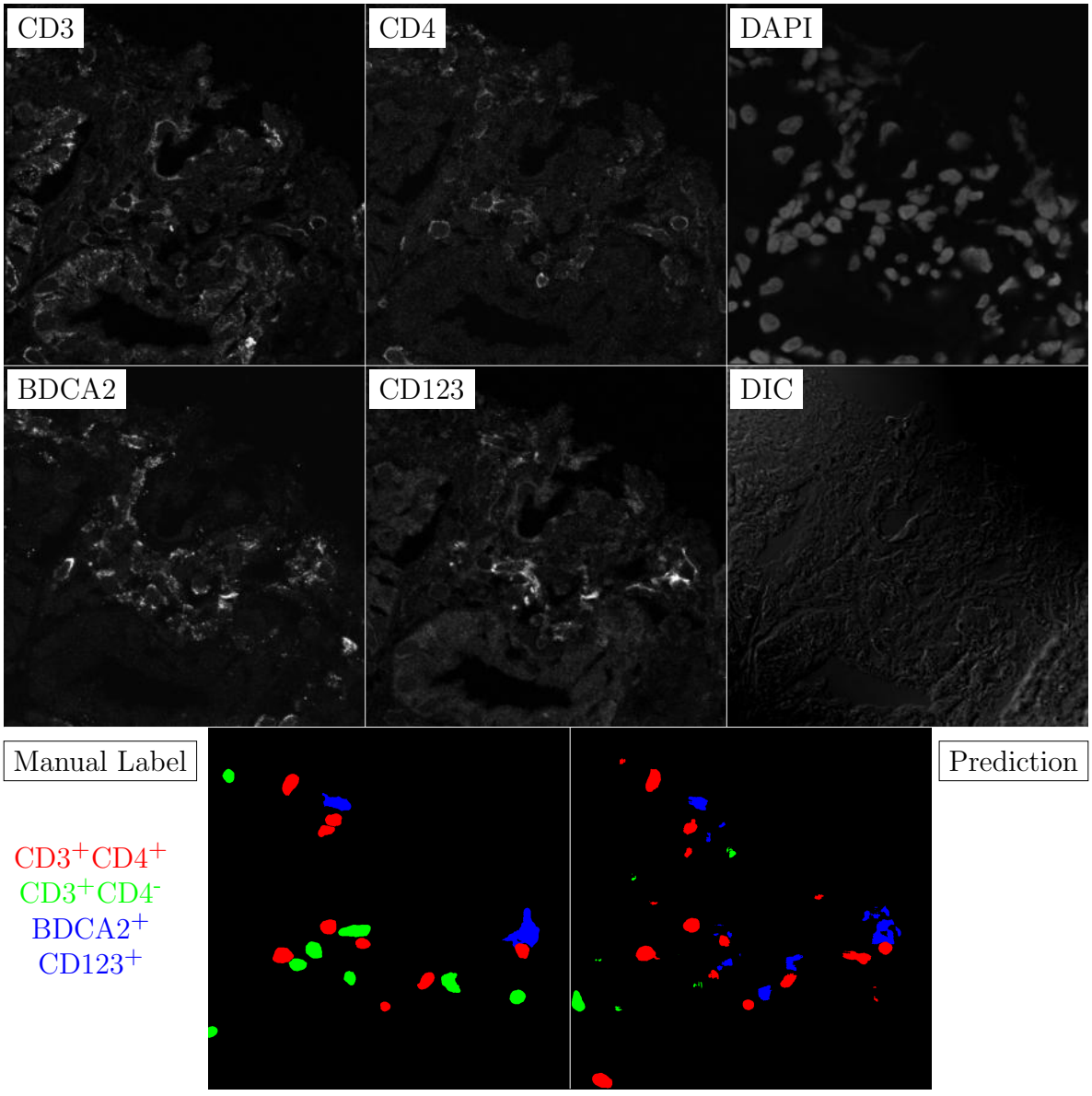

Figure G.97: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

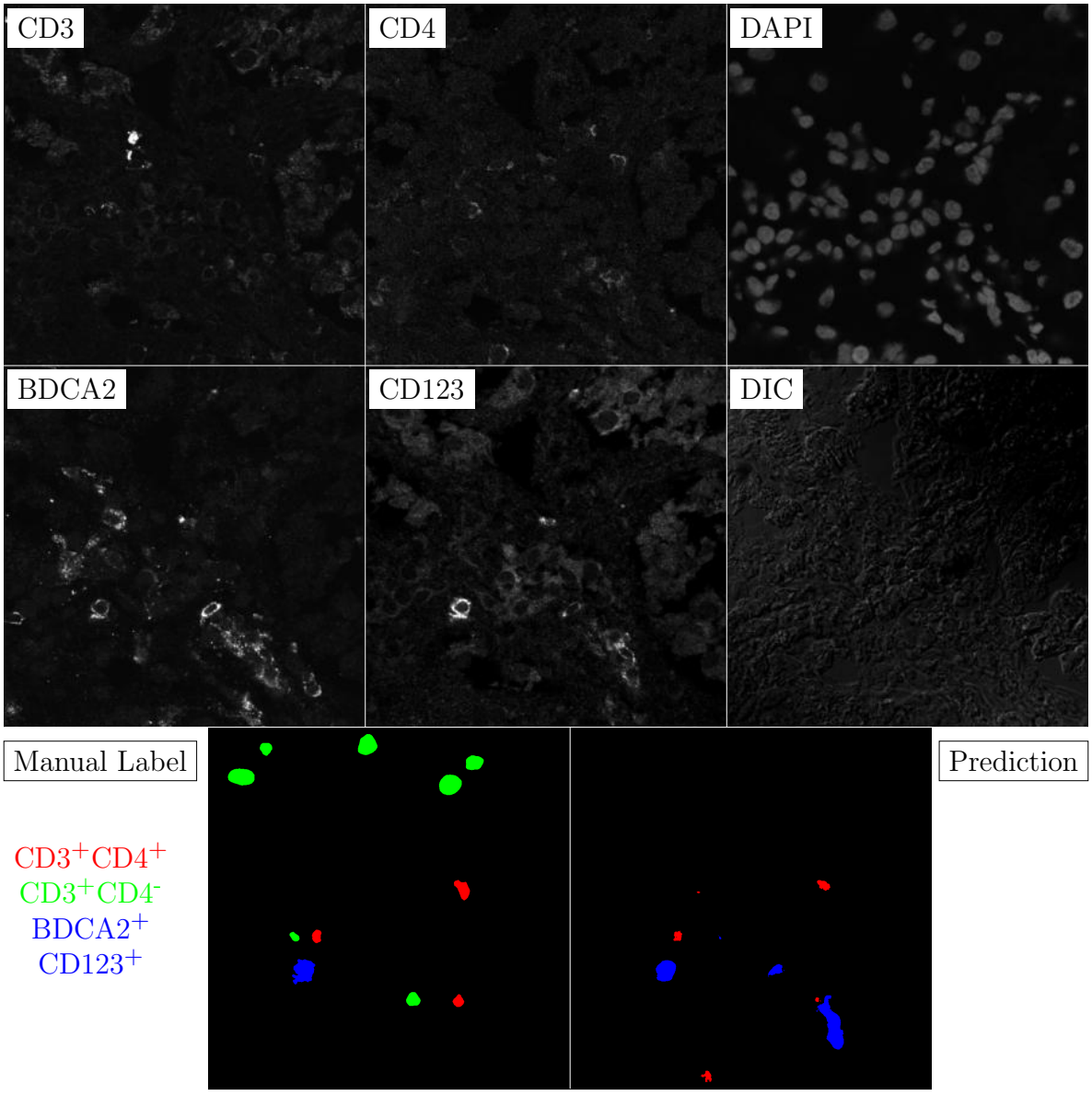

Figure G.98: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

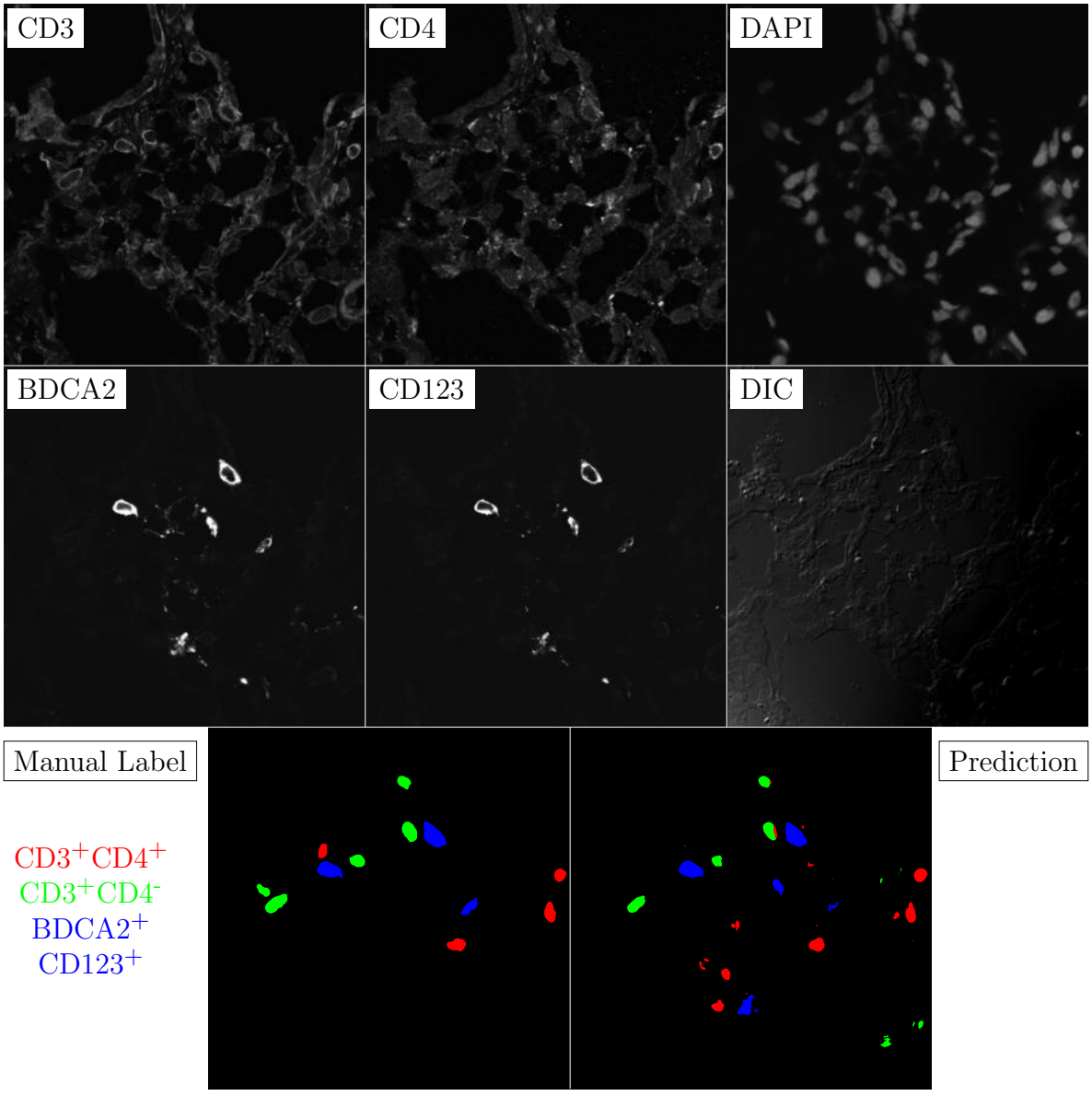

Figure G.99: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

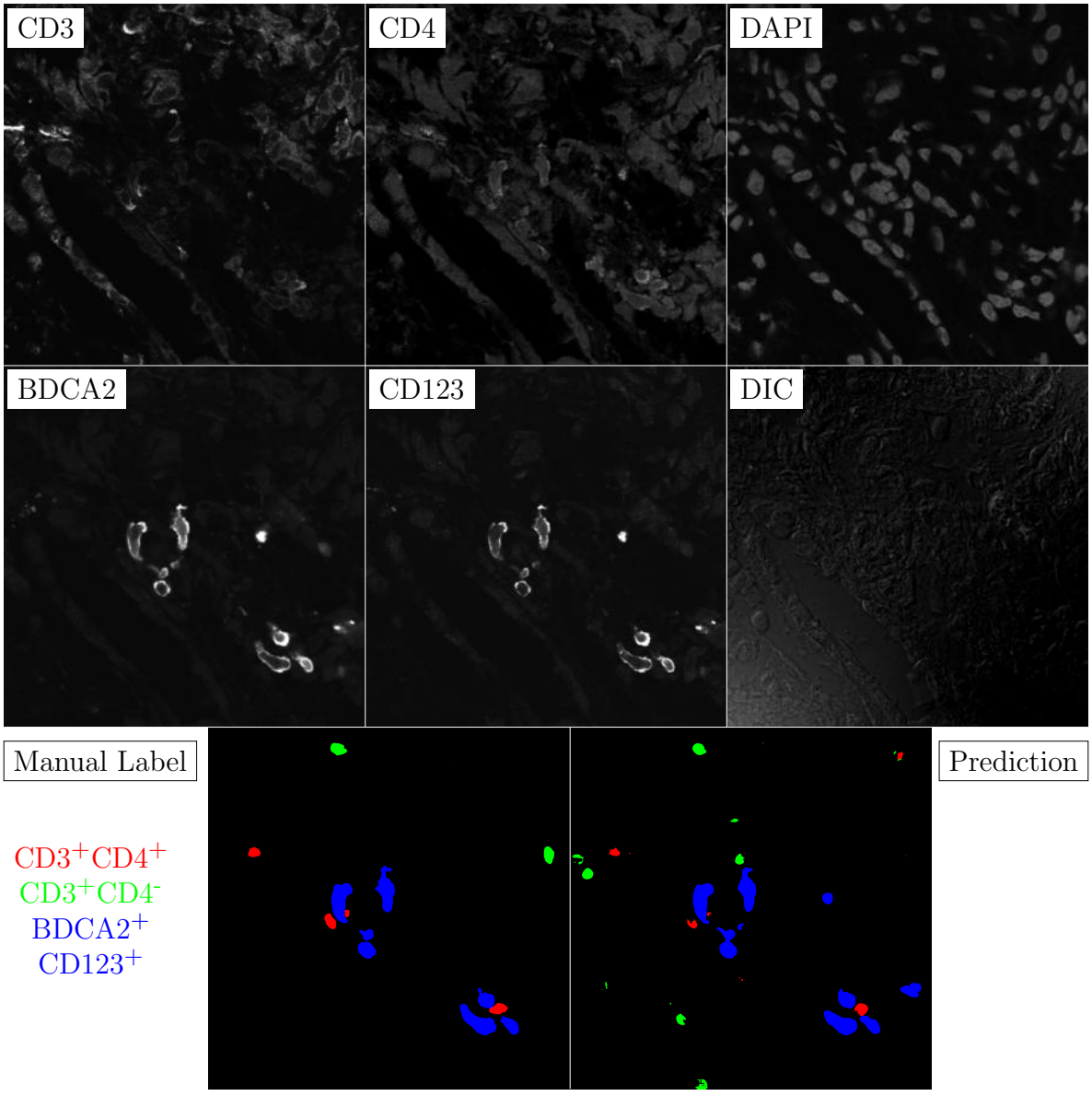

Figure G.100: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

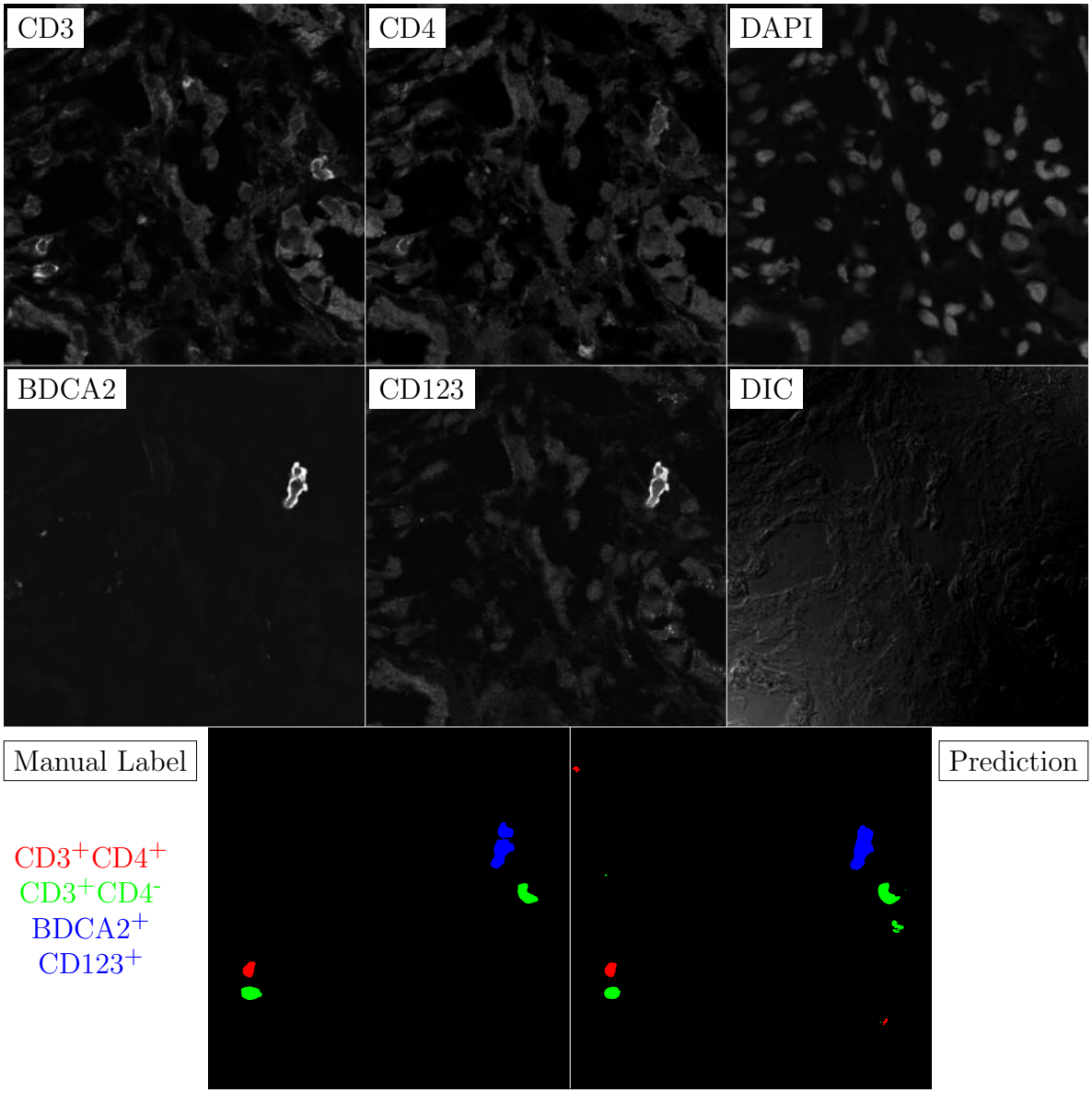

Figure G.101: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

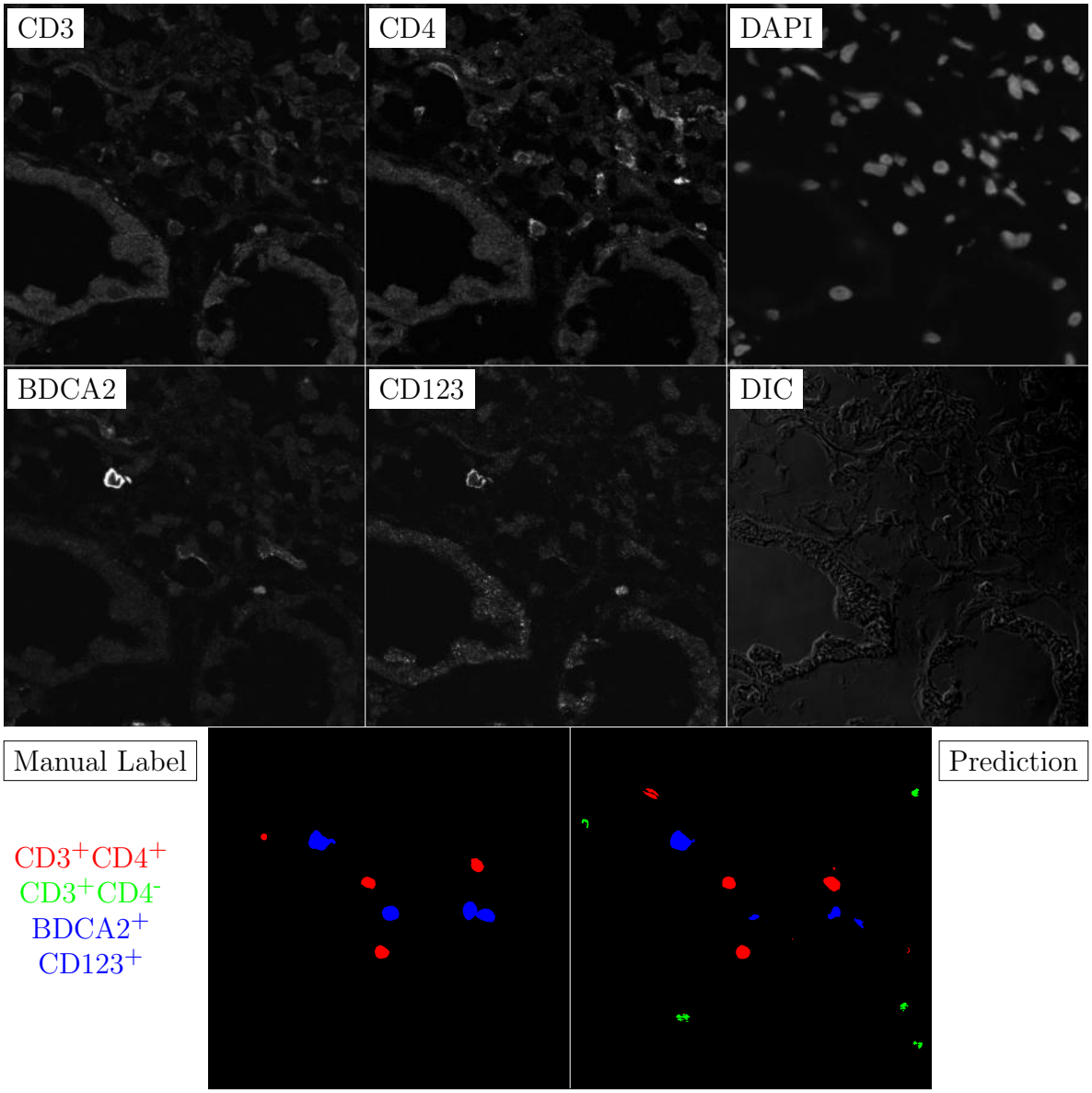

Figure G.102: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

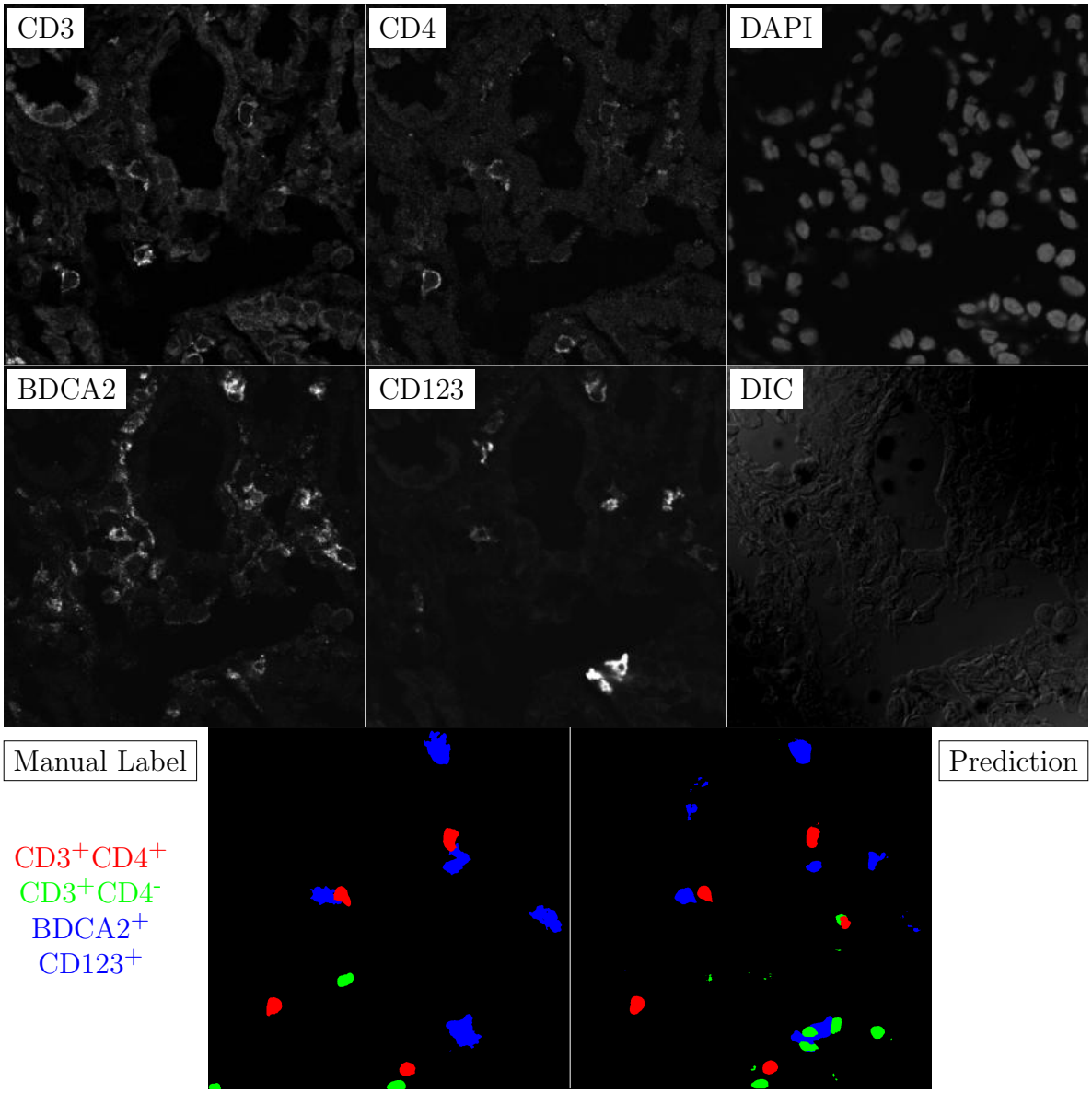

Figure G.103: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

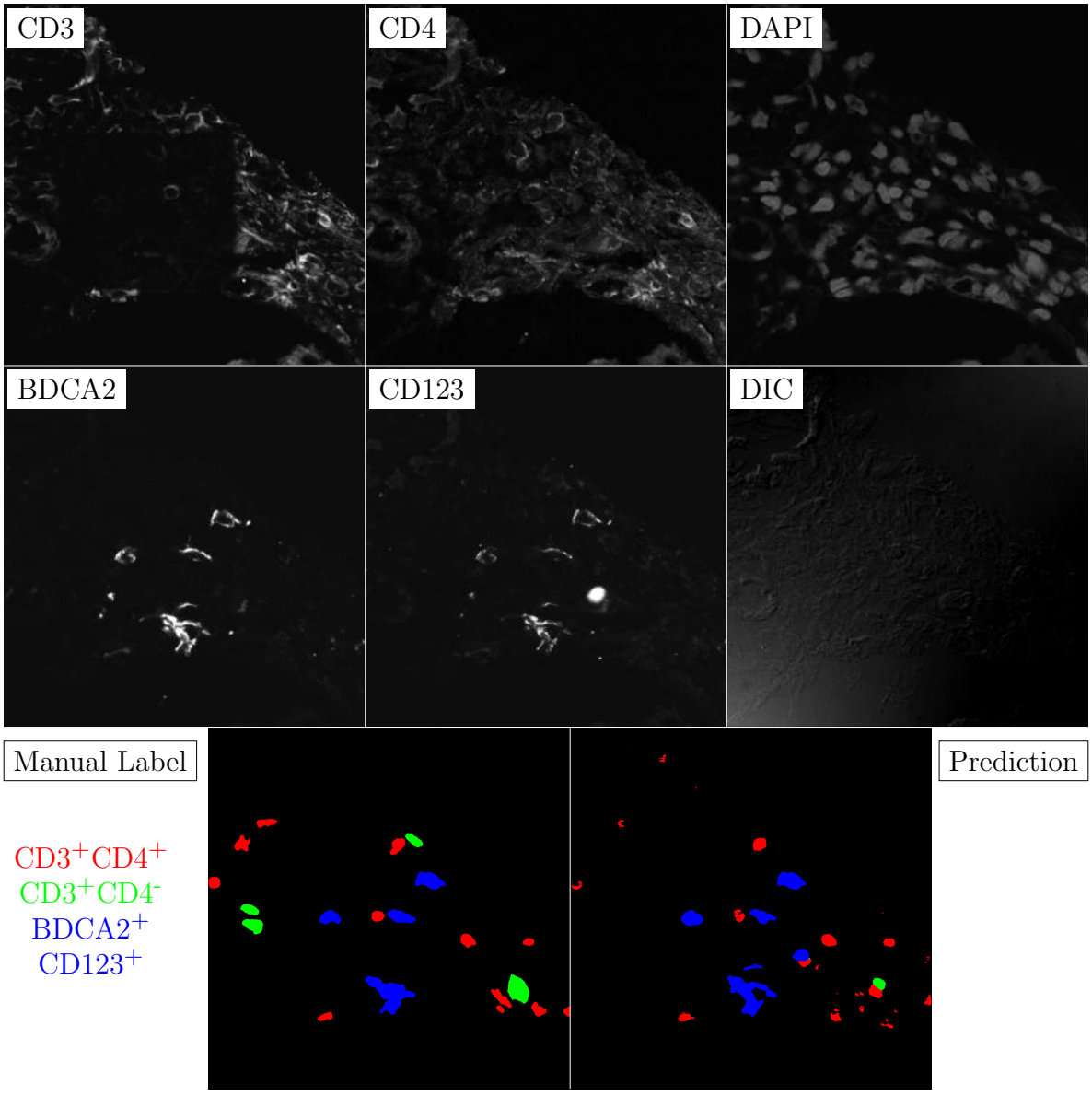

Figure G.104: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

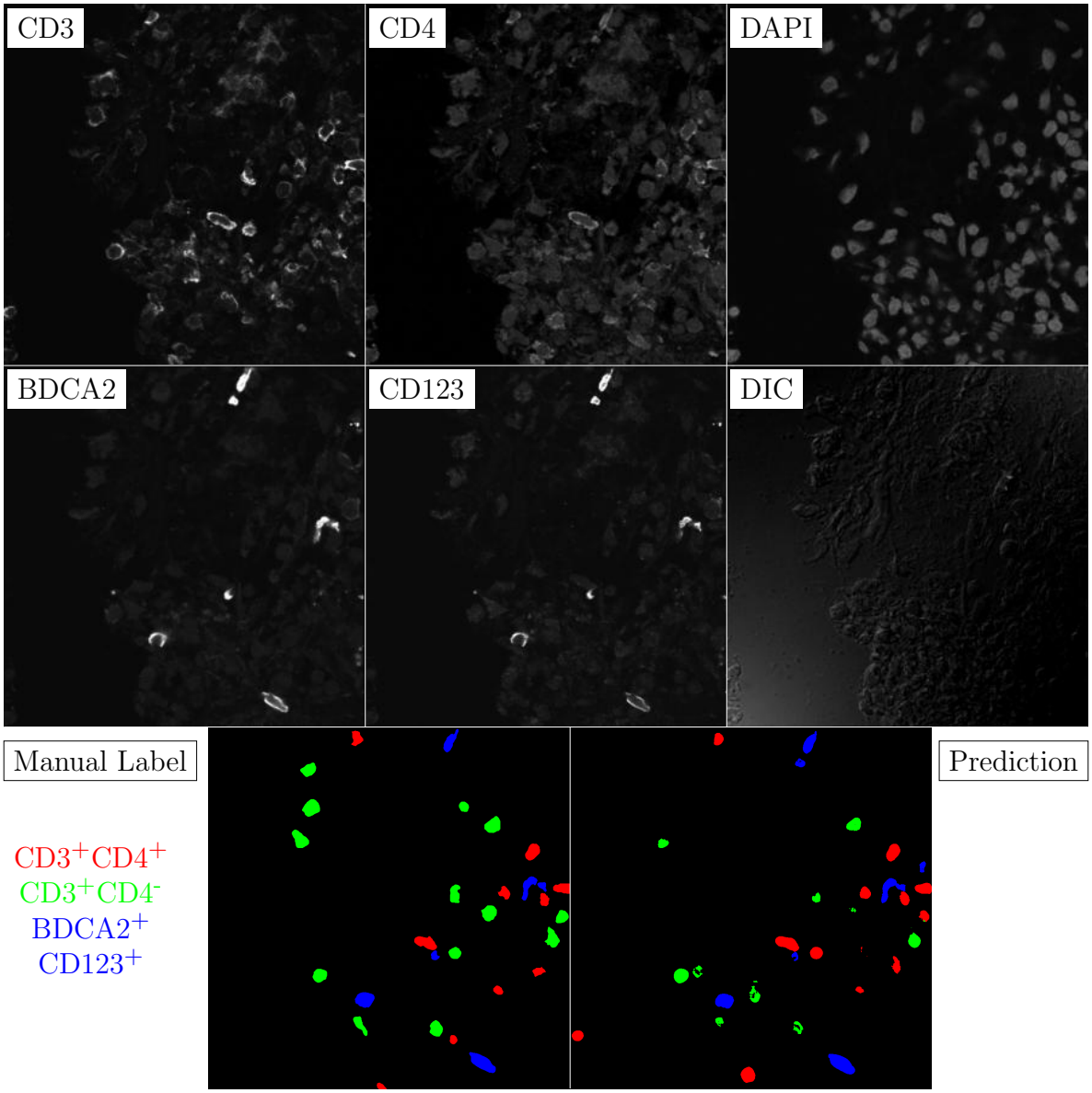

Figure G.105: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

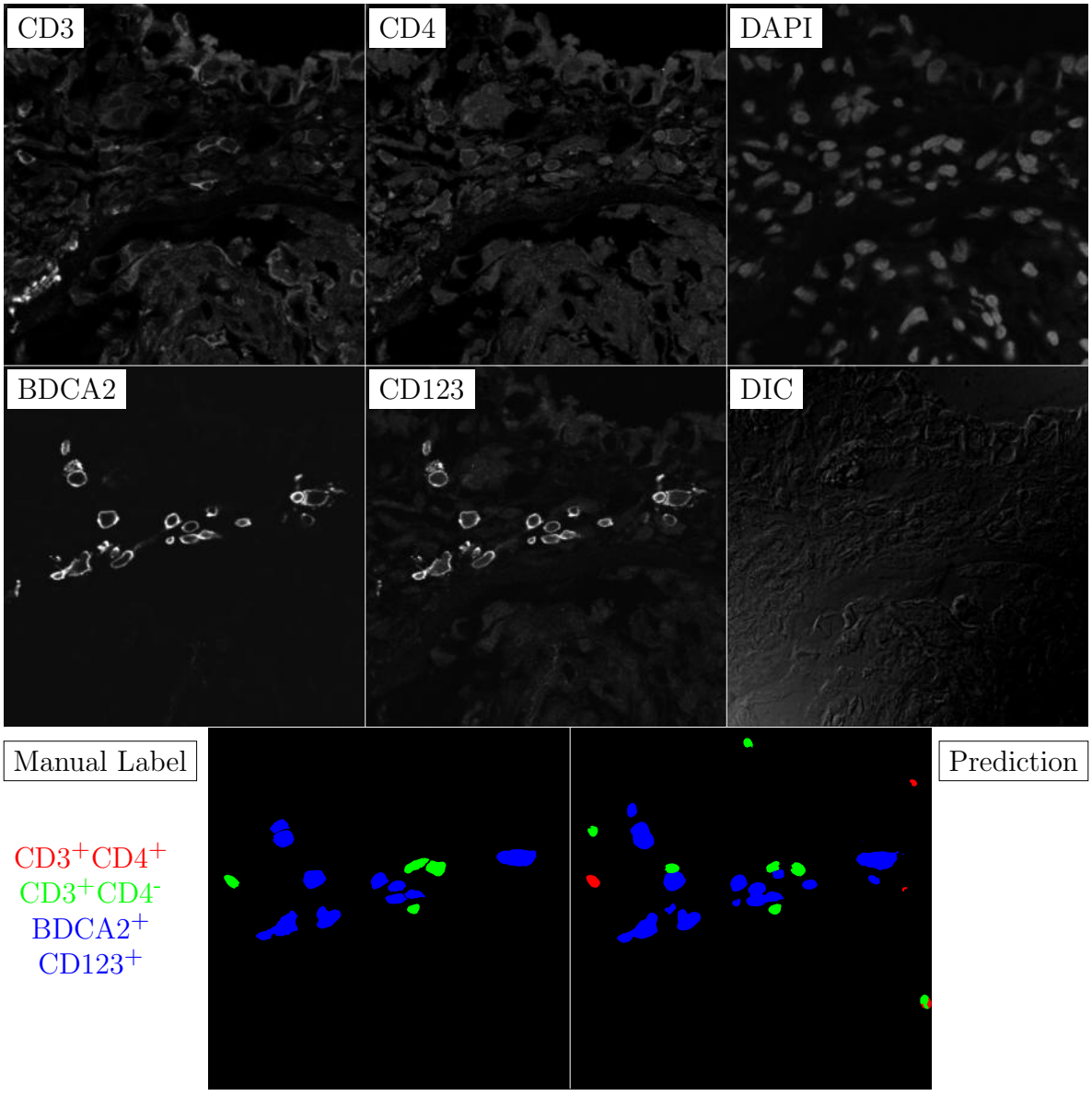

Figure G.106: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

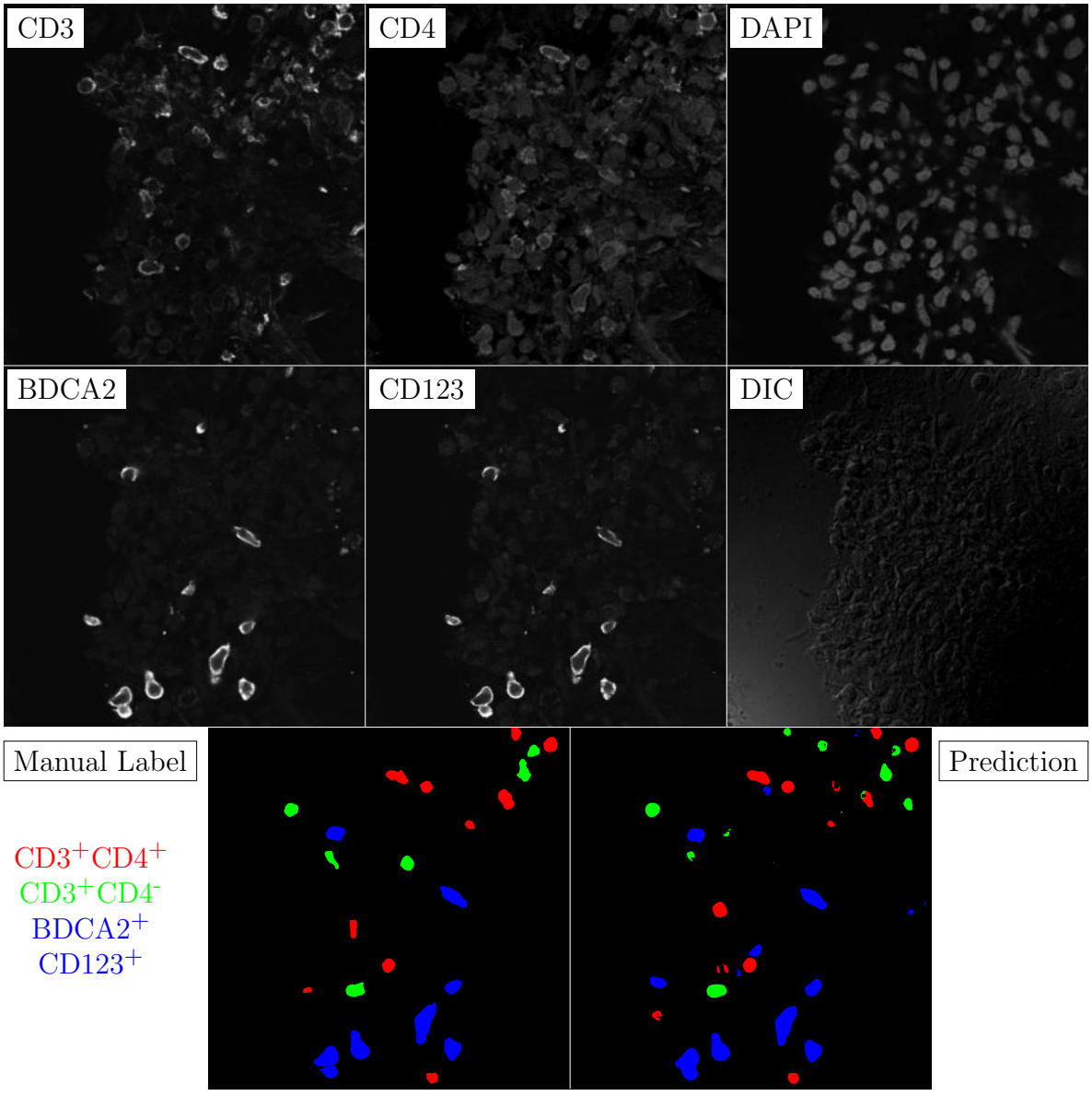

Figure G.107: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

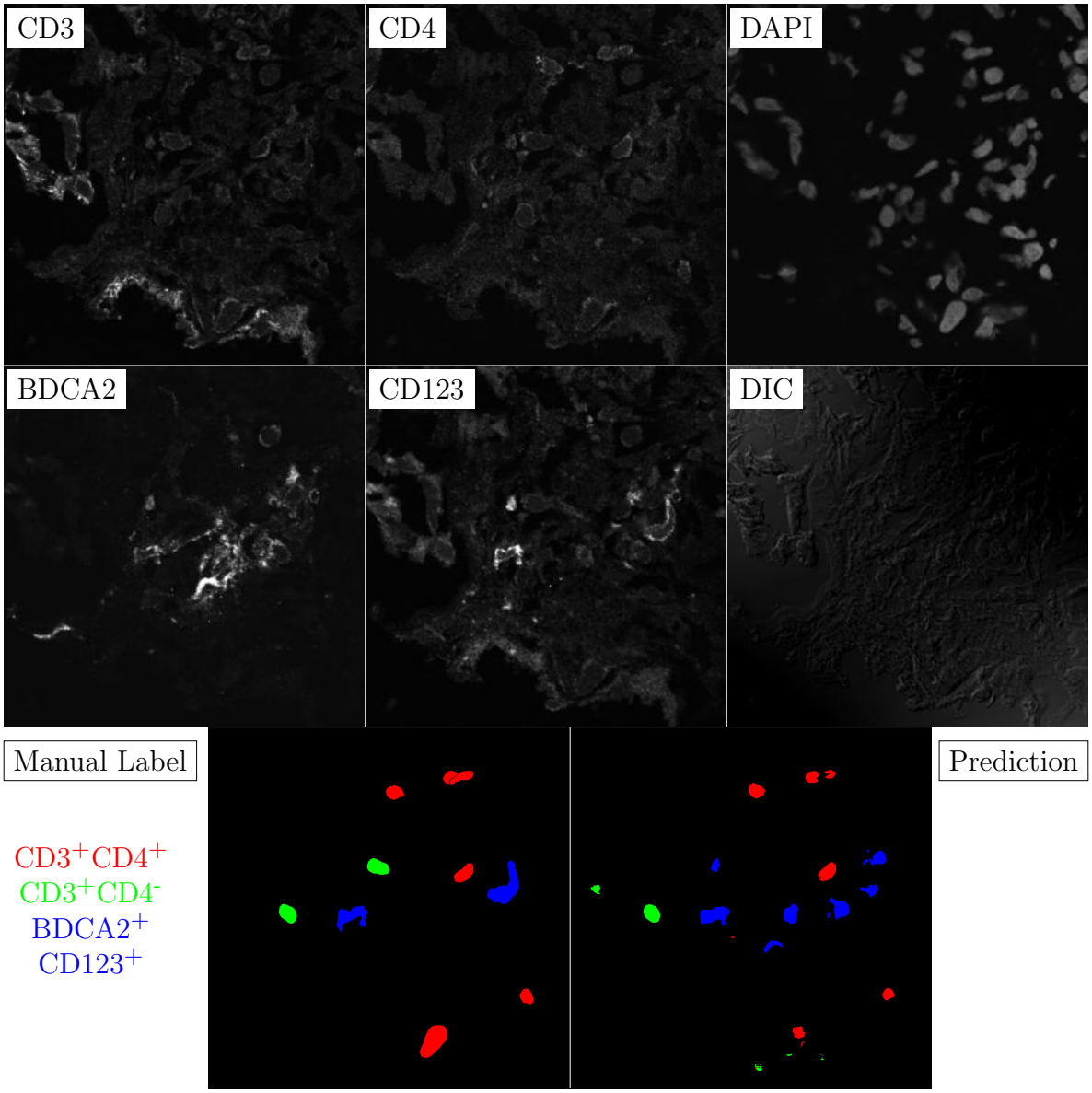

Figure G.108: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

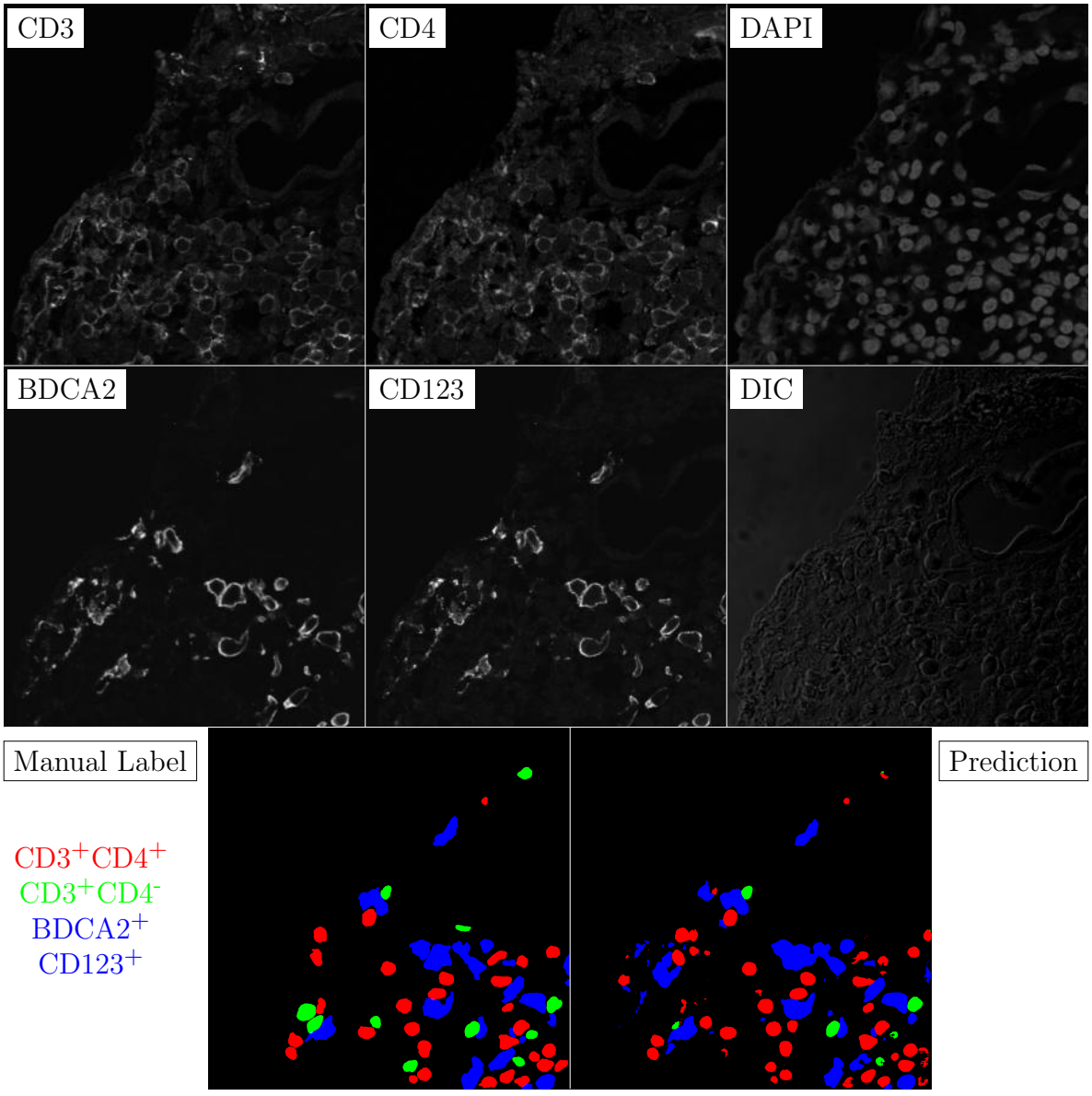

Figure G.109: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

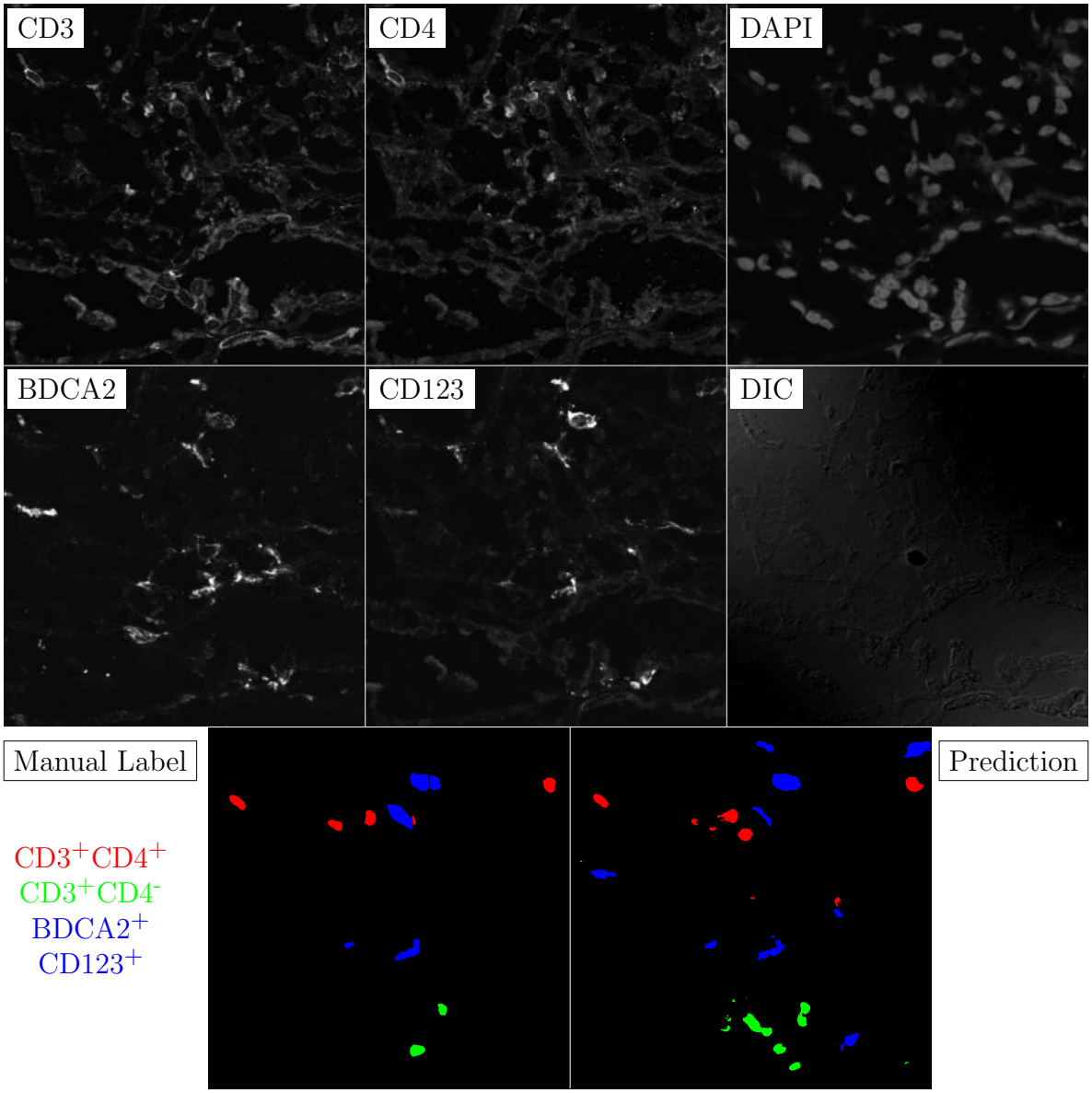

Figure G.110: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

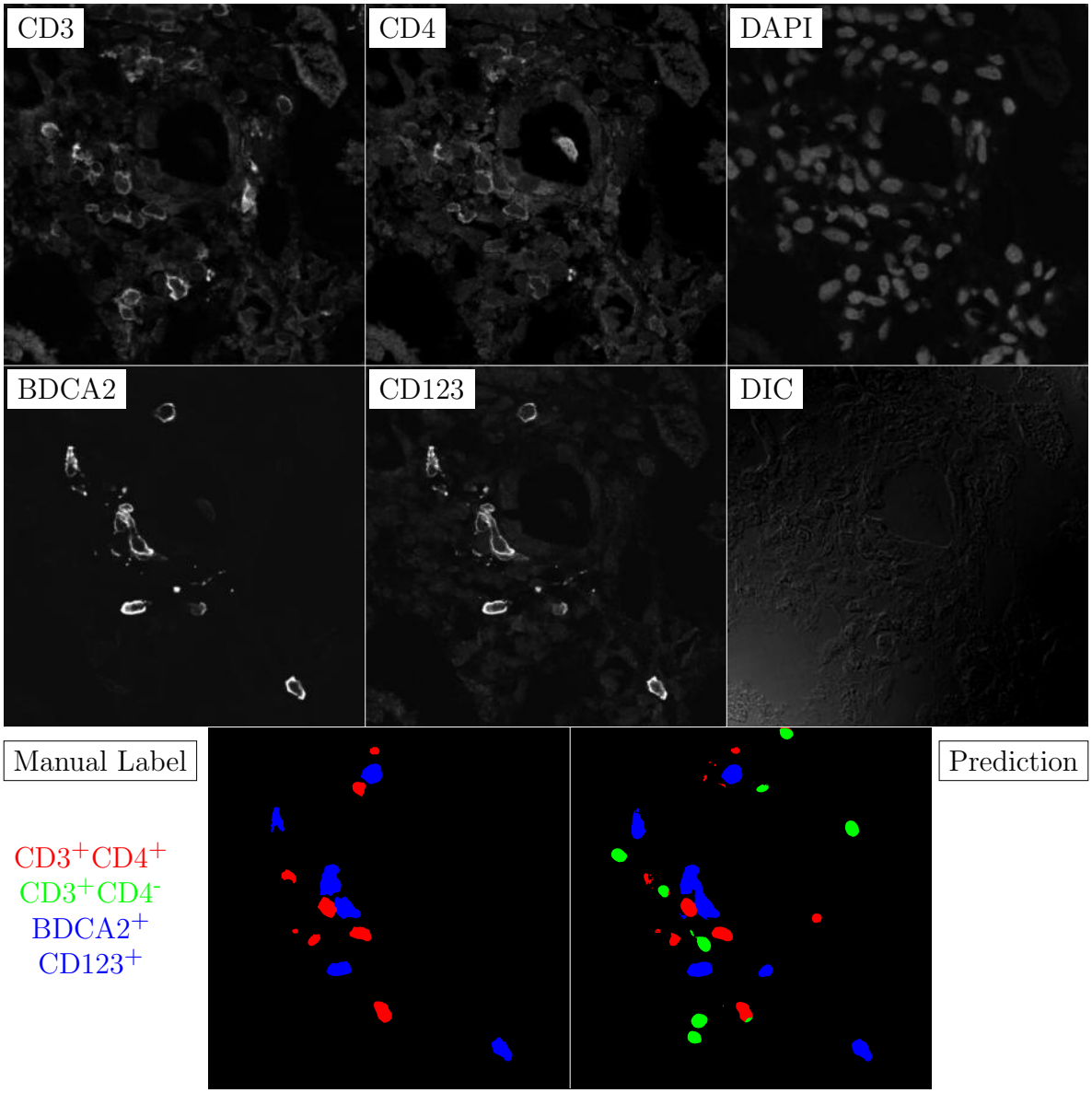

Figure G.111: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

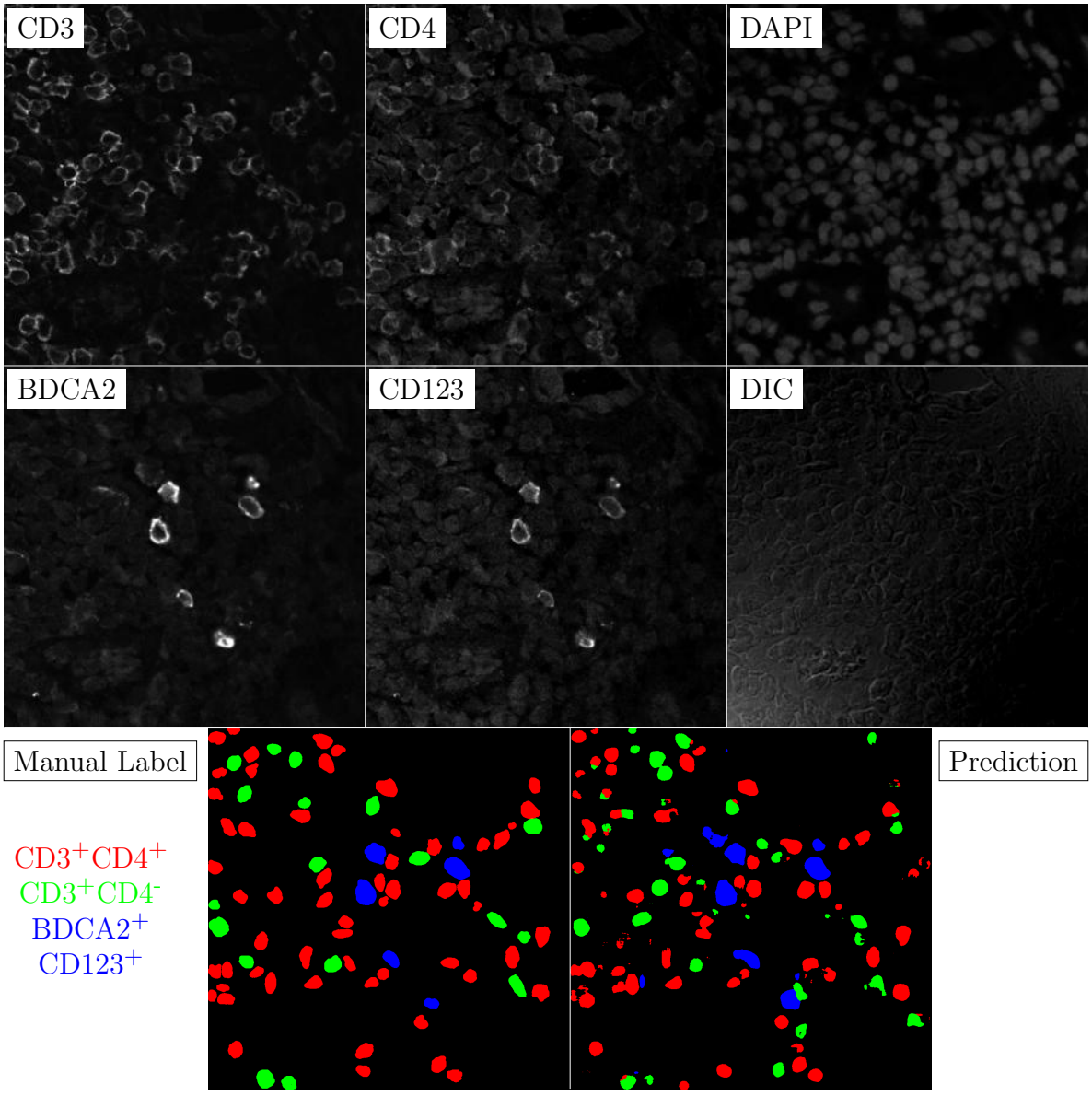

Figure G.112: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

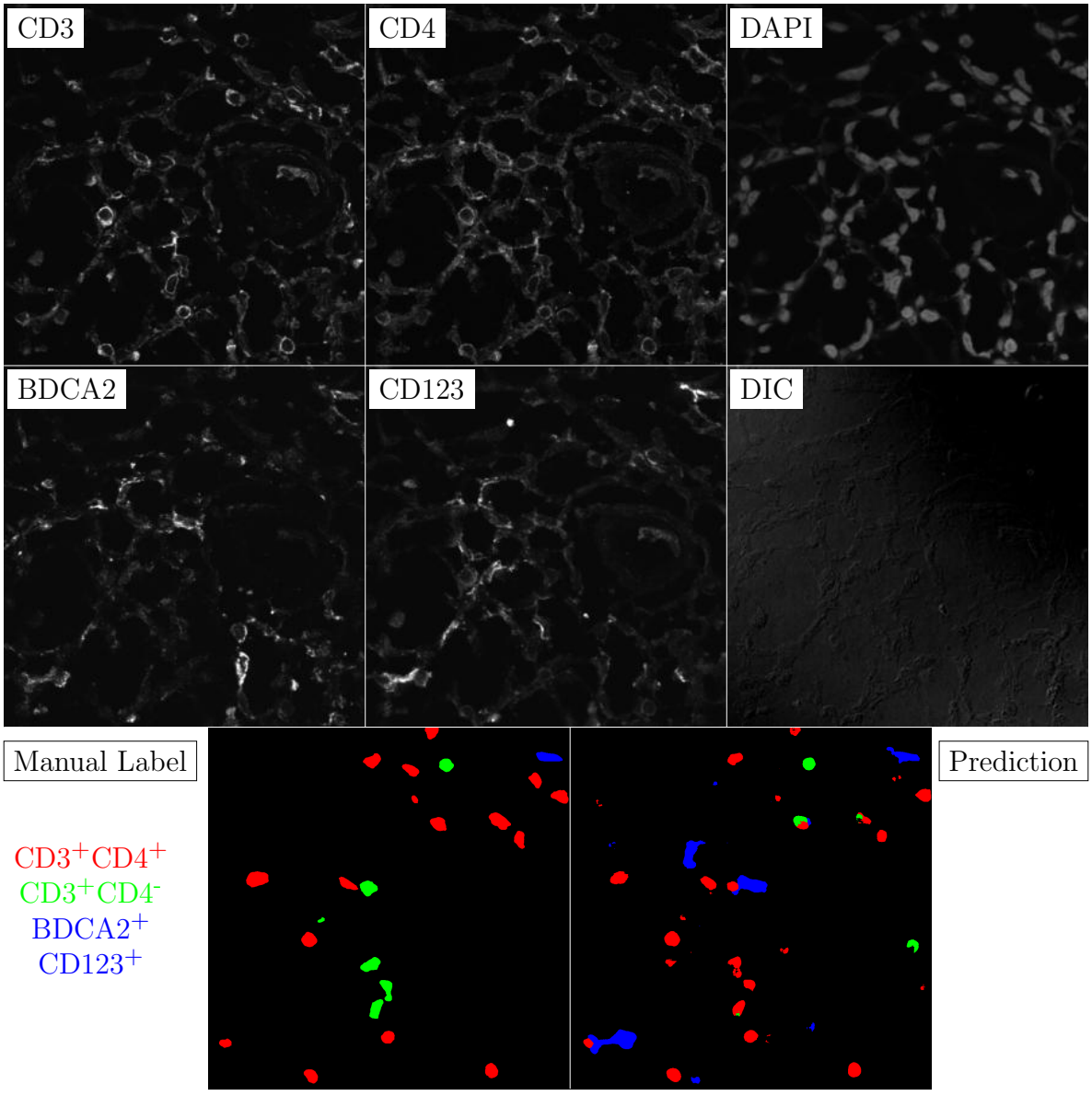

Figure G.113: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

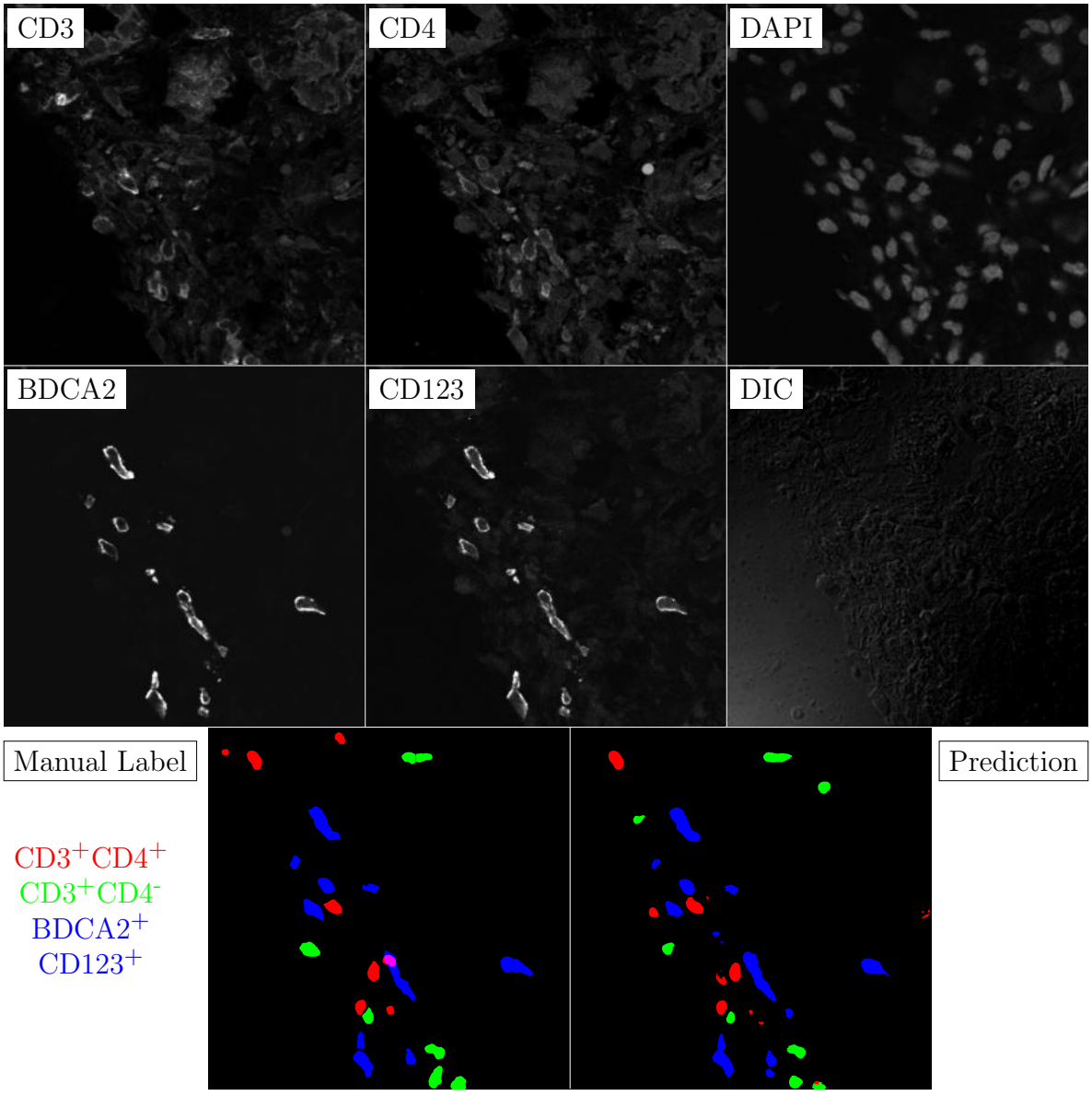

Figure G.114: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

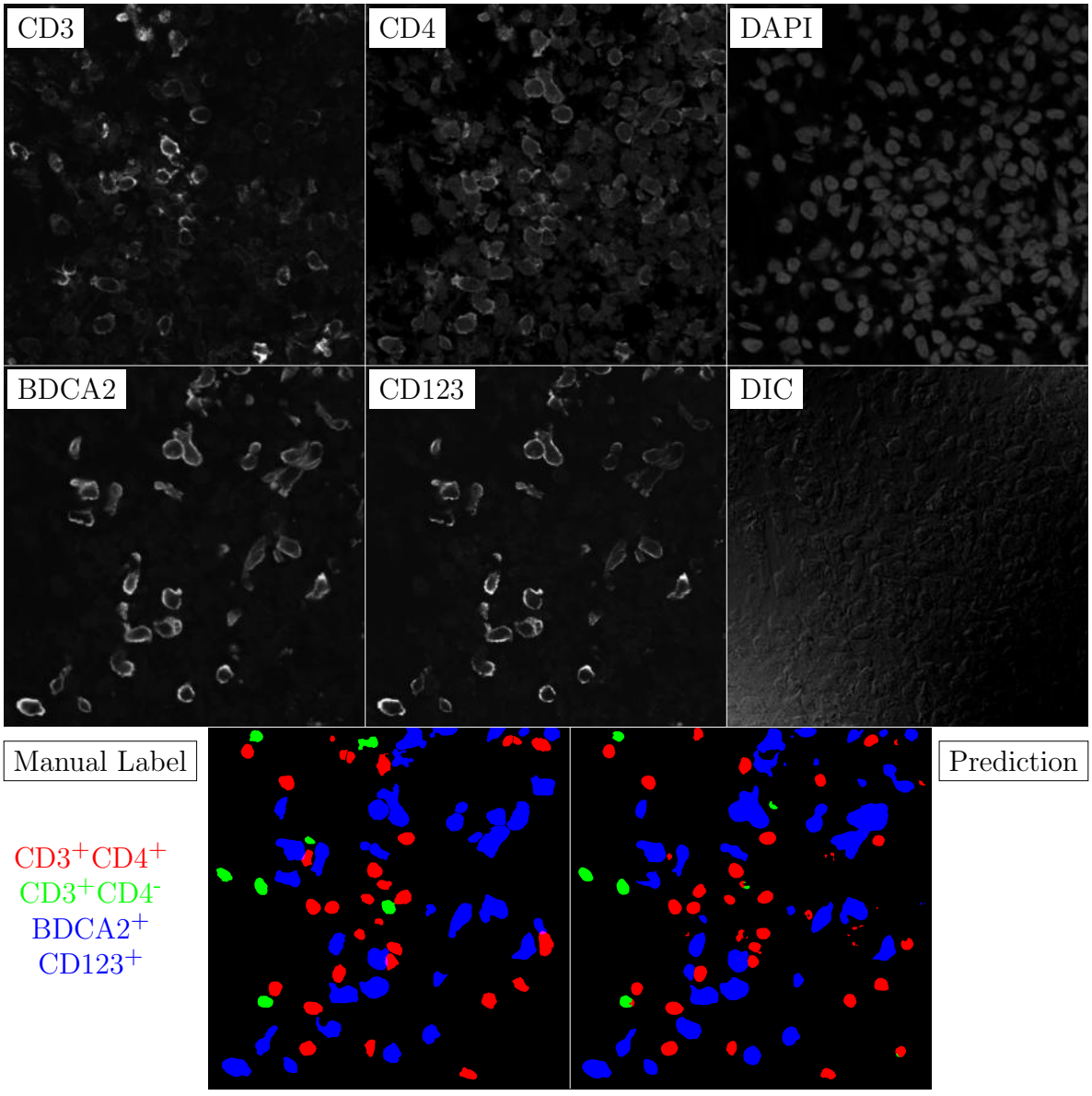

Figure G.115: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

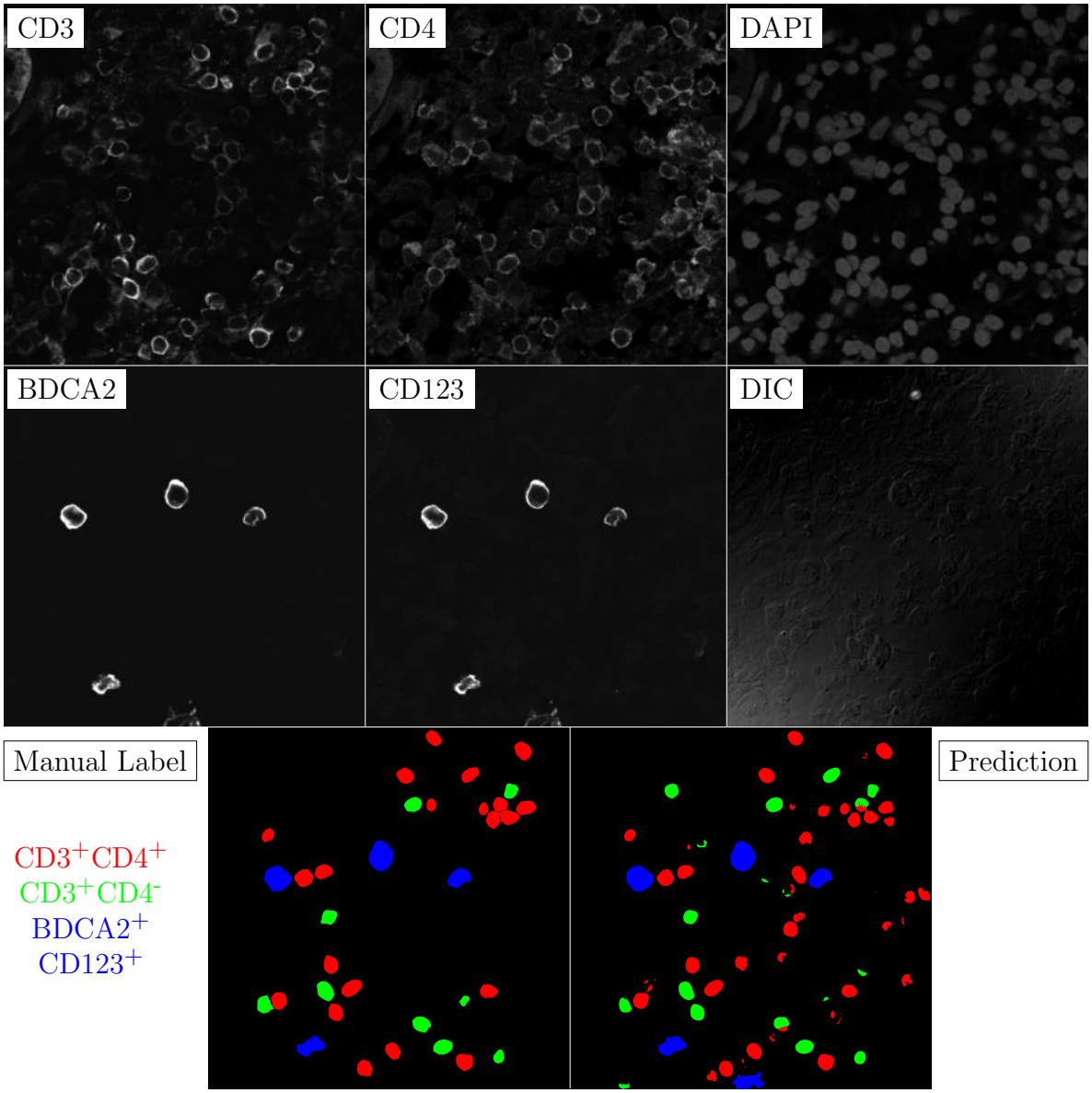

Figure G.116: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

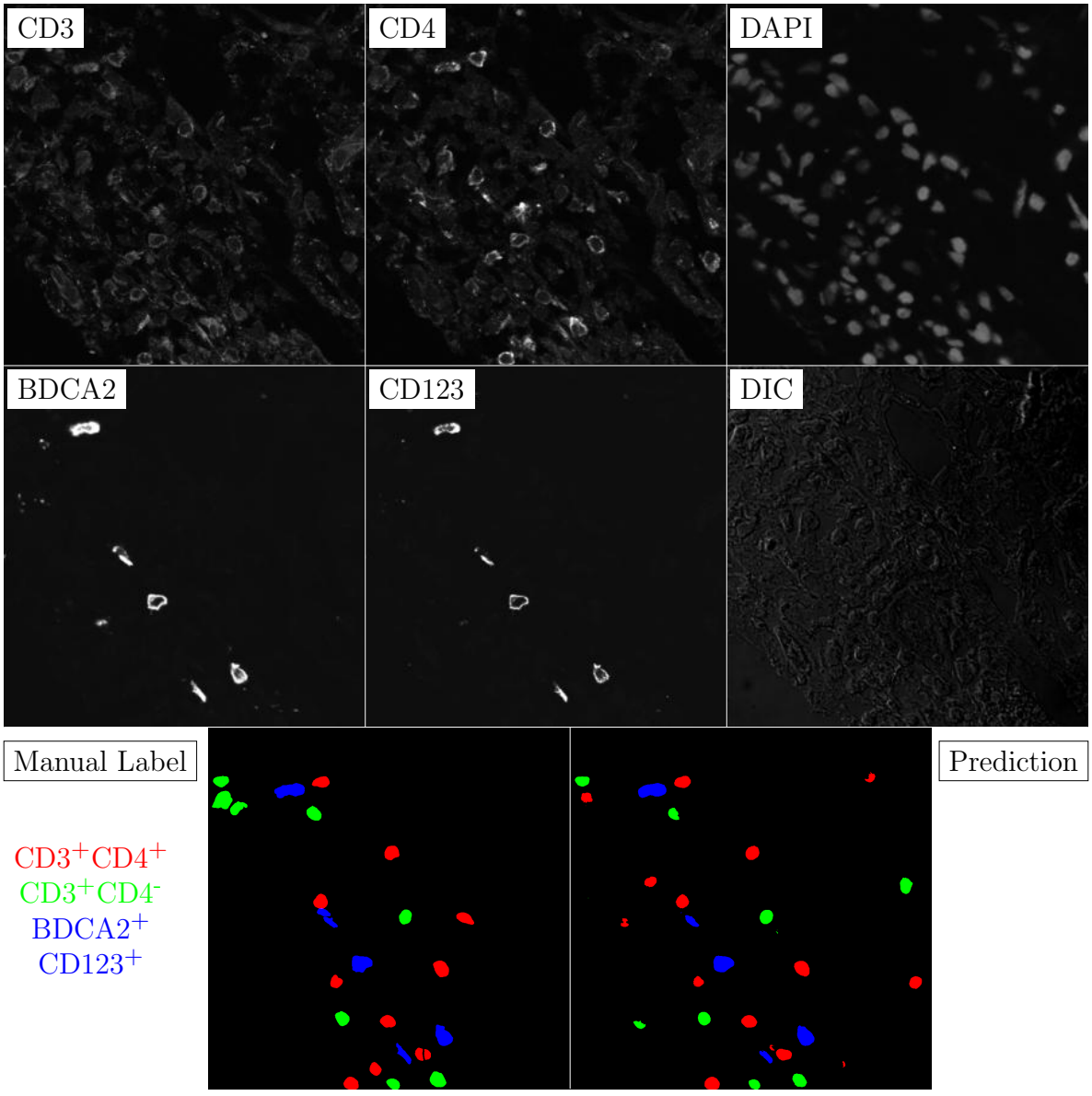

Figure G.117: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

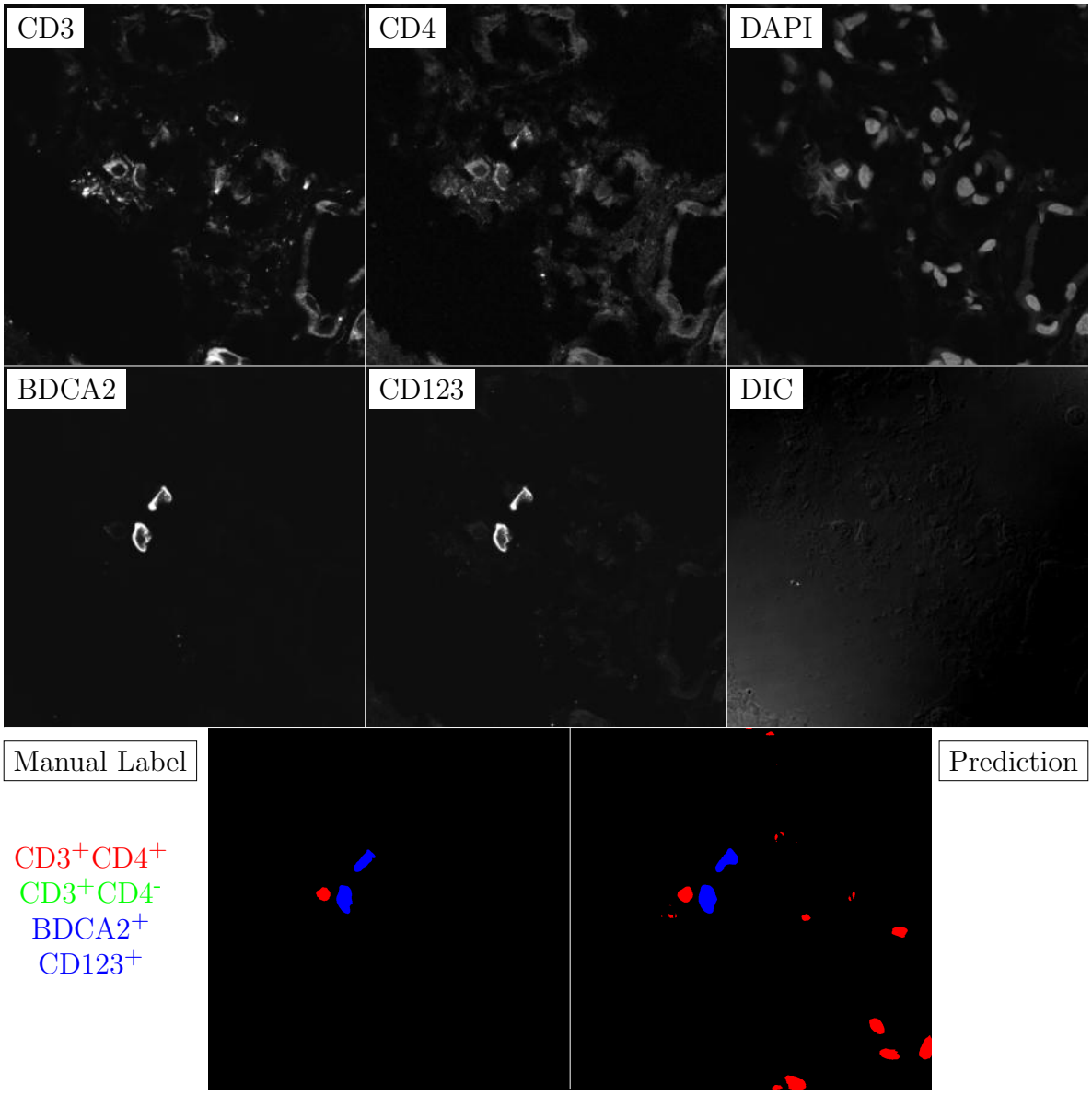

Figure G.118: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

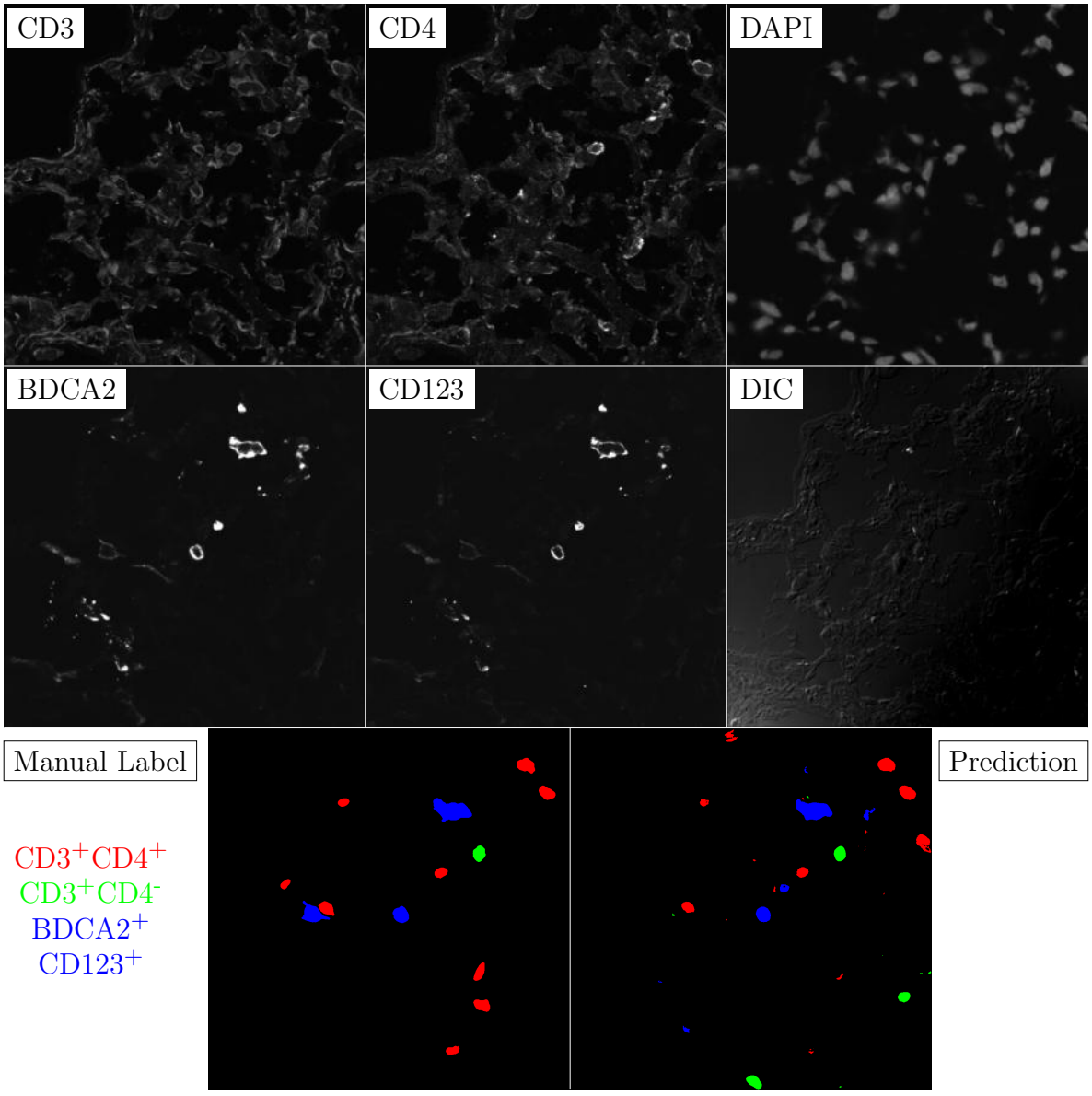

Figure G.119: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

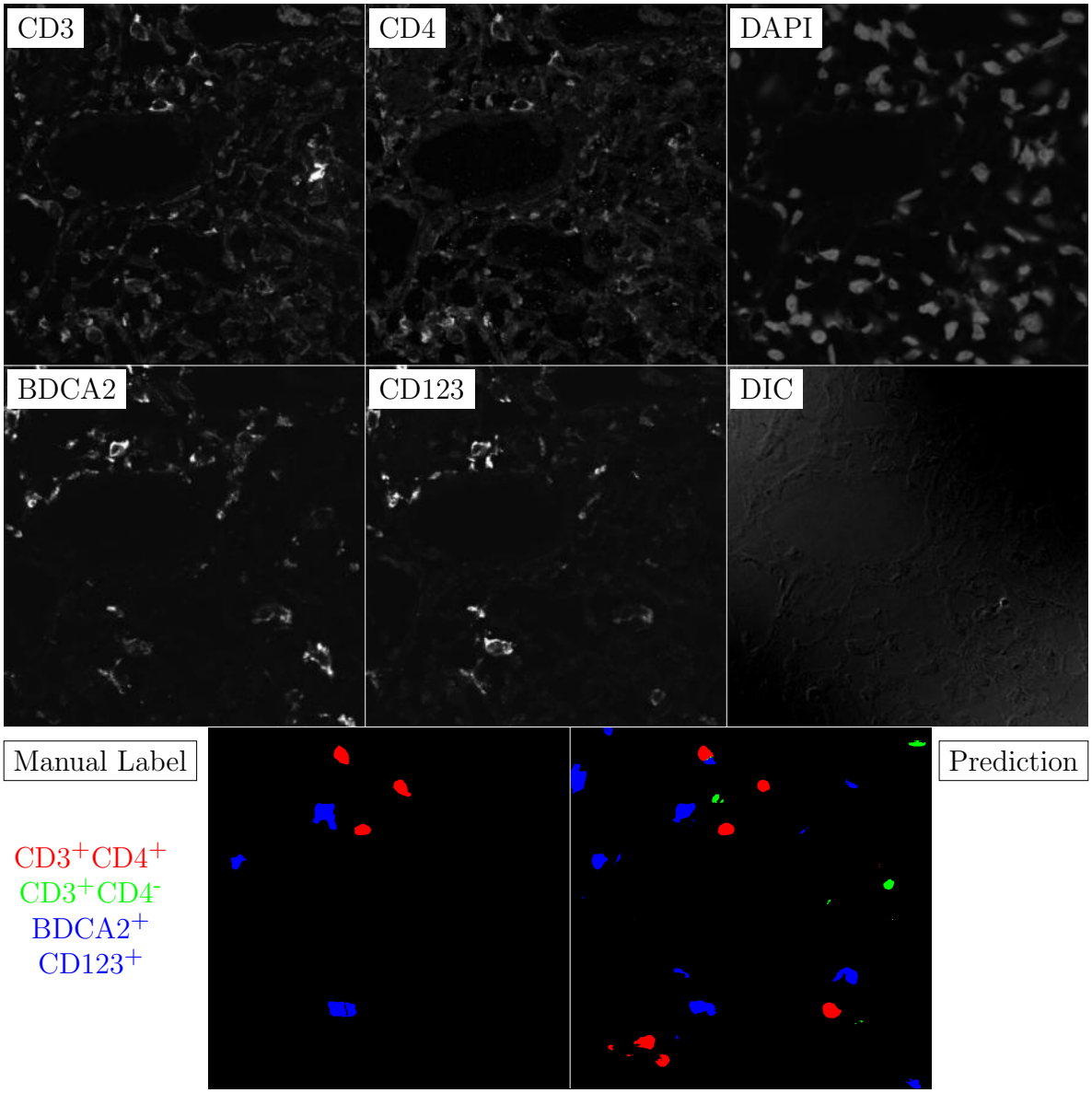

Figure G.120: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

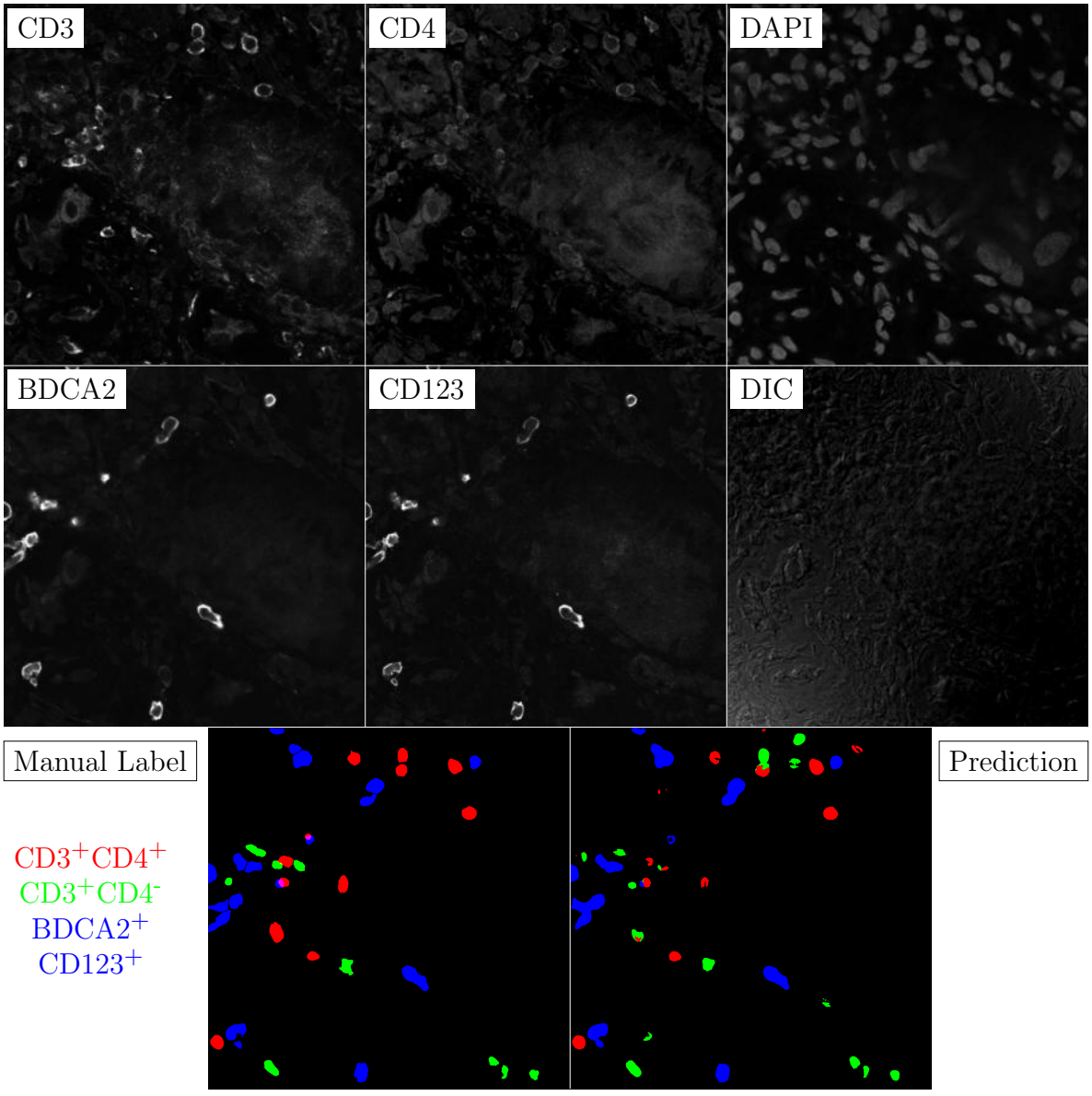

Figure G.121: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

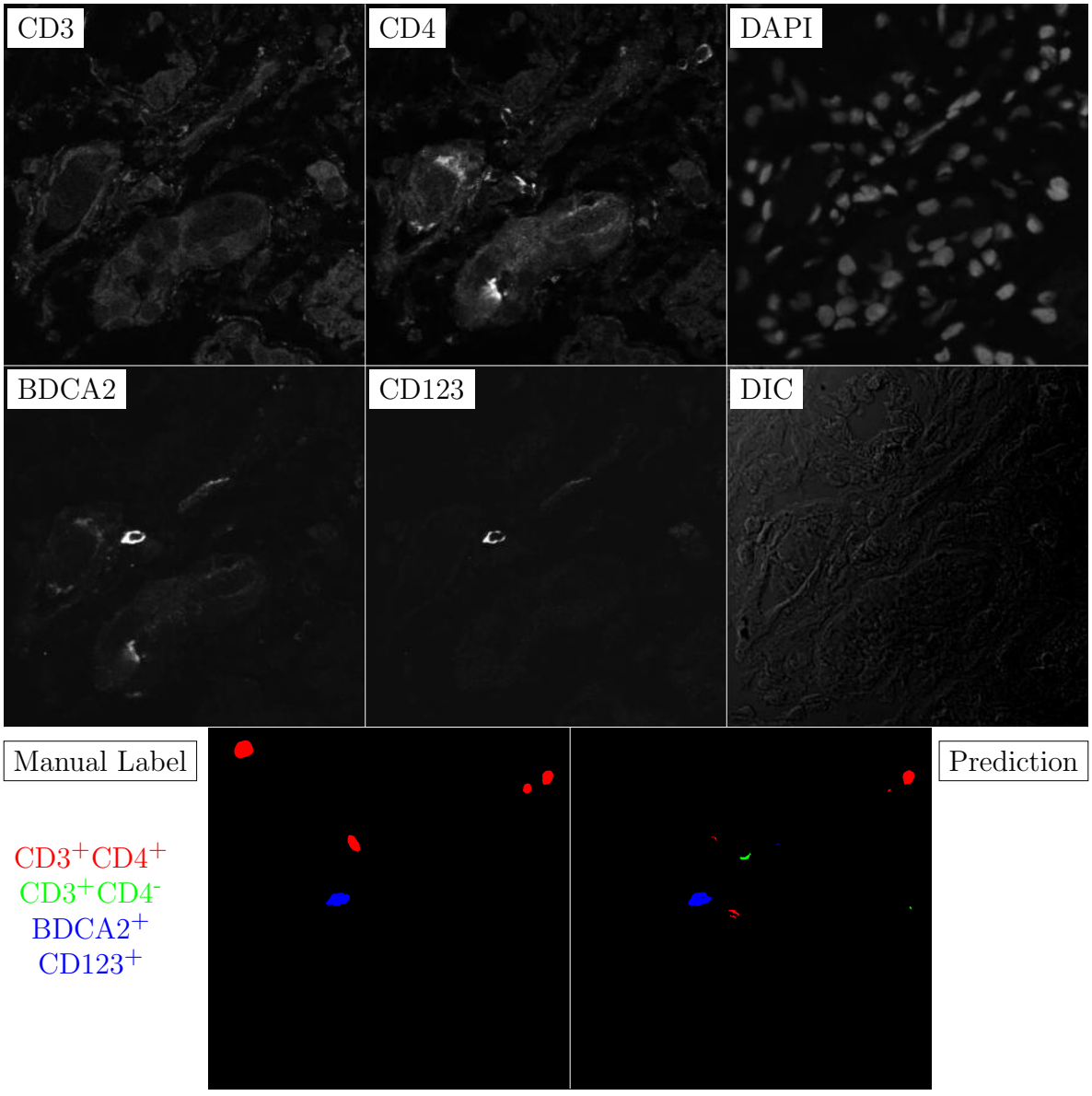

Figure G.122: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

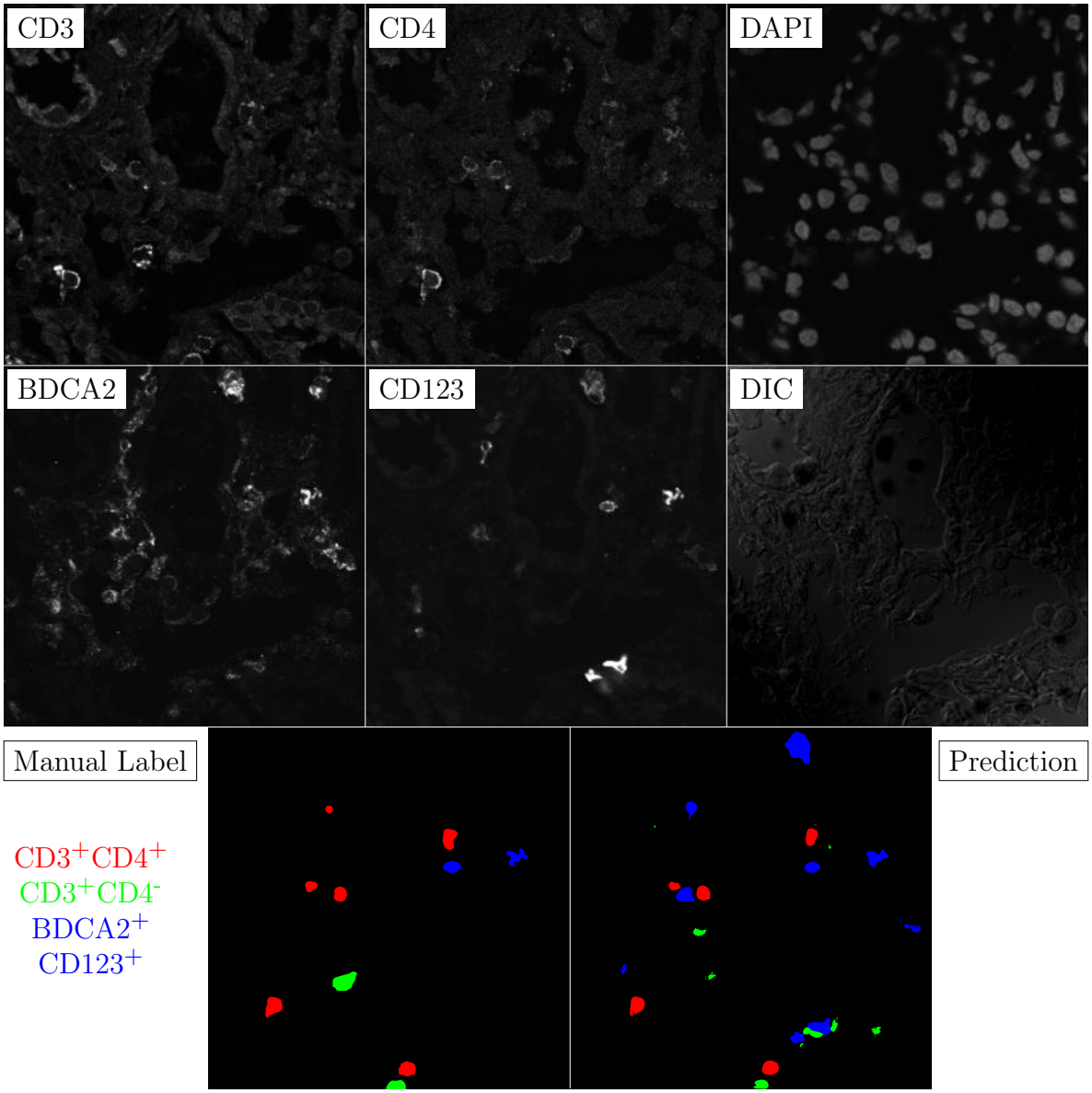

Figure G.123: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

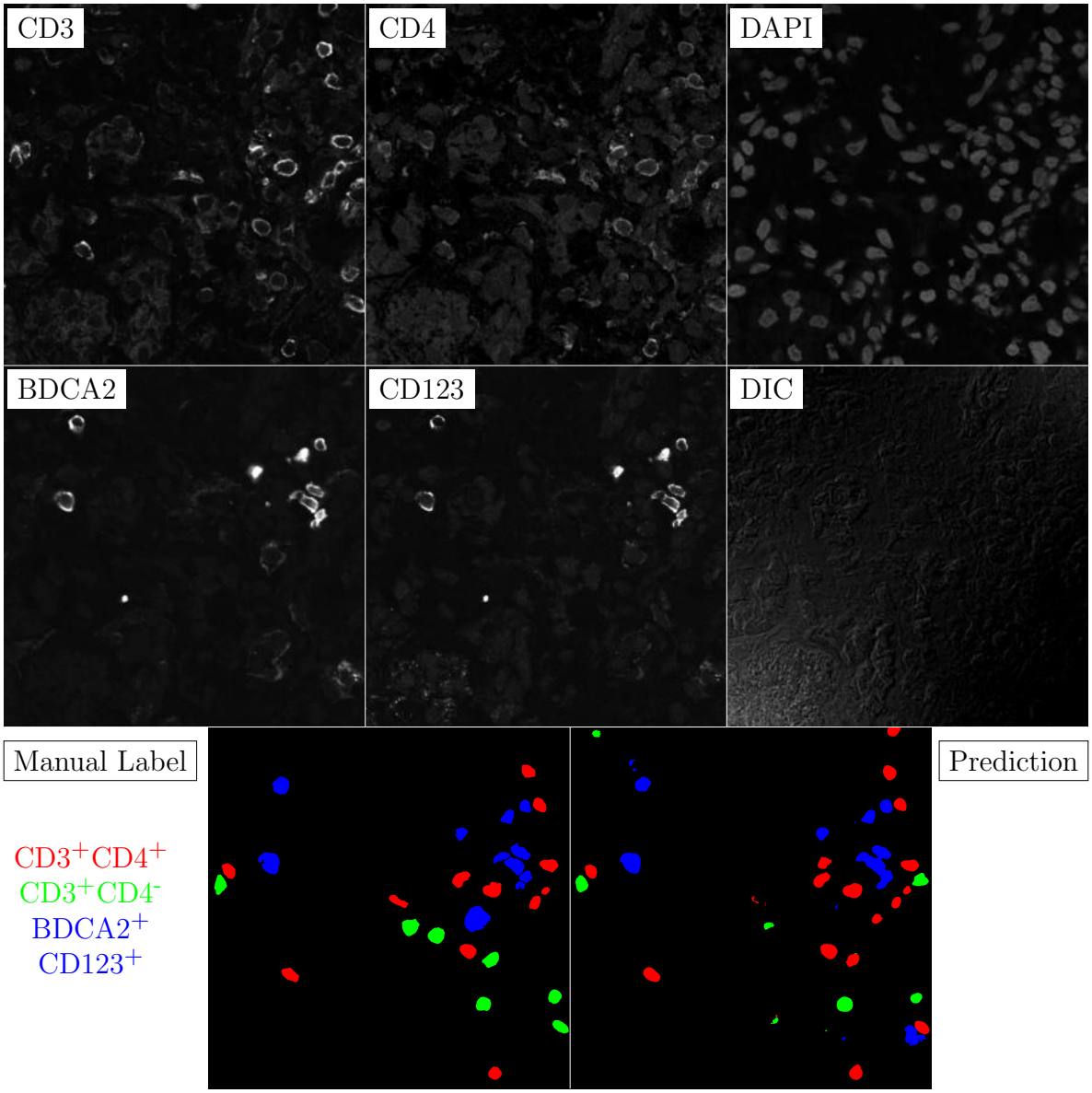

Figure G.124: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

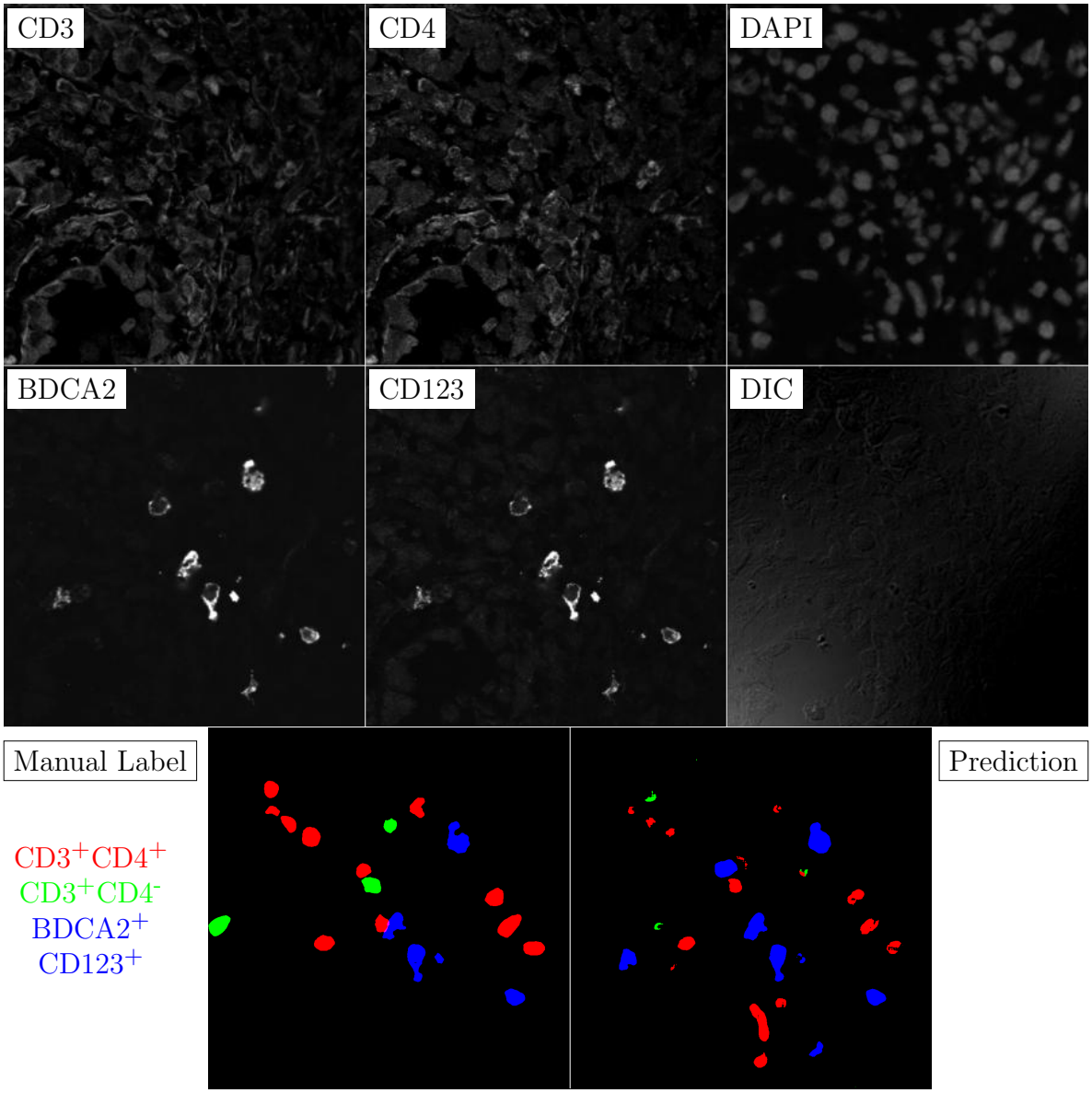

Figure G.125: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

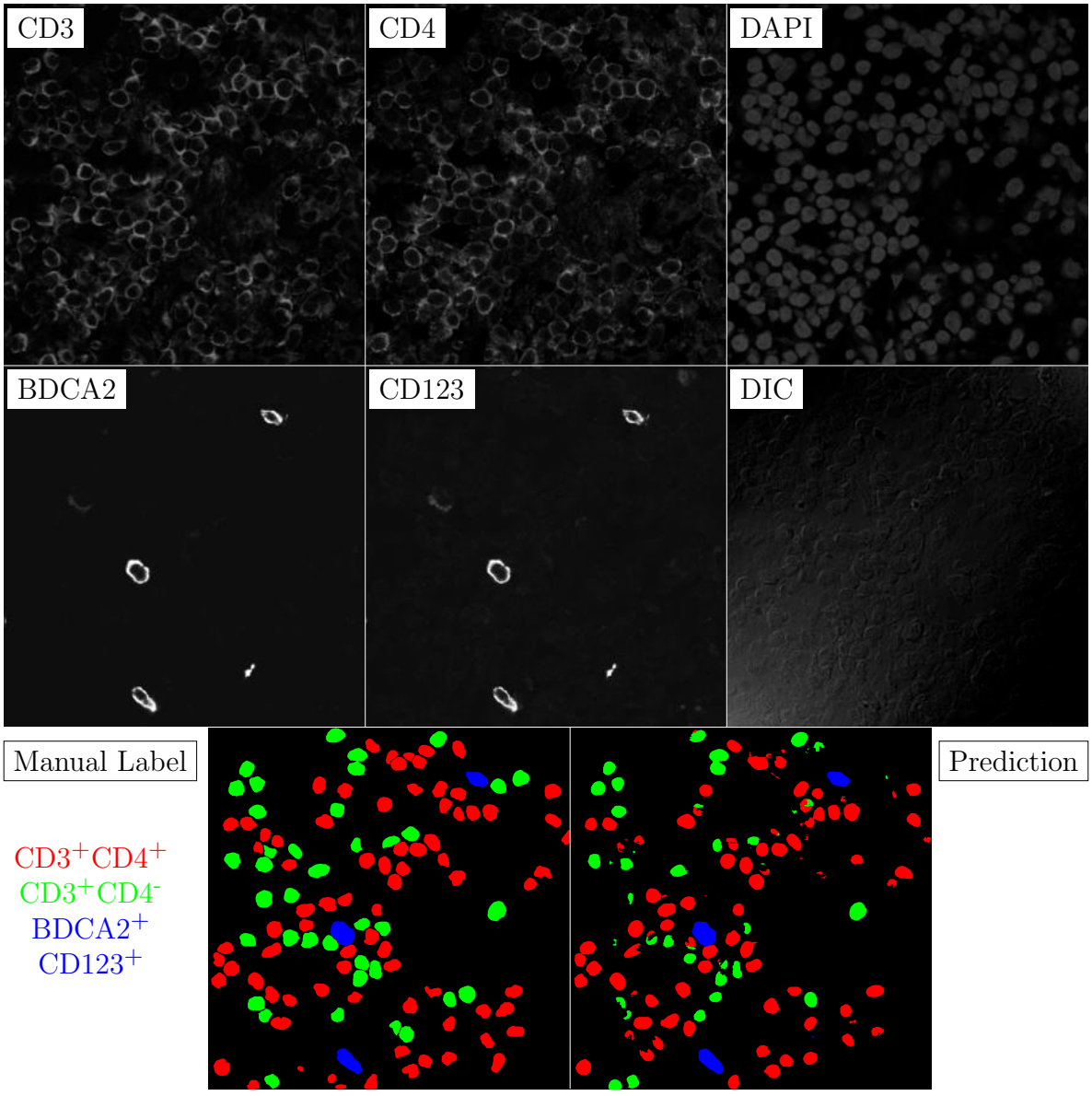

Figure G.126: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

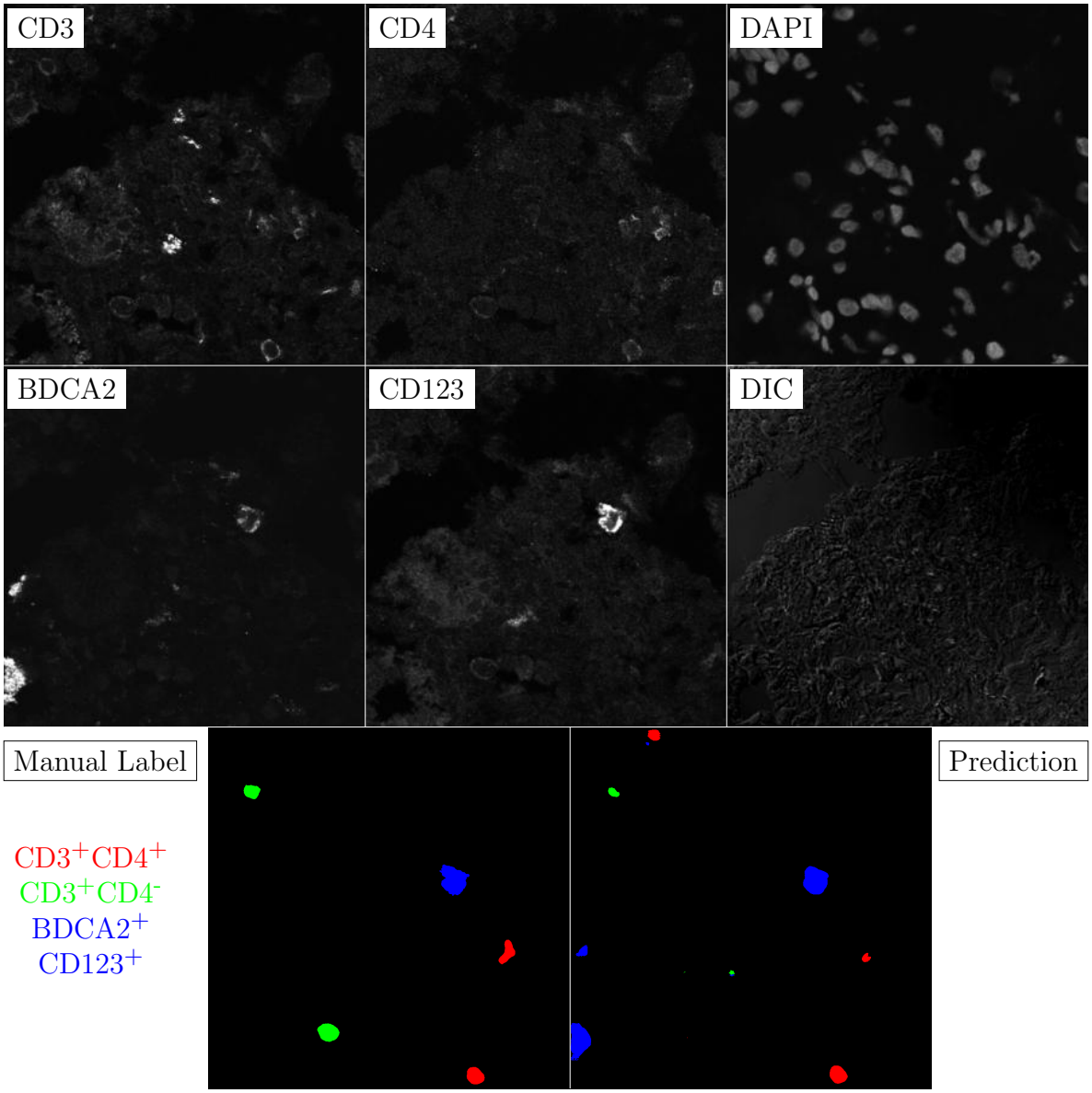

Figure G.127: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

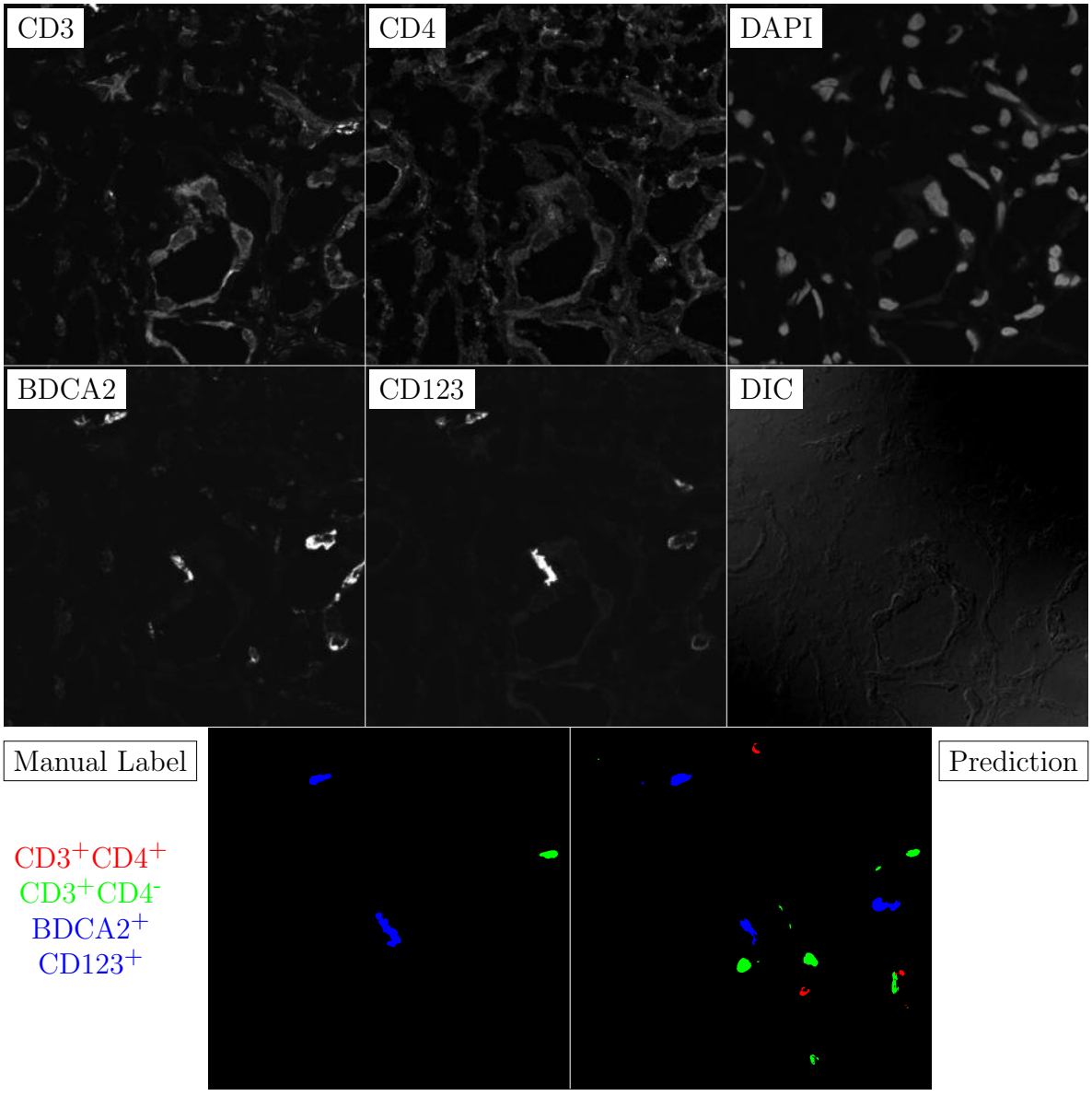

Figure G.128: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

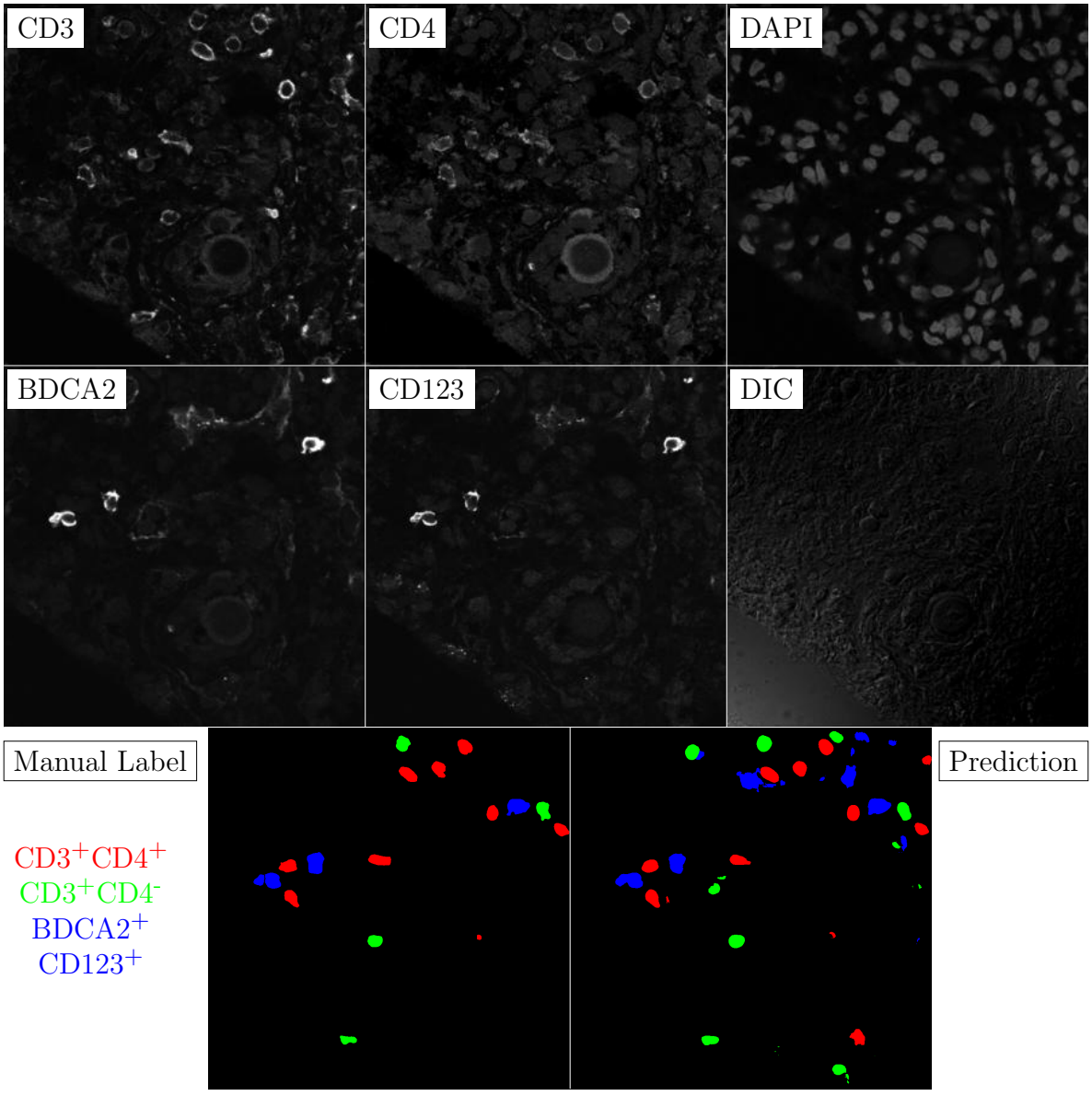

Figure G.129: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

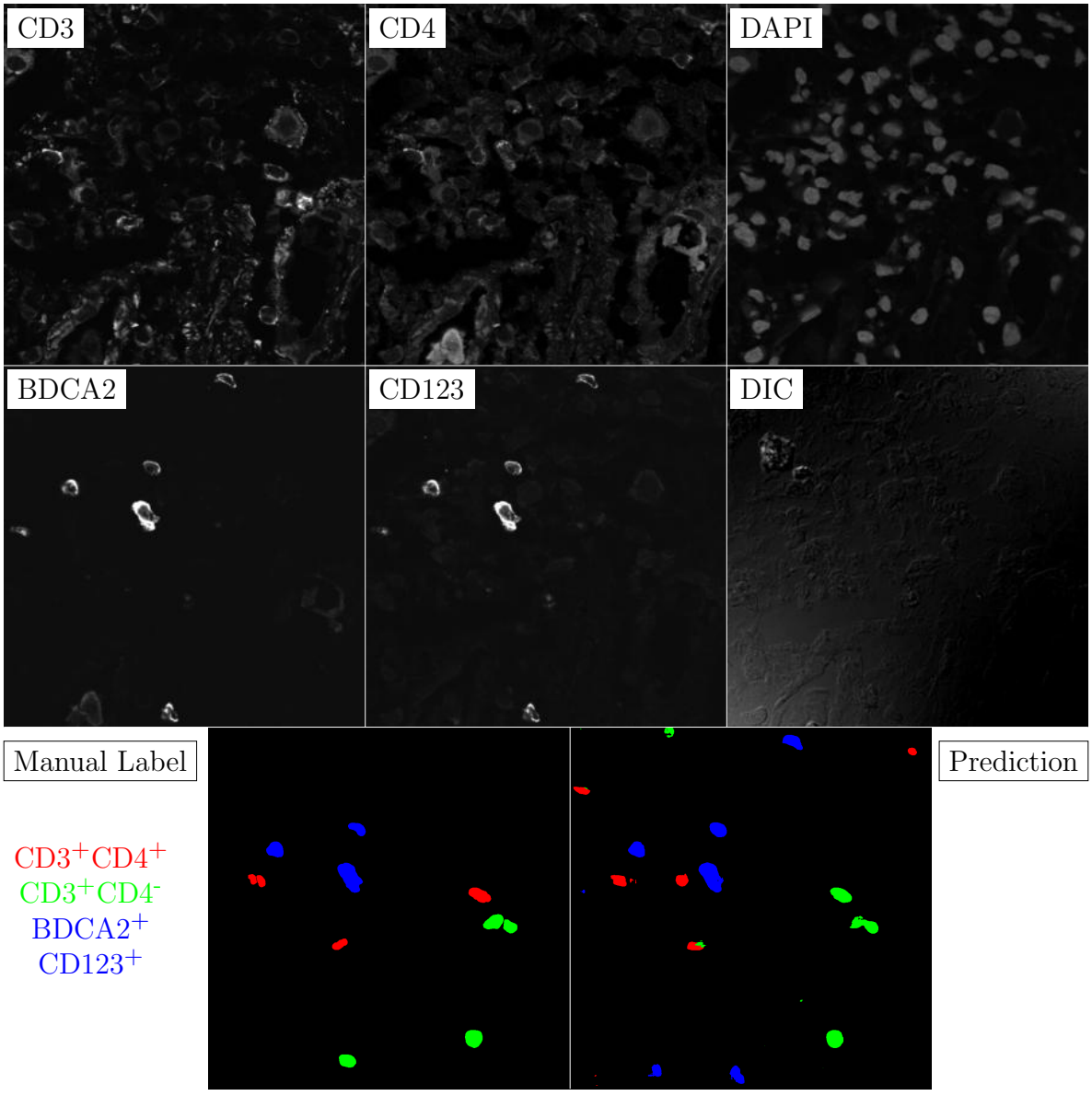

Figure G.130: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

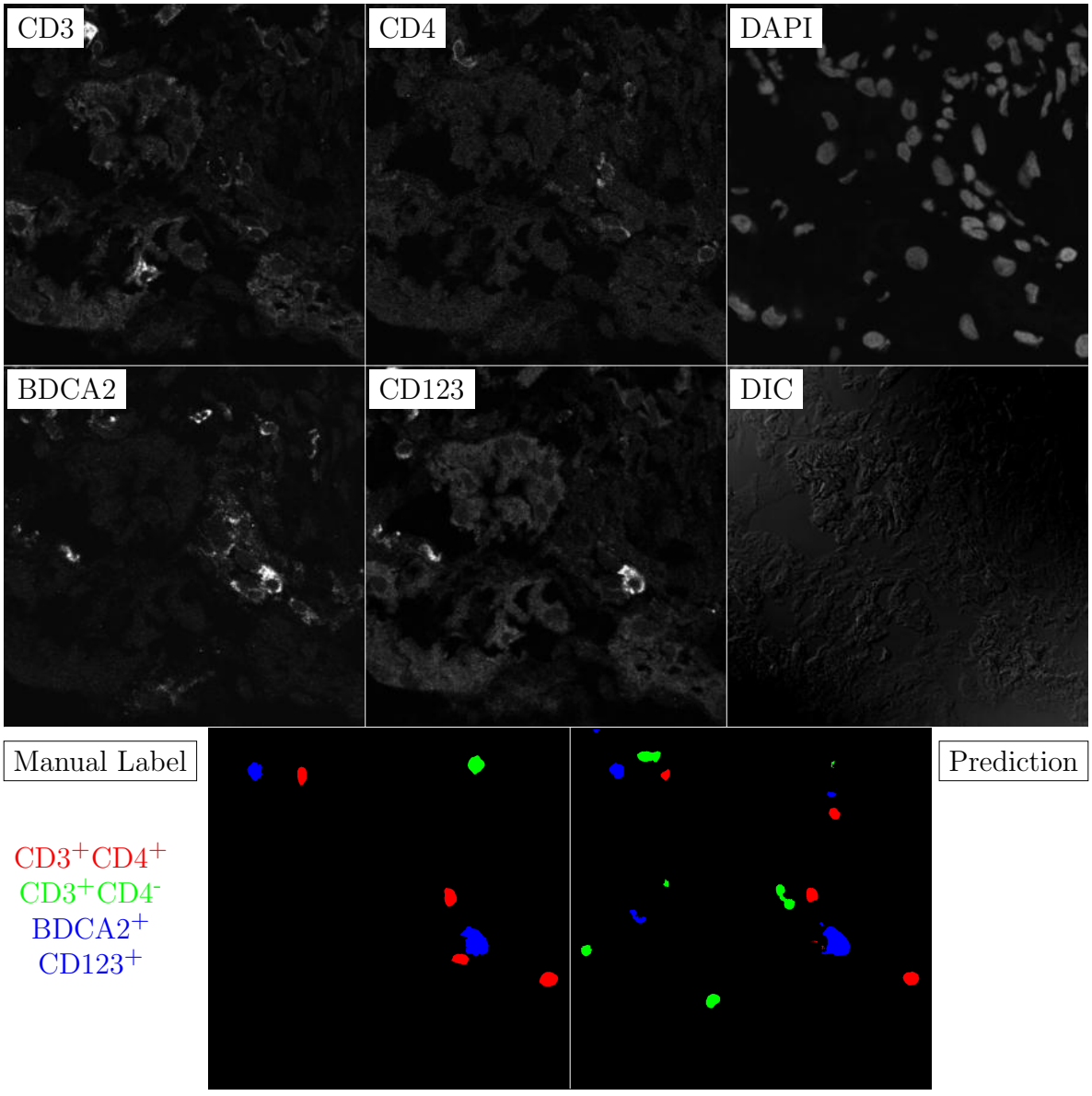

Figure G.131: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

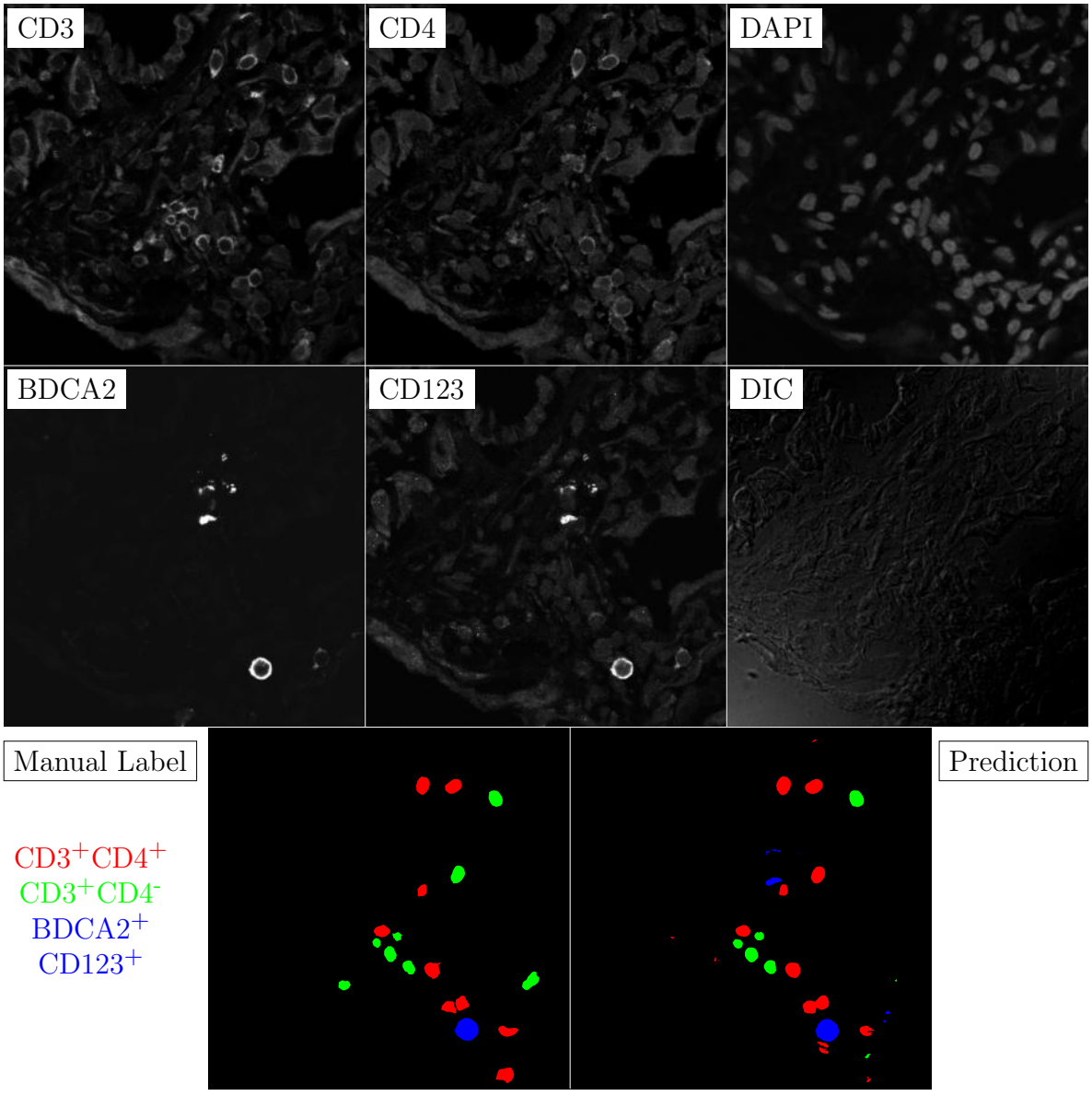

Figure G.132: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

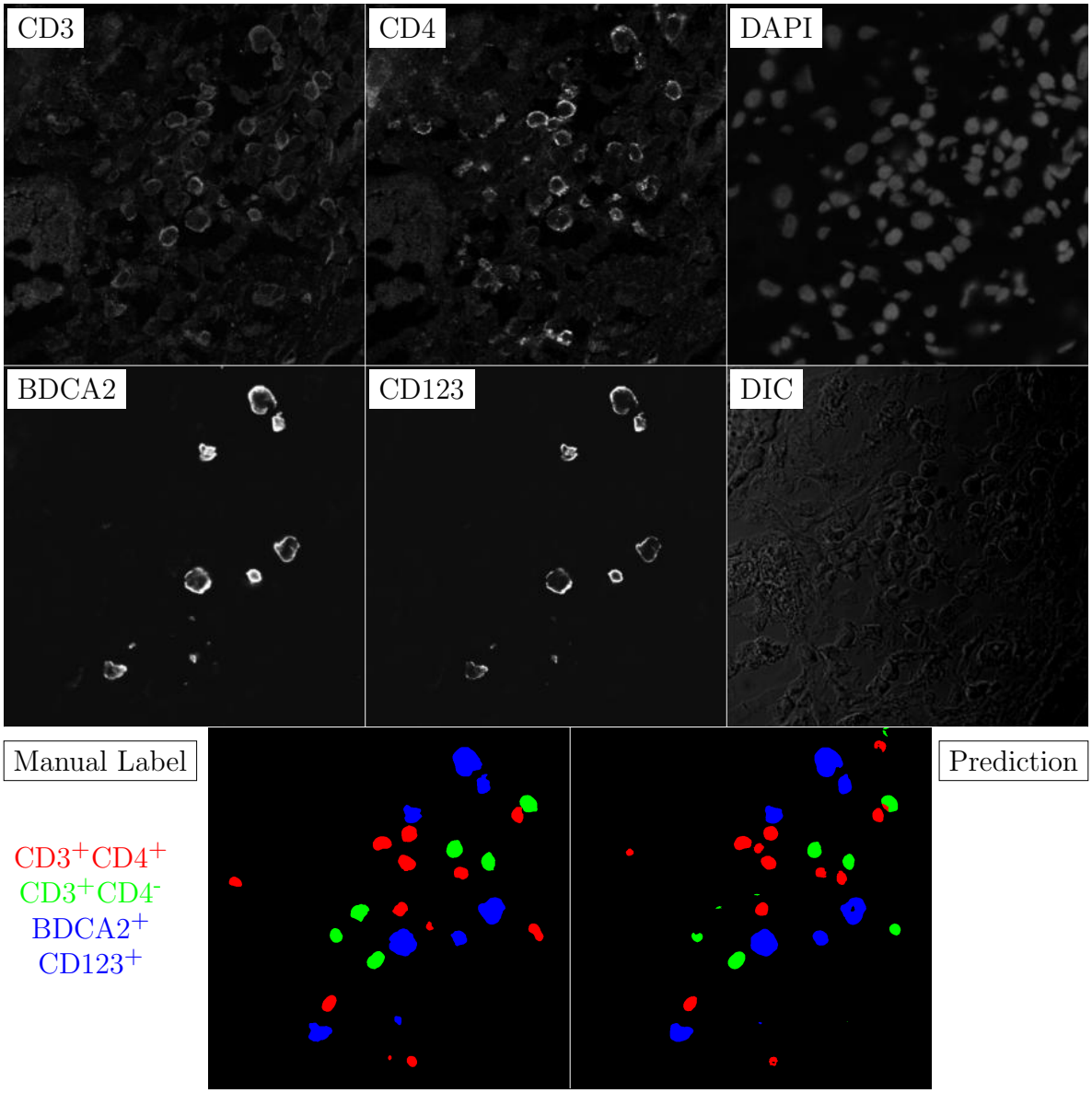

Figure G.133: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

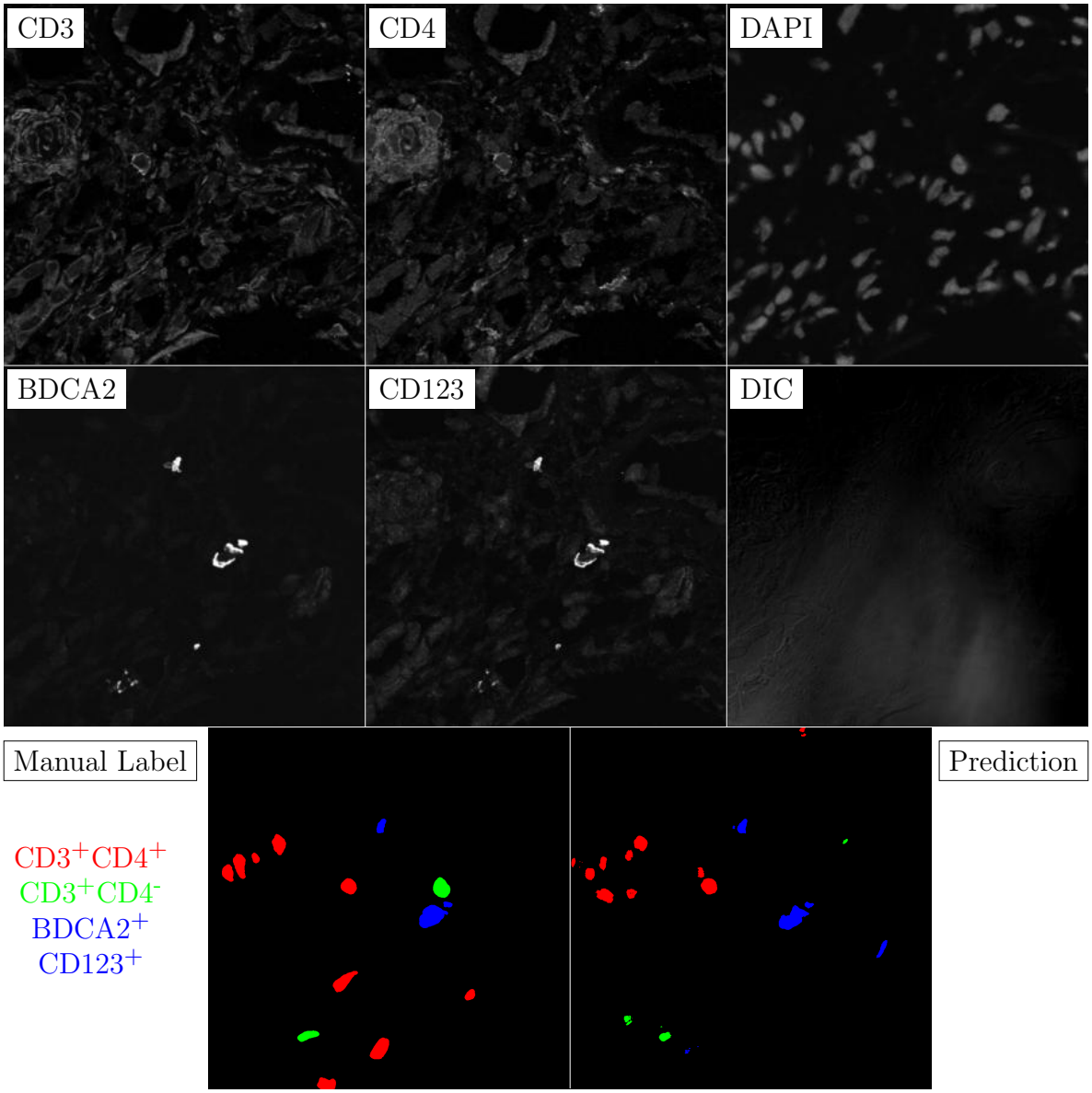

Figure G.134: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

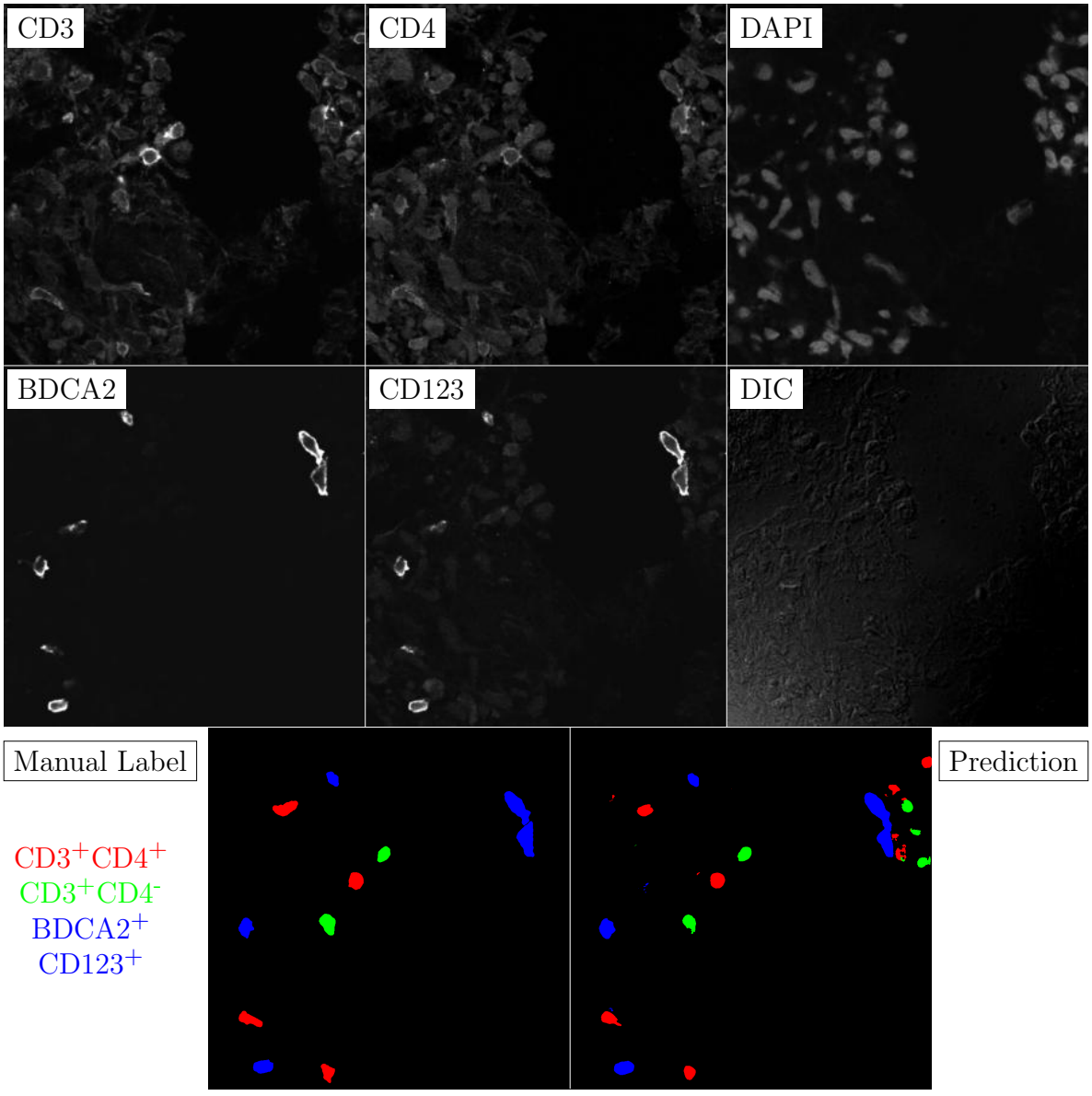

Figure G.135: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

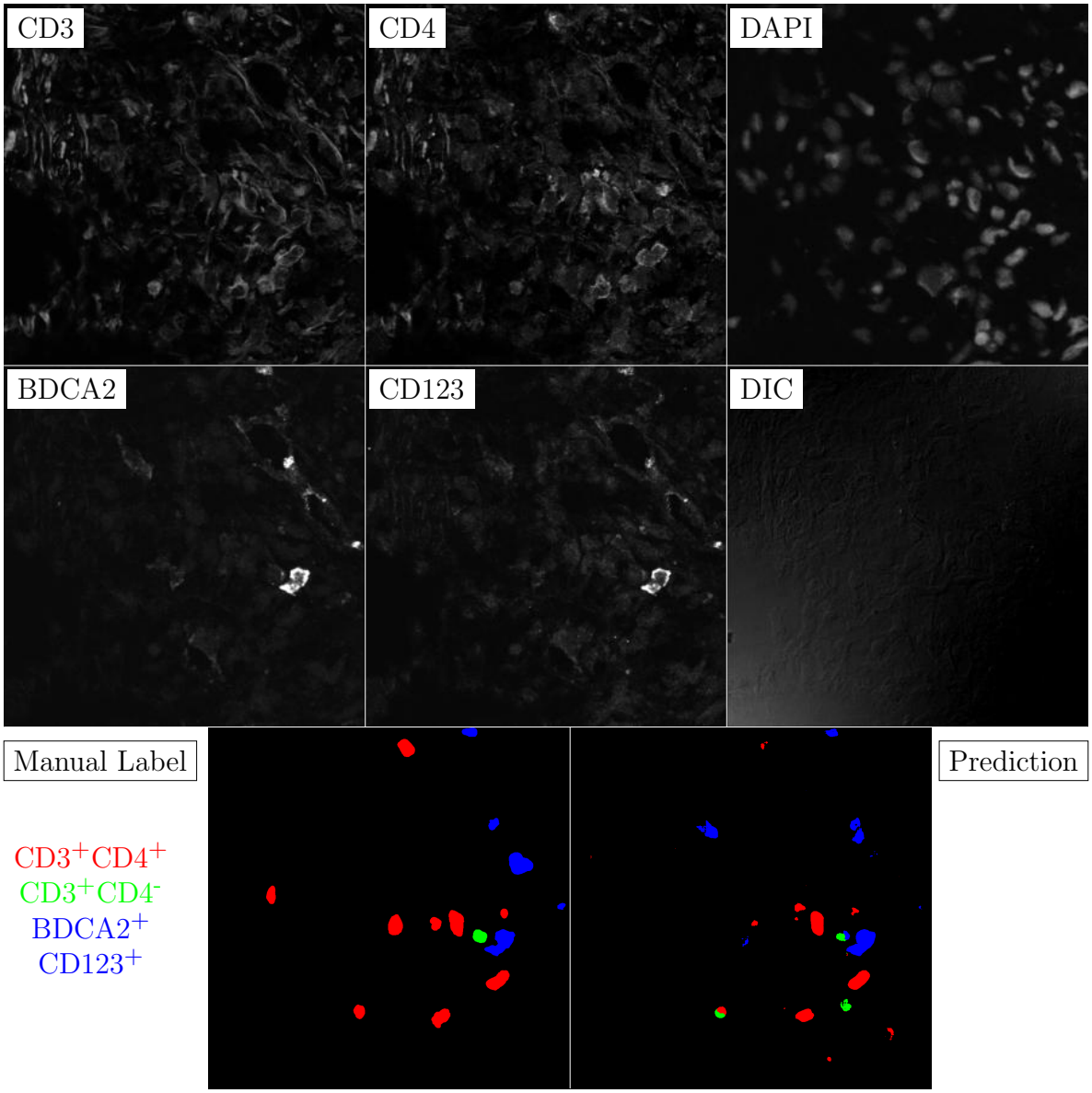

Figure G.136: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

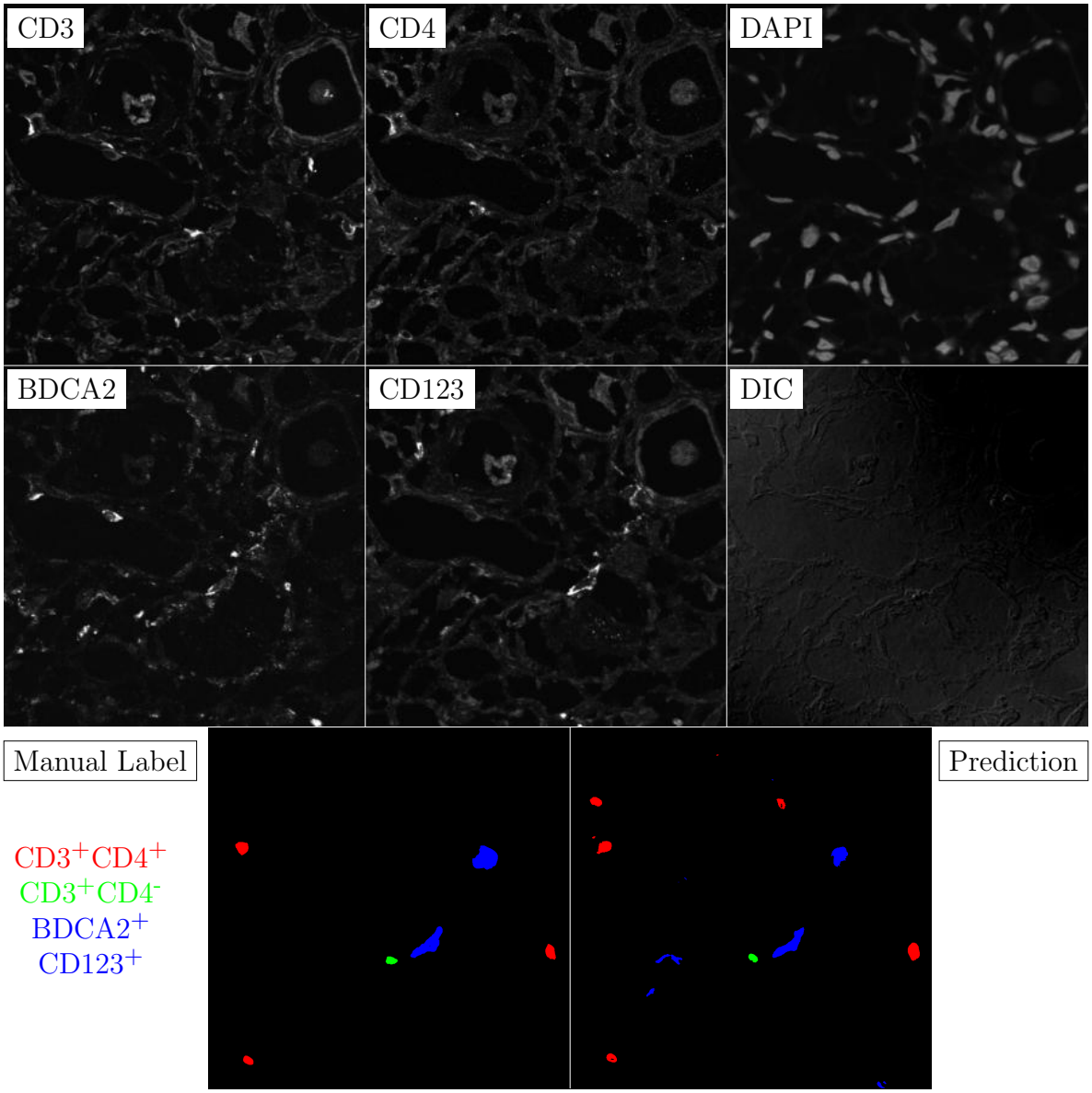

Figure G.137: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

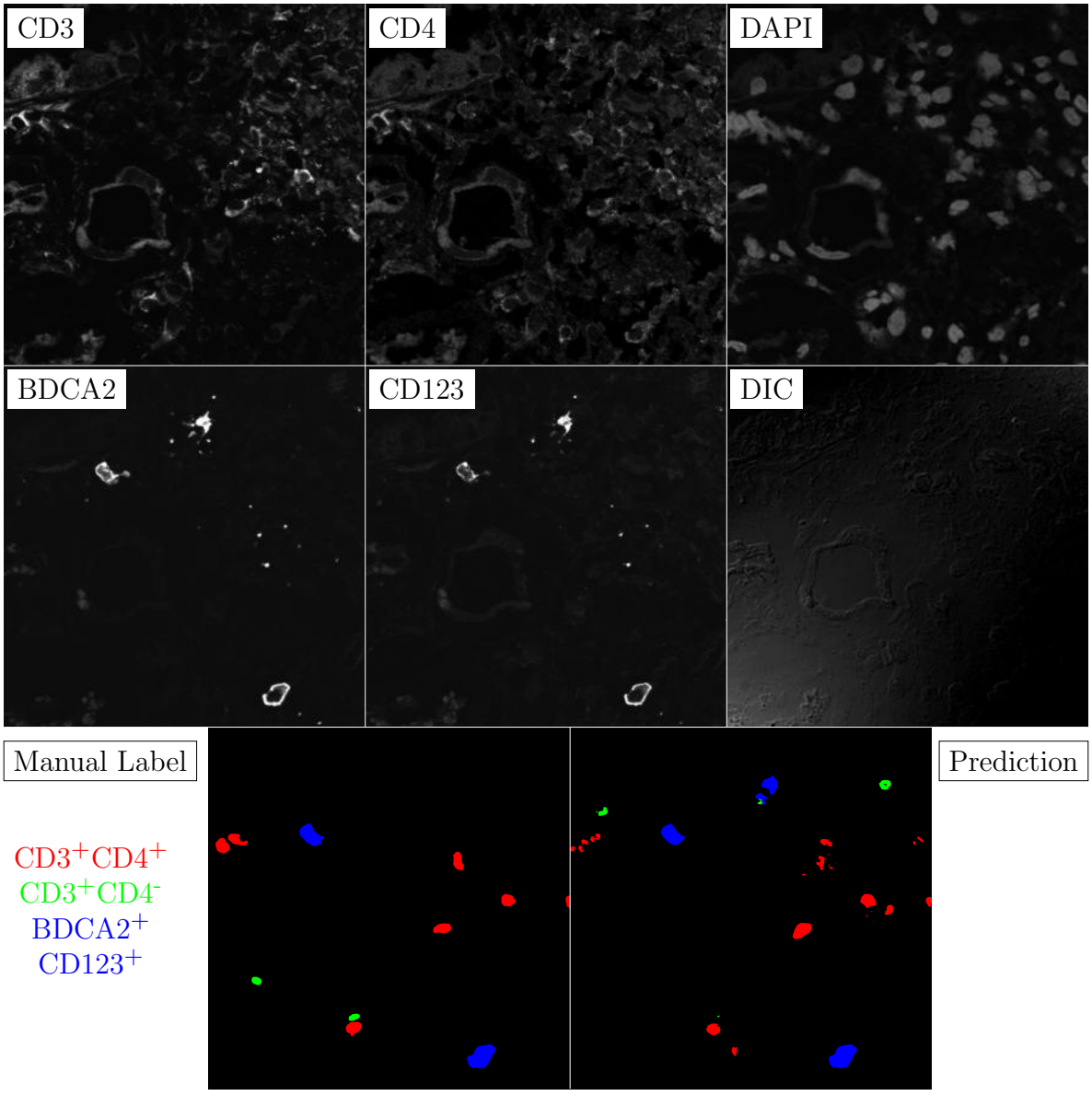

Figure G.138: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

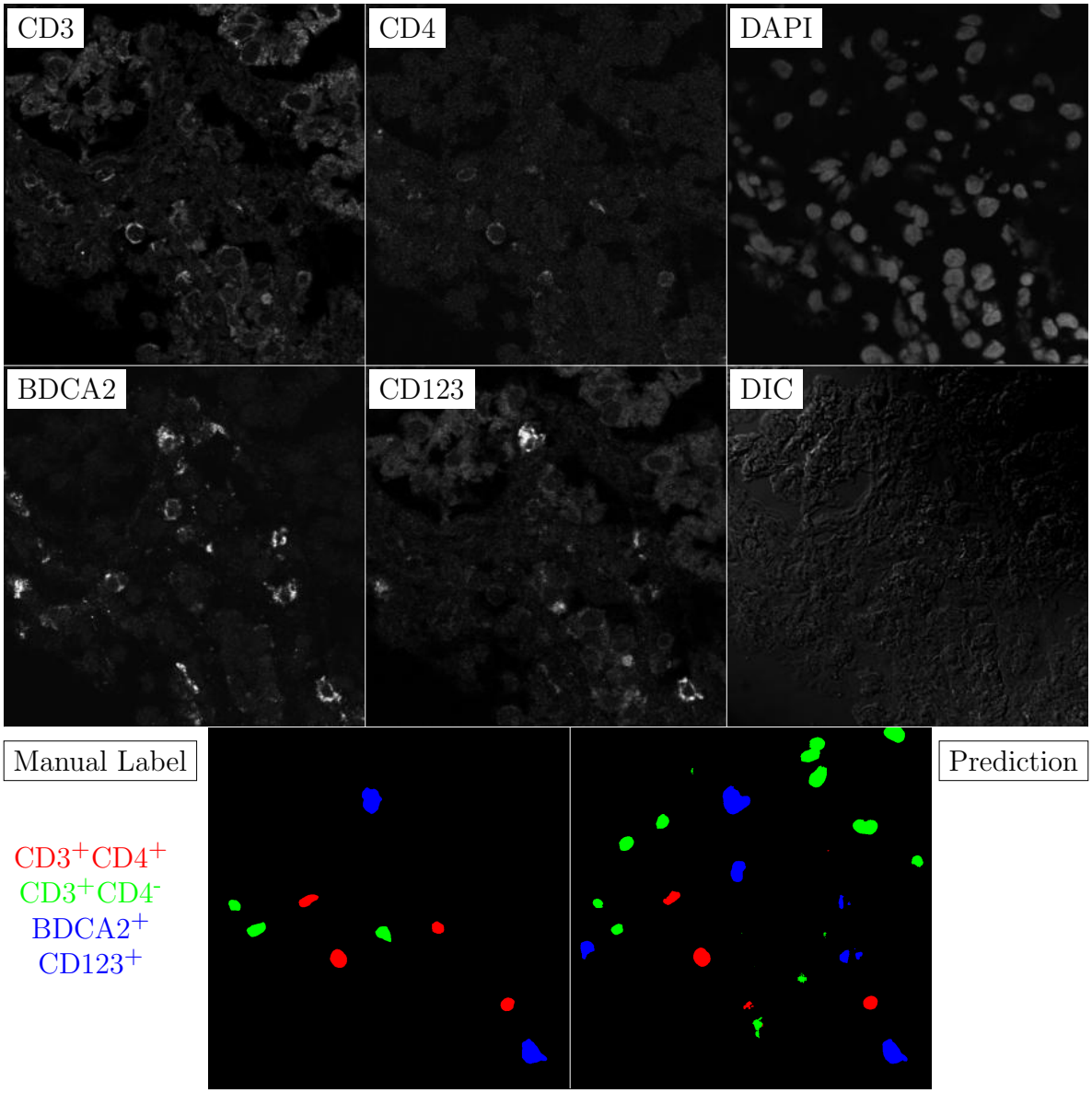

Figure G.139: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

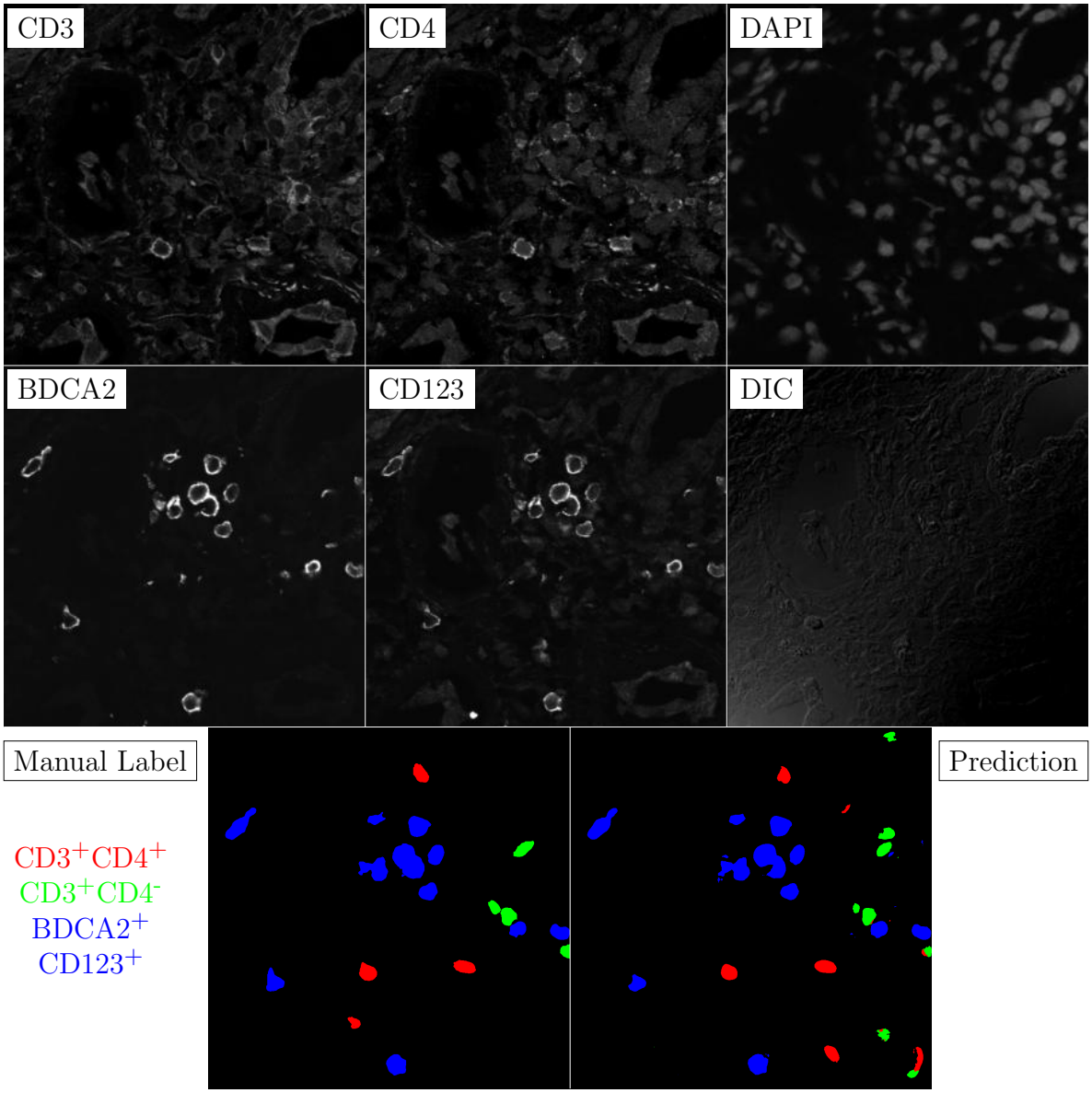

Figure G.140: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

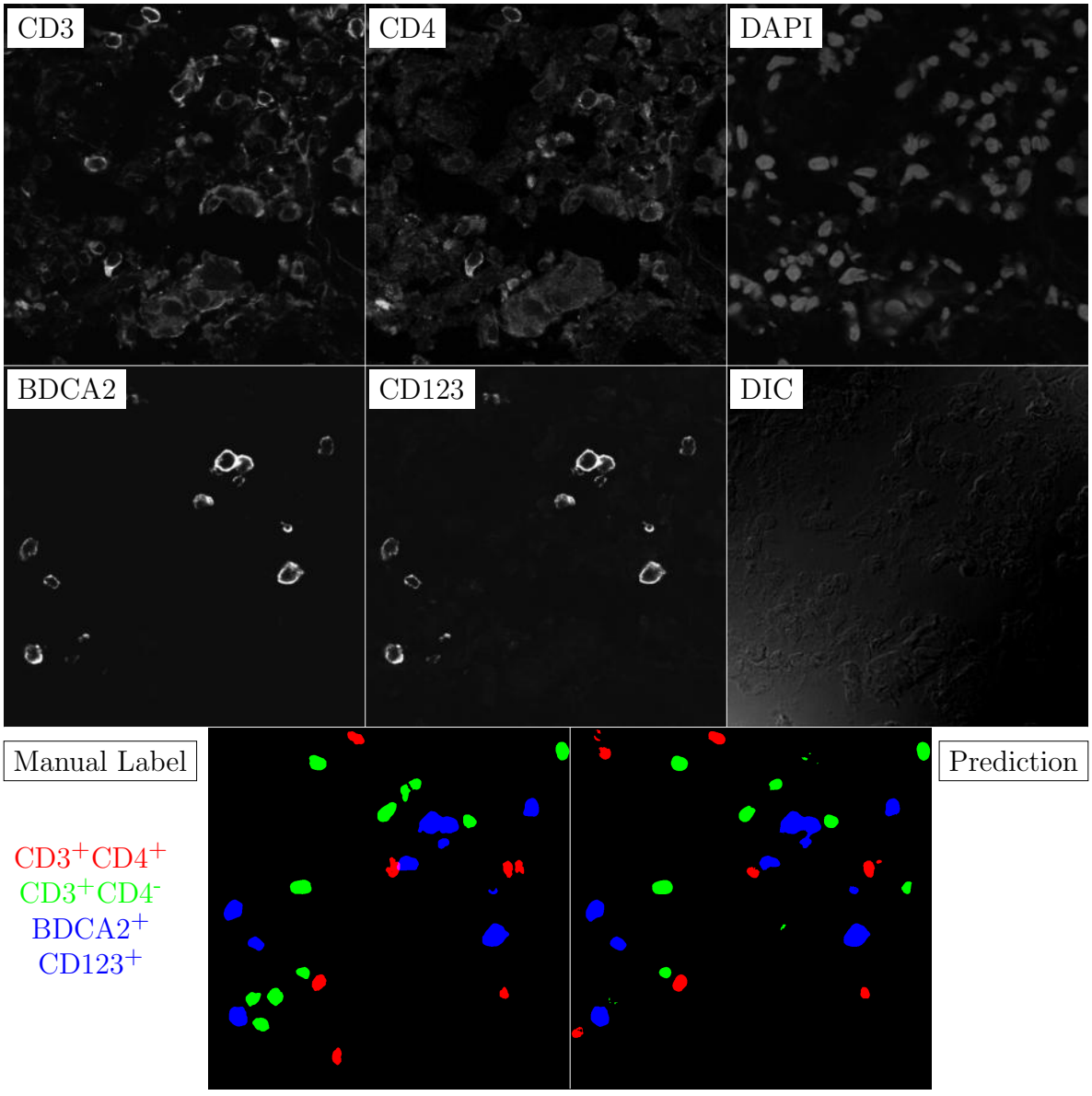

Figure G.141: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

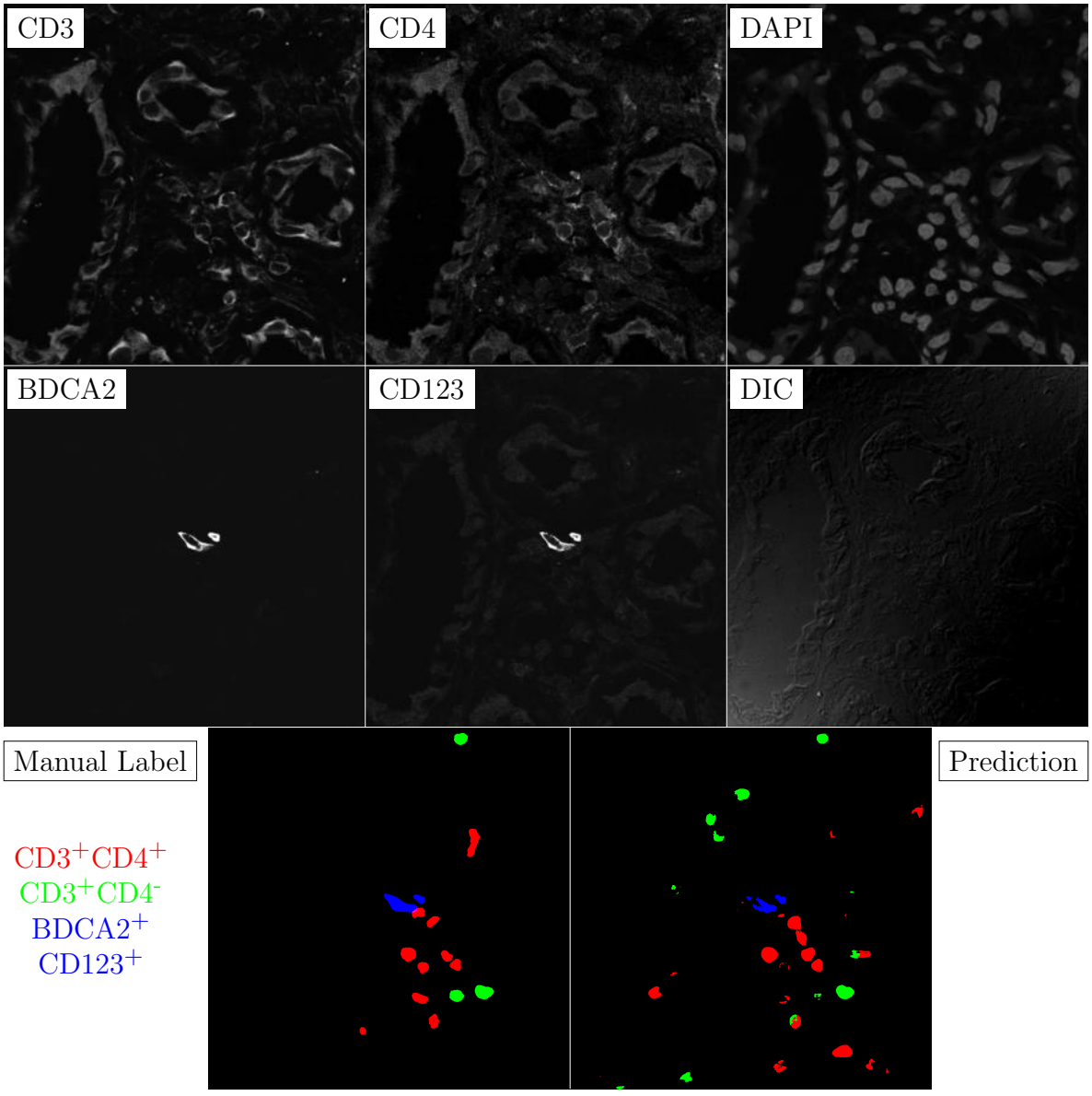

Figure G.142: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

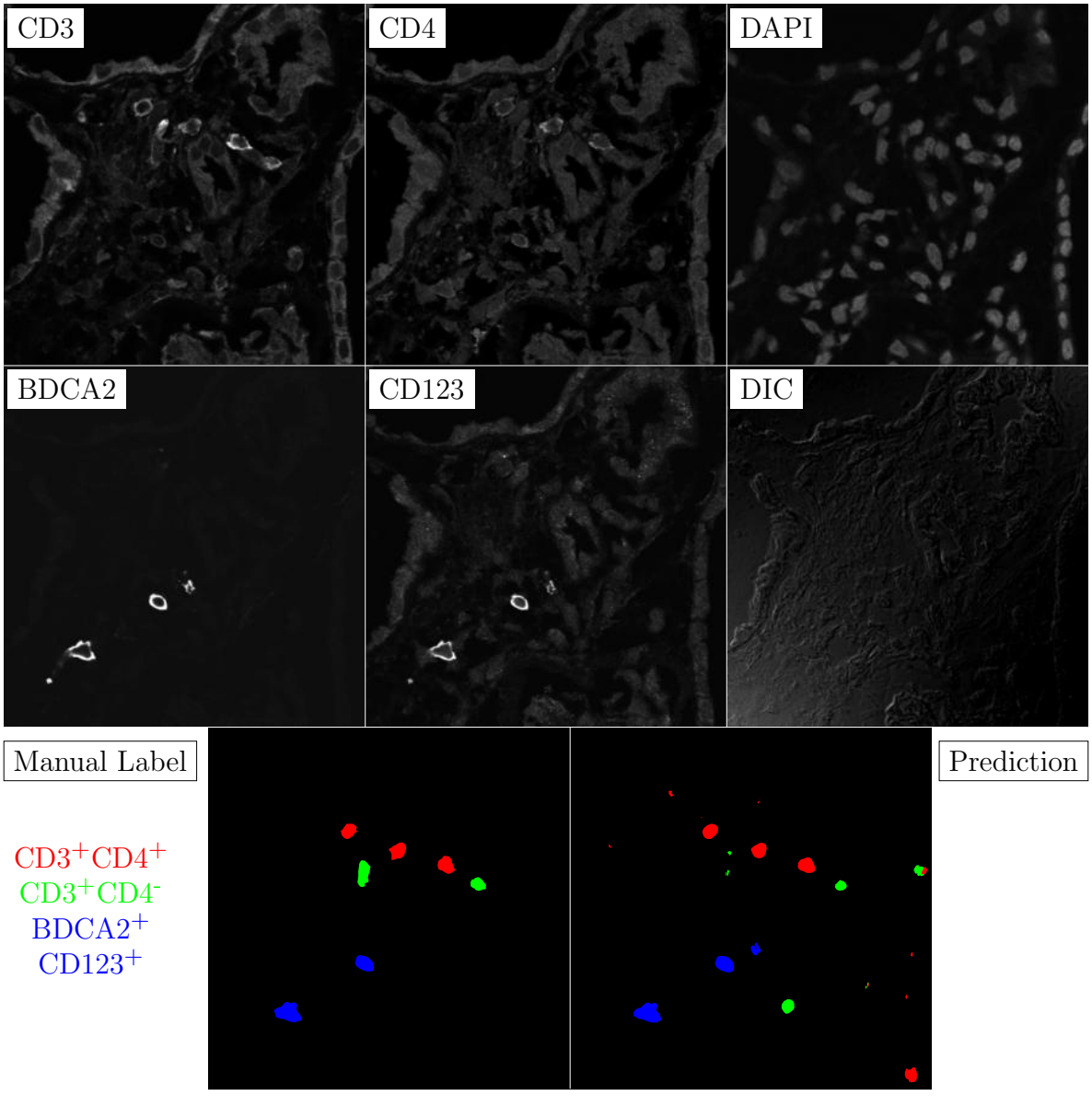

Figure G.143: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

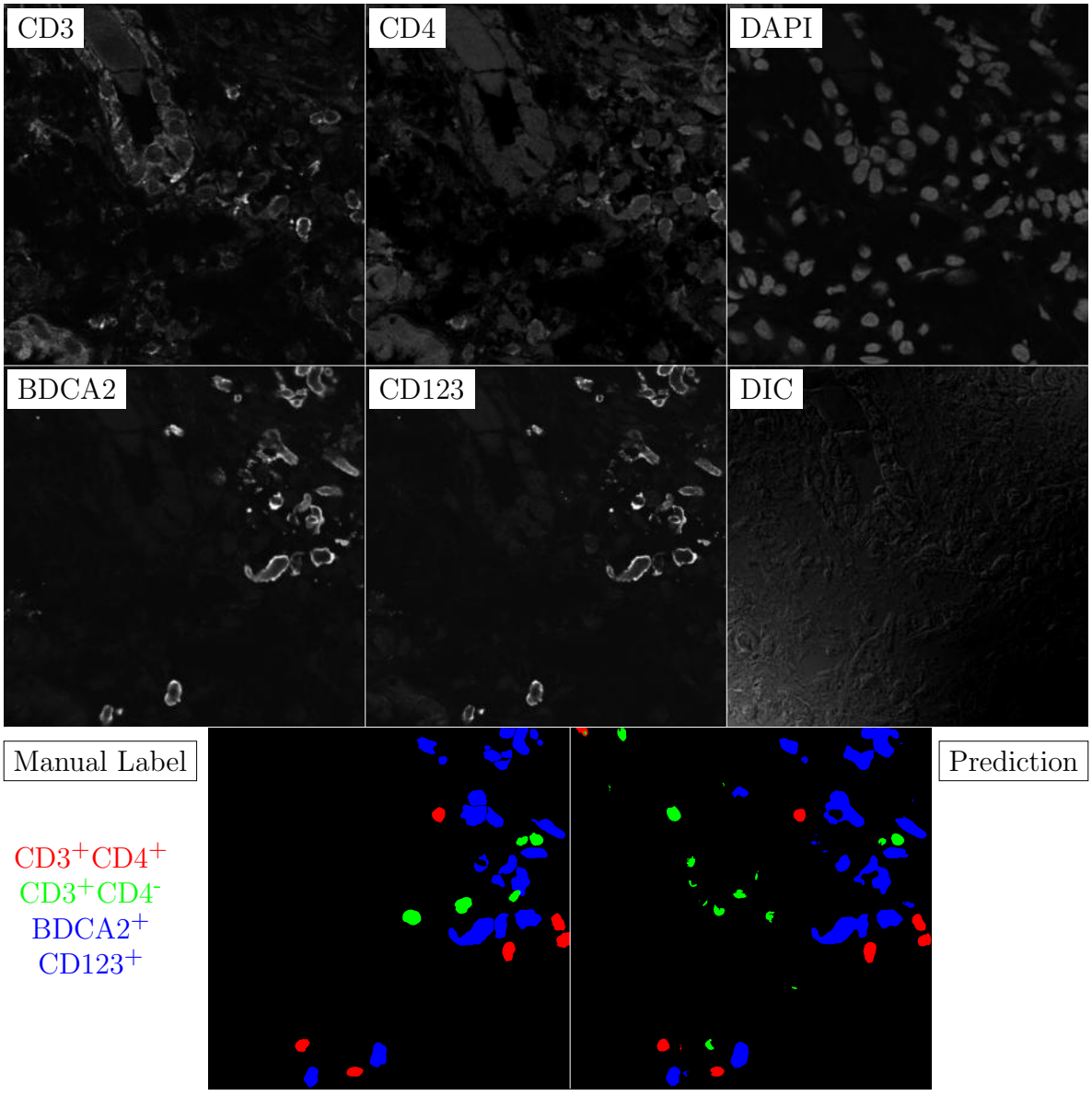

Figure G.144: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

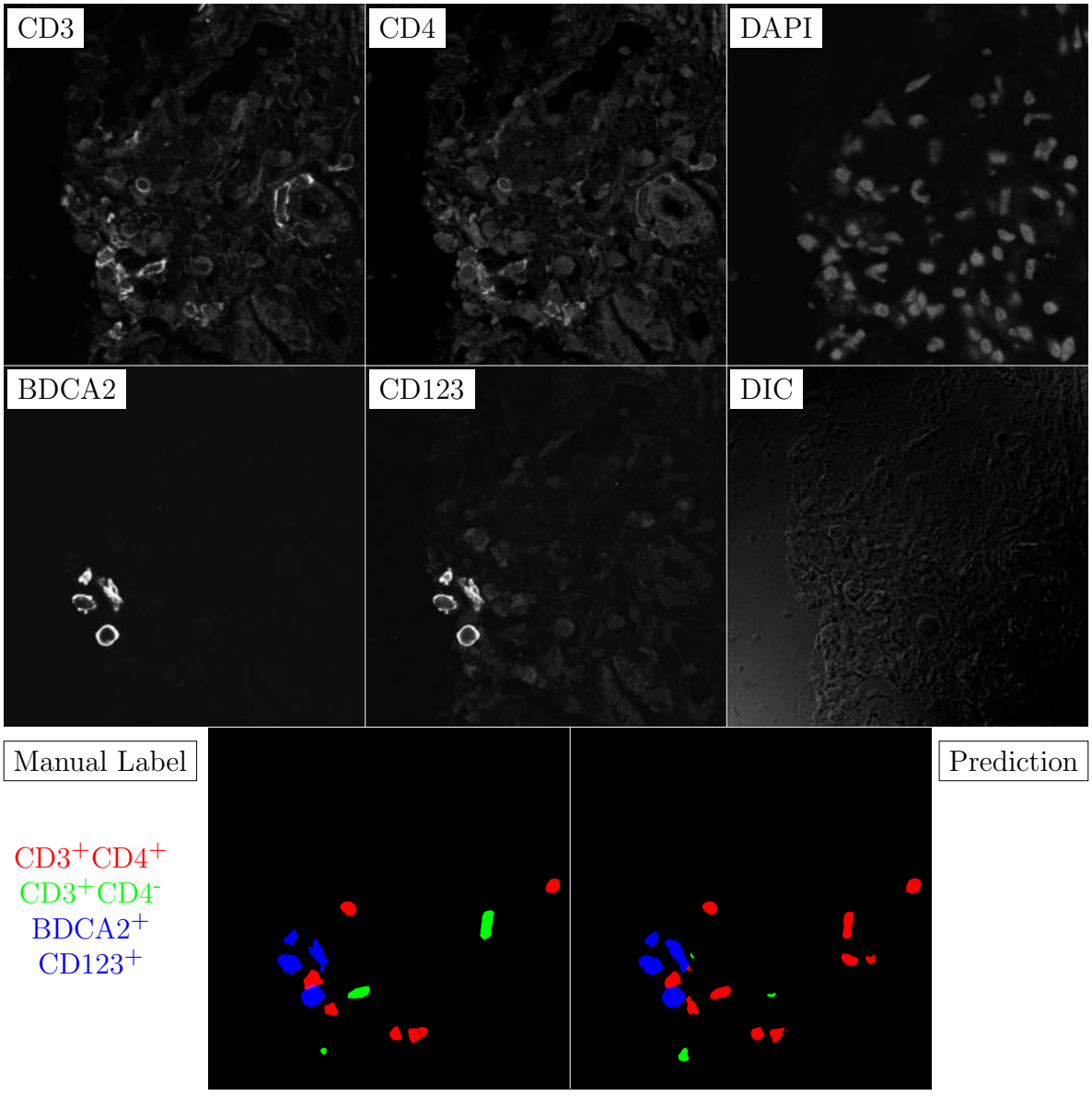

Figure G.145: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

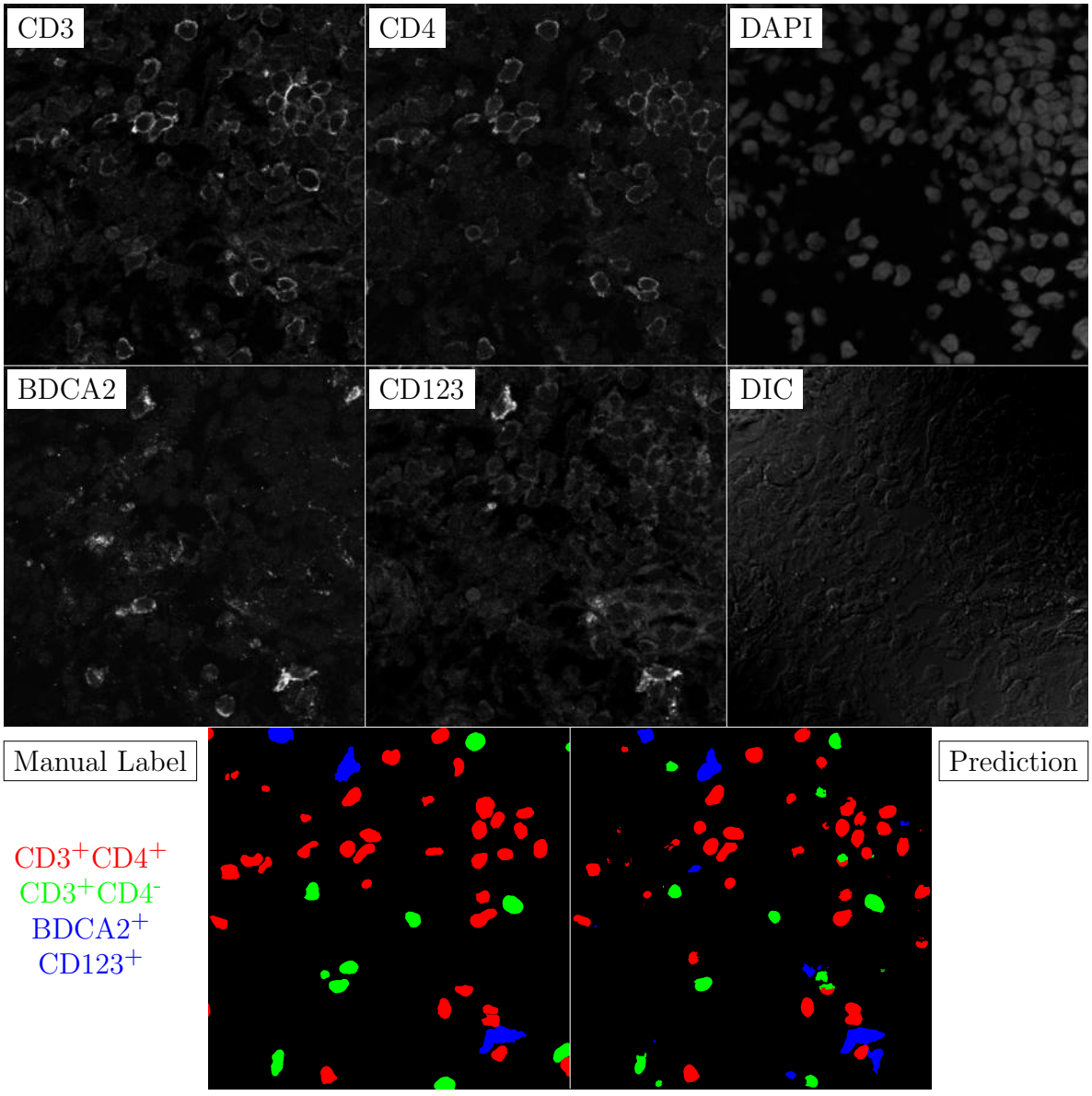

Figure G.146: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

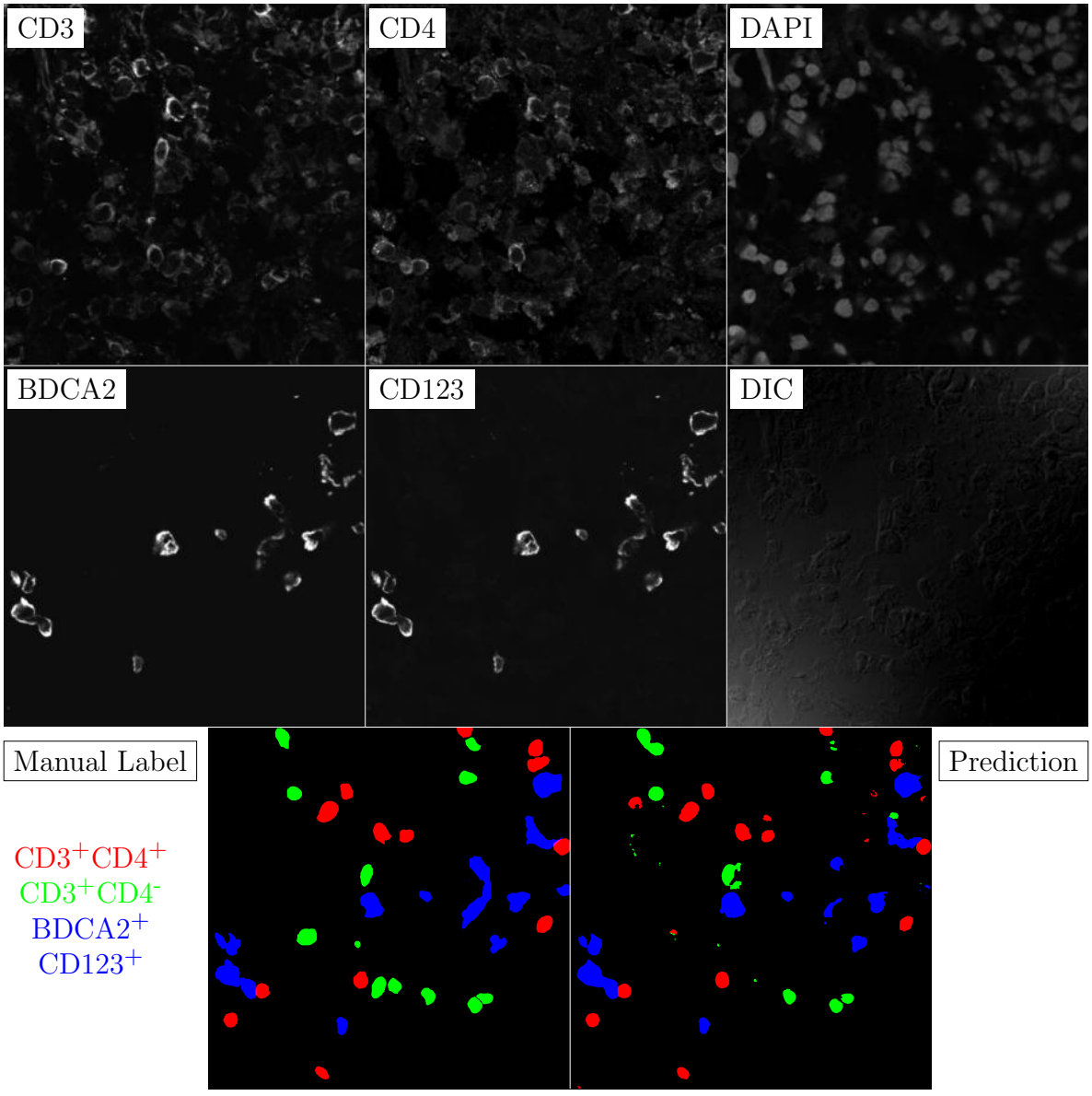

Figure G.147: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

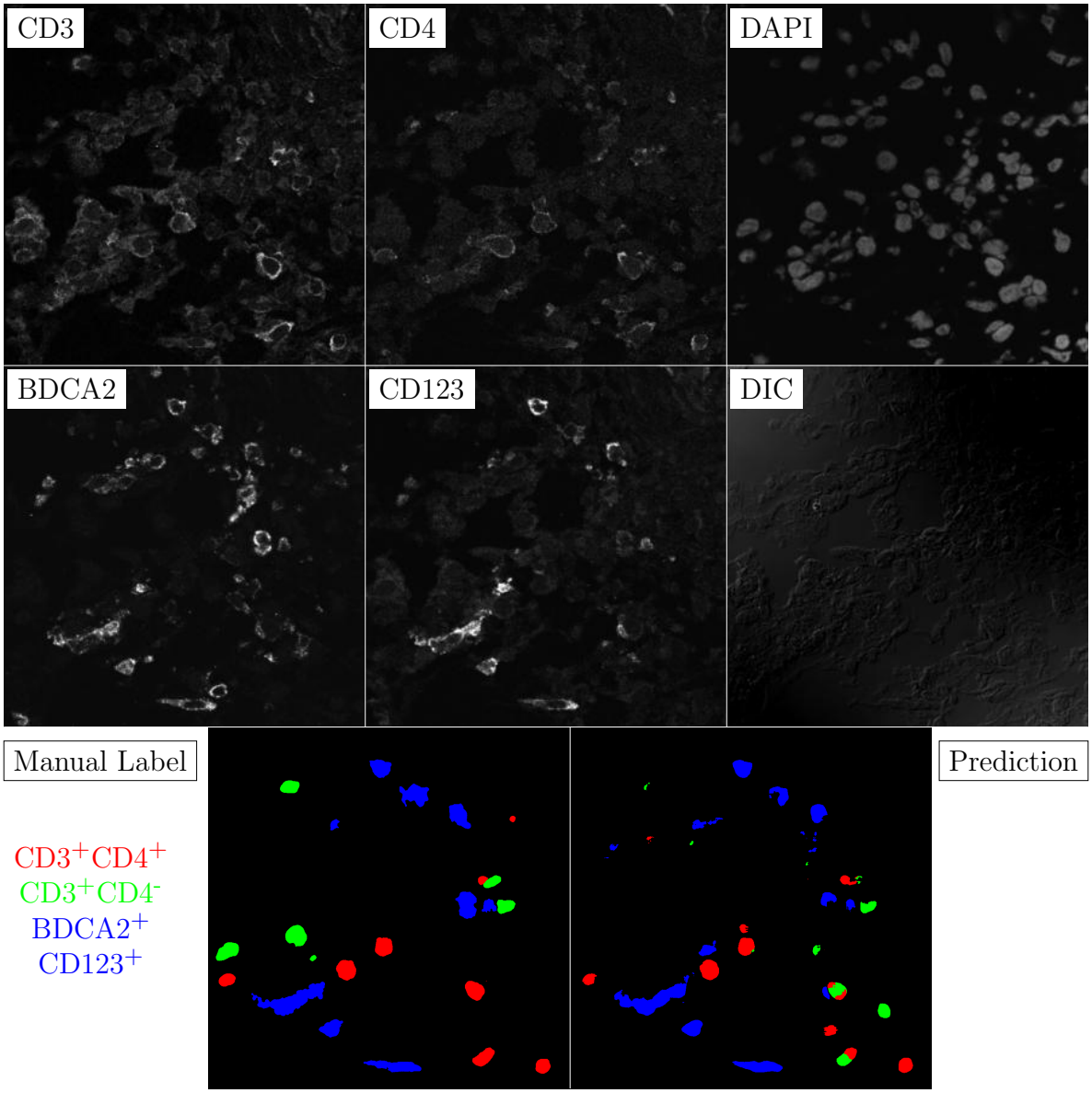

Figure G.148: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

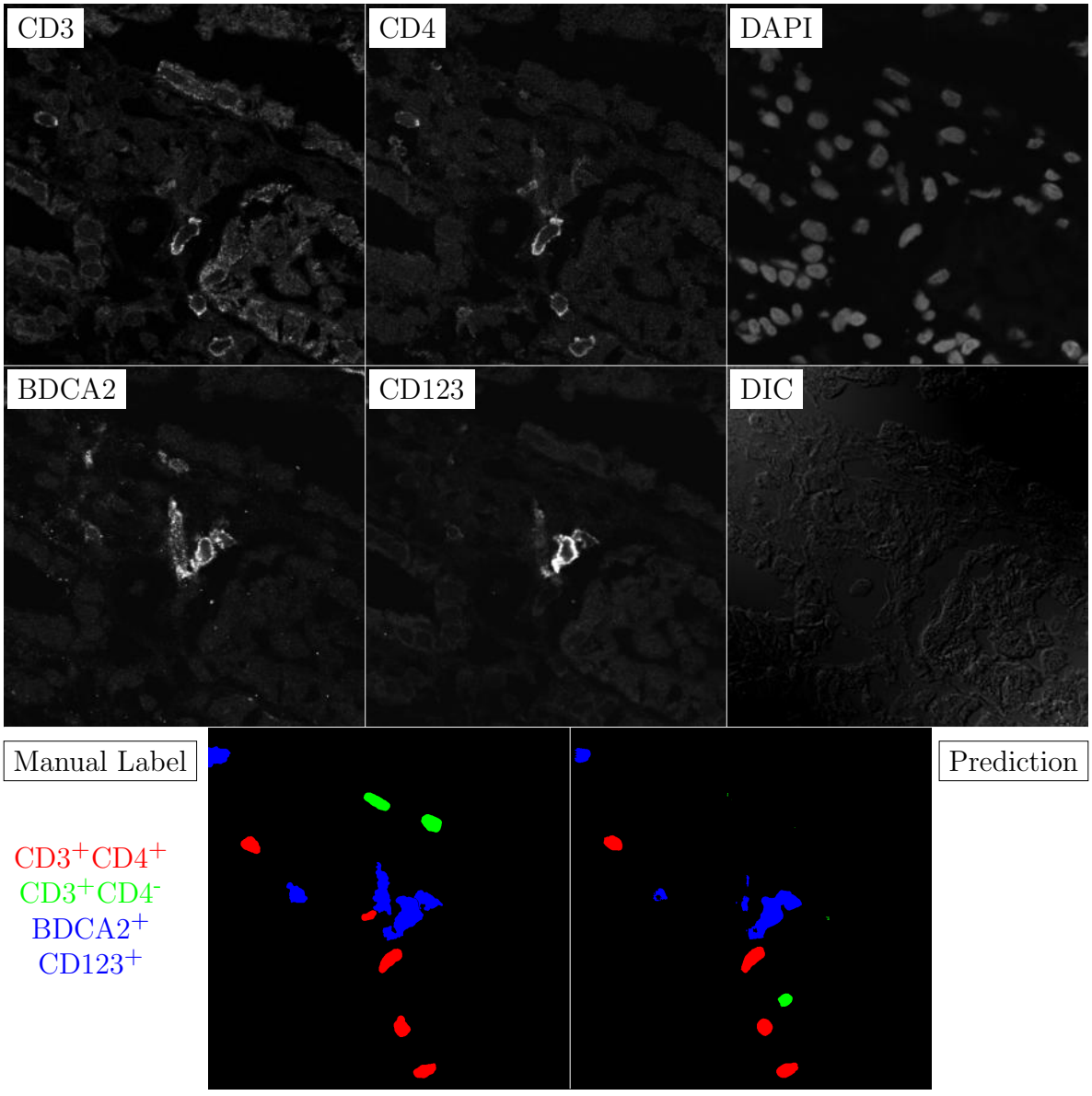

Figure G.149: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

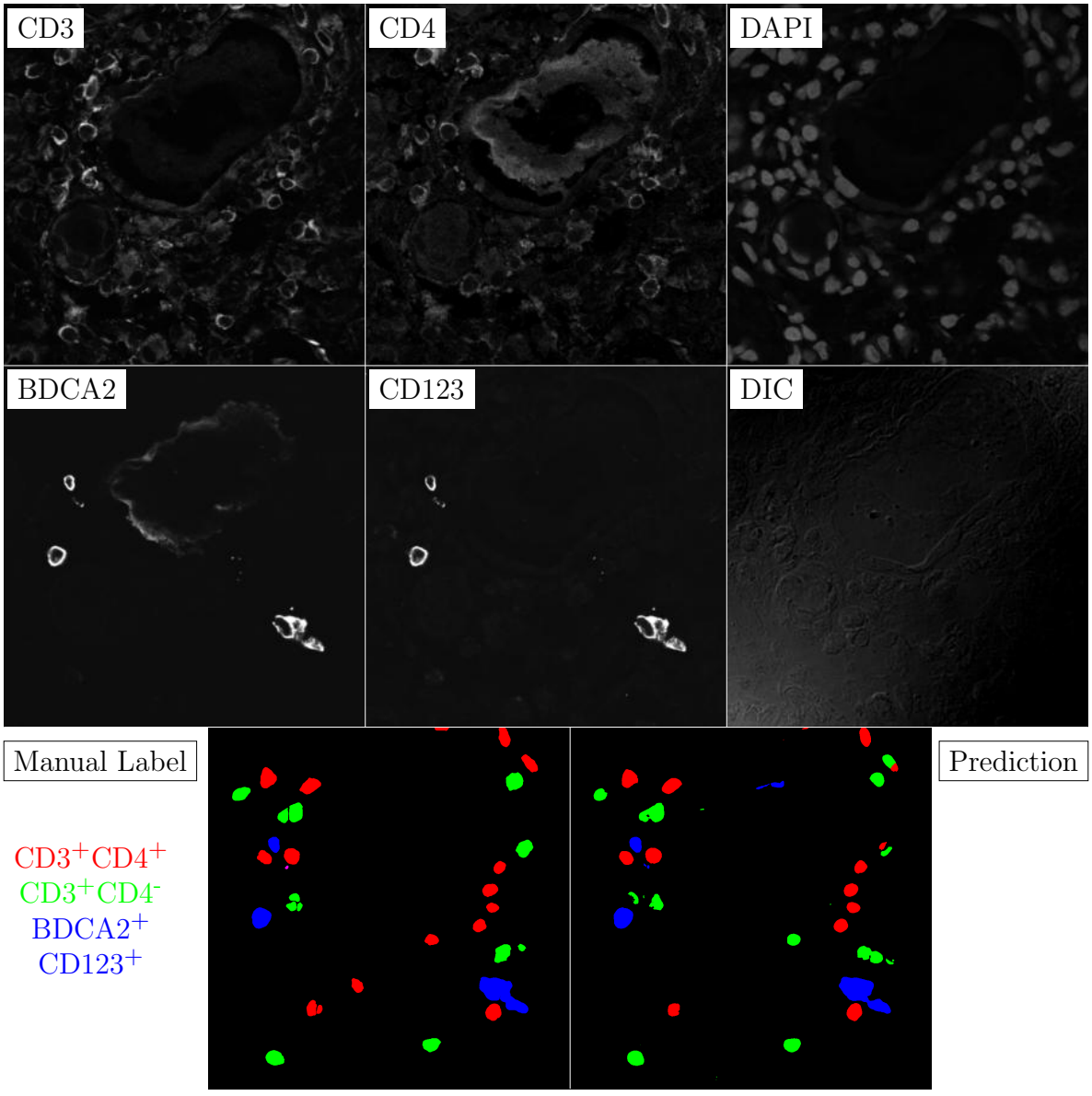

Figure G.150: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

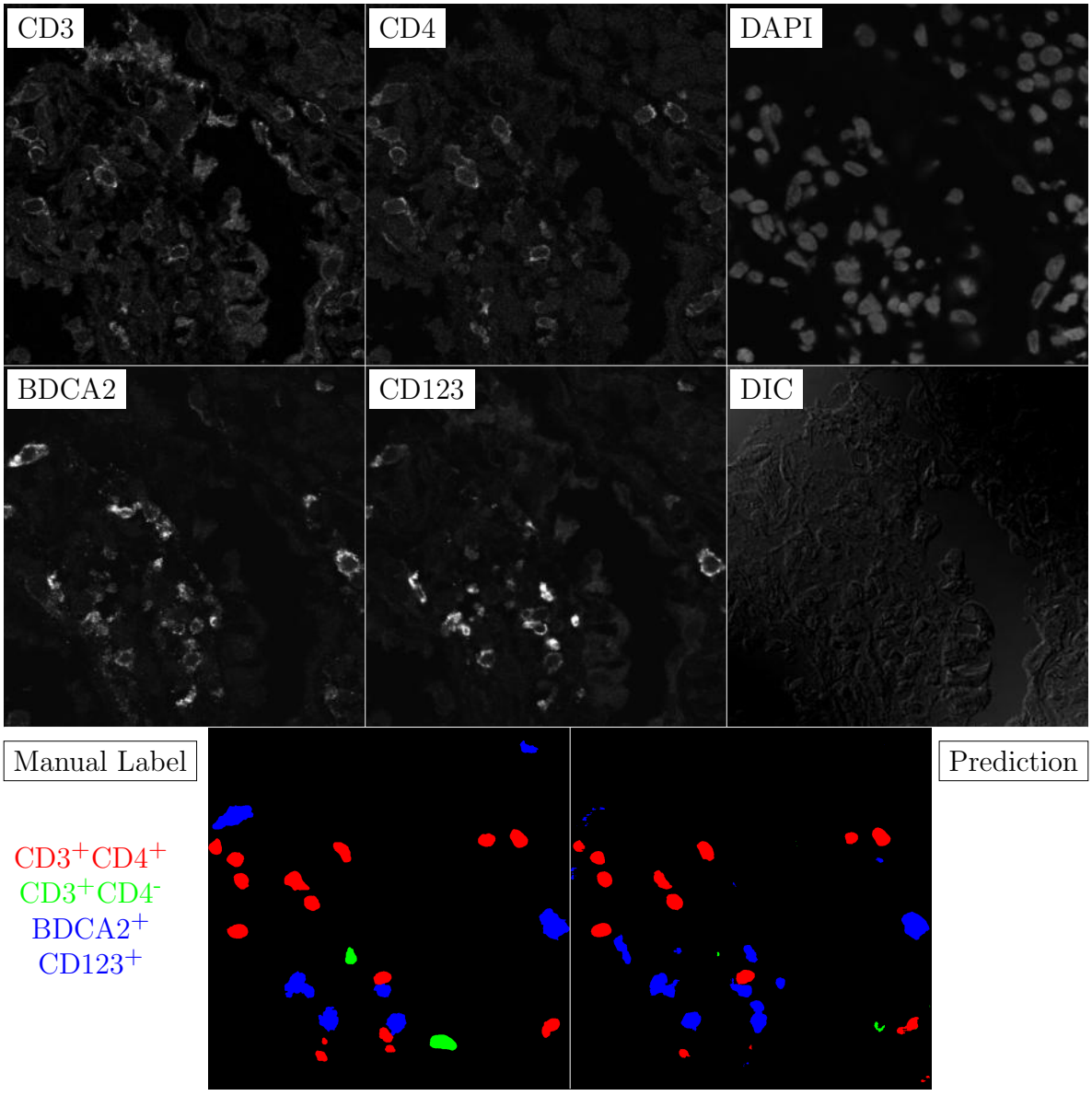

Figure G.151: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

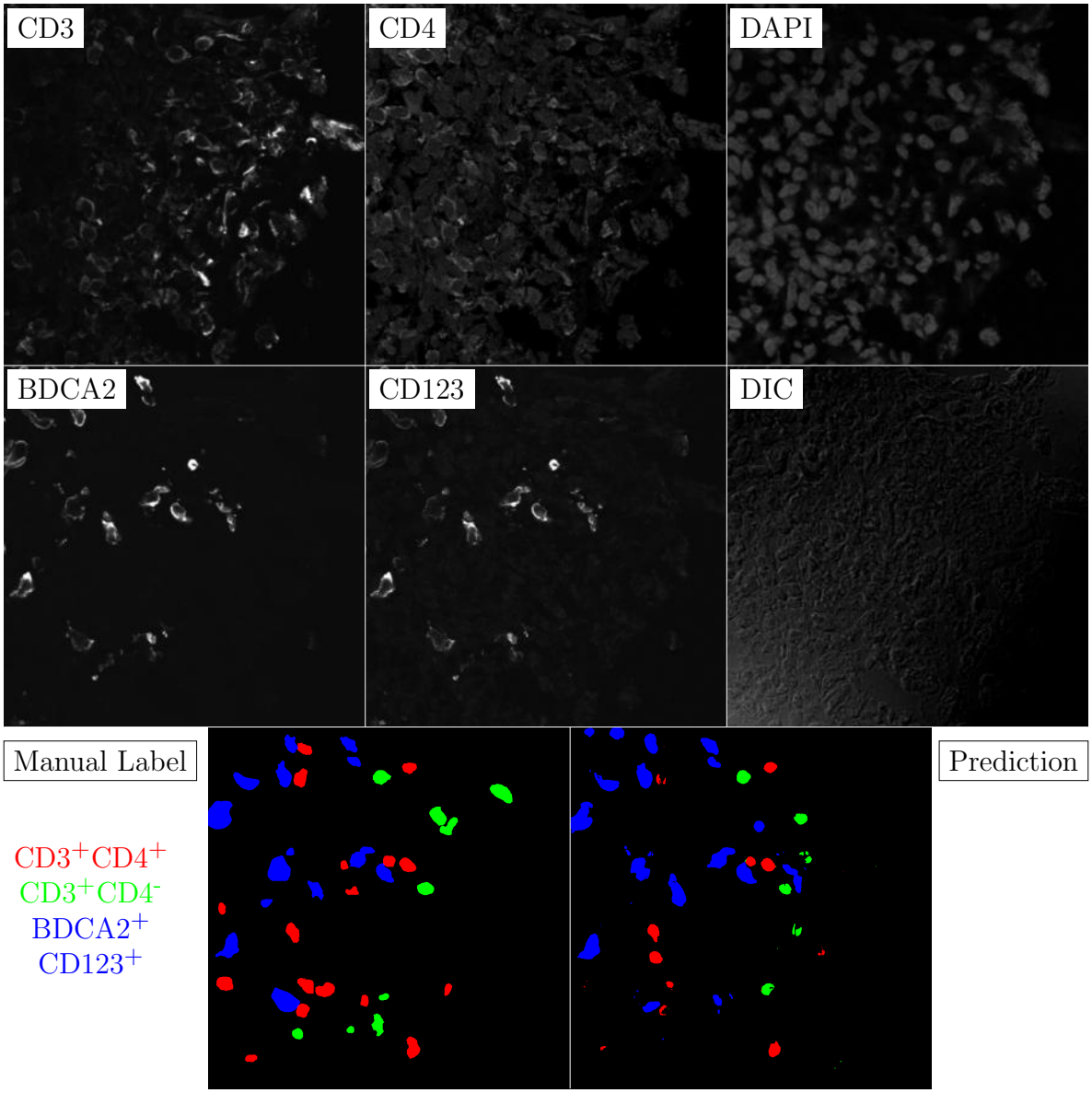

Figure G.152: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

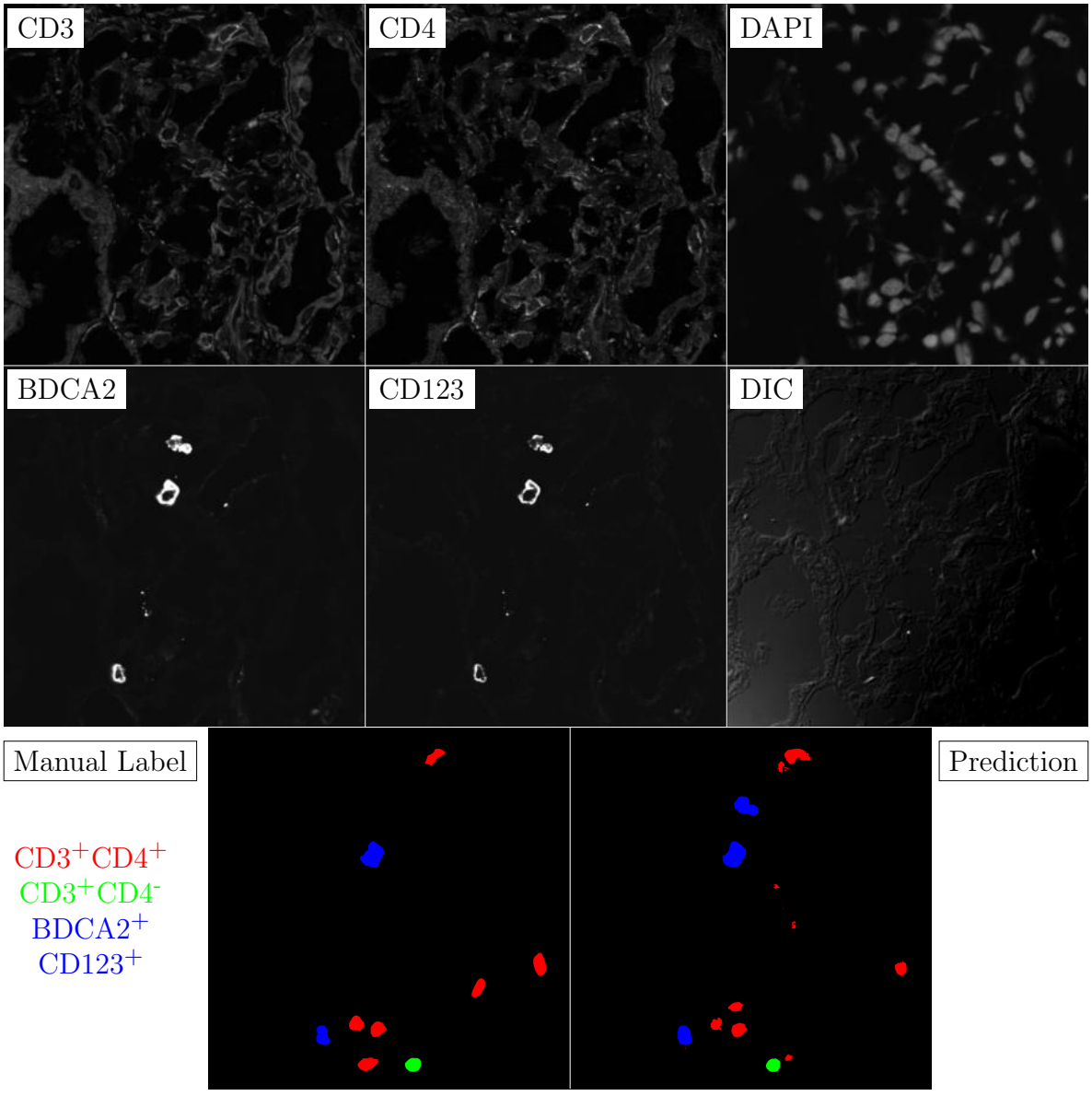

Figure G.153: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

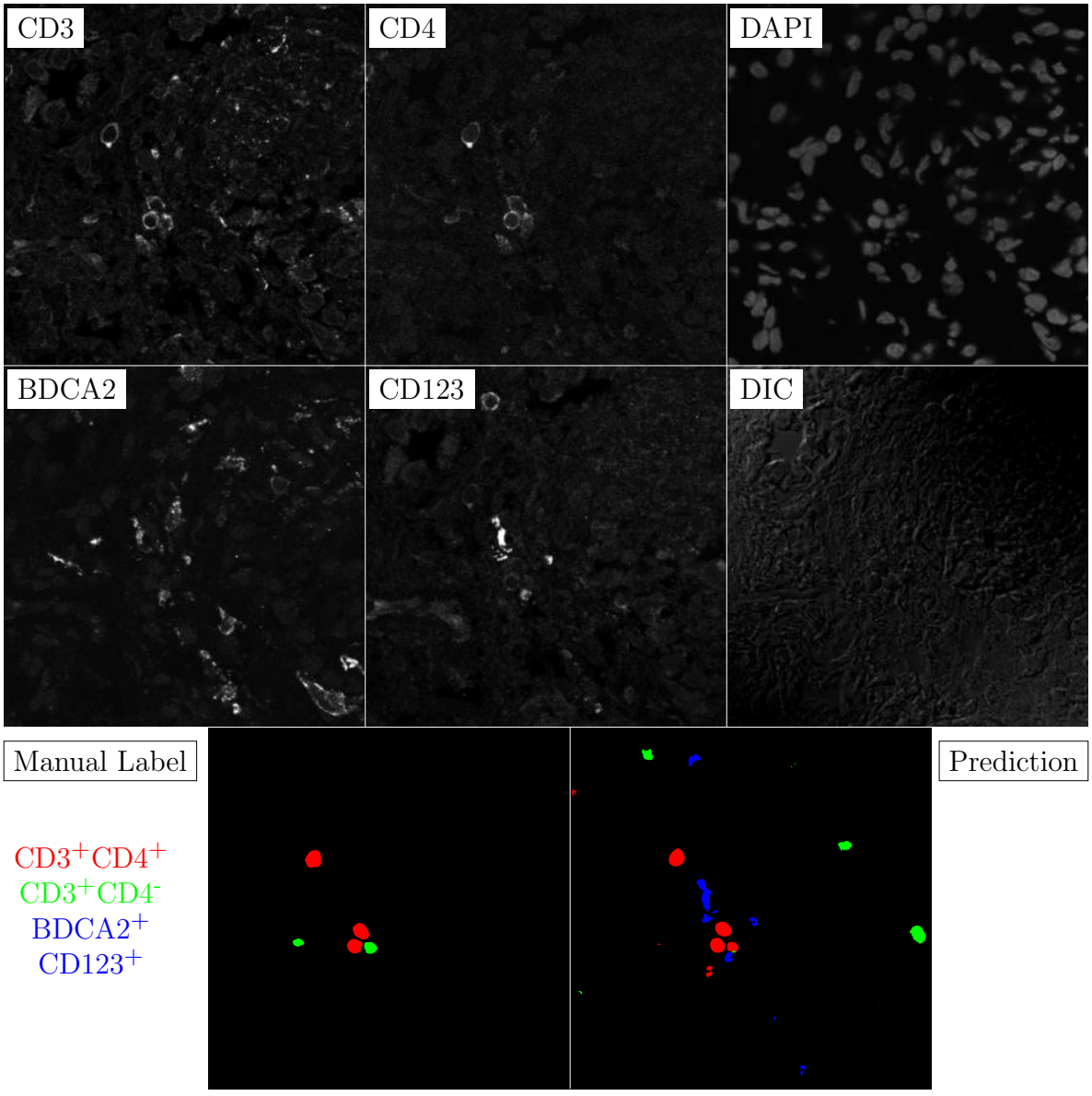

Figure G.154: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

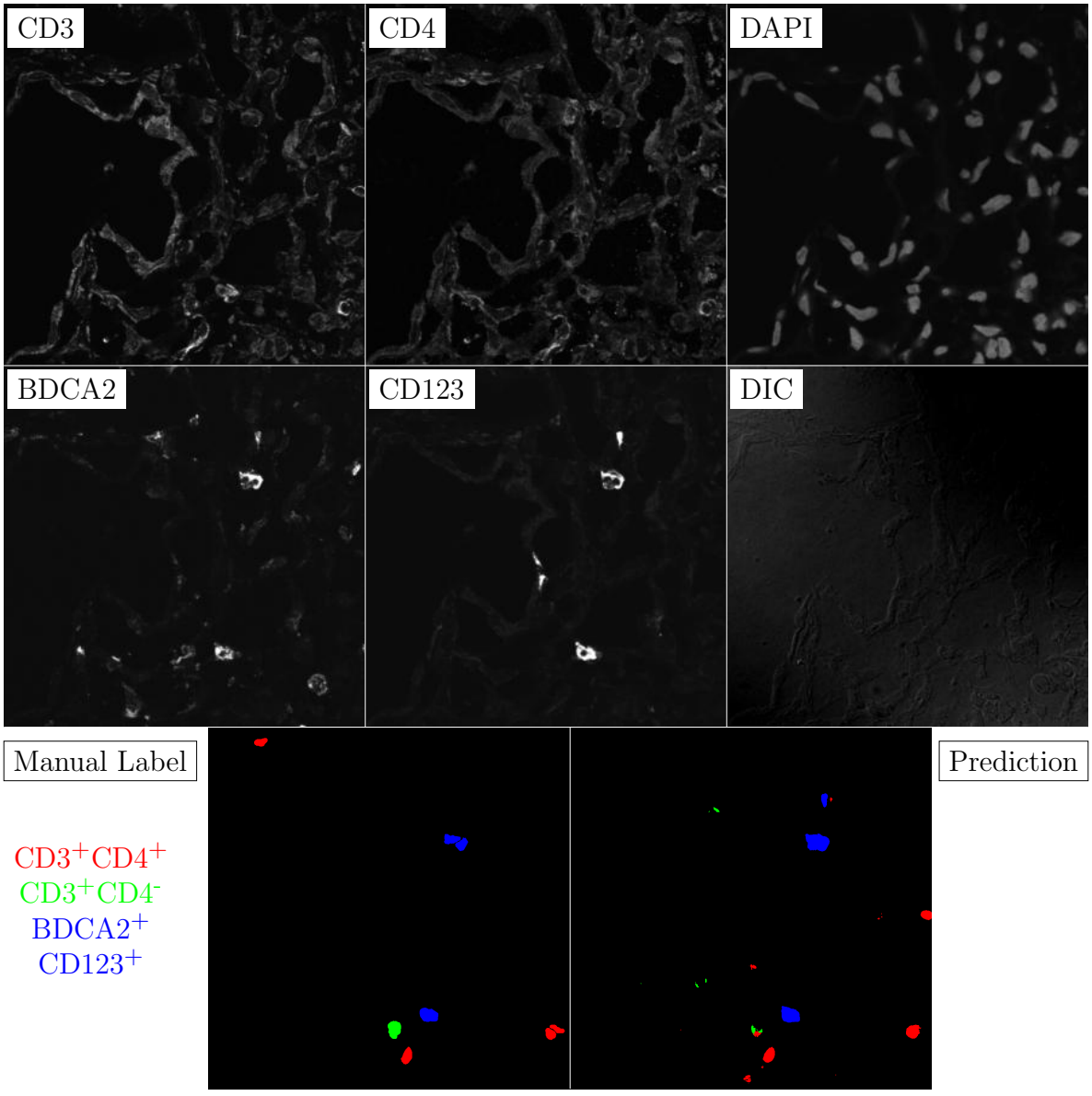

Figure G.155: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

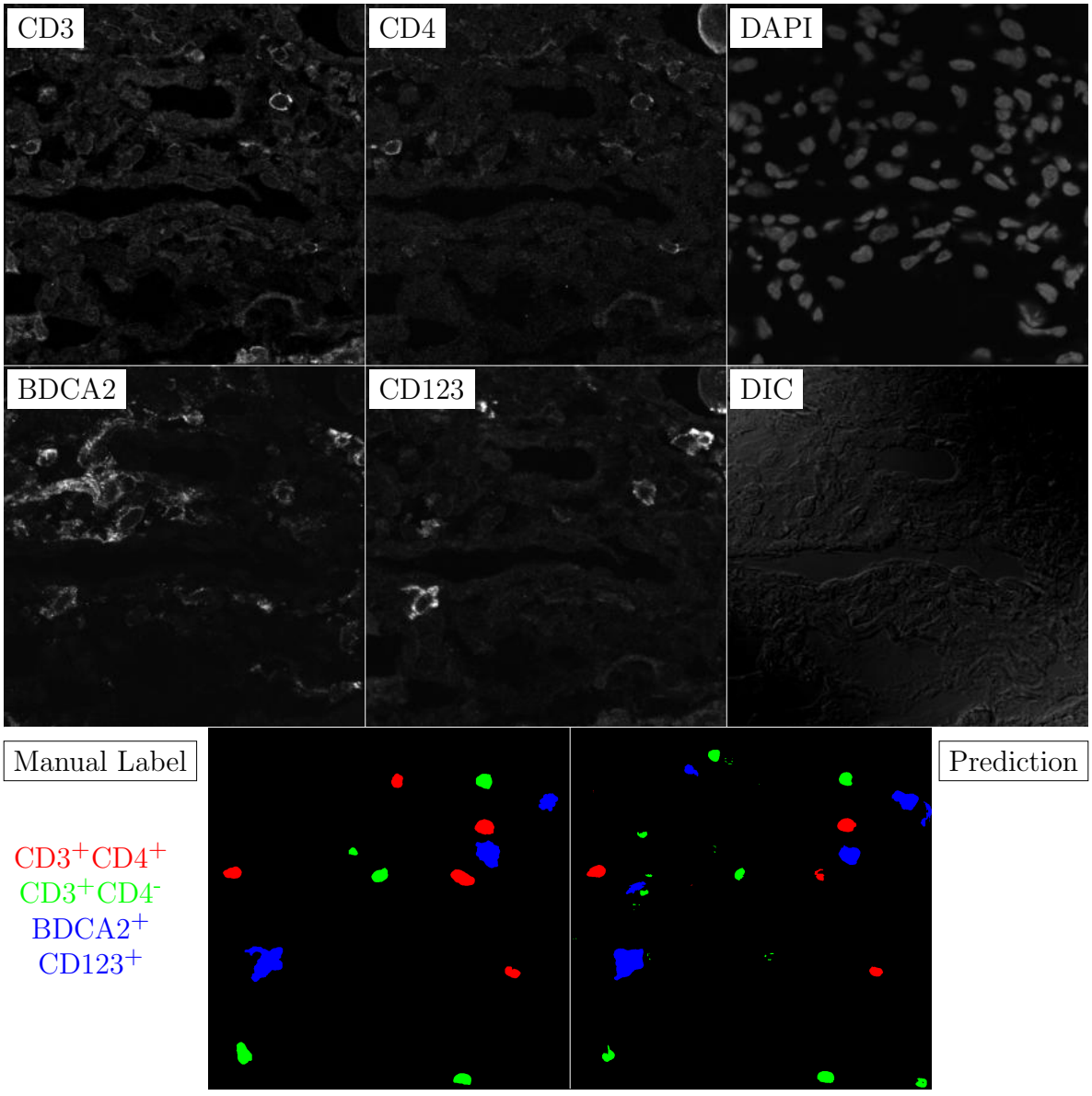

Figure G.156: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

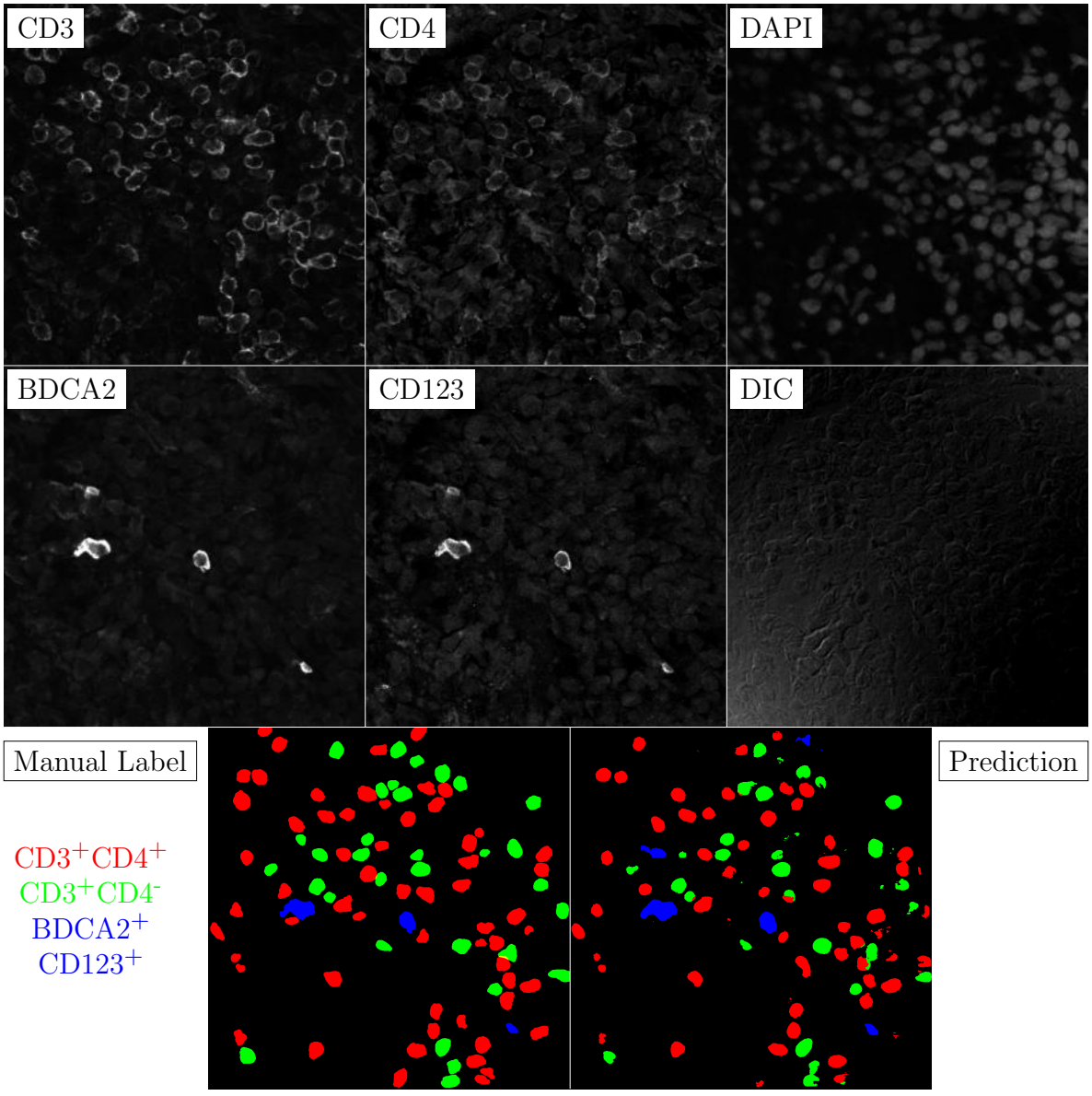

Figure G.157: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

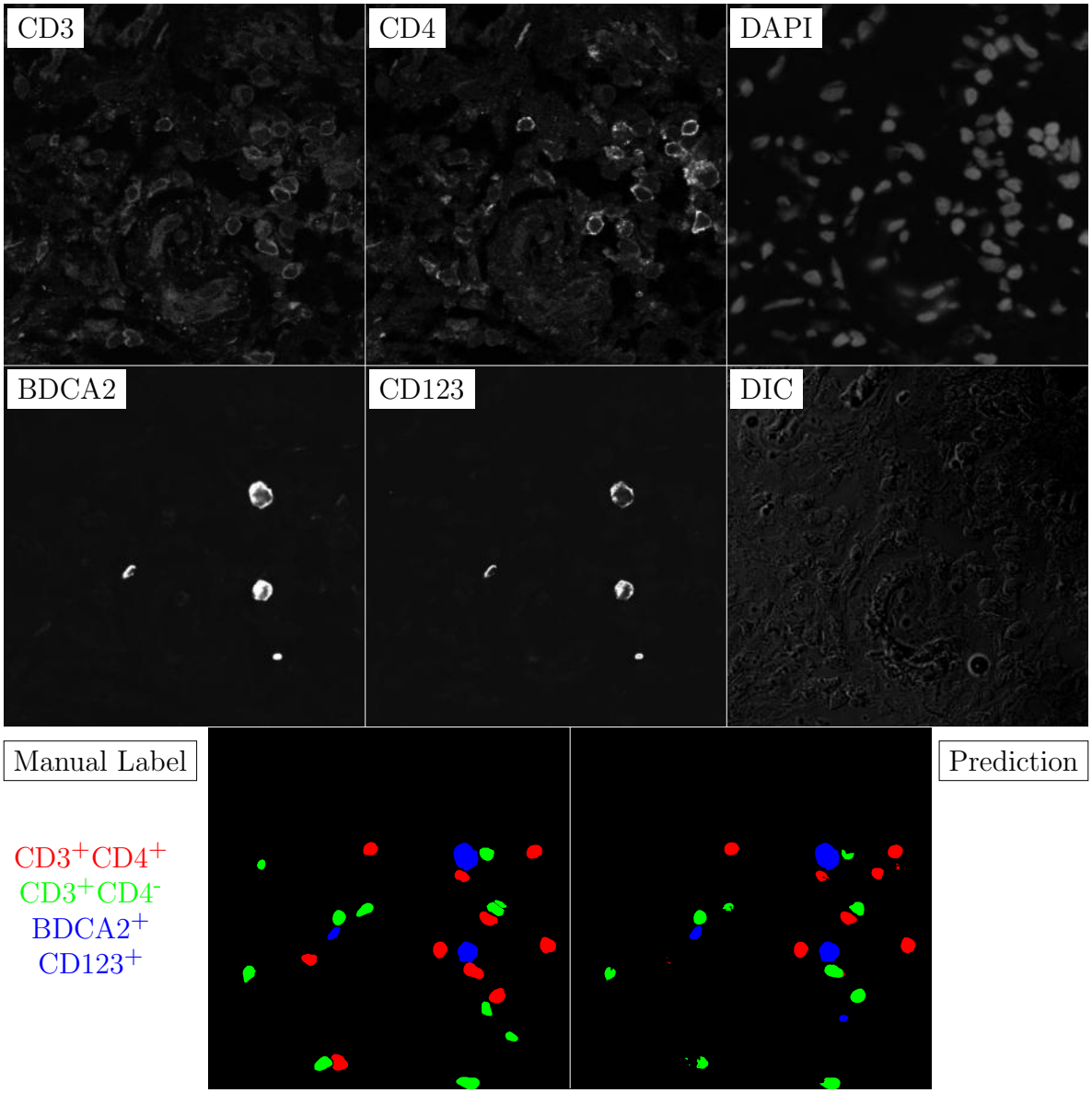

Figure G.158: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

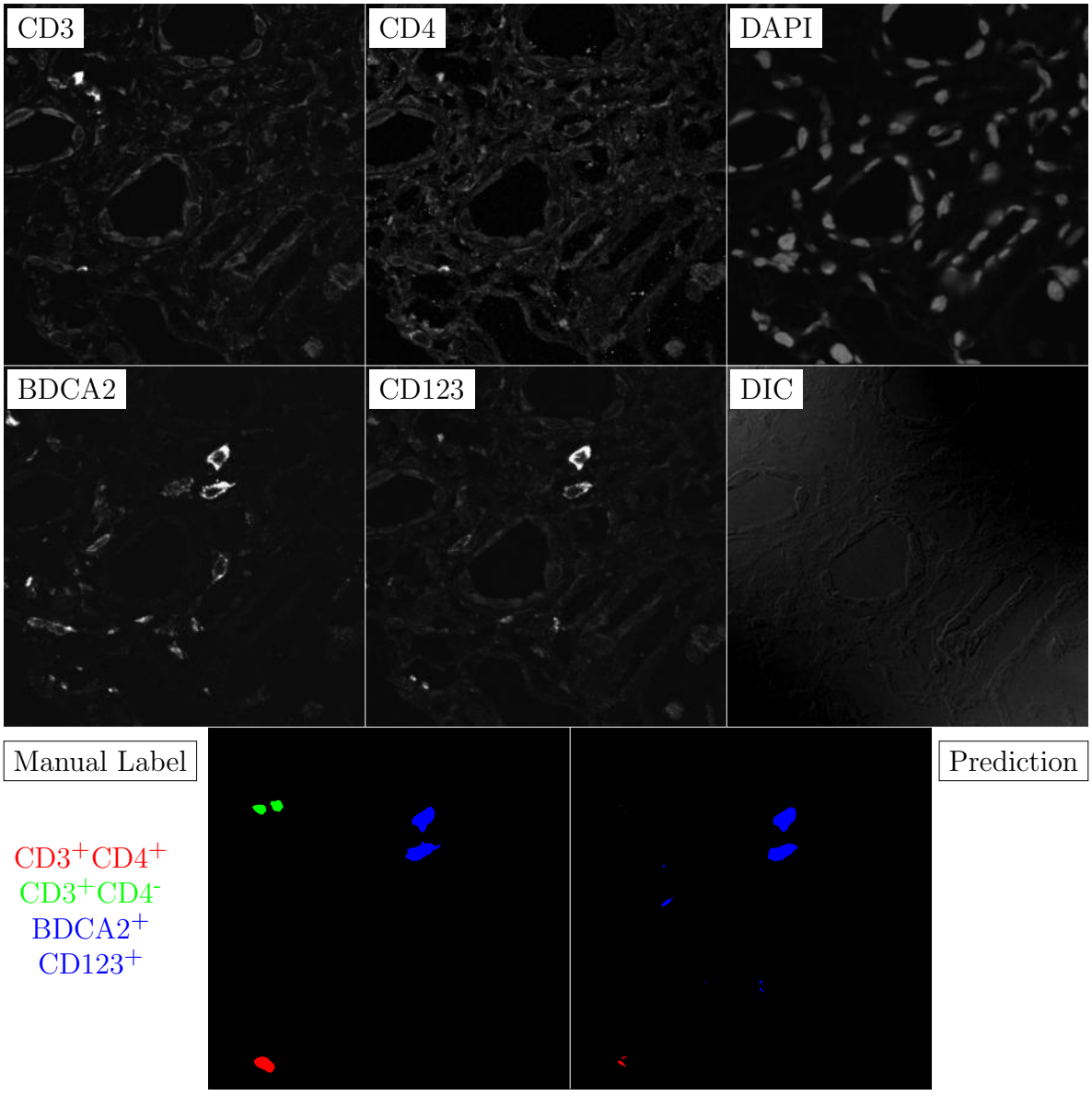

Figure G.159: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

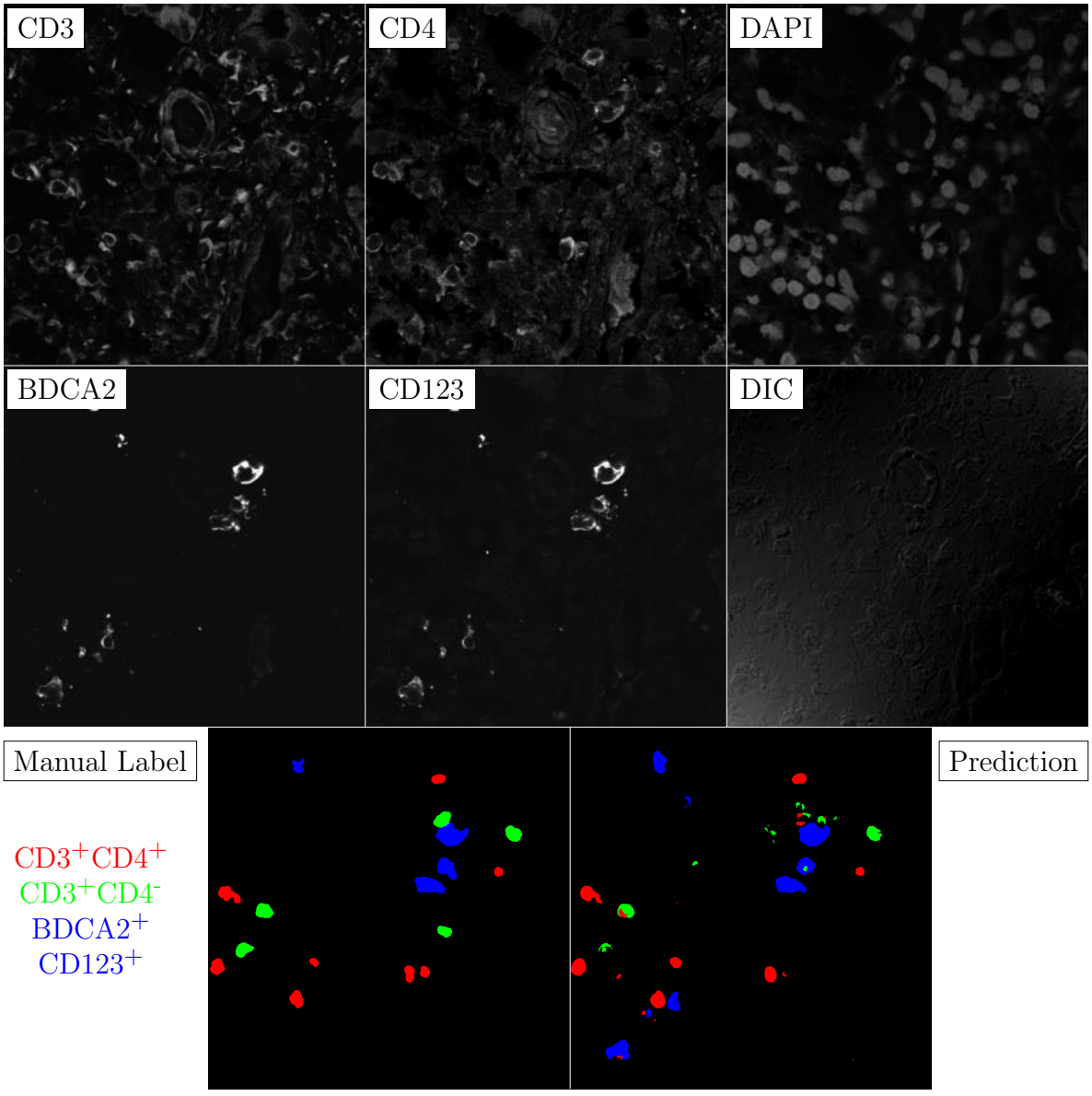

Figure G.160: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

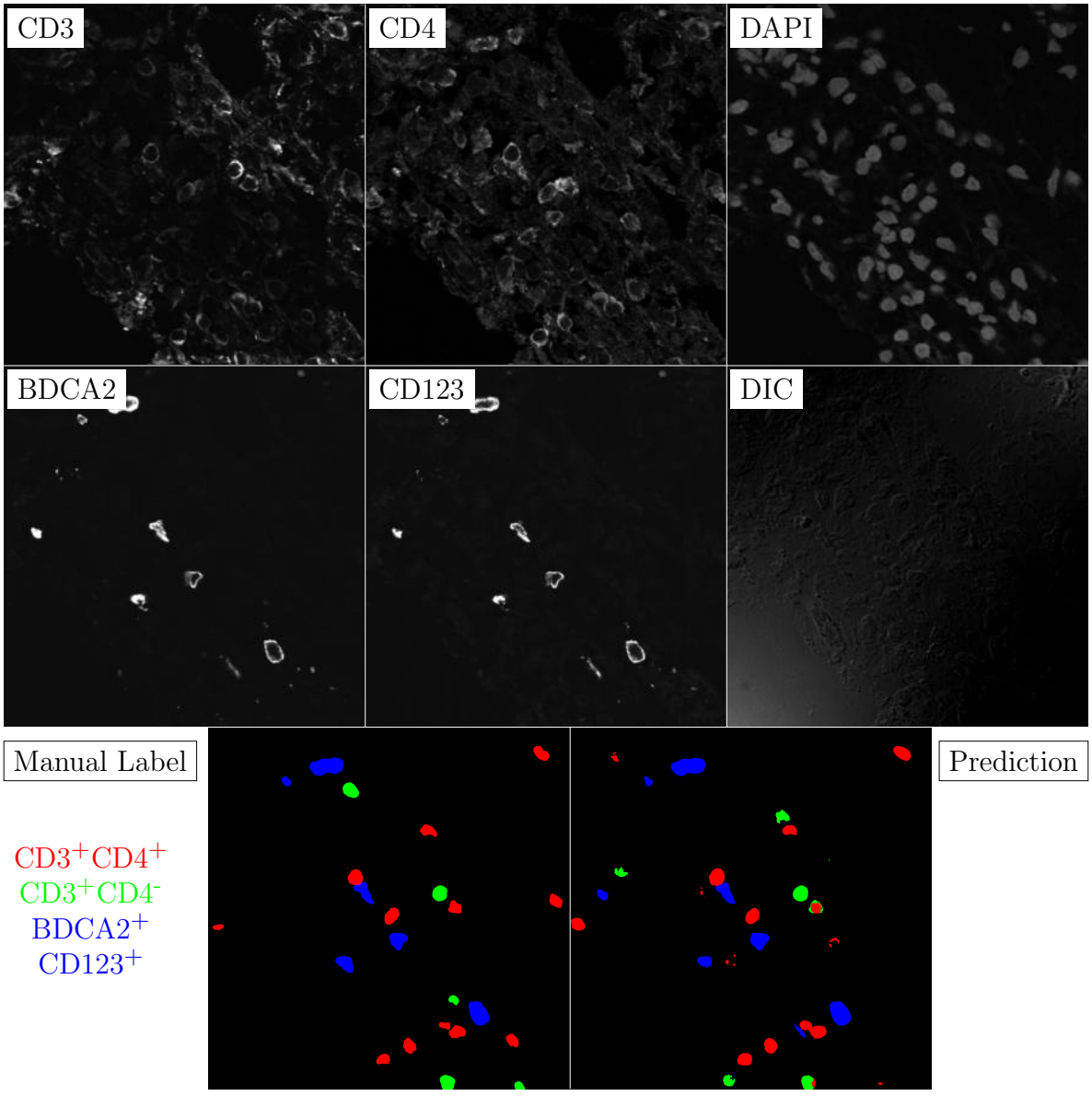

Figure G.161: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

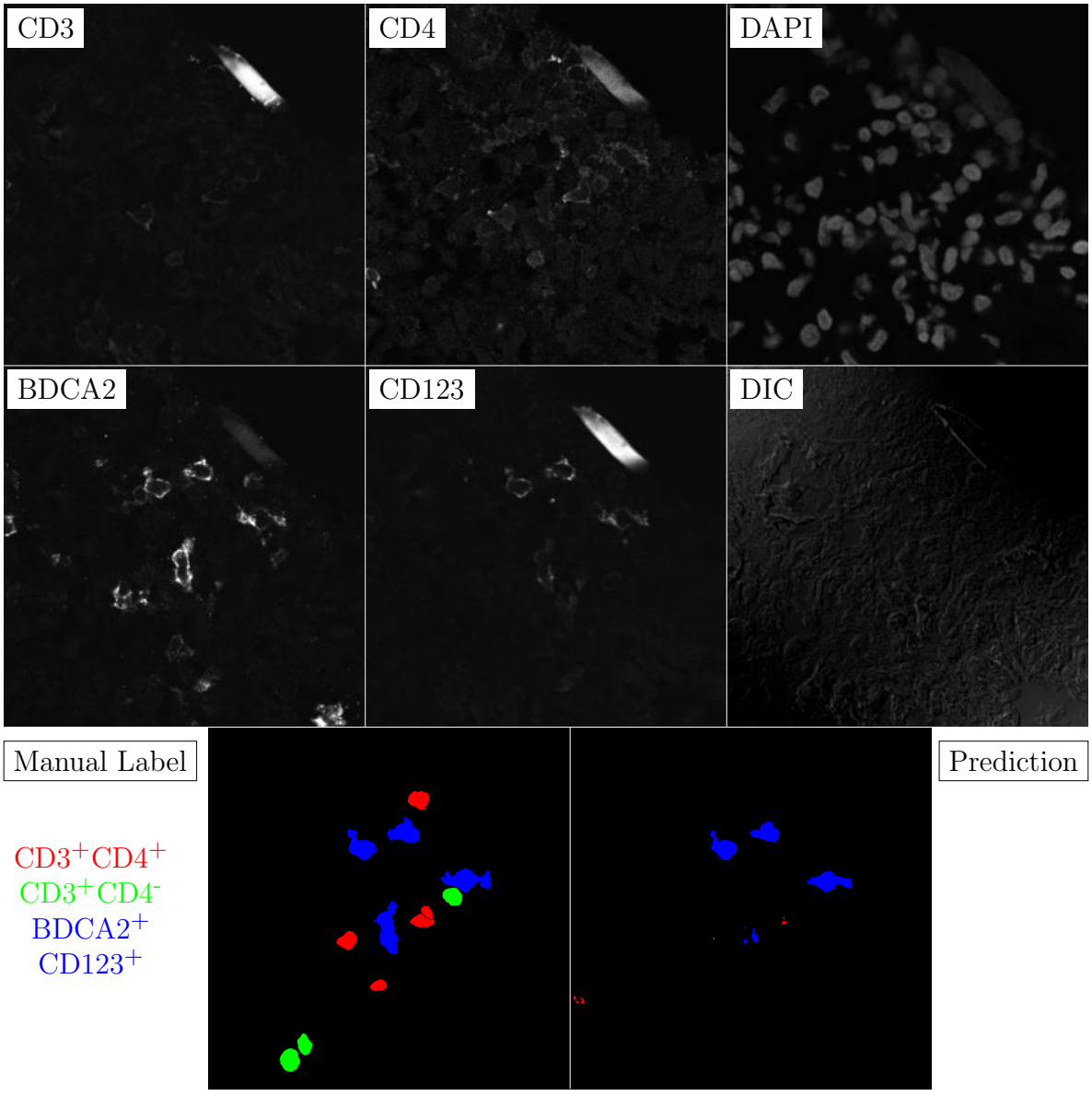

Figure G.162: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

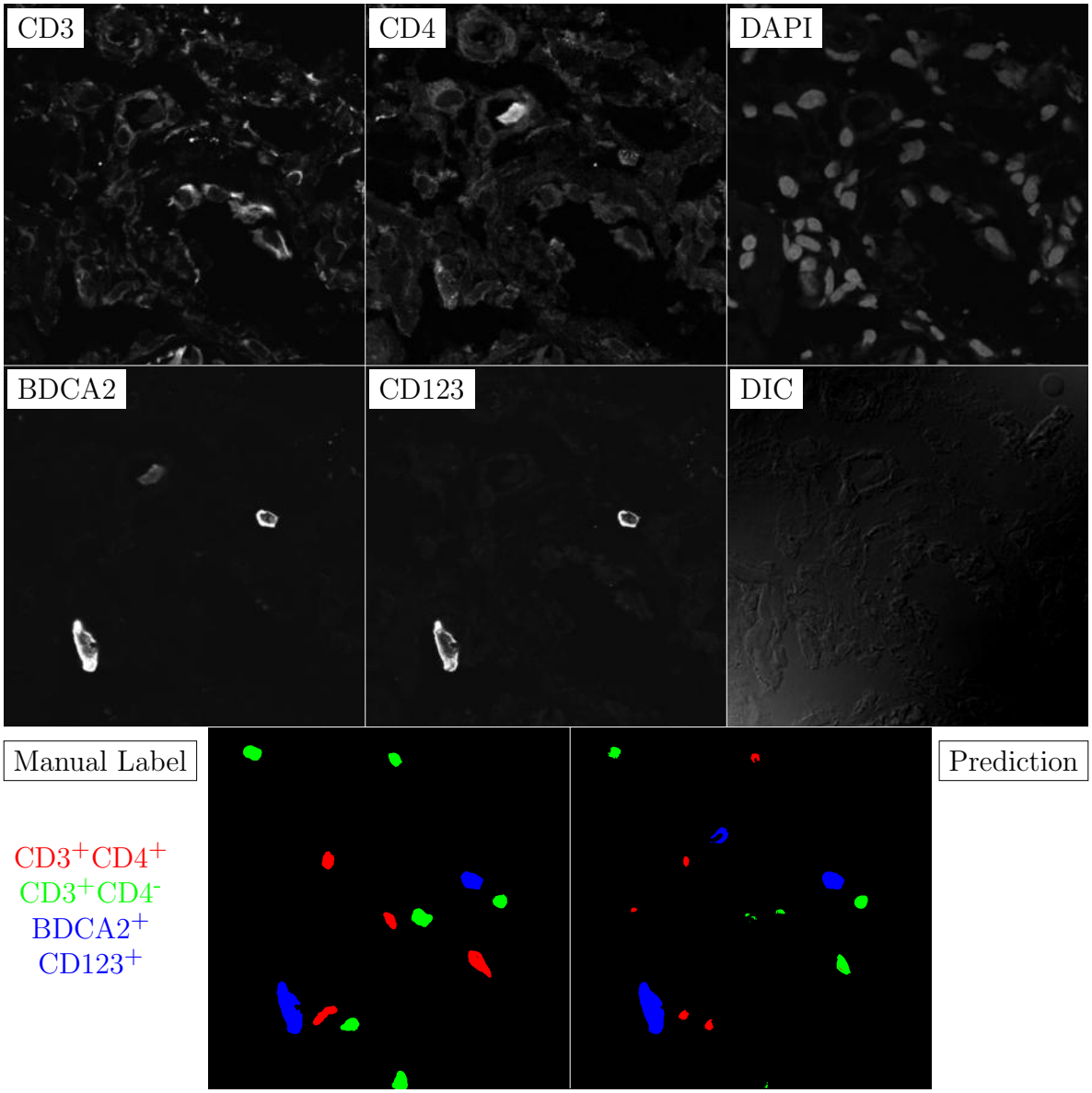

Figure G.163: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

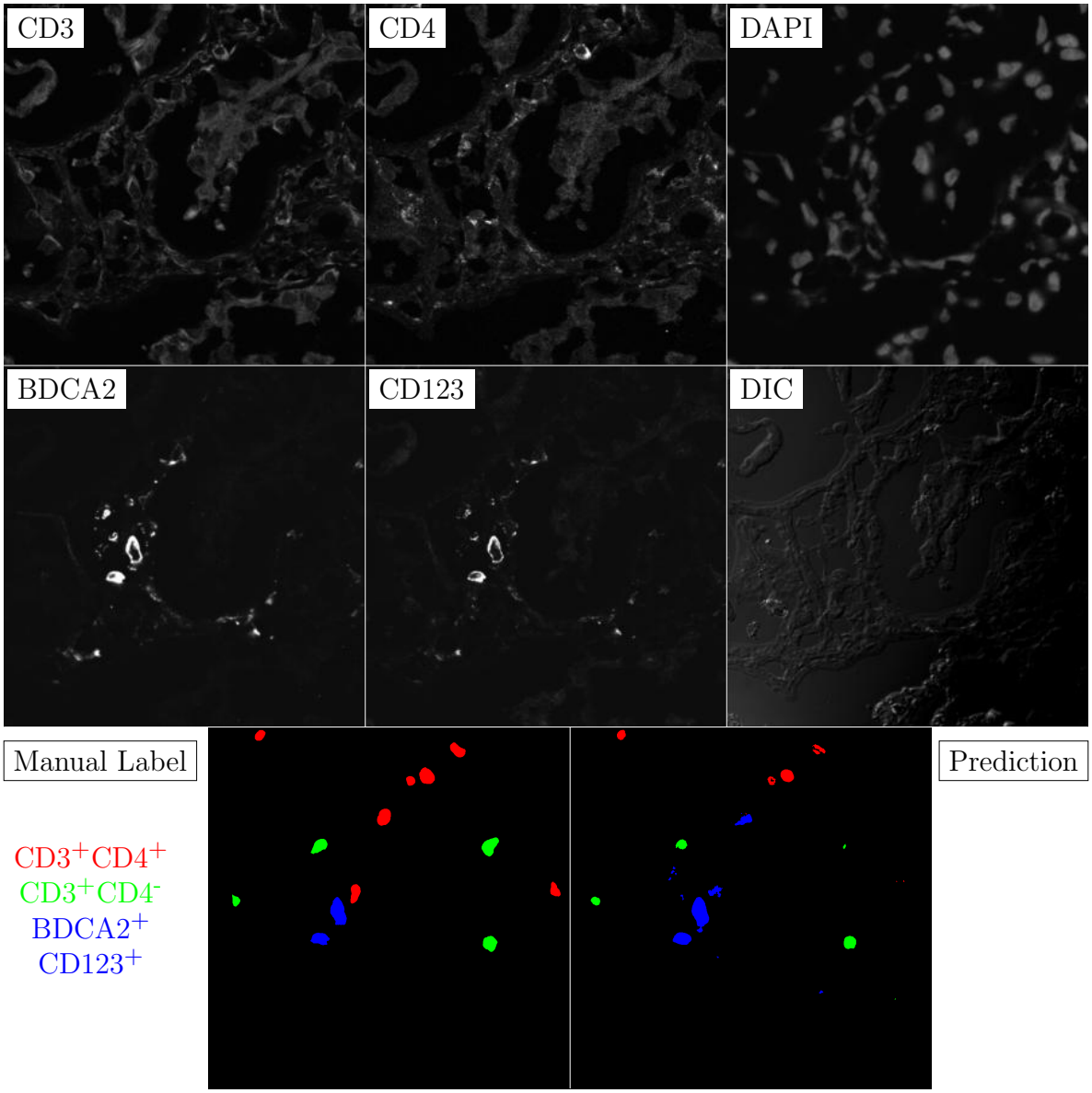

Figure G.164: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

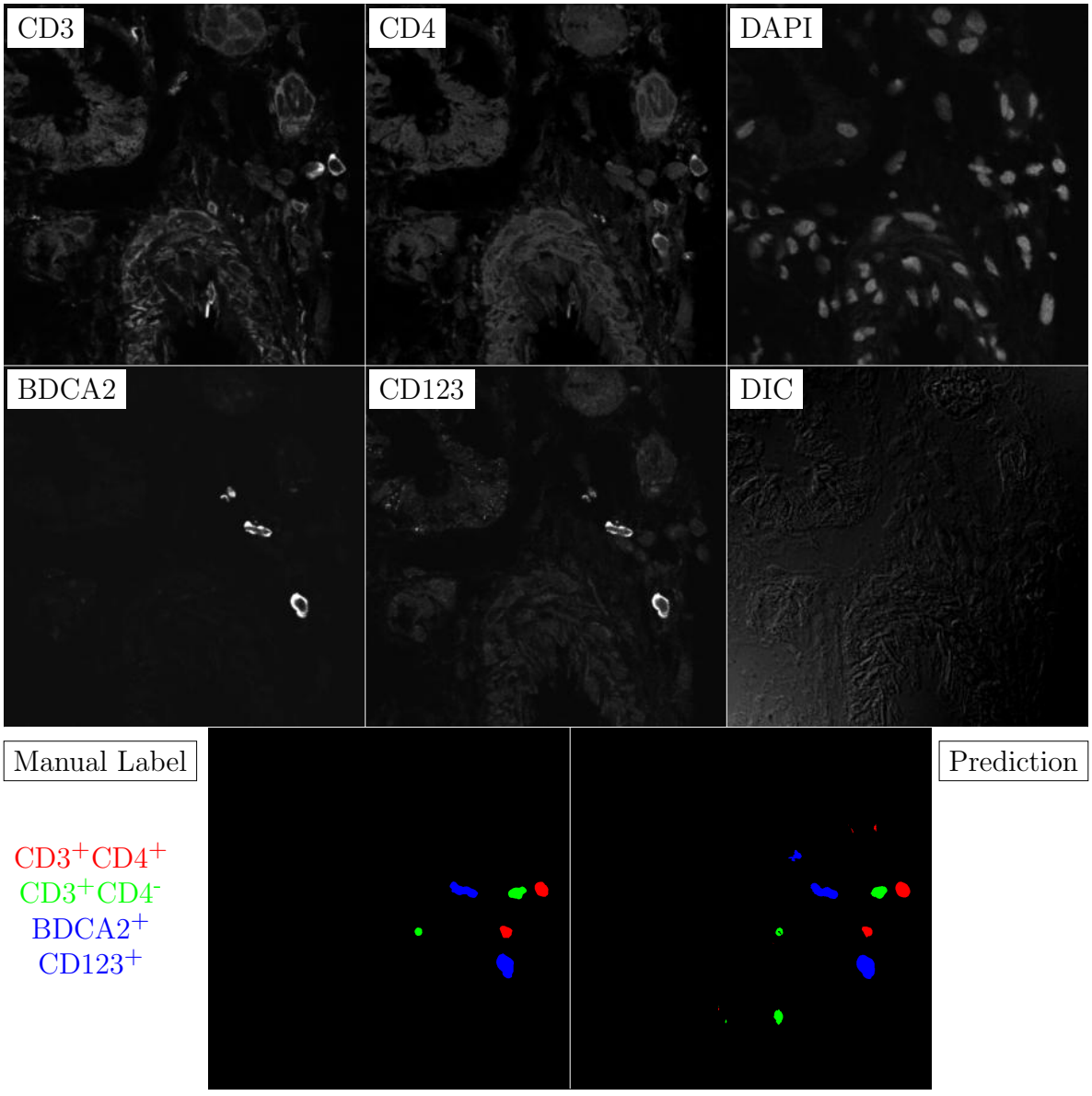

Figure G.165: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

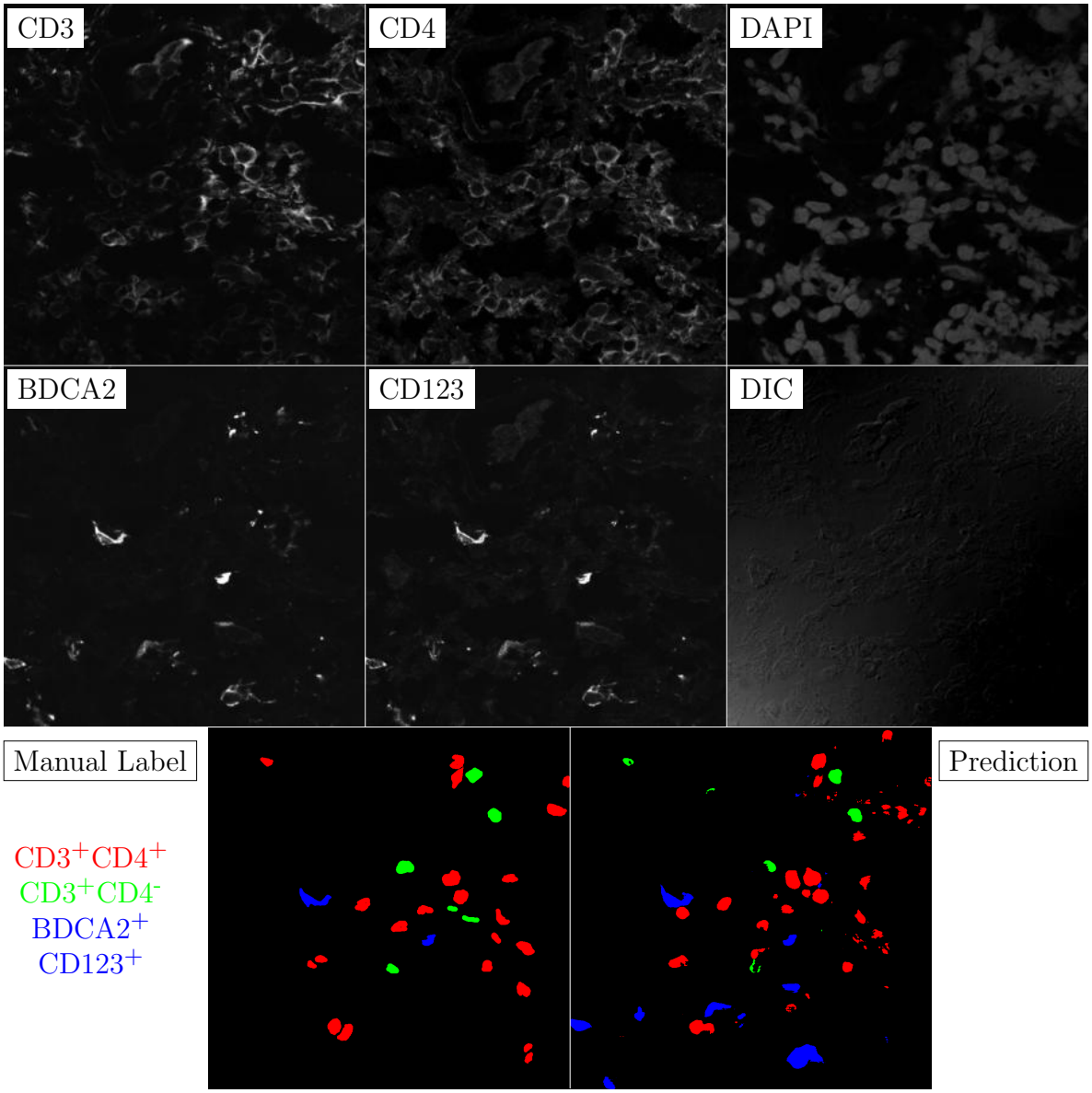

Figure G.166: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

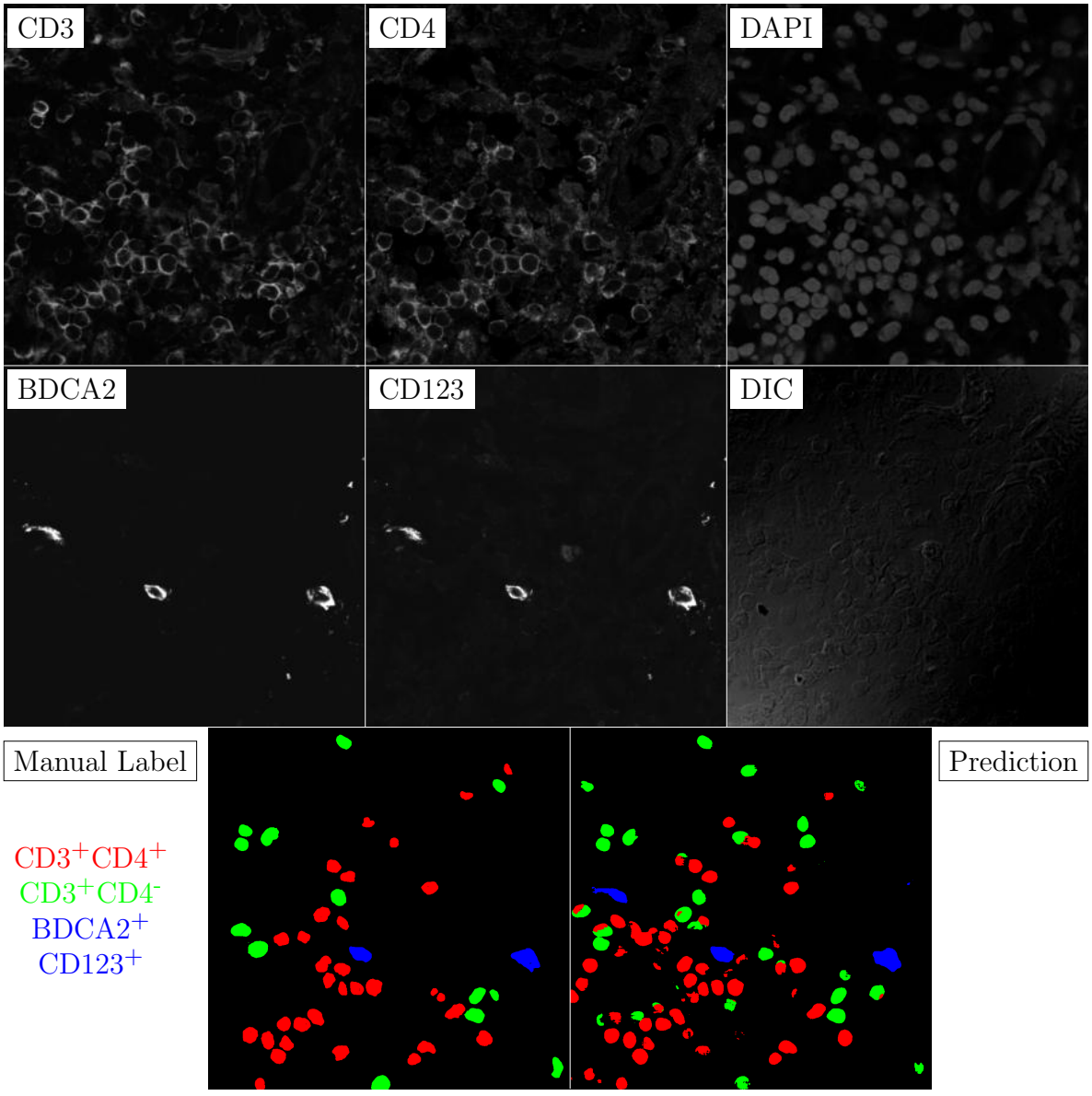

Figure G.167: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

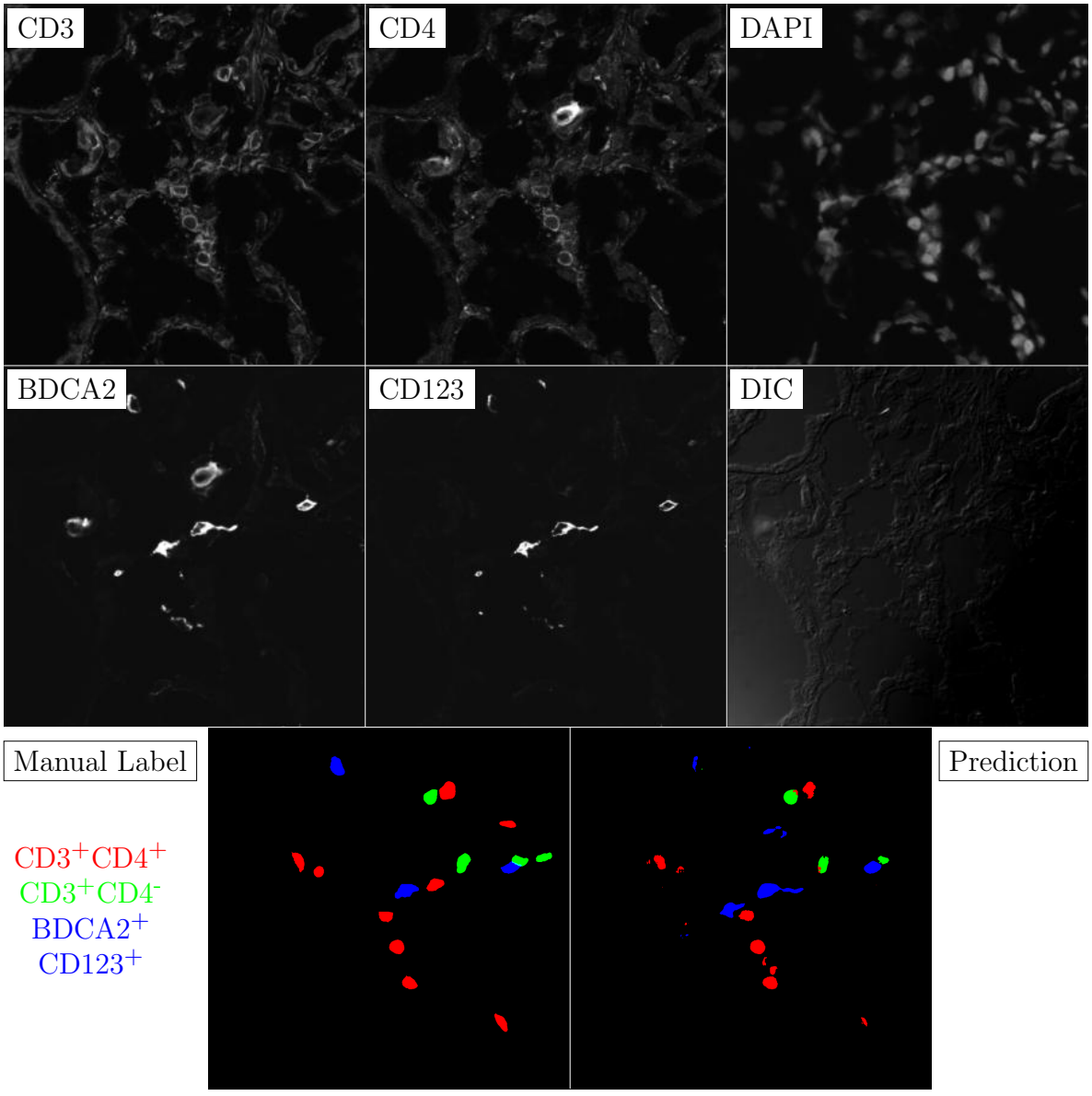

Figure G.168: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

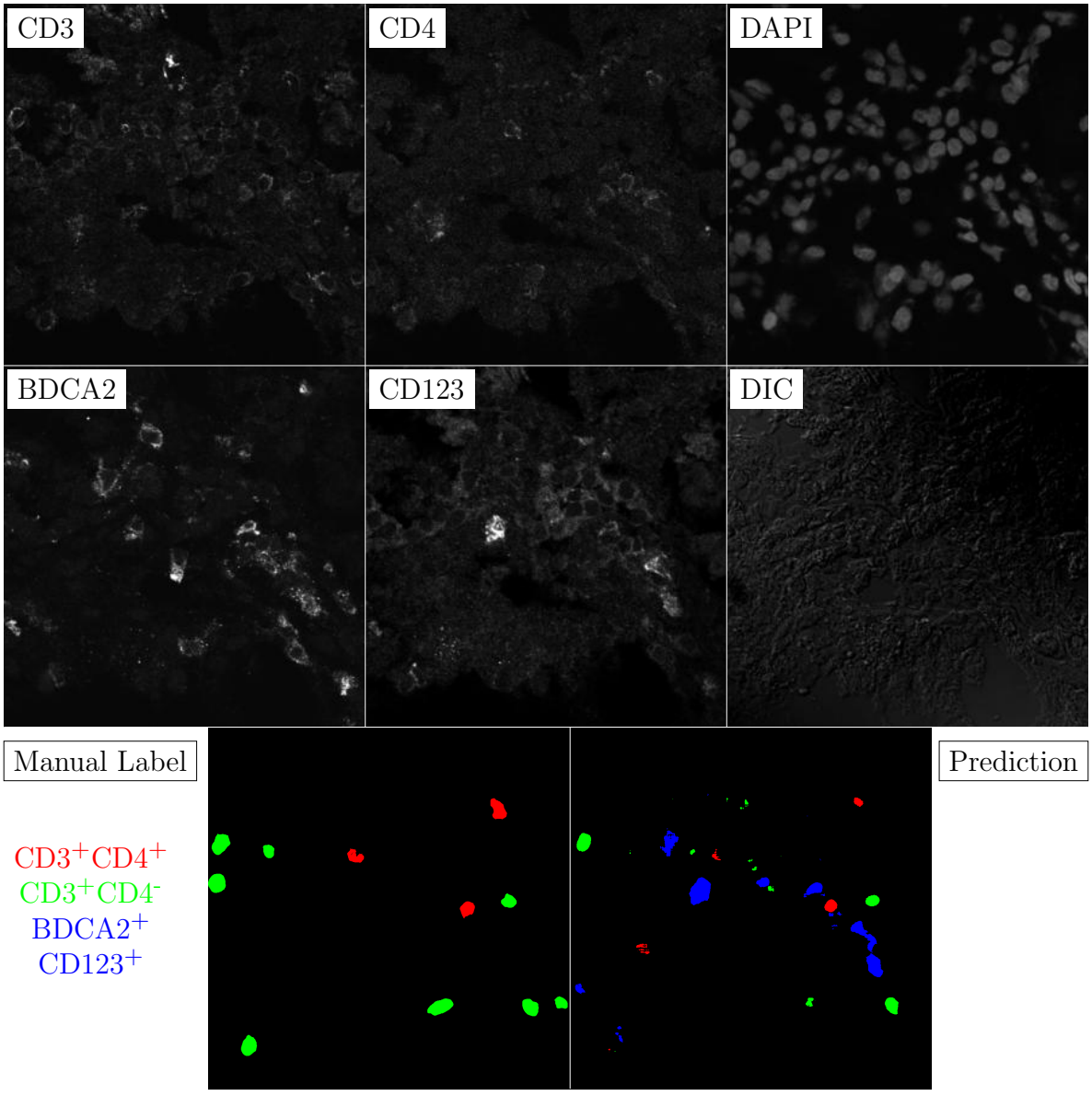

Figure G.169: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

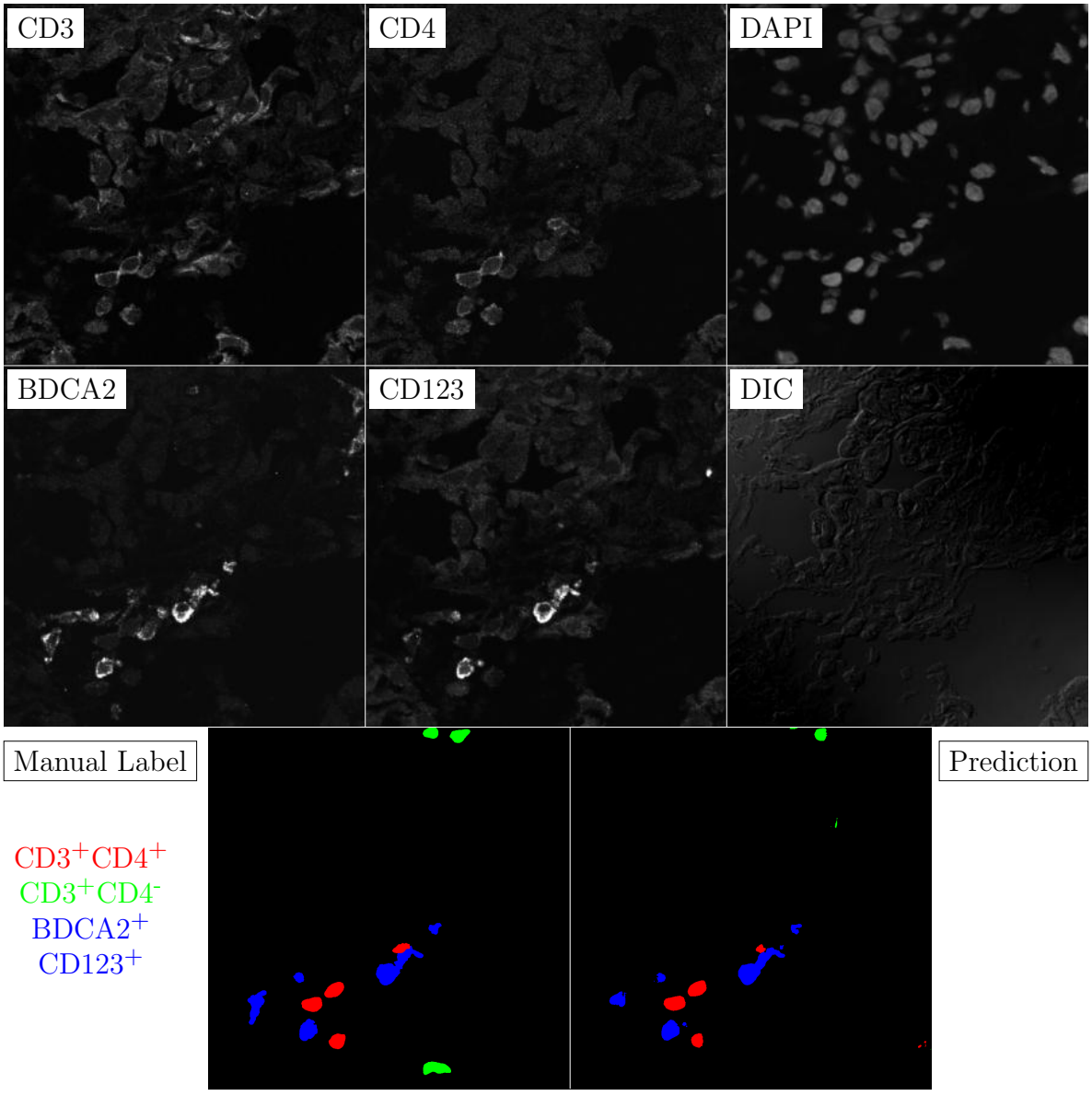

Figure G.170: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

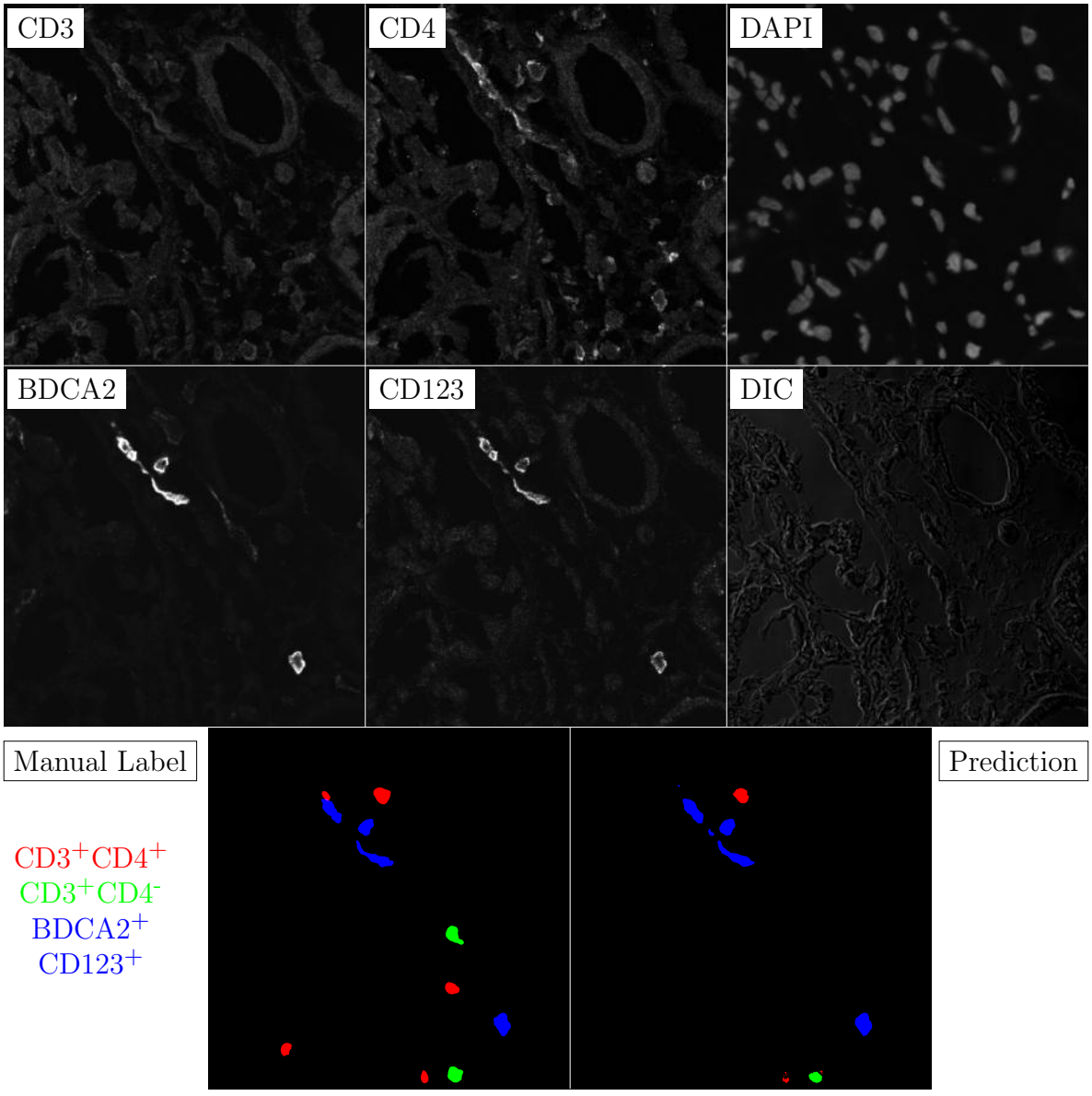

Figure G.171: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

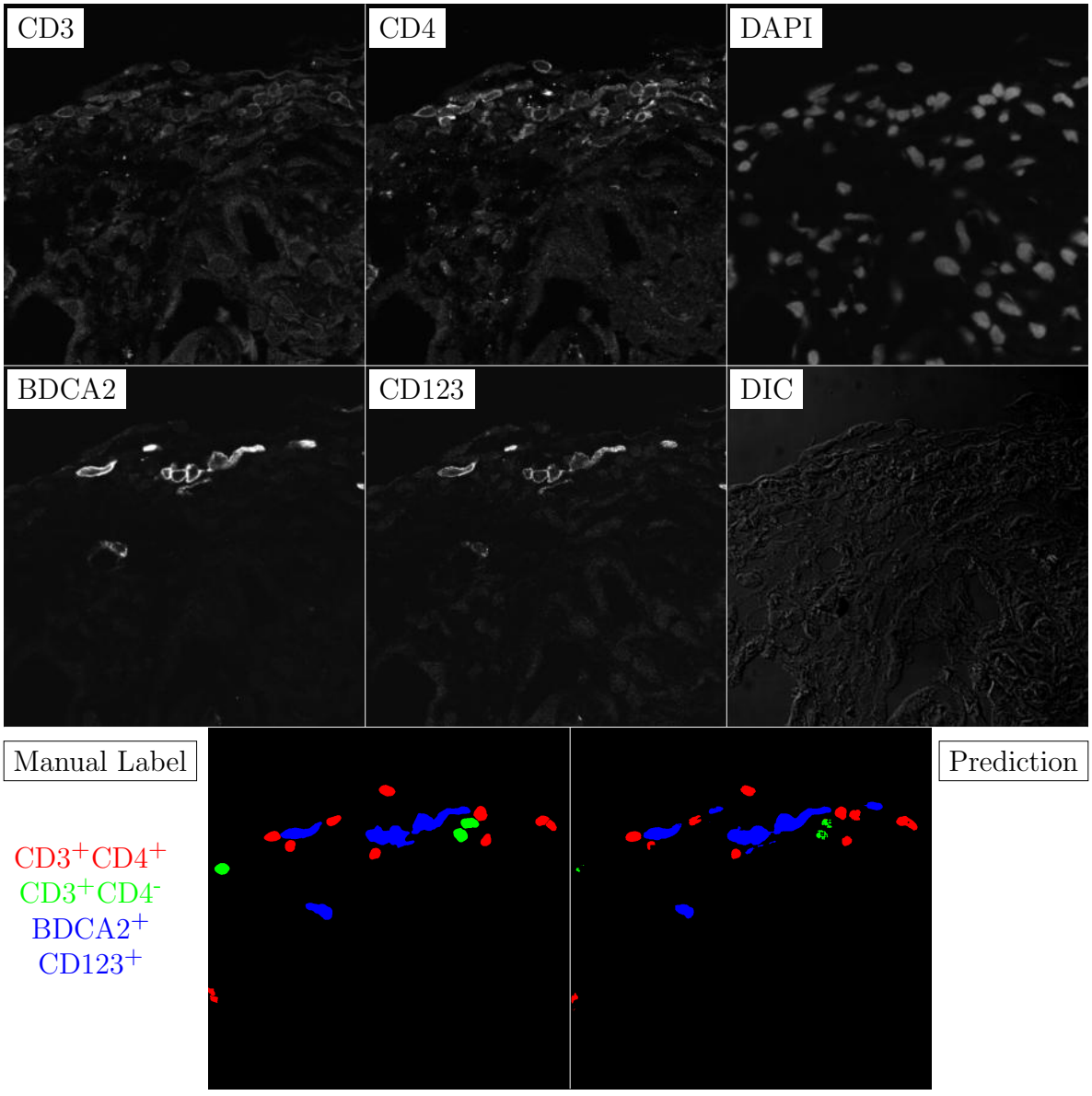

Figure G.172: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

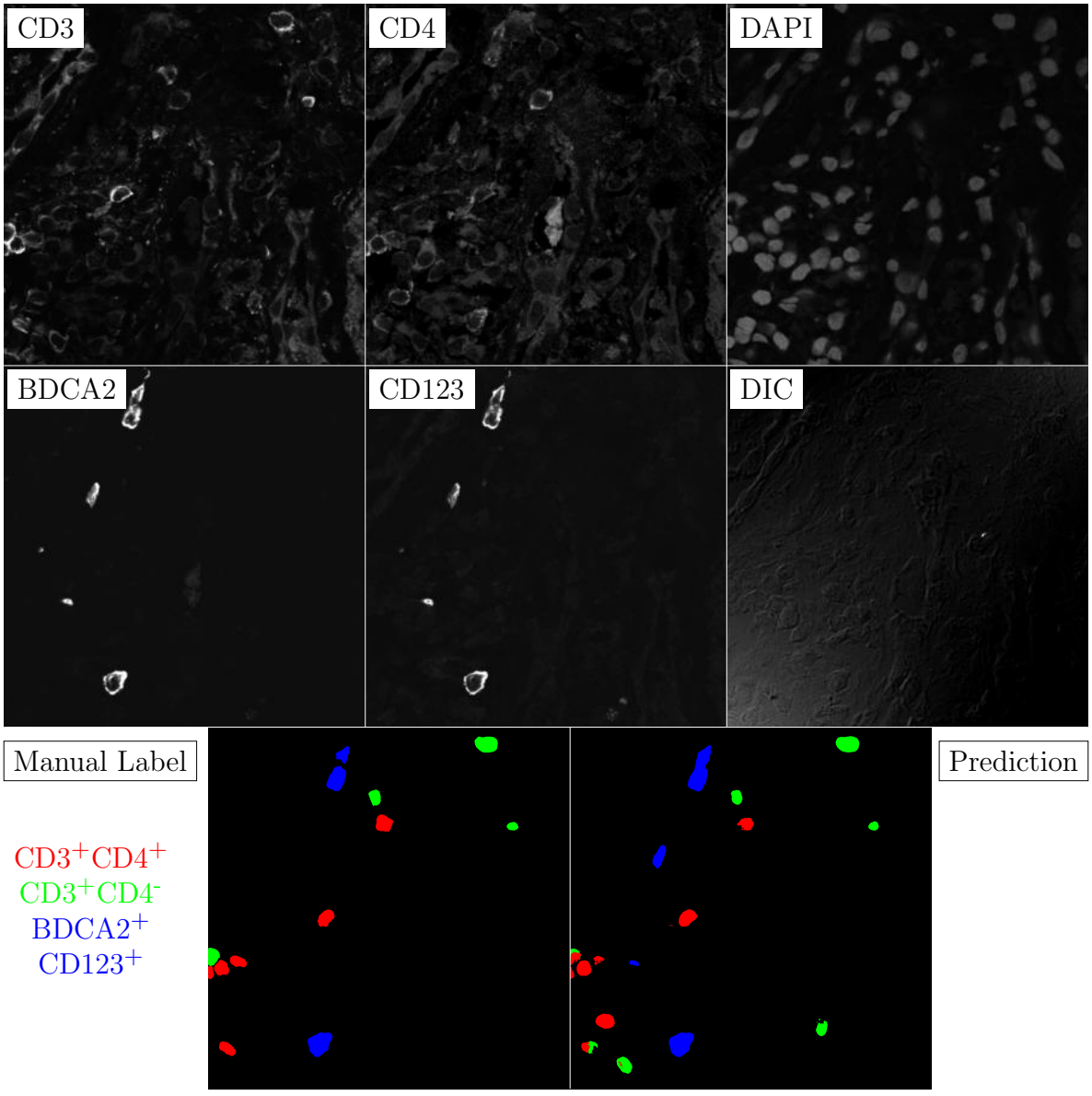

Figure G.173: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

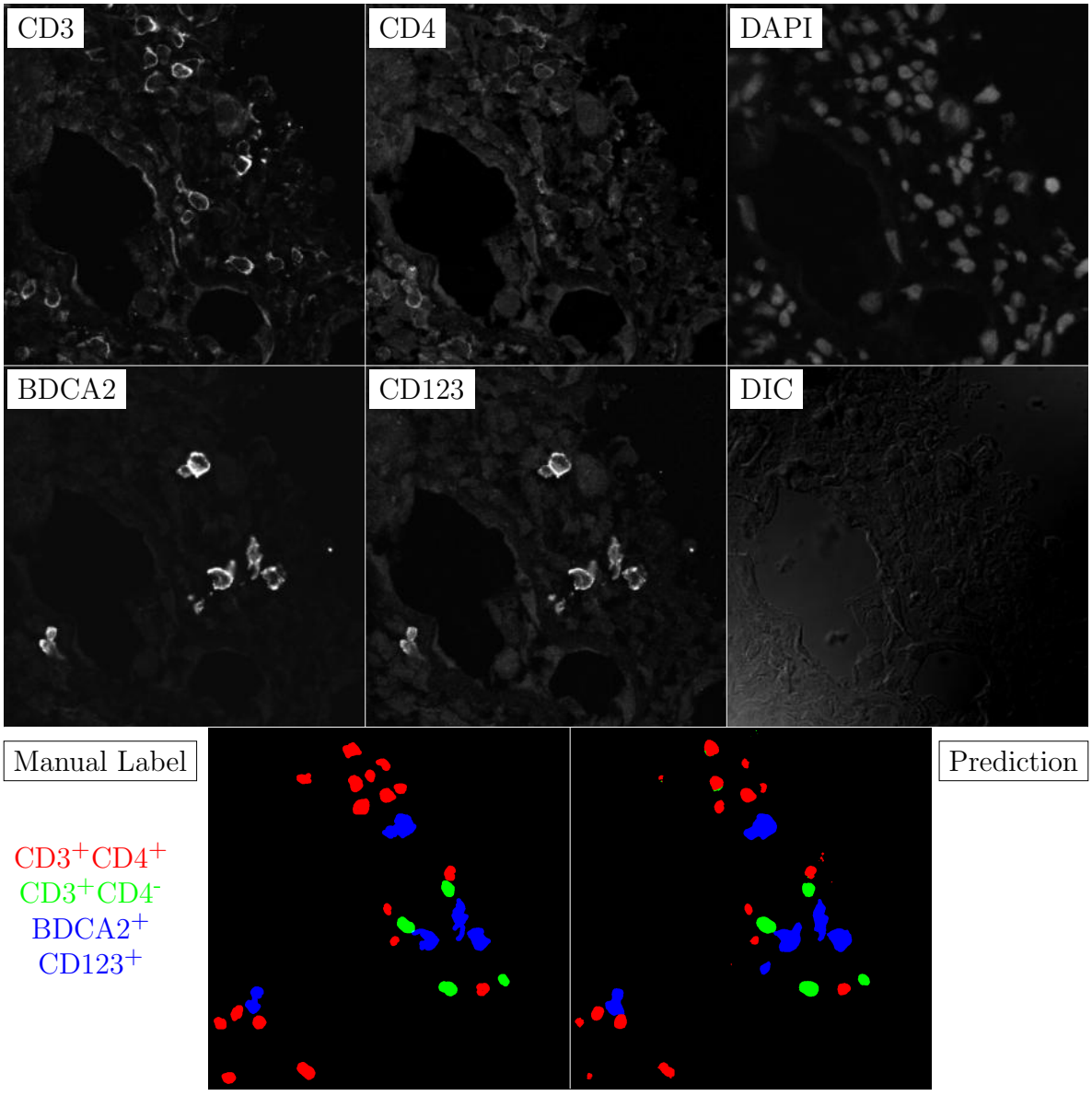

Figure G.174: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

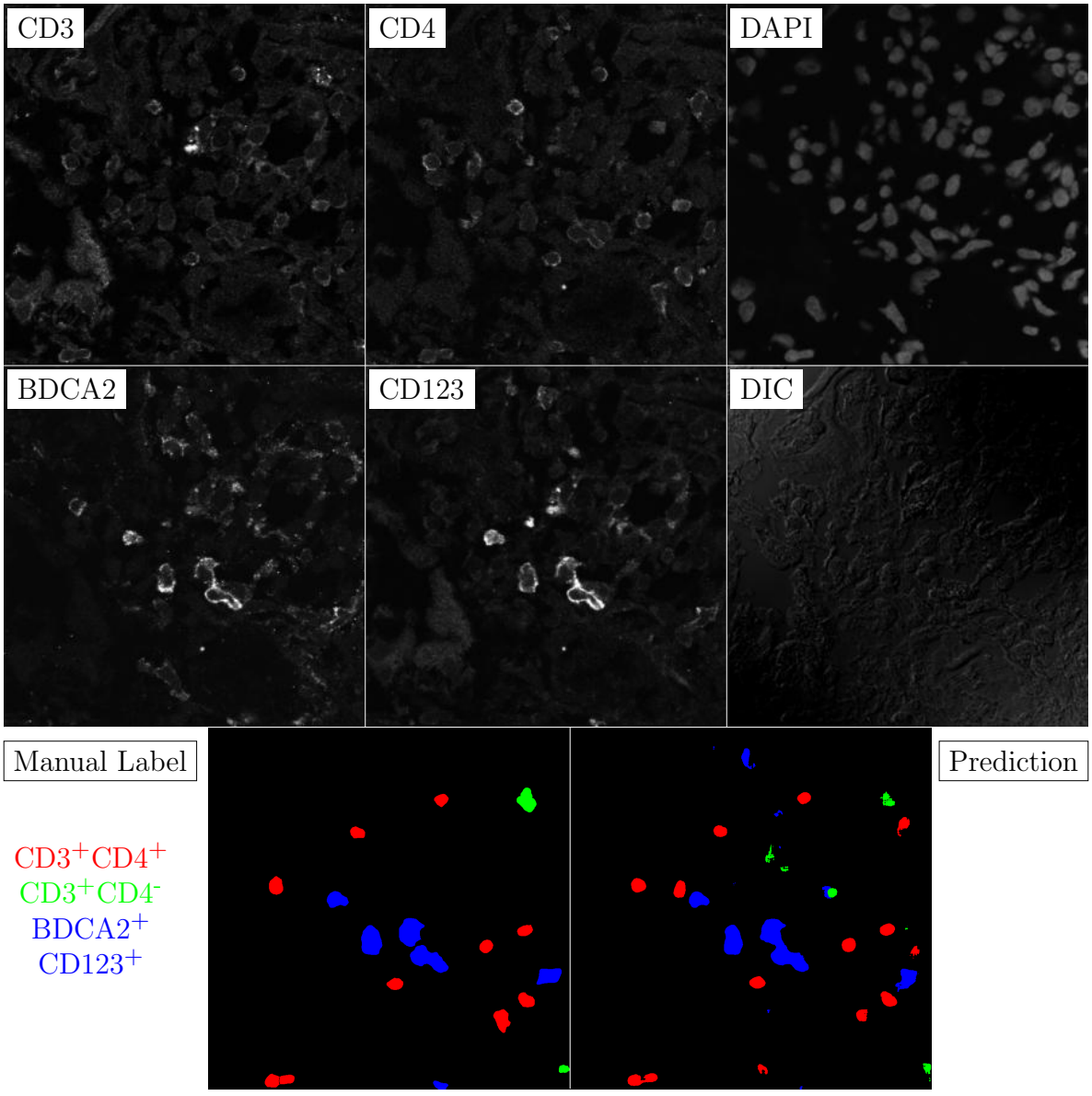

Figure G.175: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

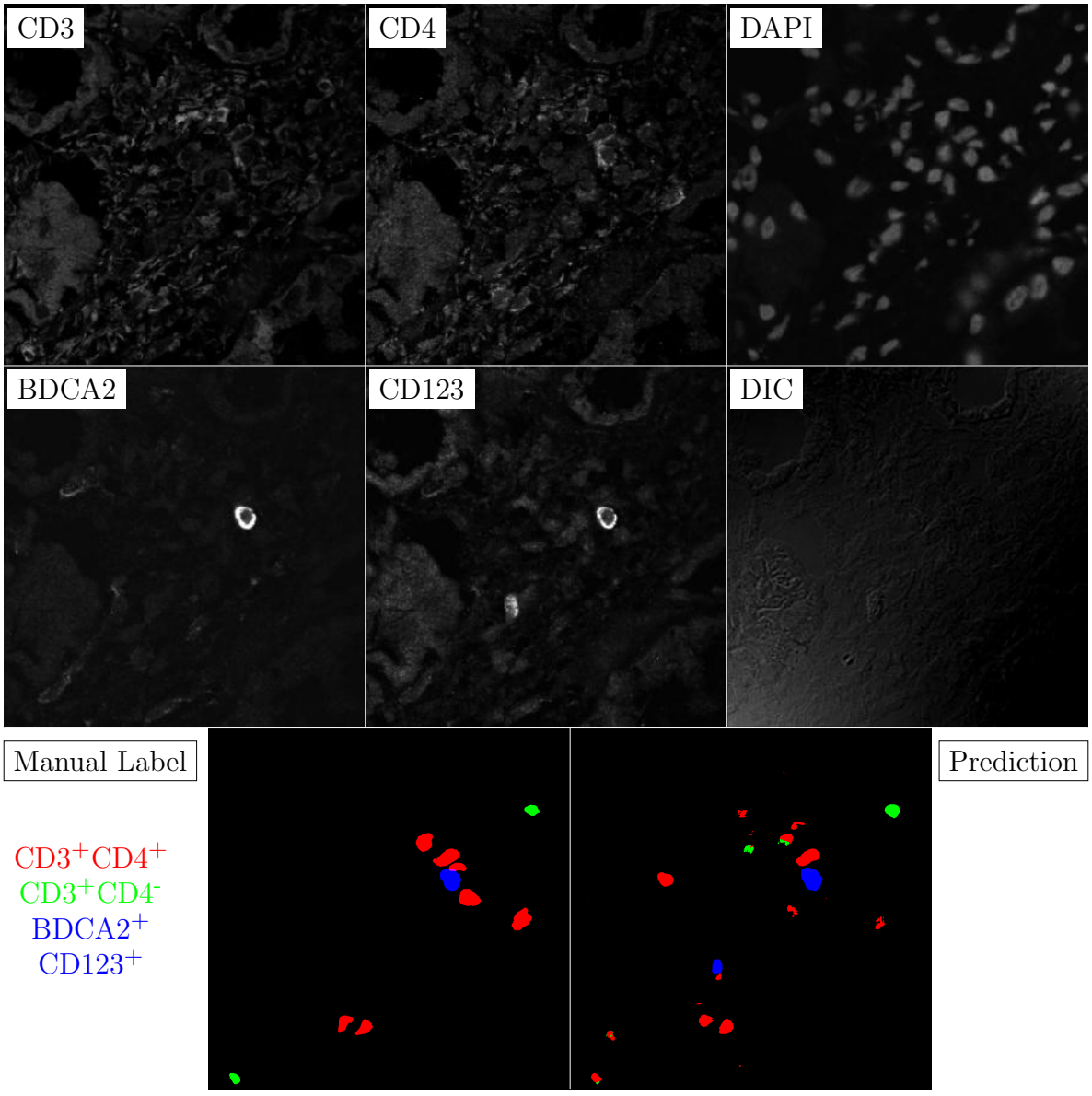

Figure G.176: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

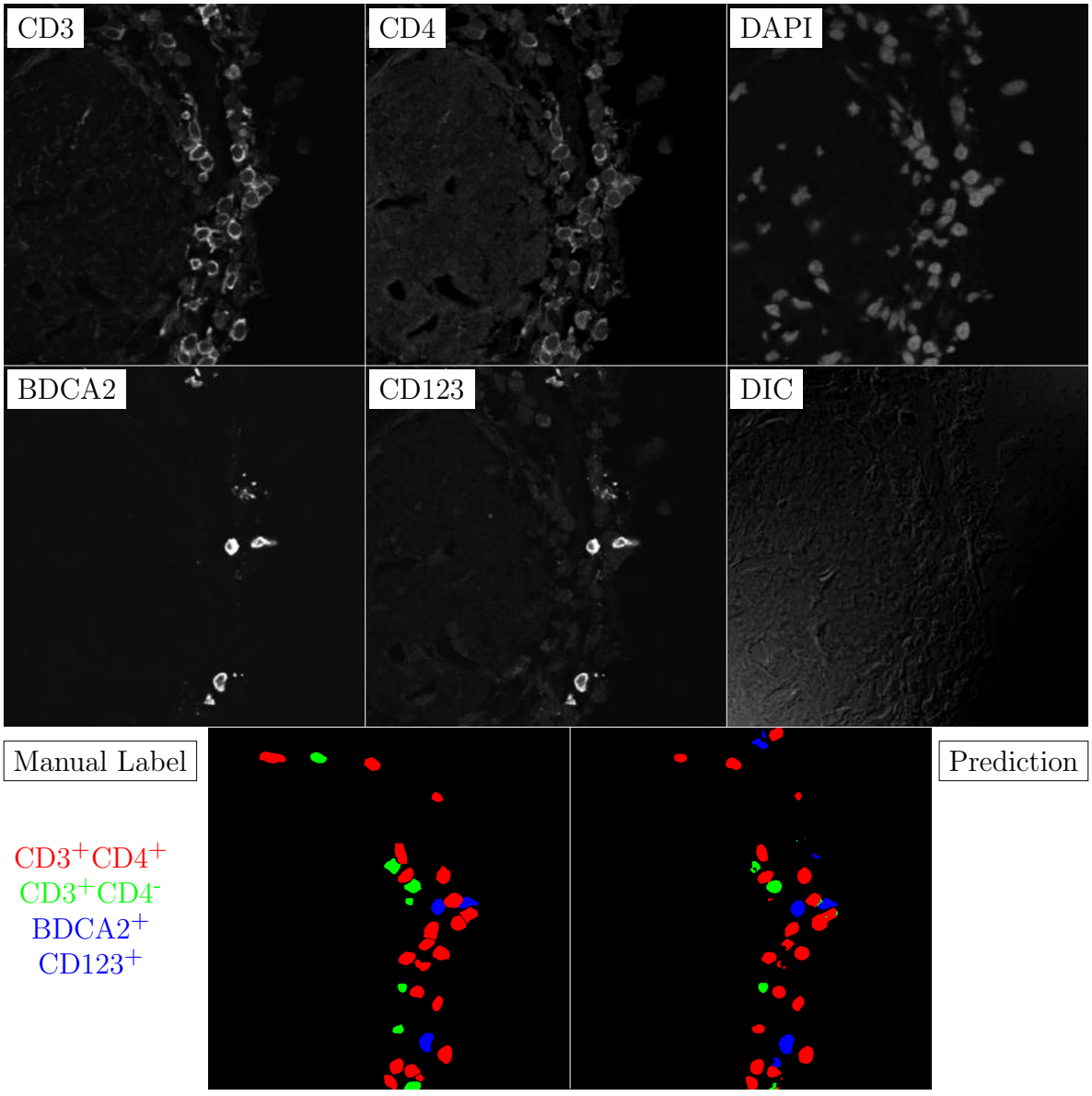

Figure G.177: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

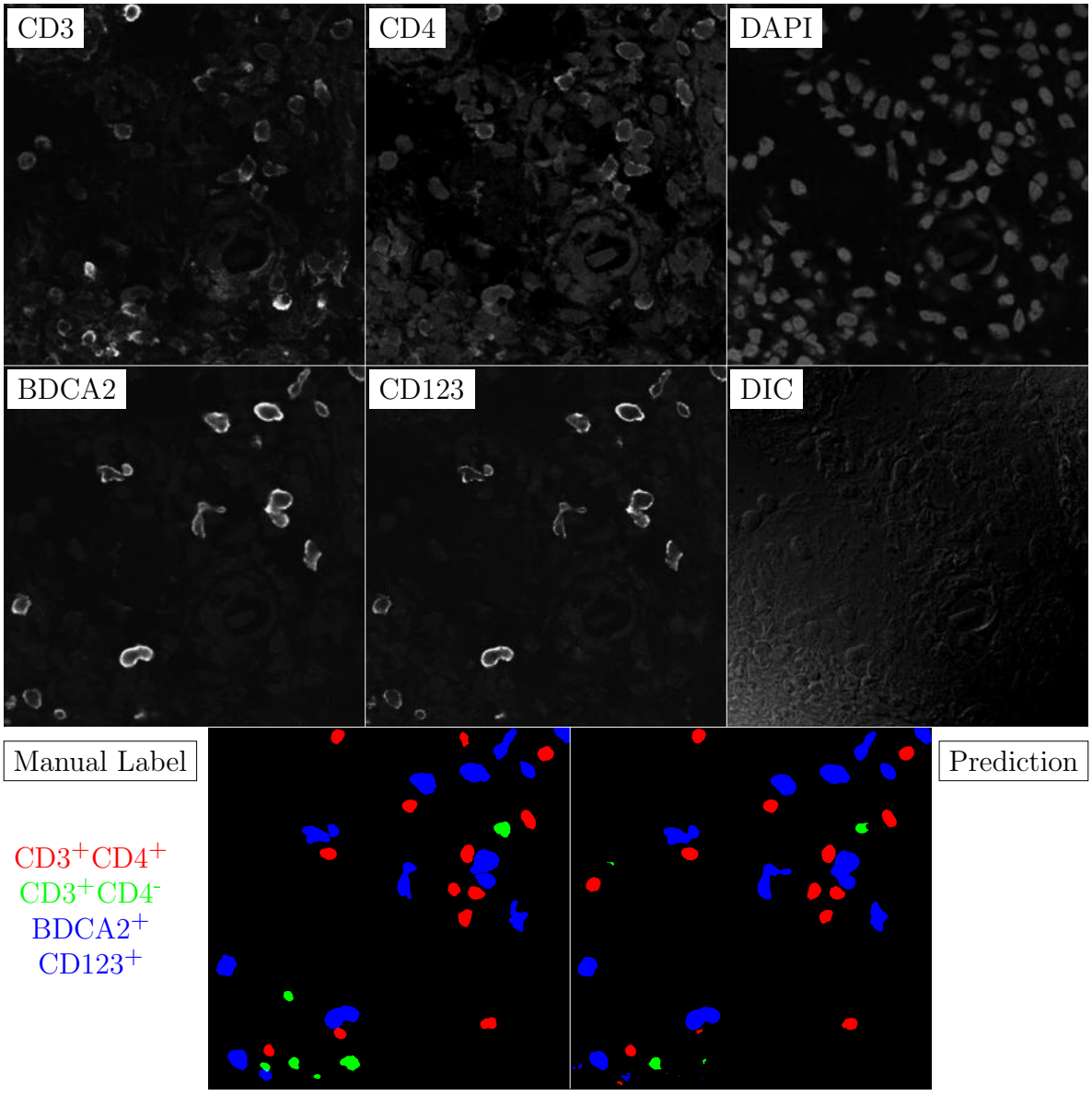

Figure G.178: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

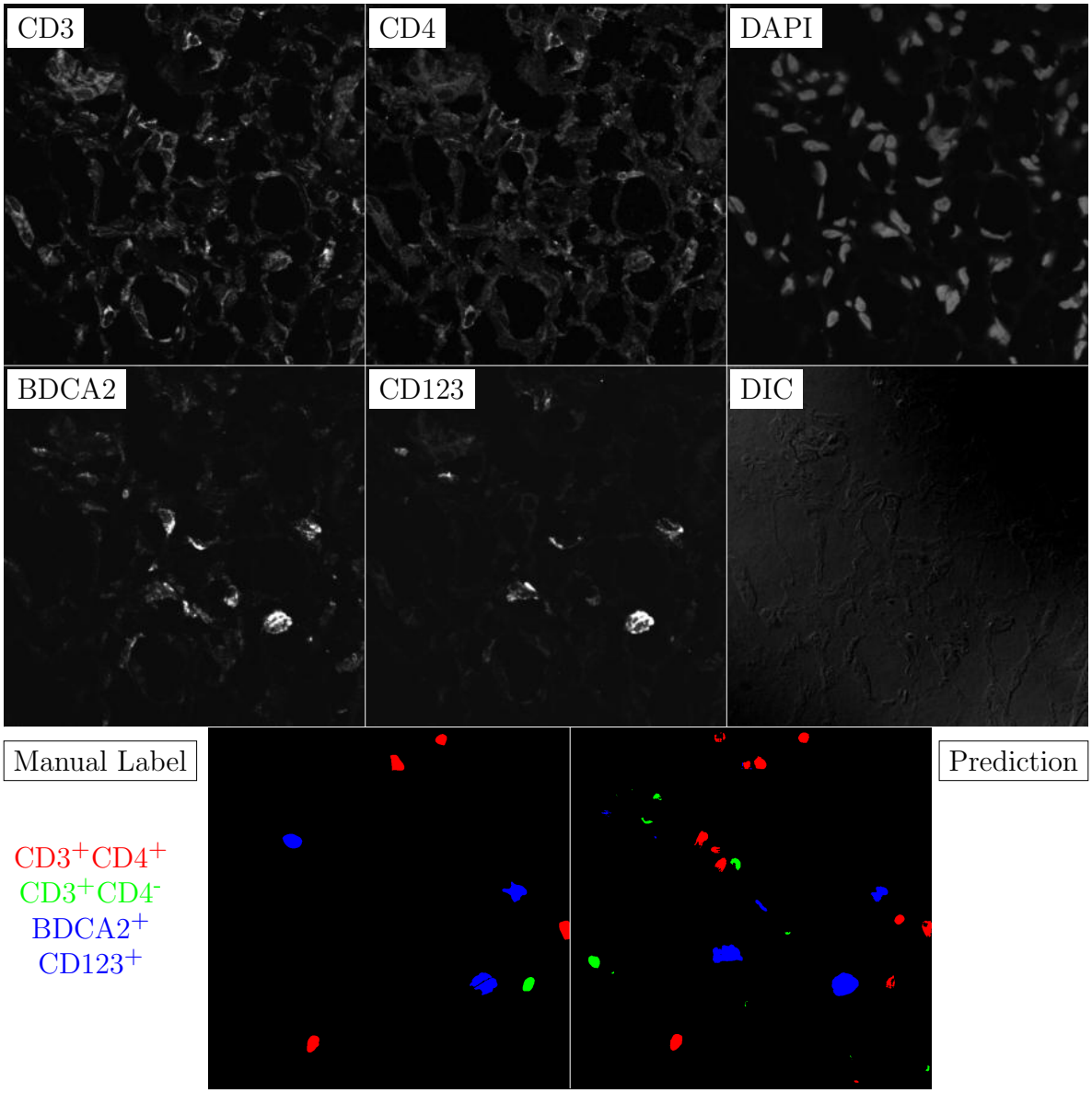

Figure G.179: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

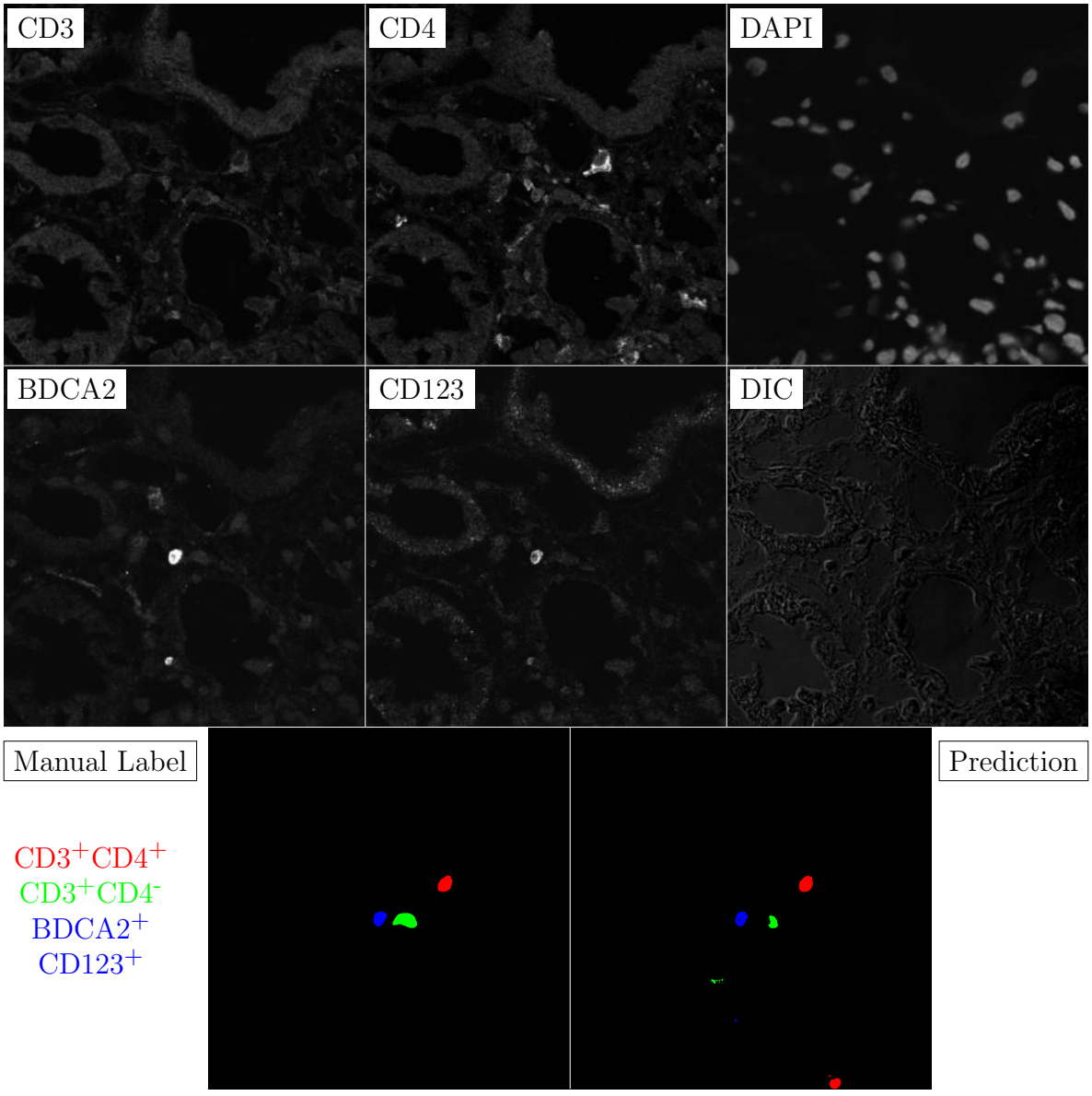

Figure G.180: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

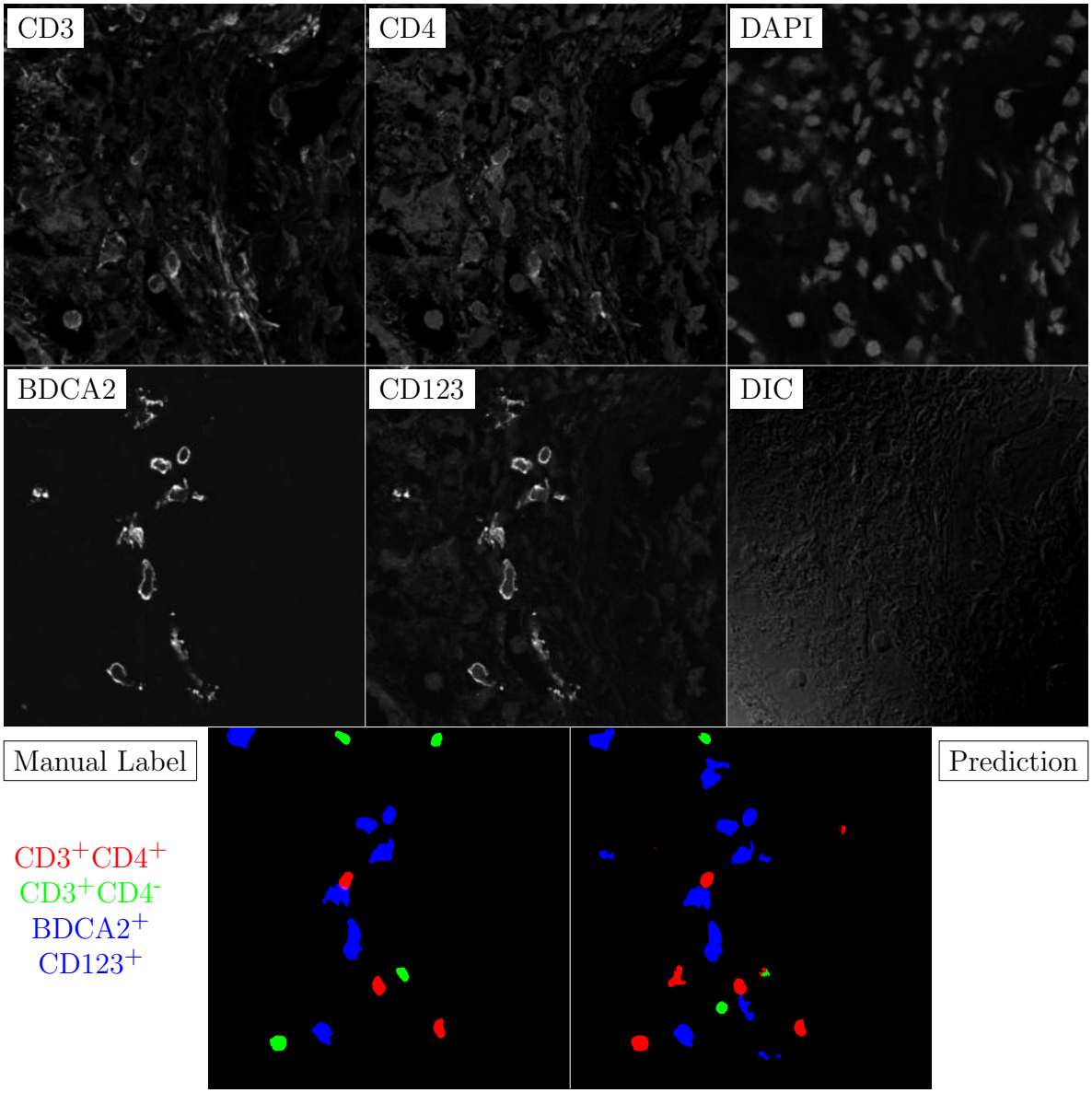

Figure G.181: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

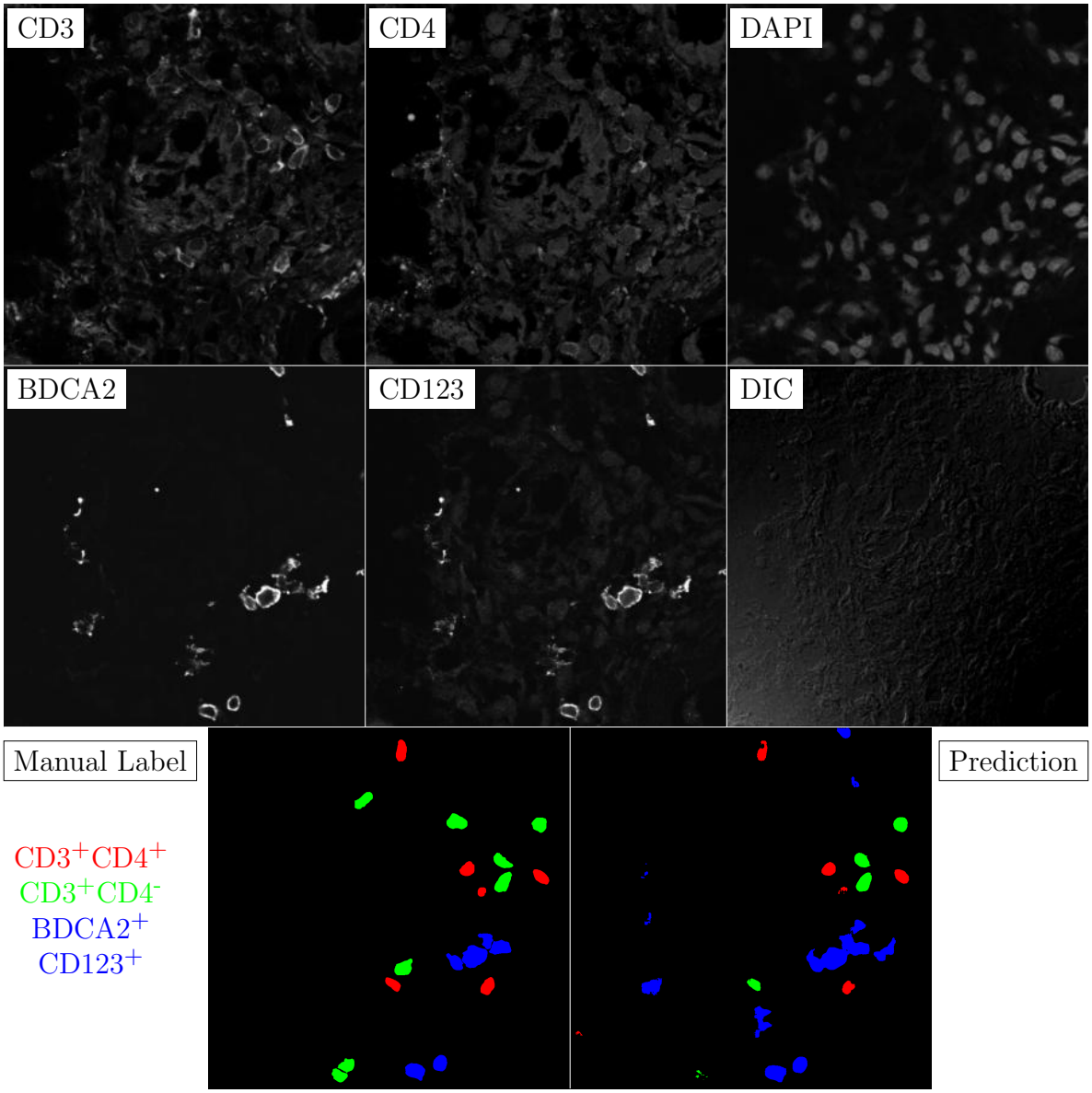

Figure G.182: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

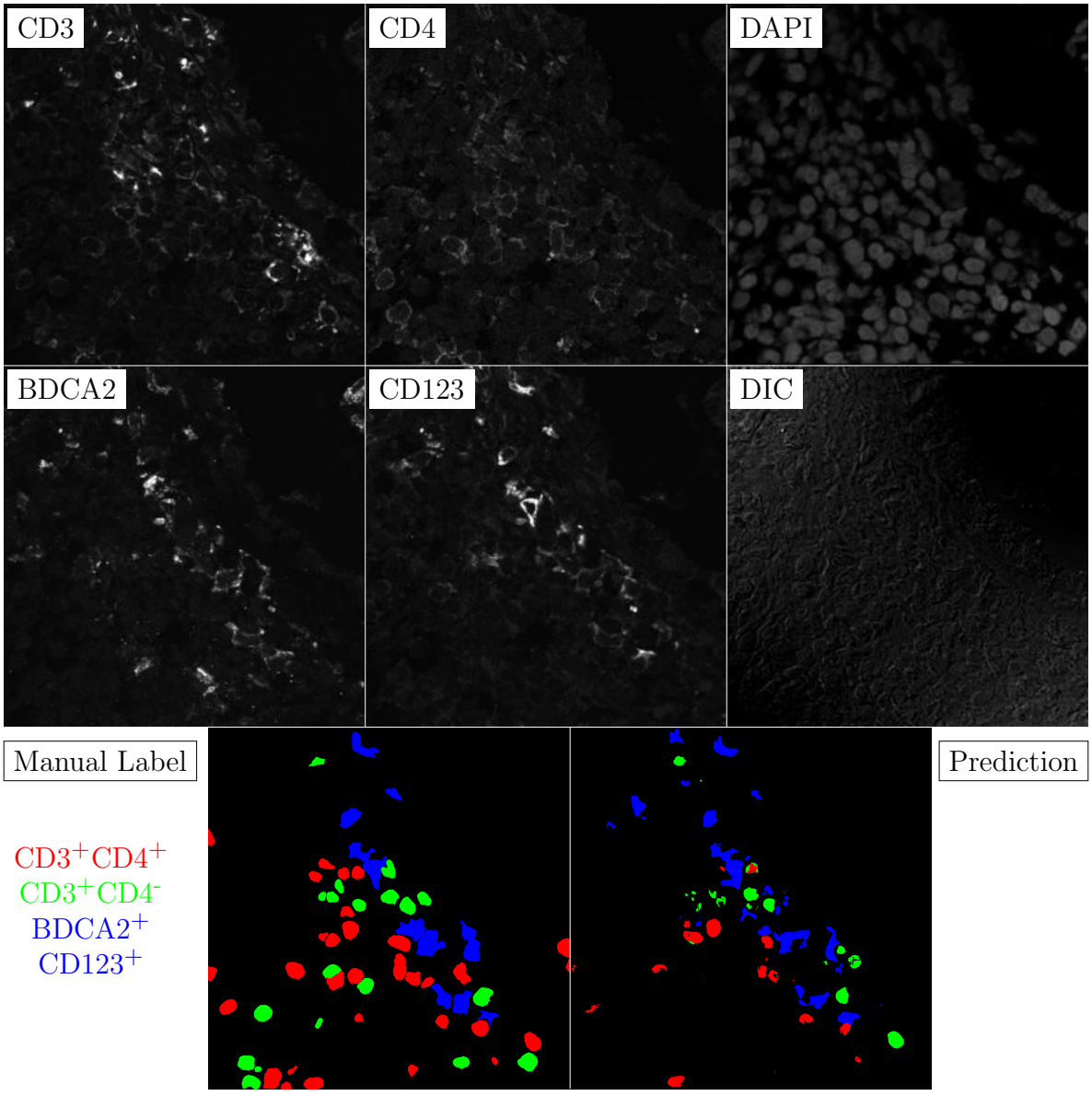

Figure G.183: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

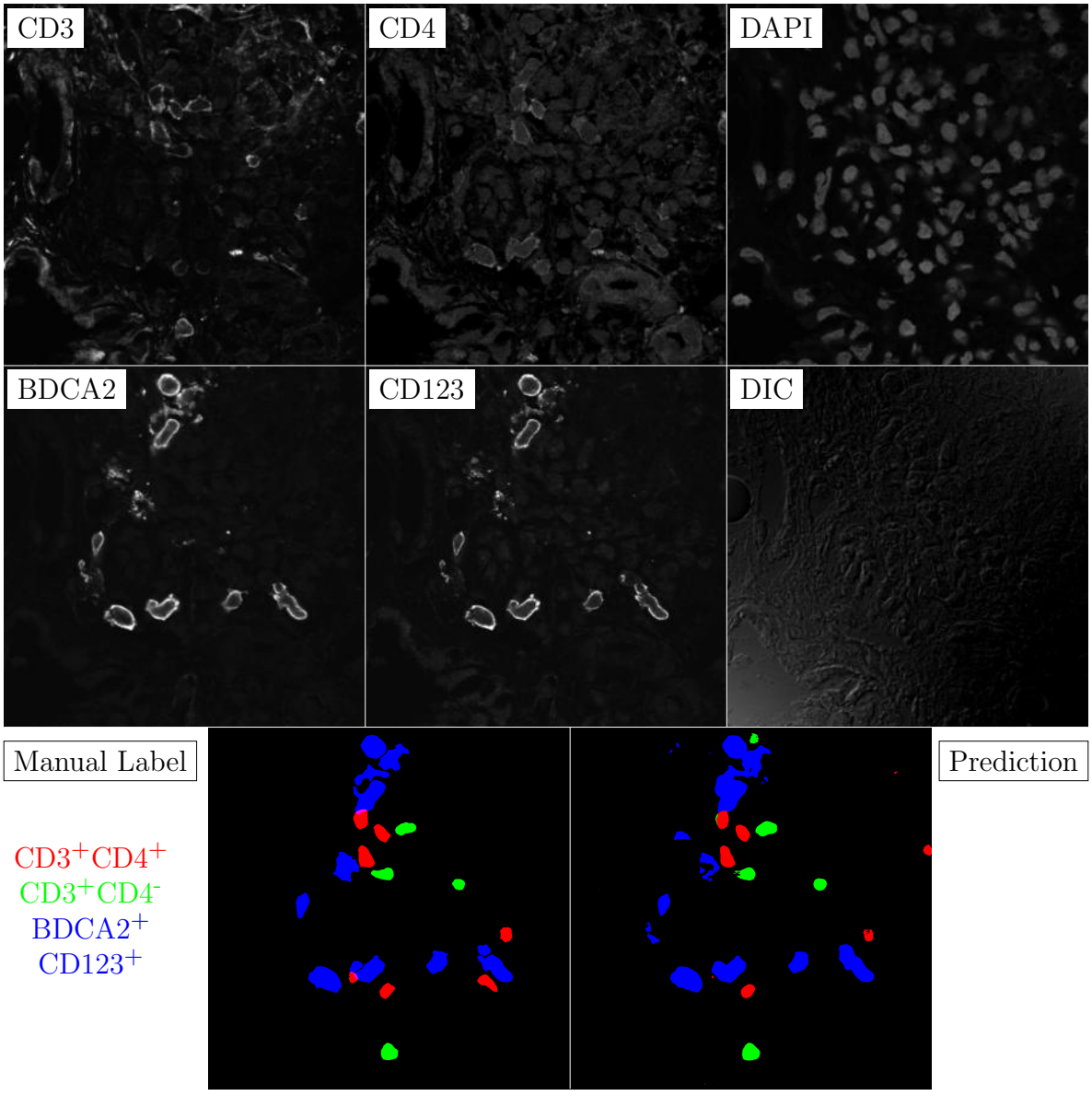

Figure G.184: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

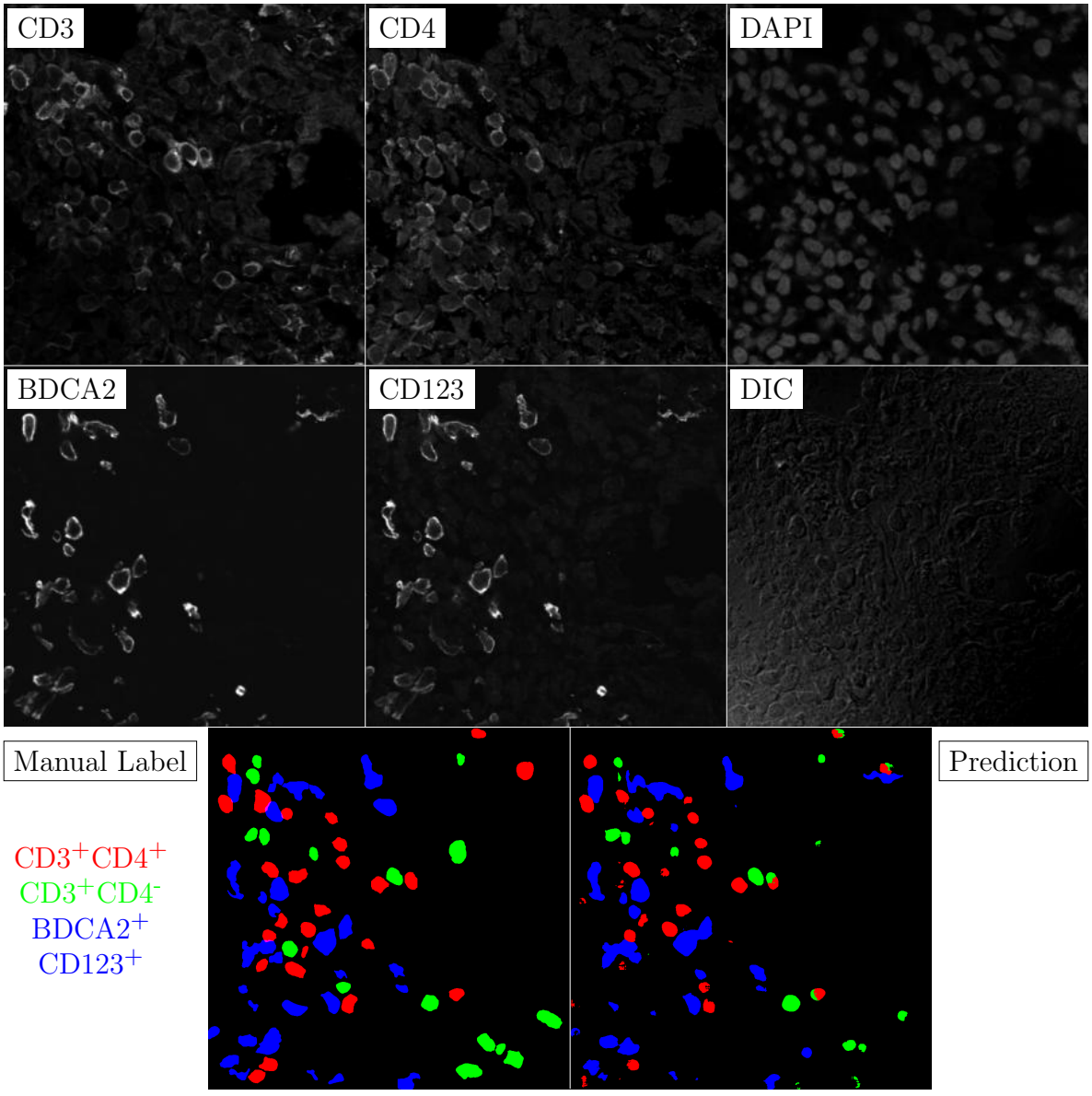

Figure G.185: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

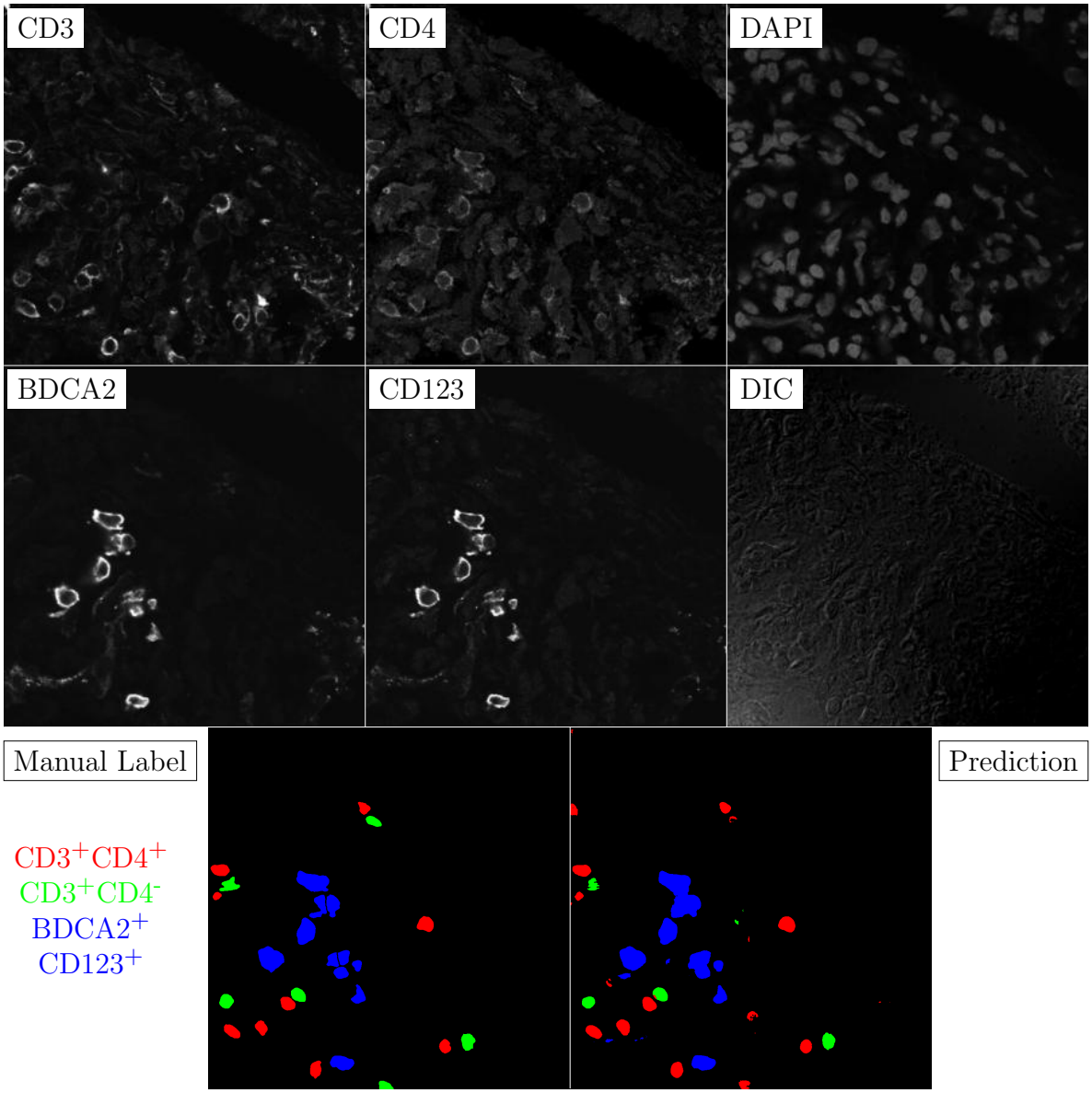

Figure G.186: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

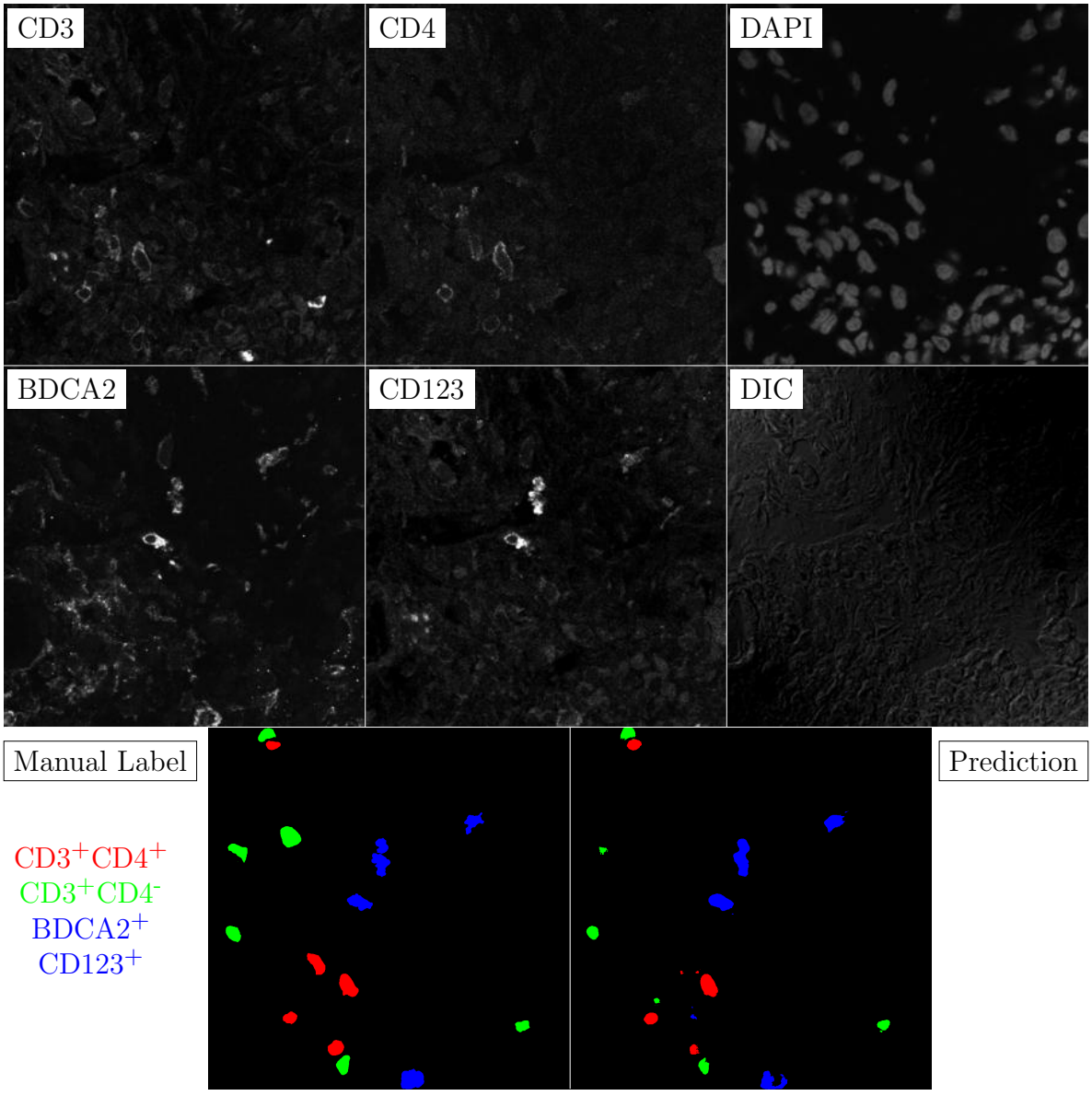

Figure G.187: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

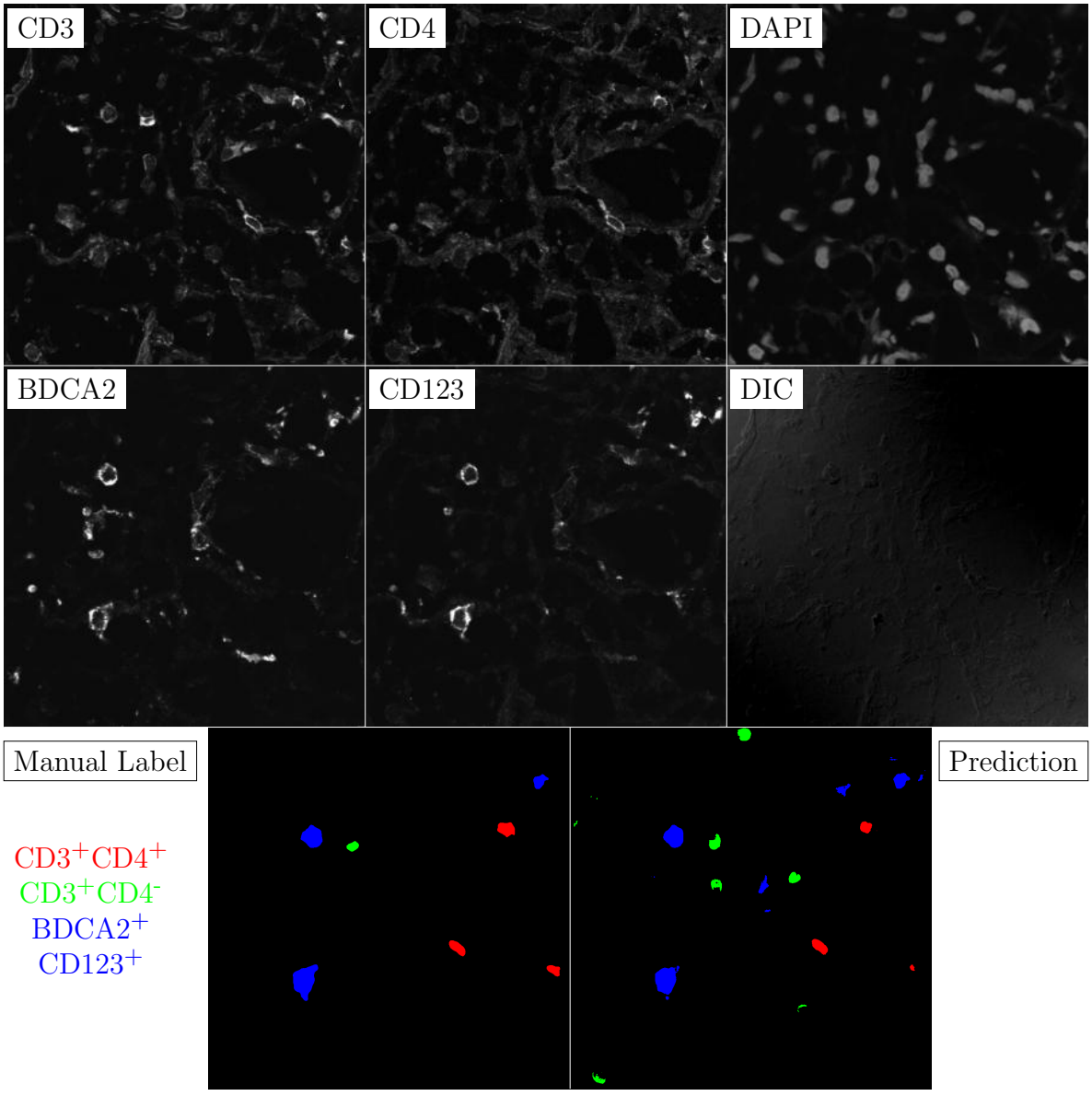

Figure G.188: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

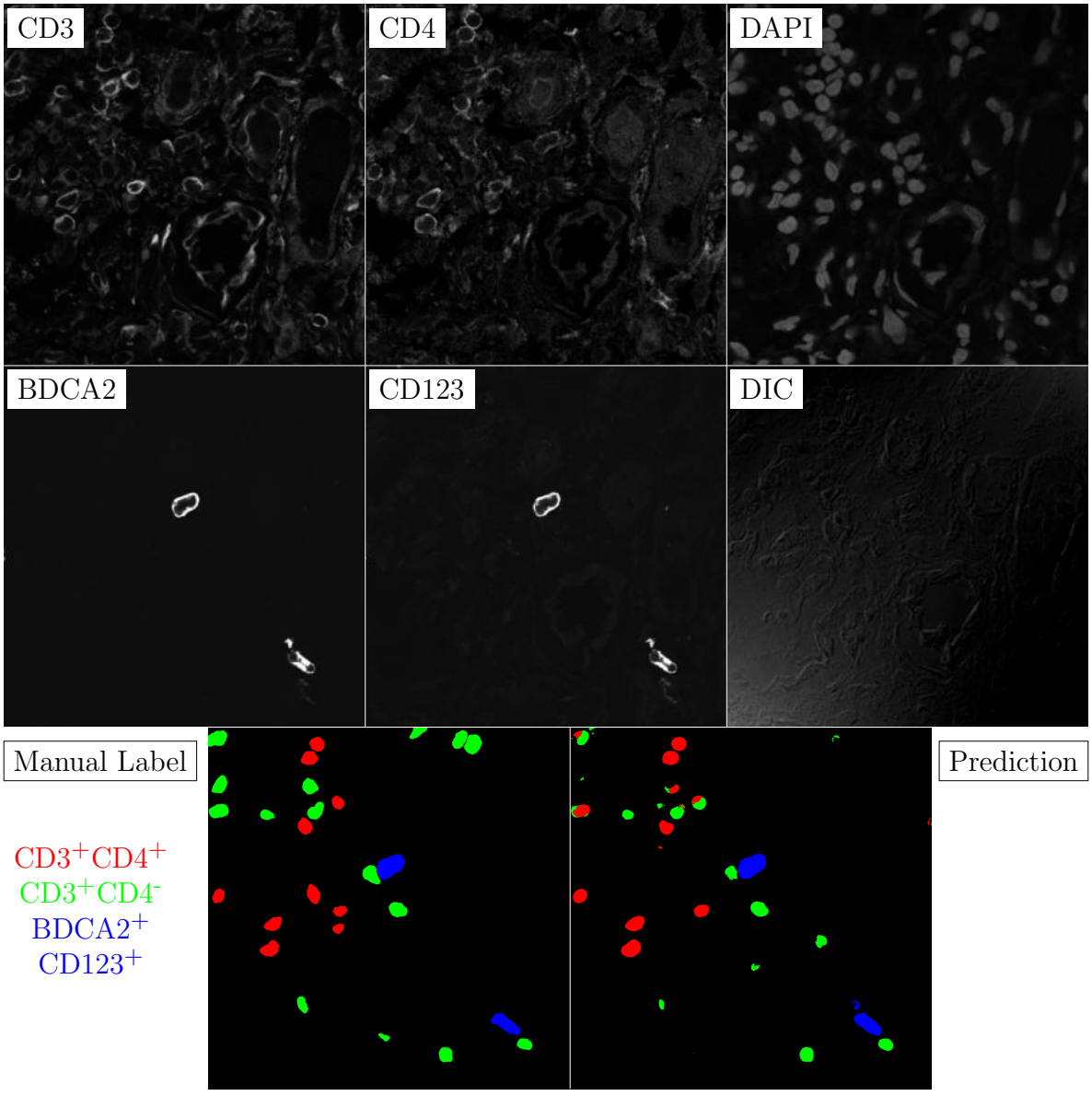

Figure G.189: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

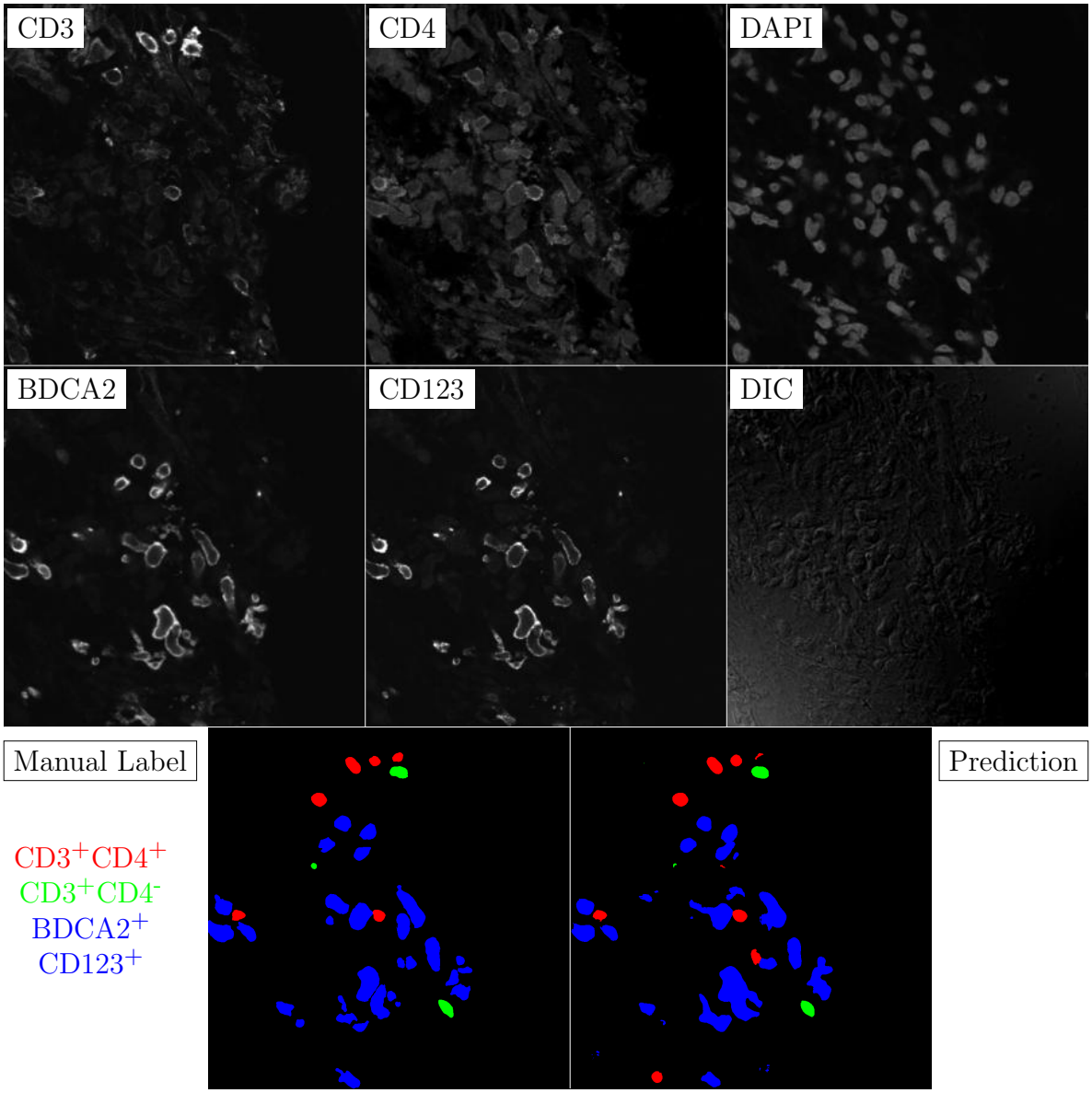

Figure G.190: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

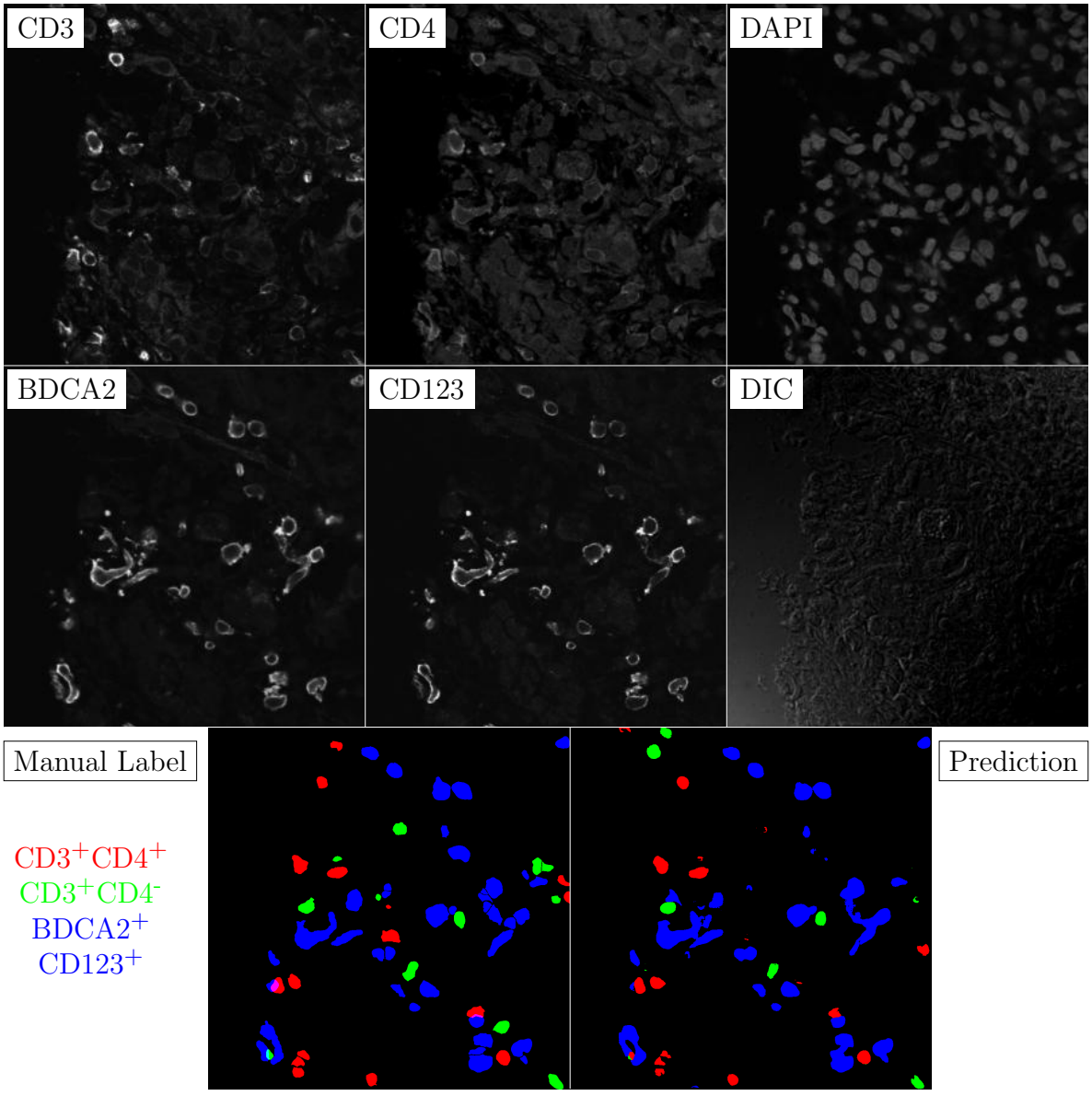

Figure G.191: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

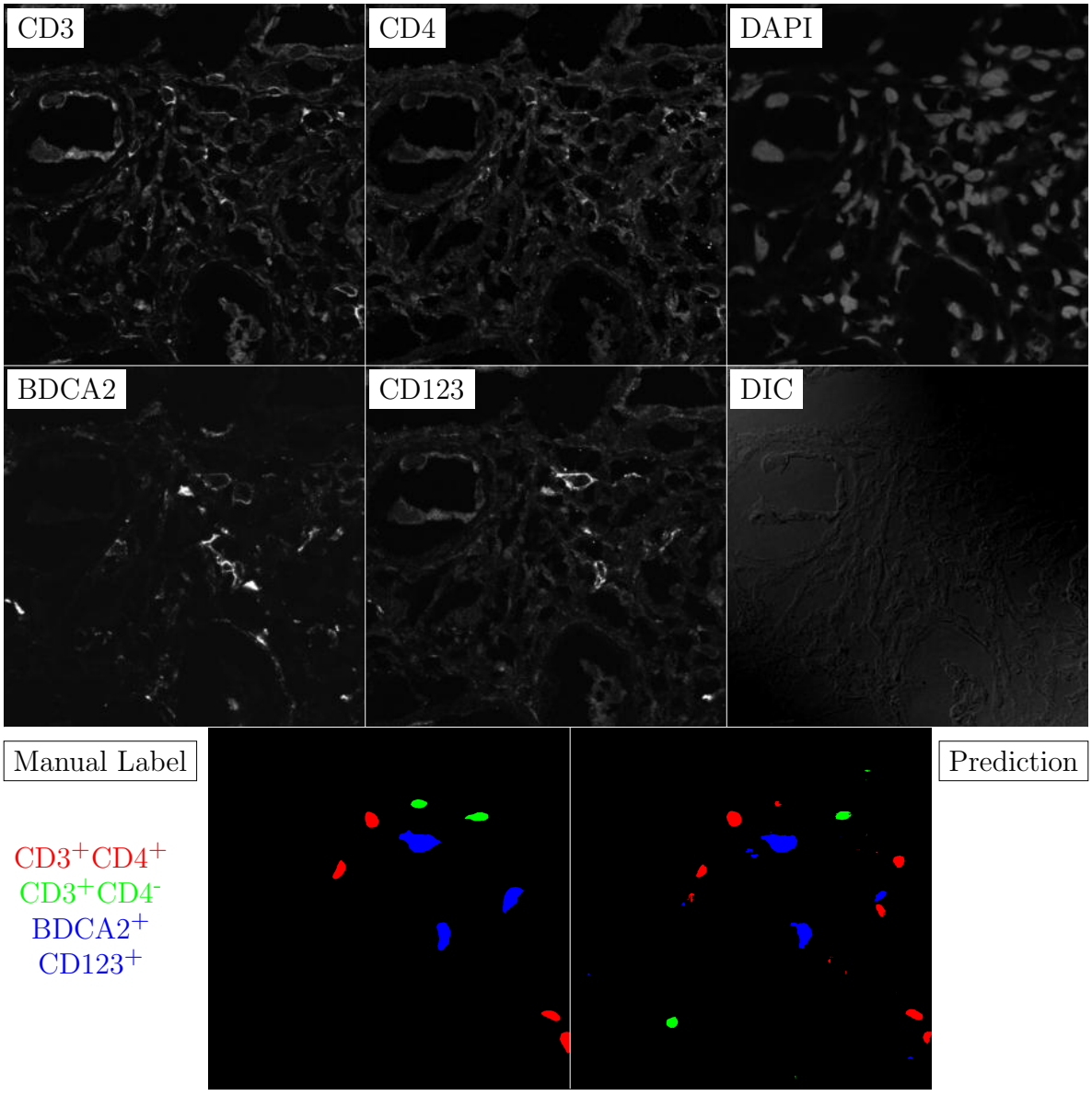

Figure G.192: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

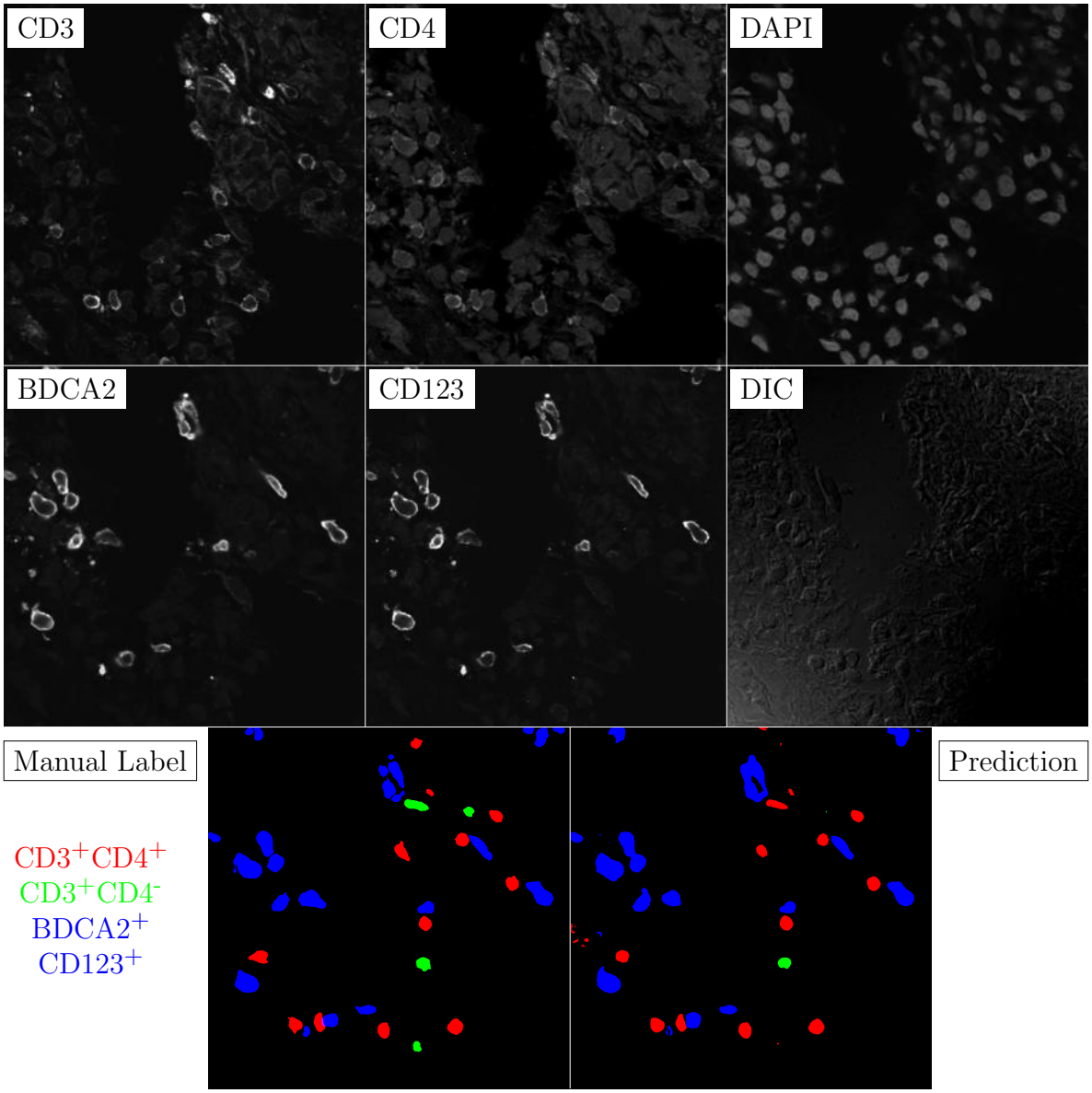

Figure G.193: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

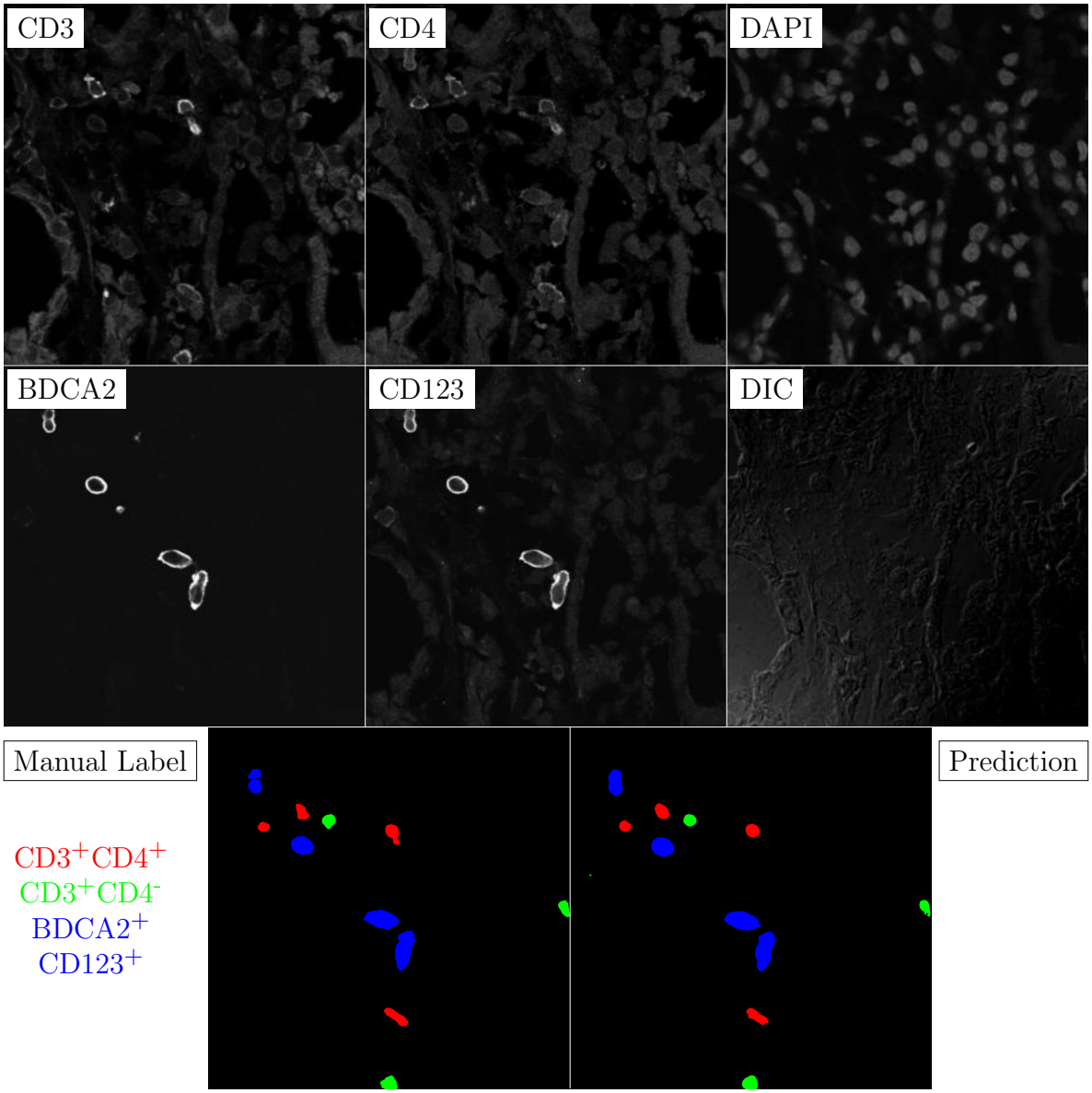

Figure G.194: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

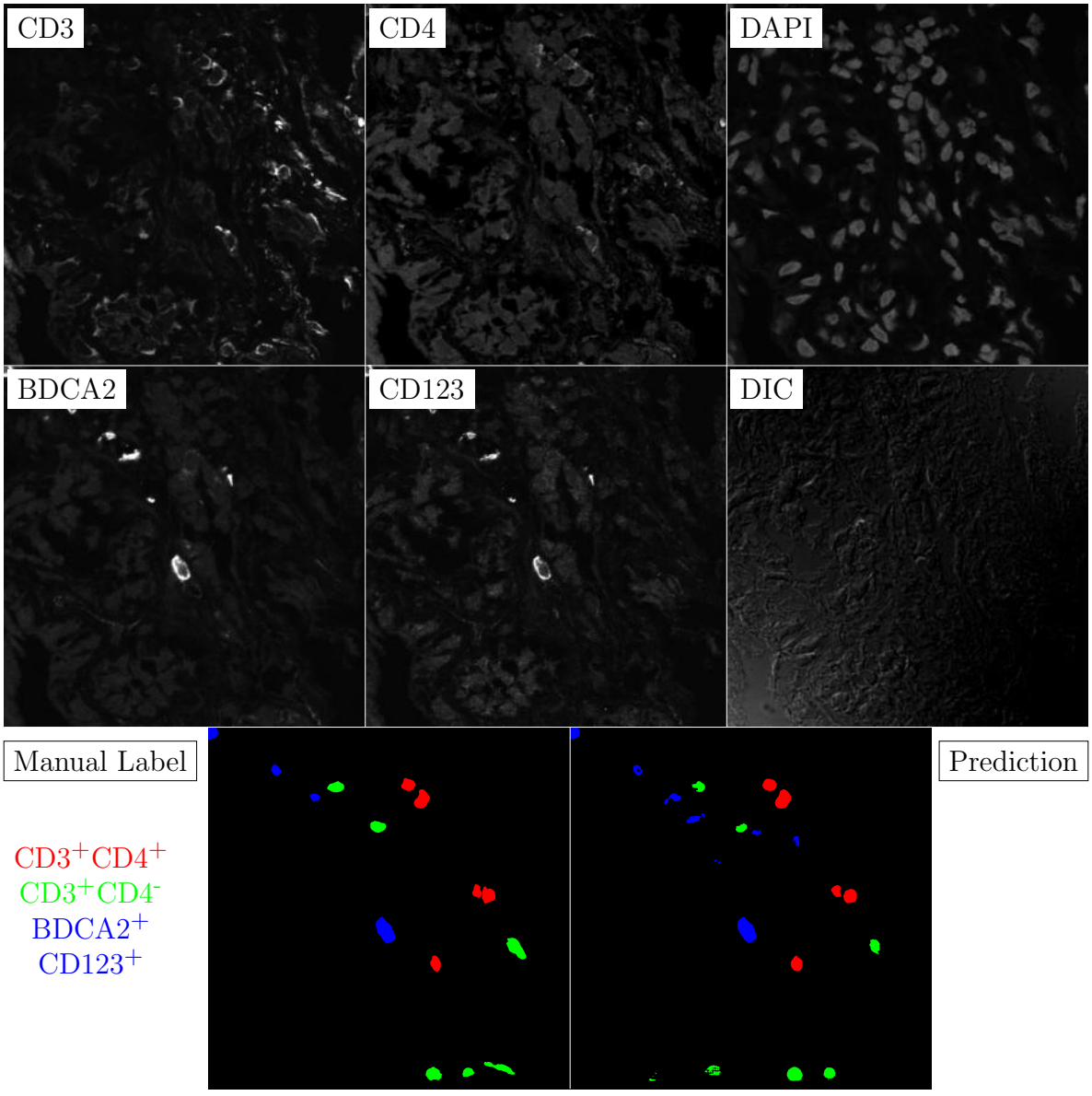

Figure G.195: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

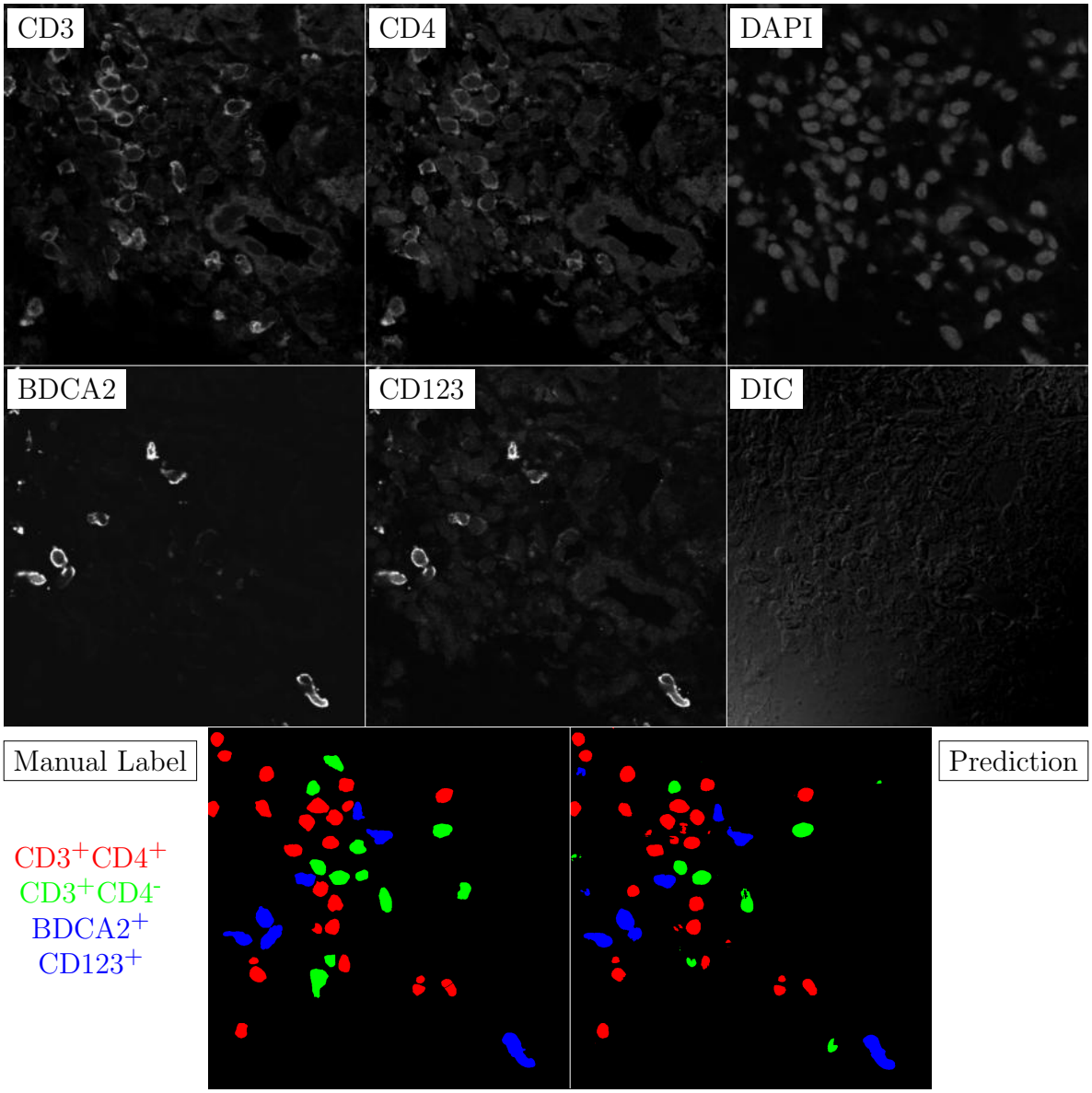

Figure G.196: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

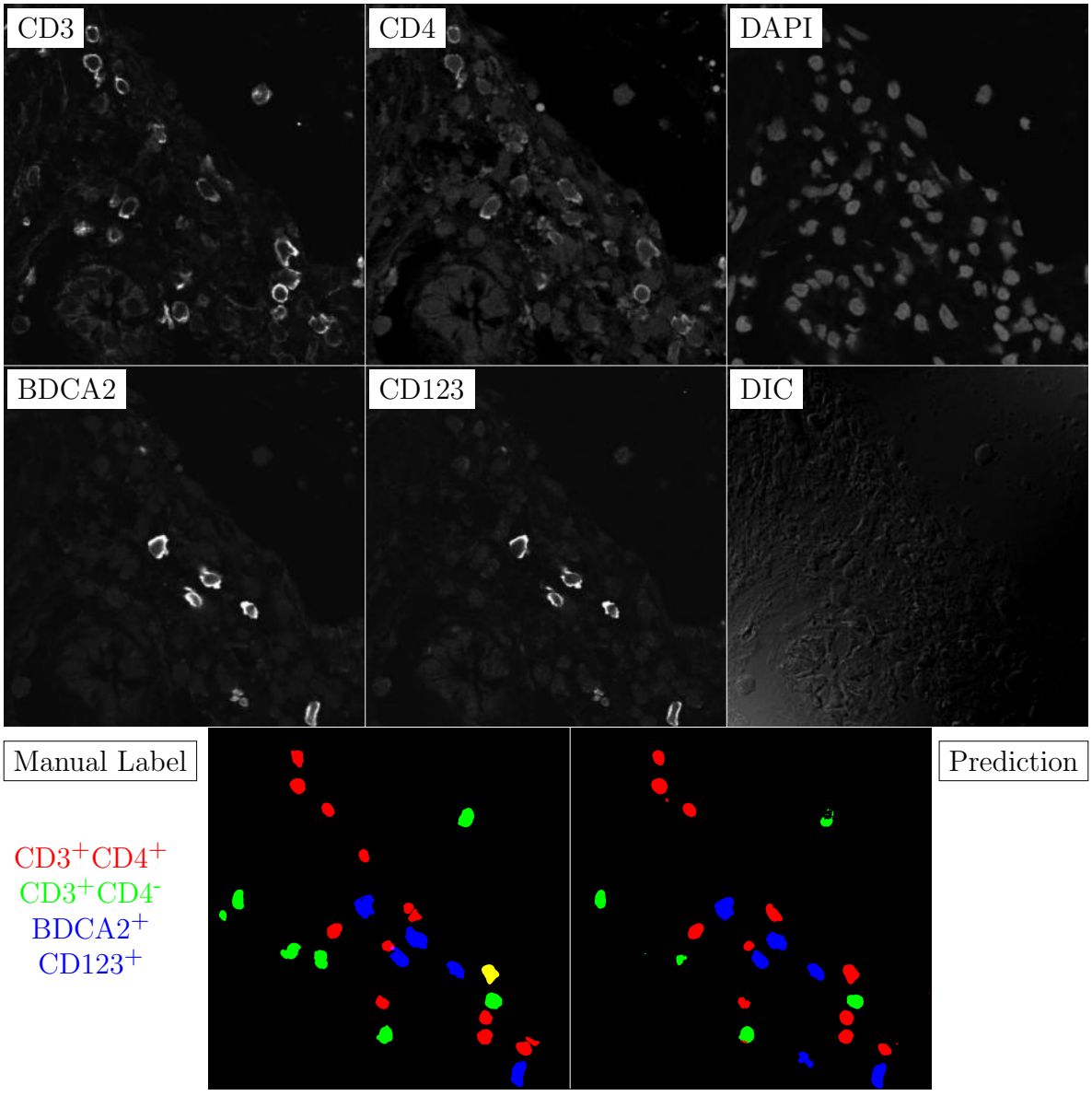

Figure G.197: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

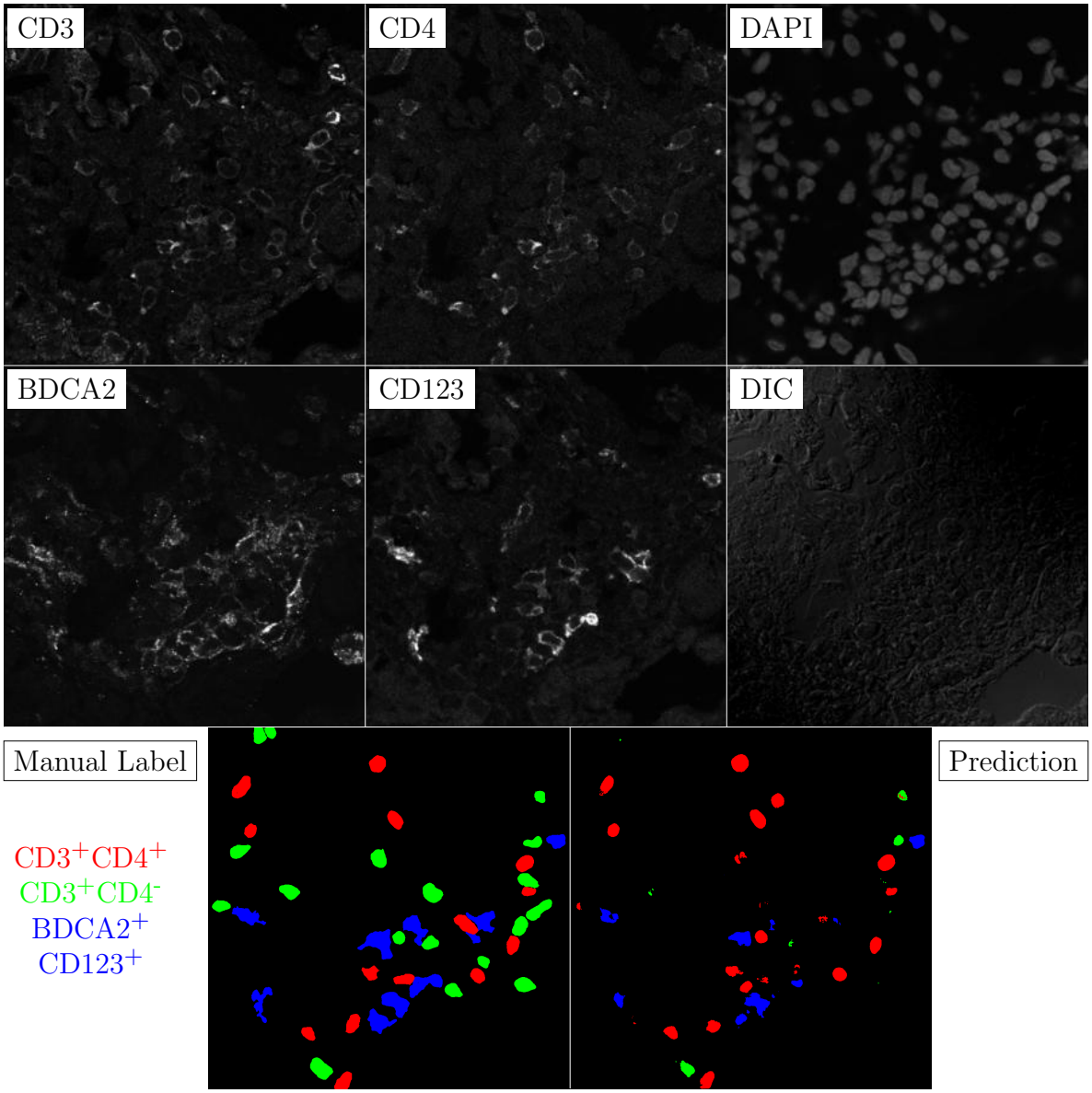

Figure G.198: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

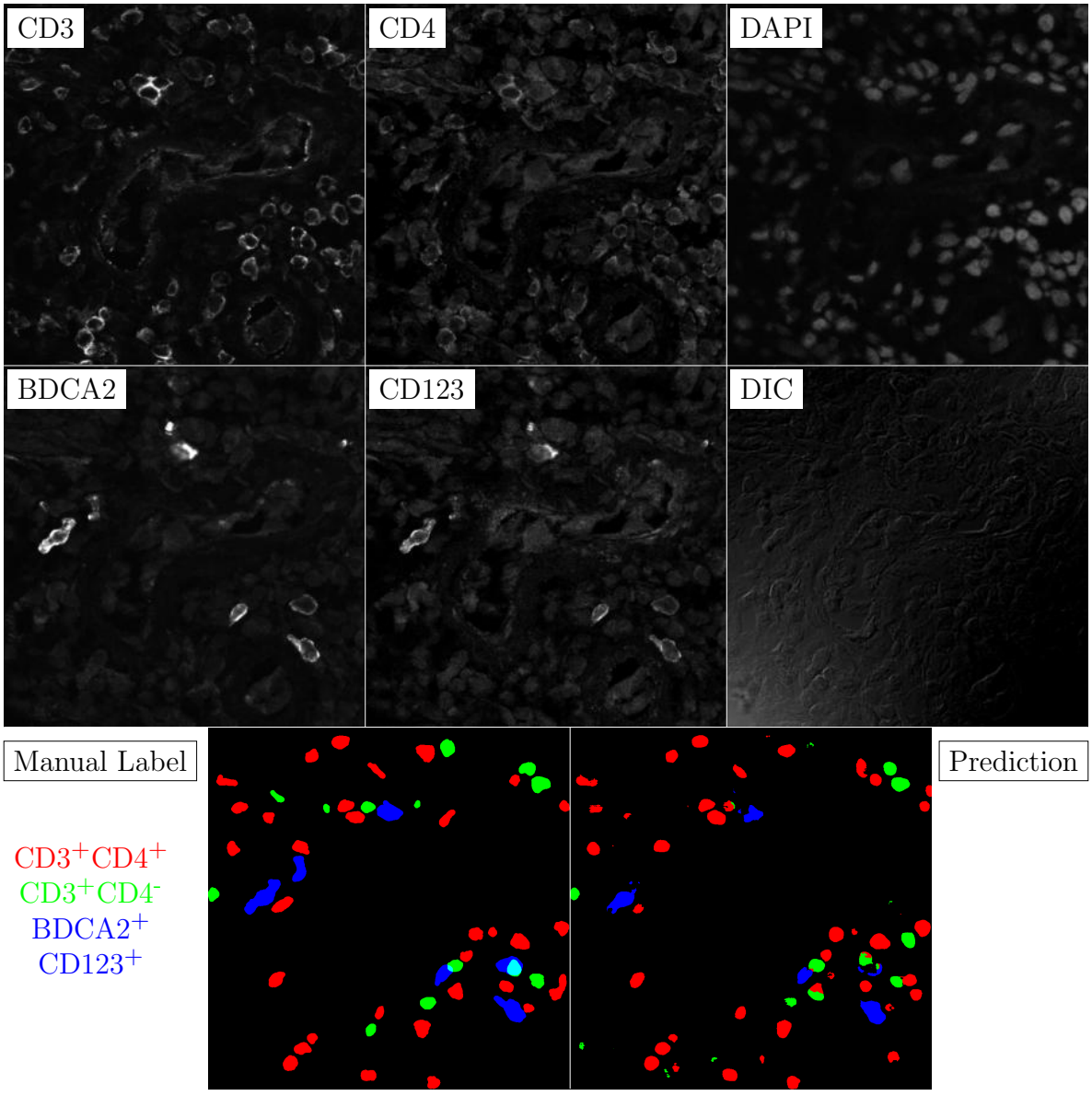

Figure G.199: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

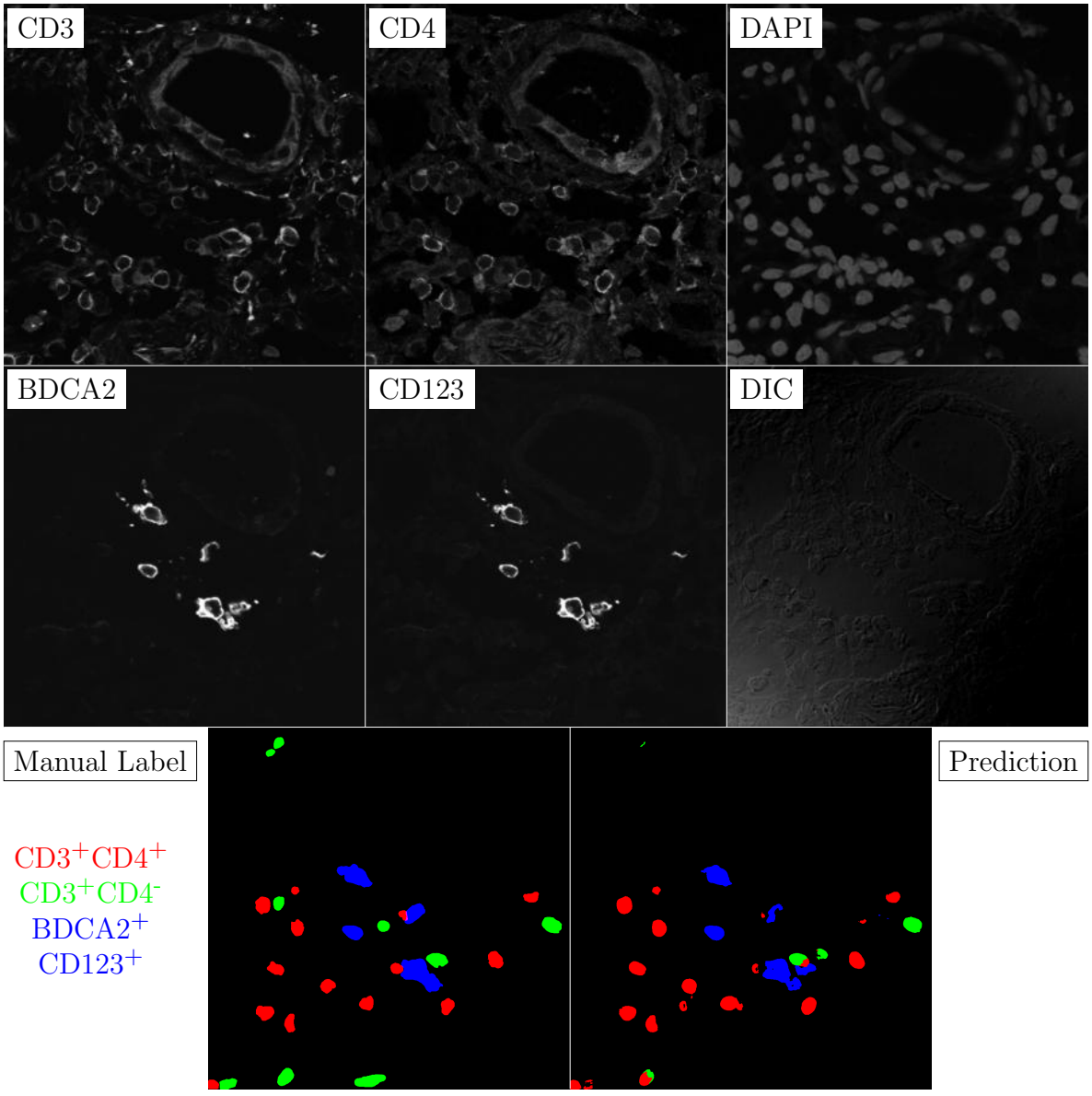

Figure G.200: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

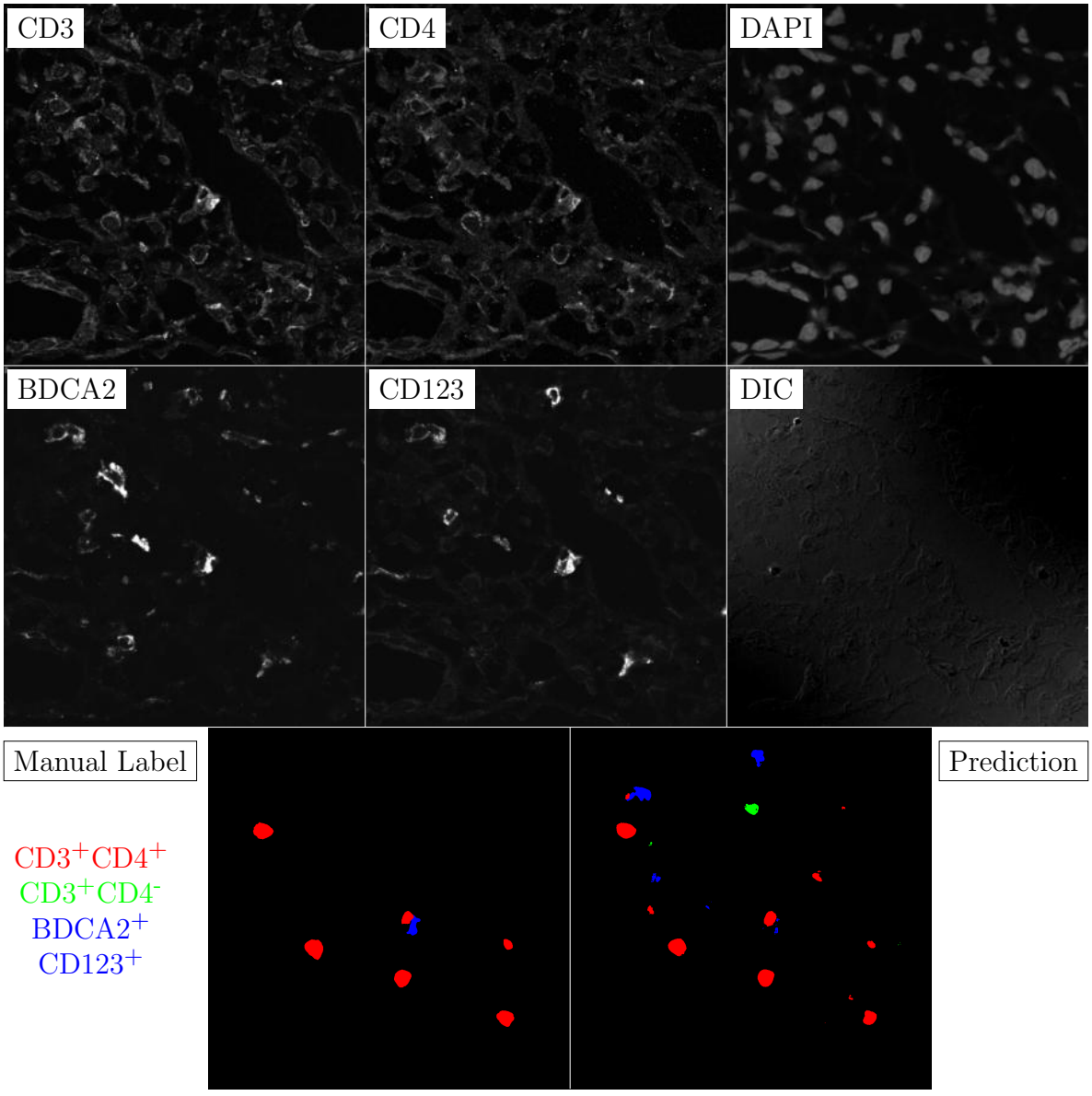

Figure G.201: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

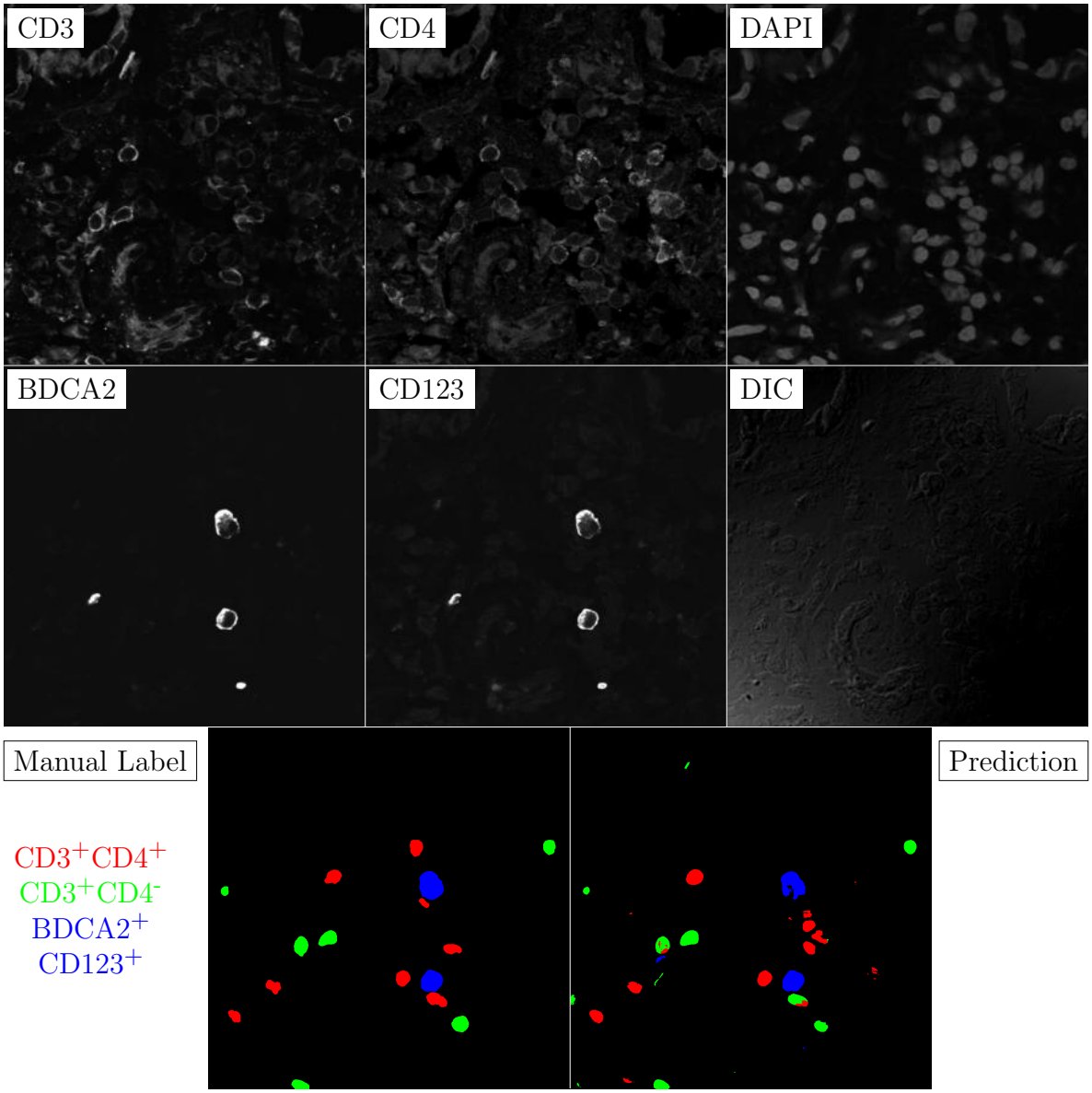

Figure G.202: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

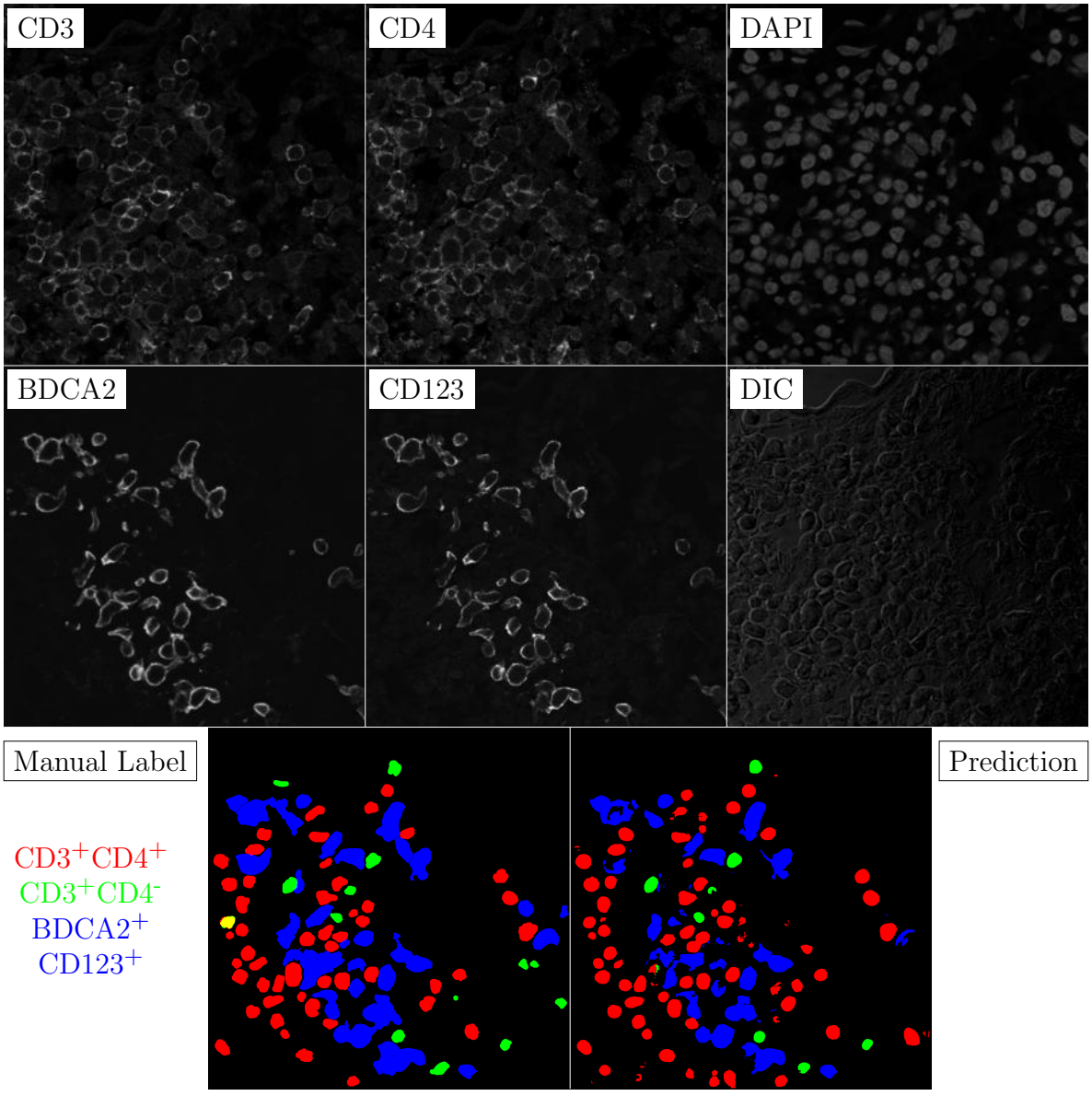

Figure G.203: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

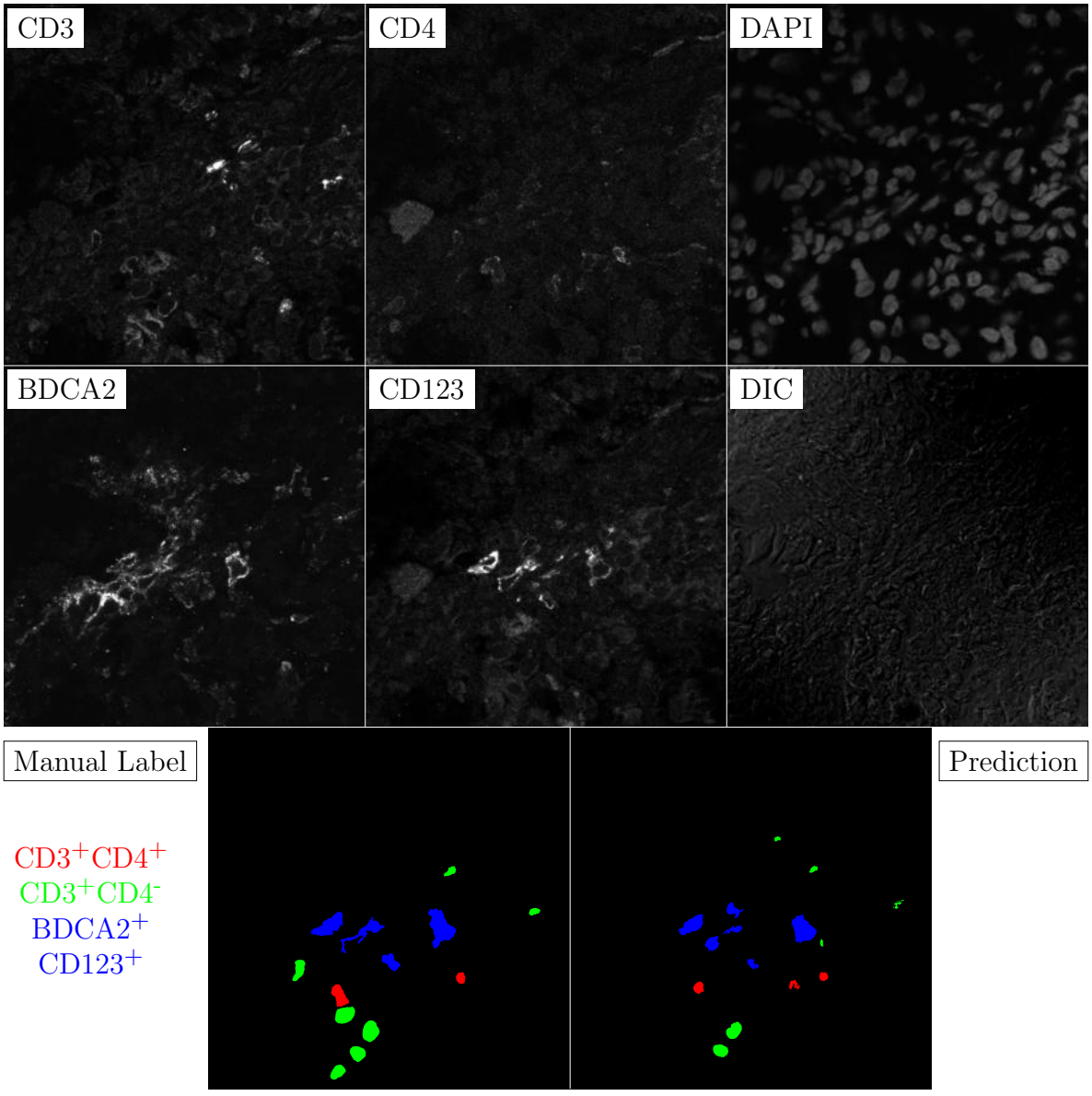

Figure G.204: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

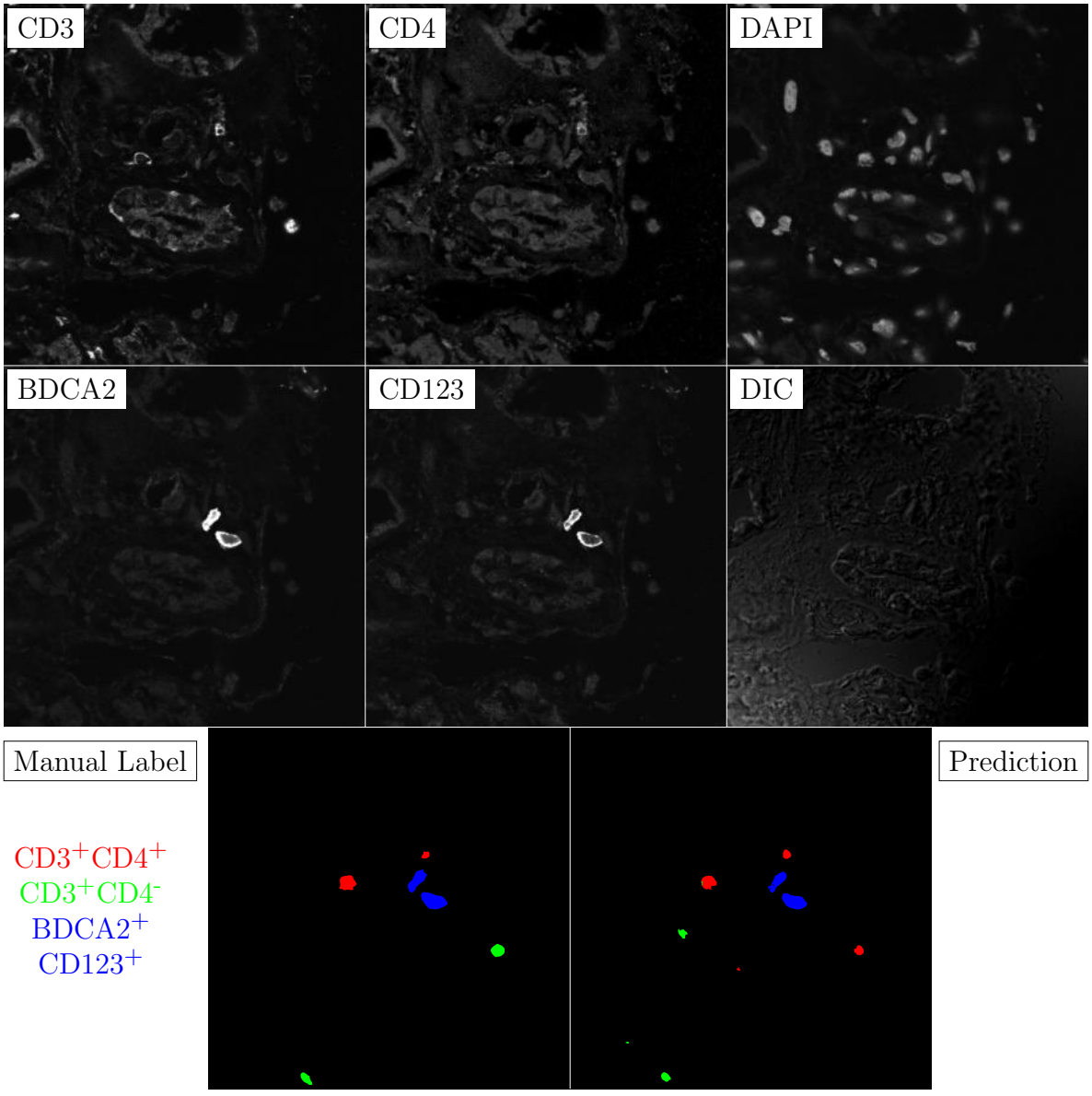

Figure G.205: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

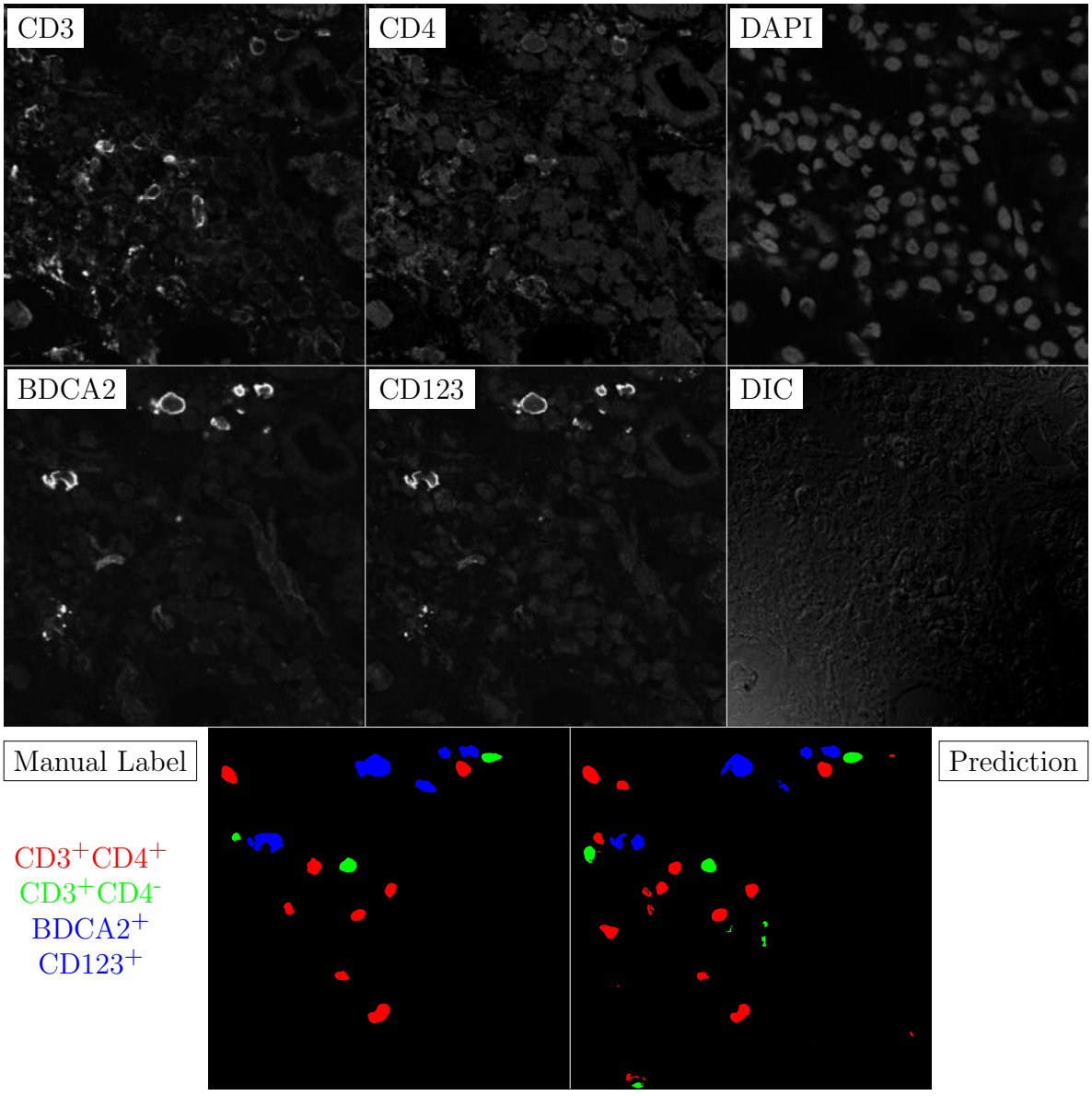

Figure G.206: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

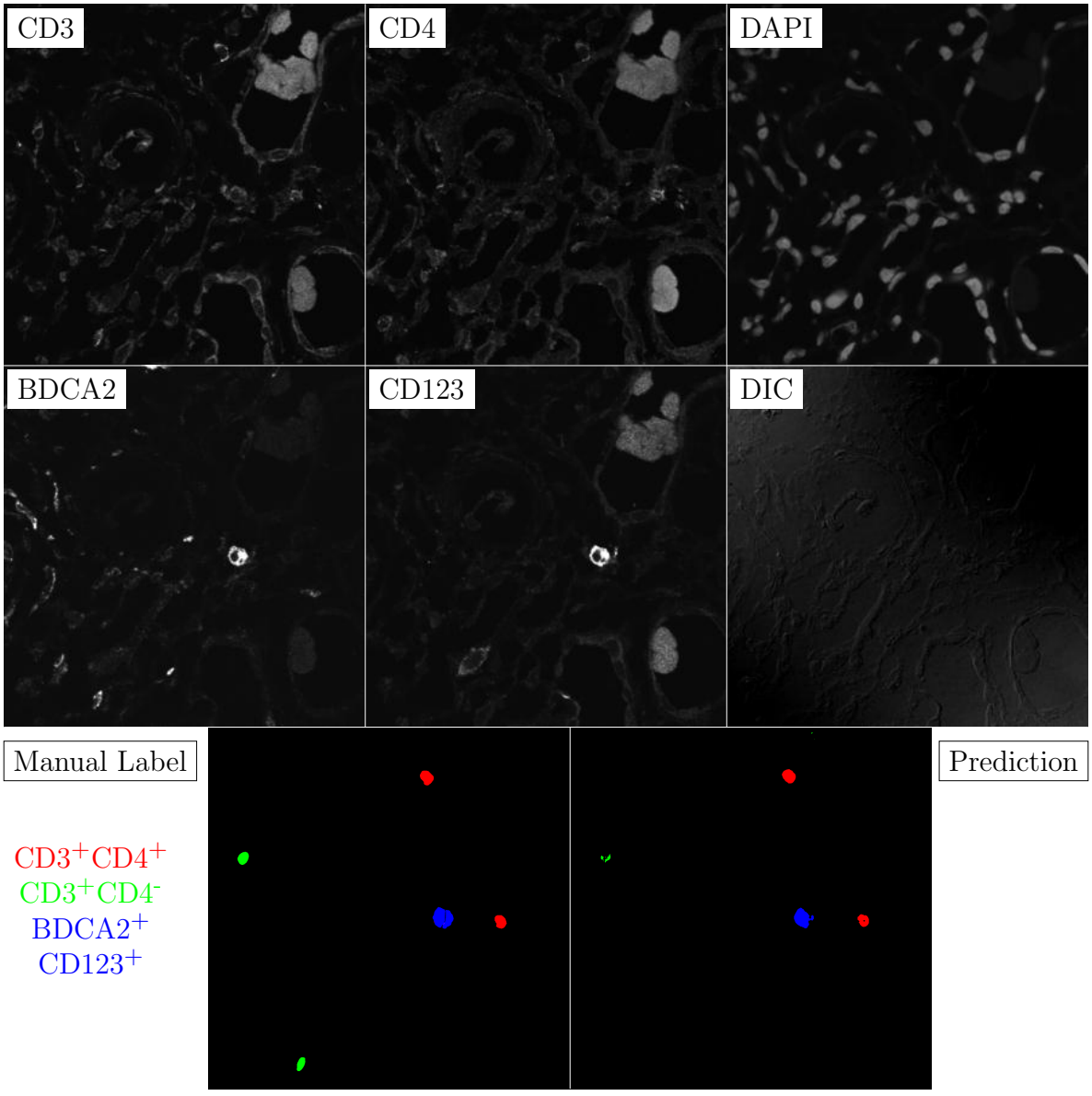

Figure G.207: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

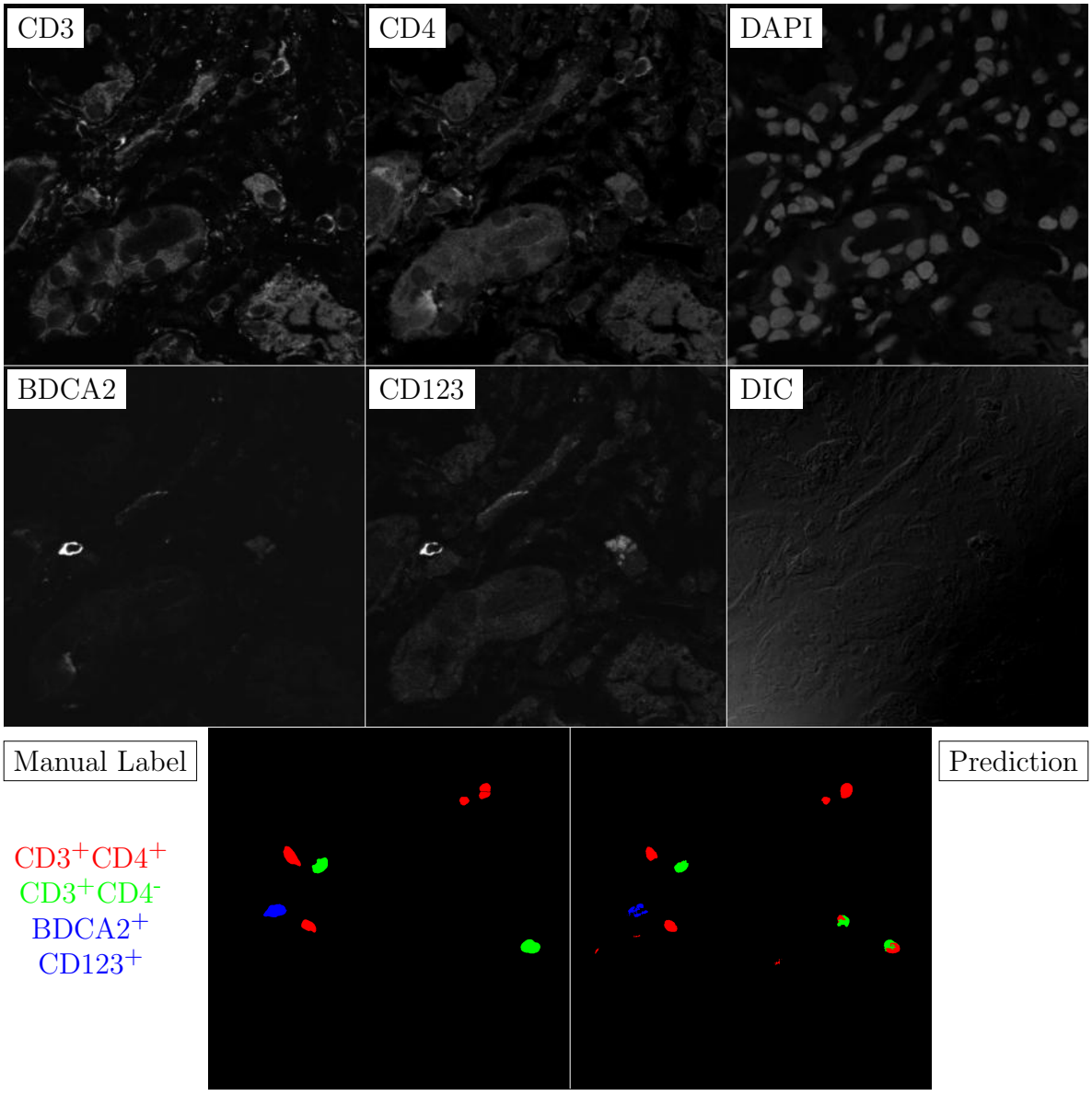

Figure G.208: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

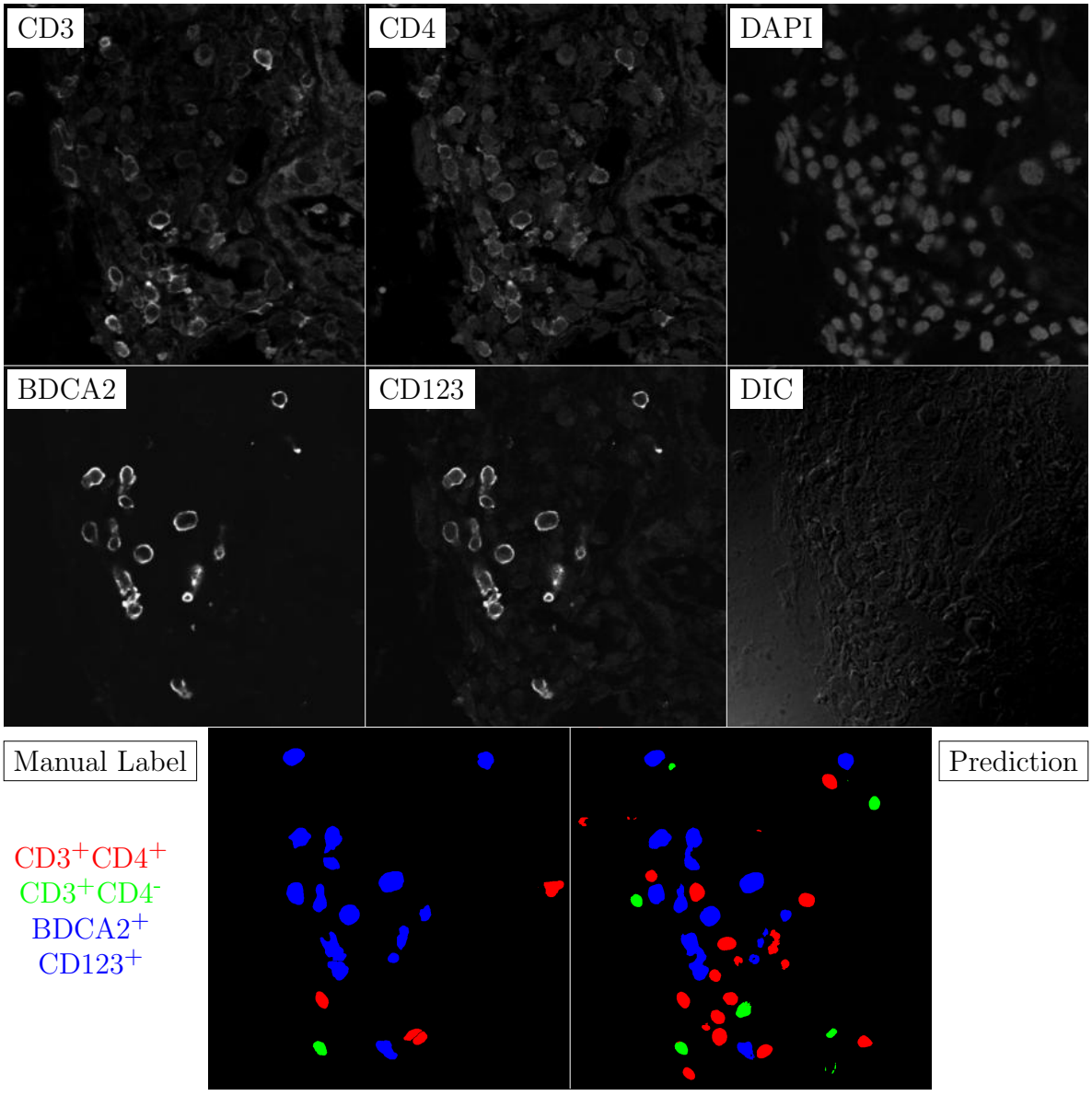

Figure G.209: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

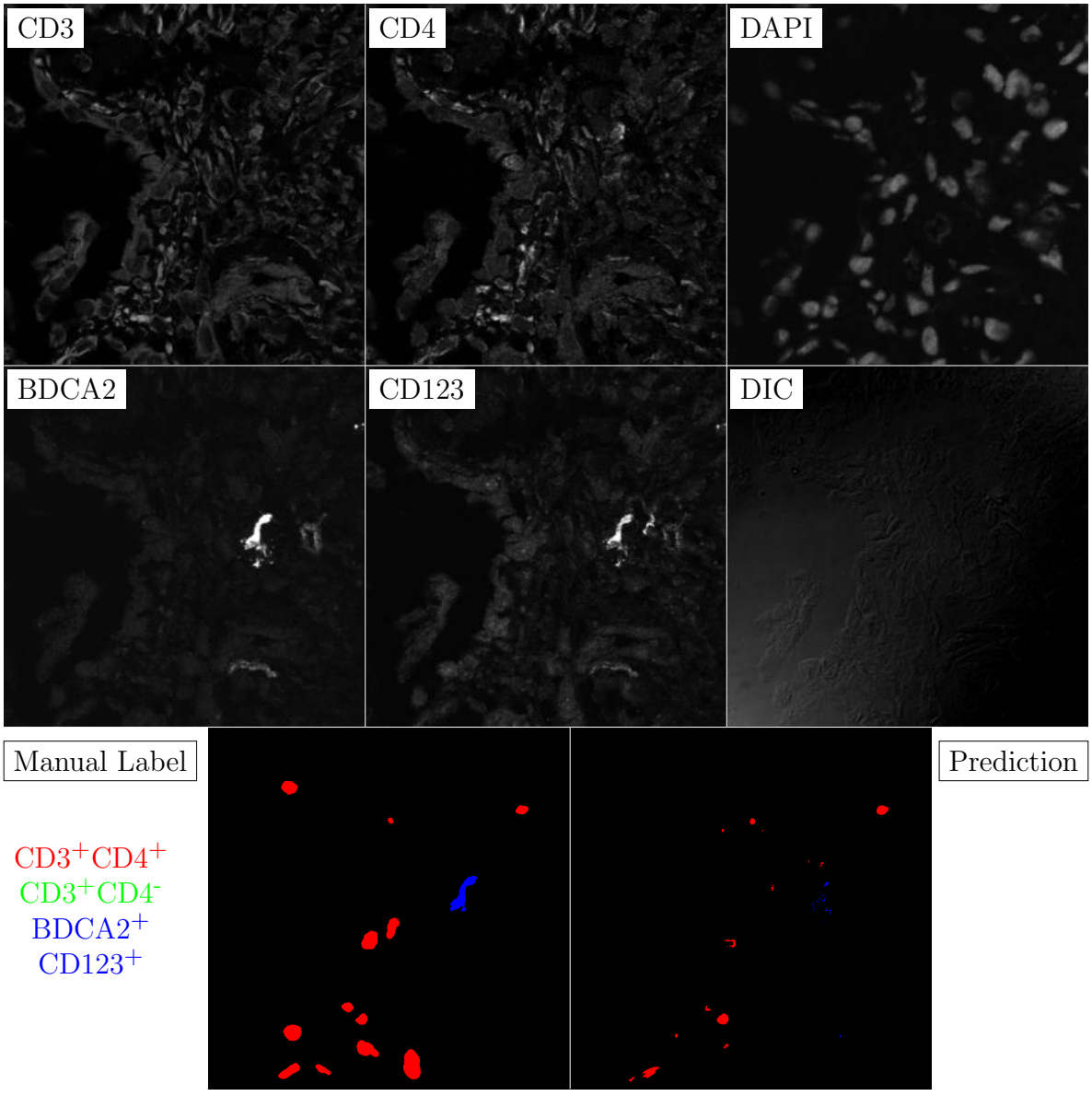

Figure G.210: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

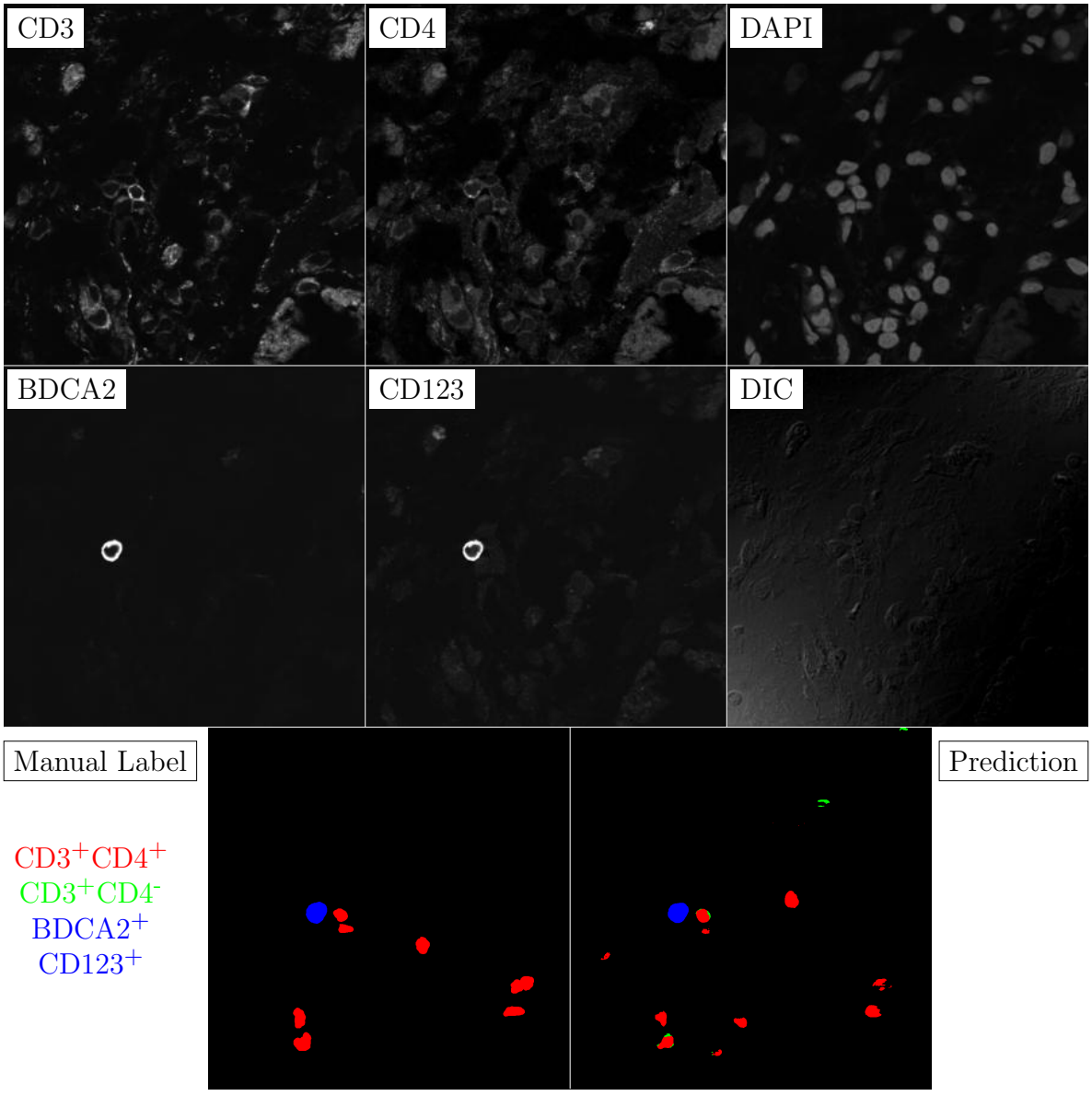

Figure G.211: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

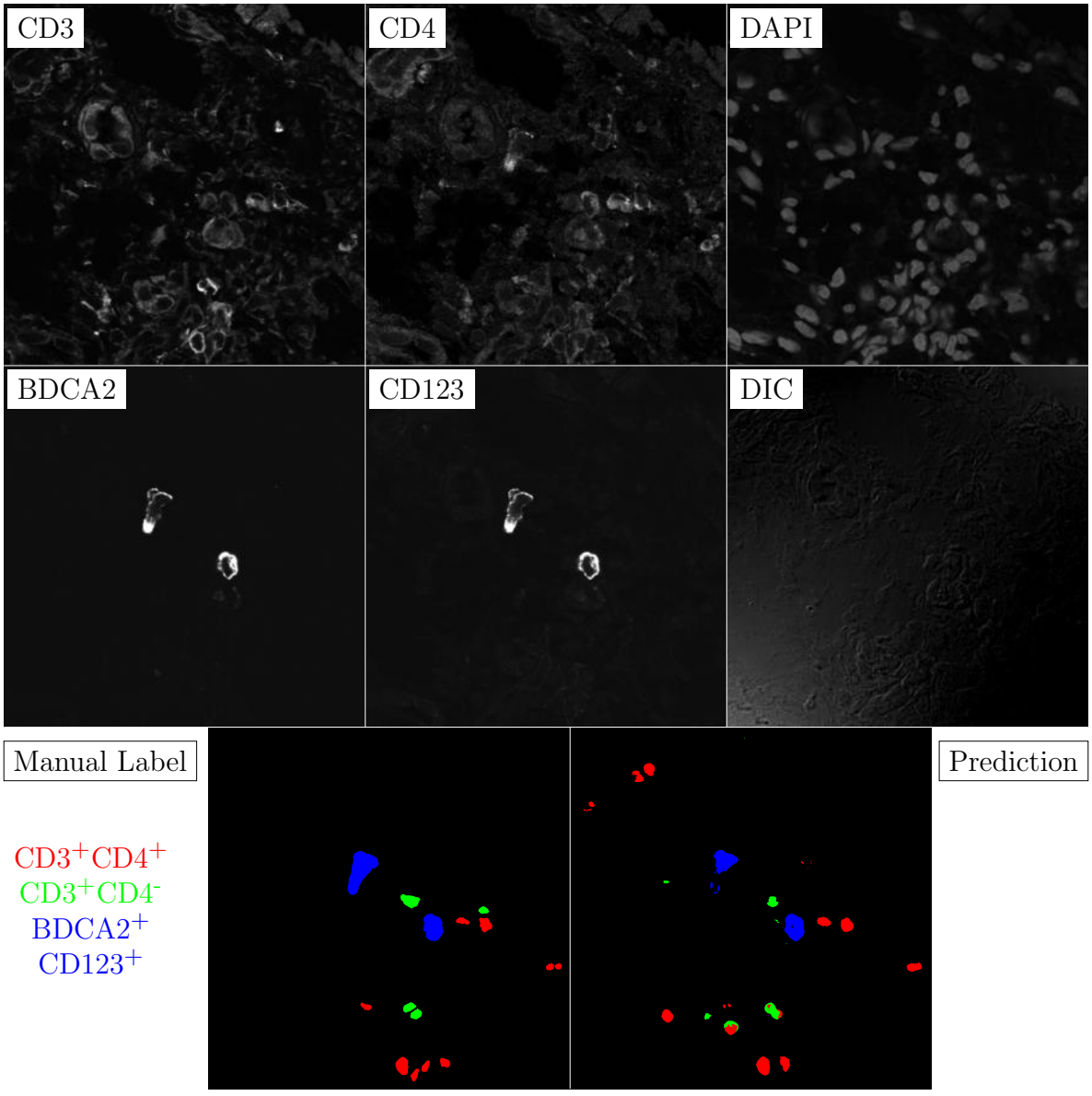

Figure G.212: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

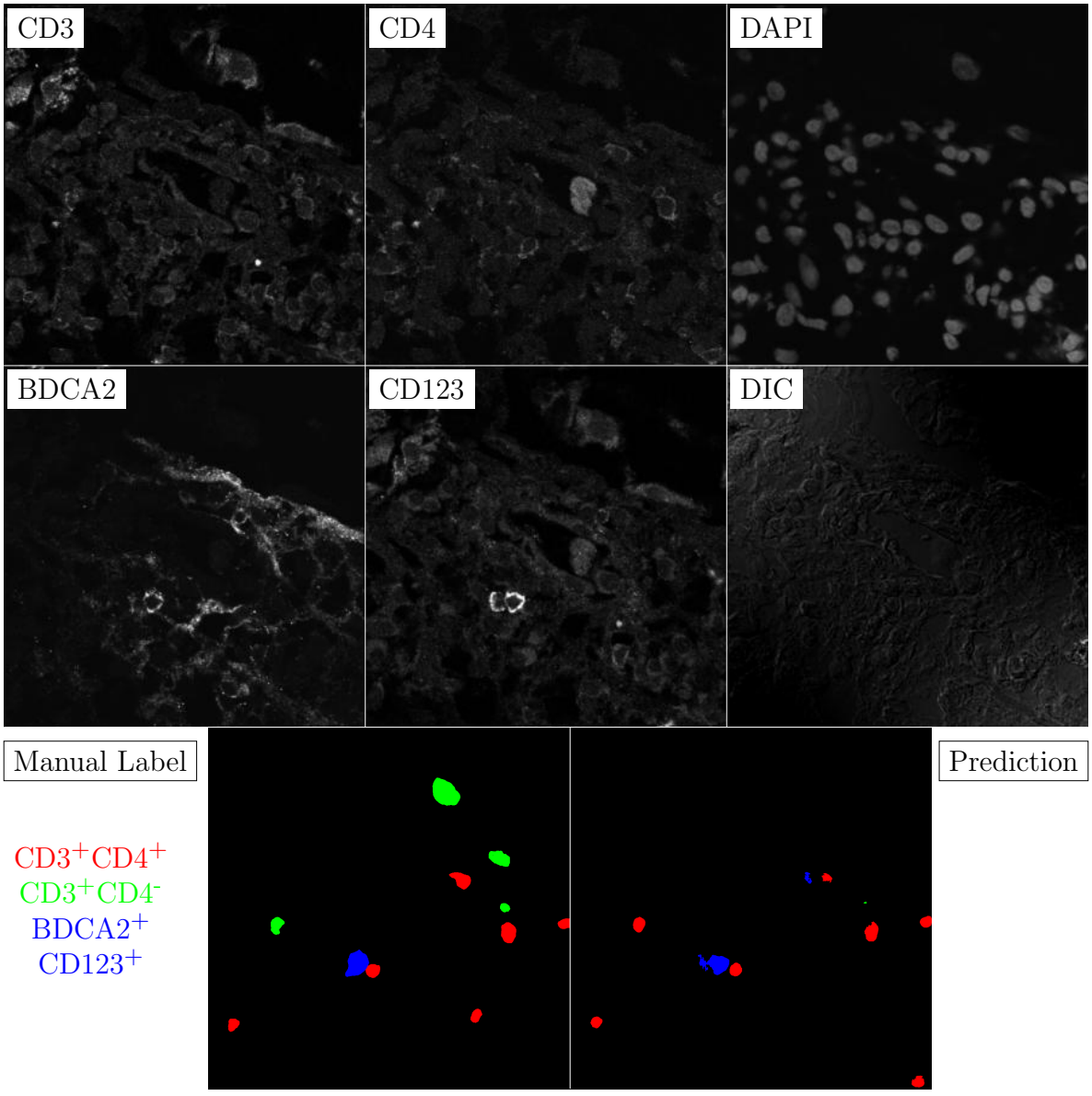

Figure G.213: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

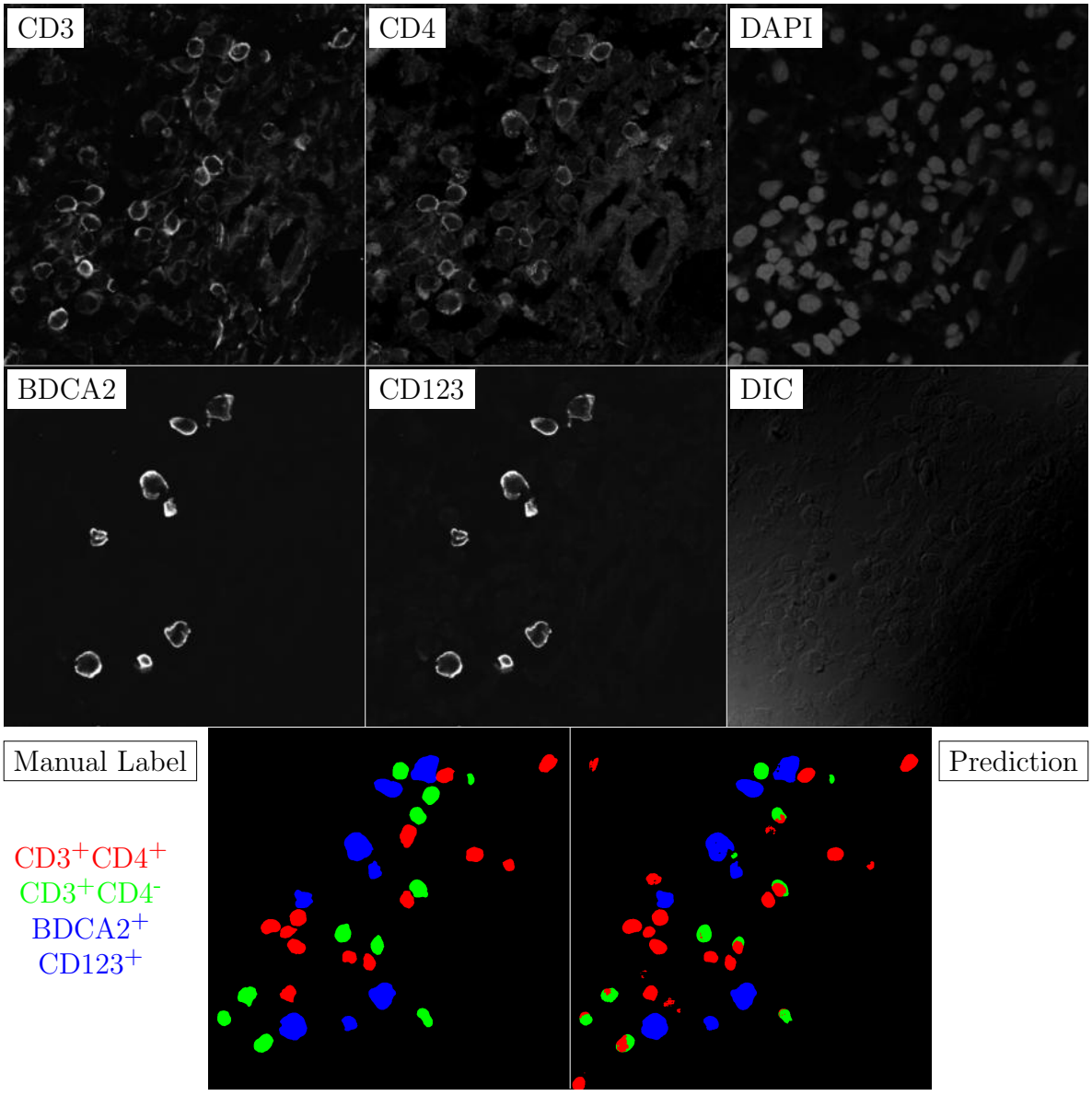

Figure G.214: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

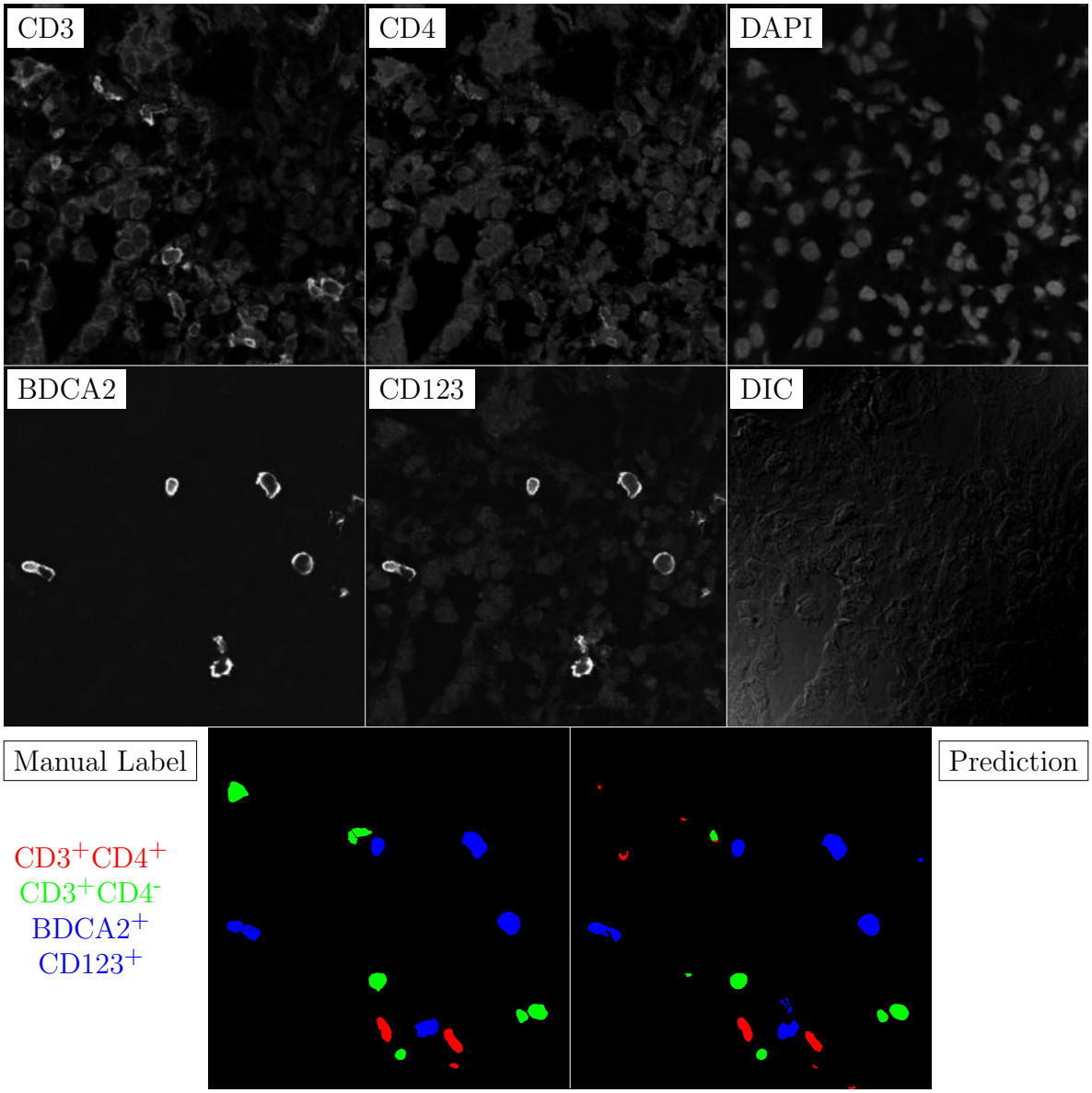

Figure G.215: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

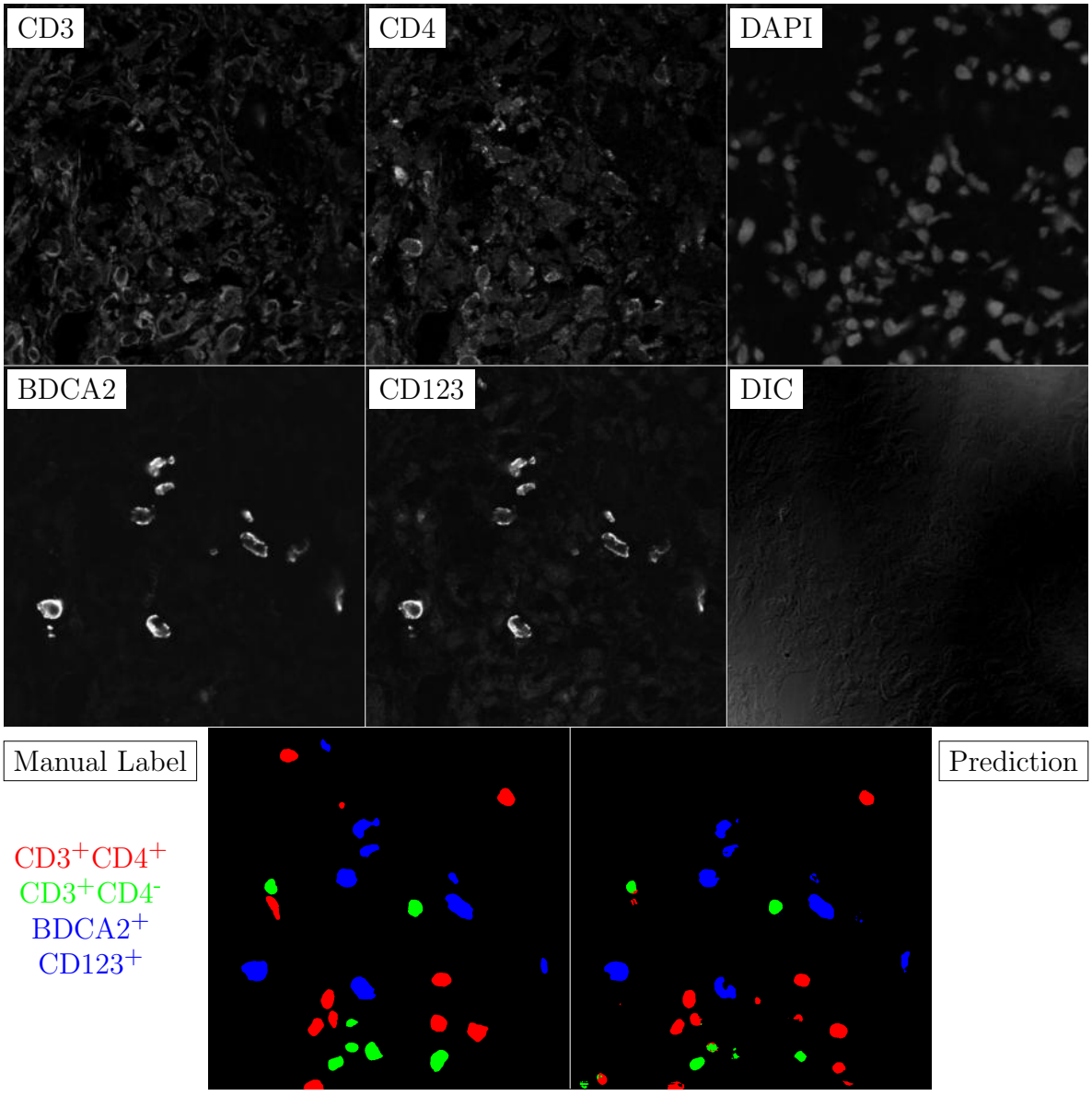

Figure G.216: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

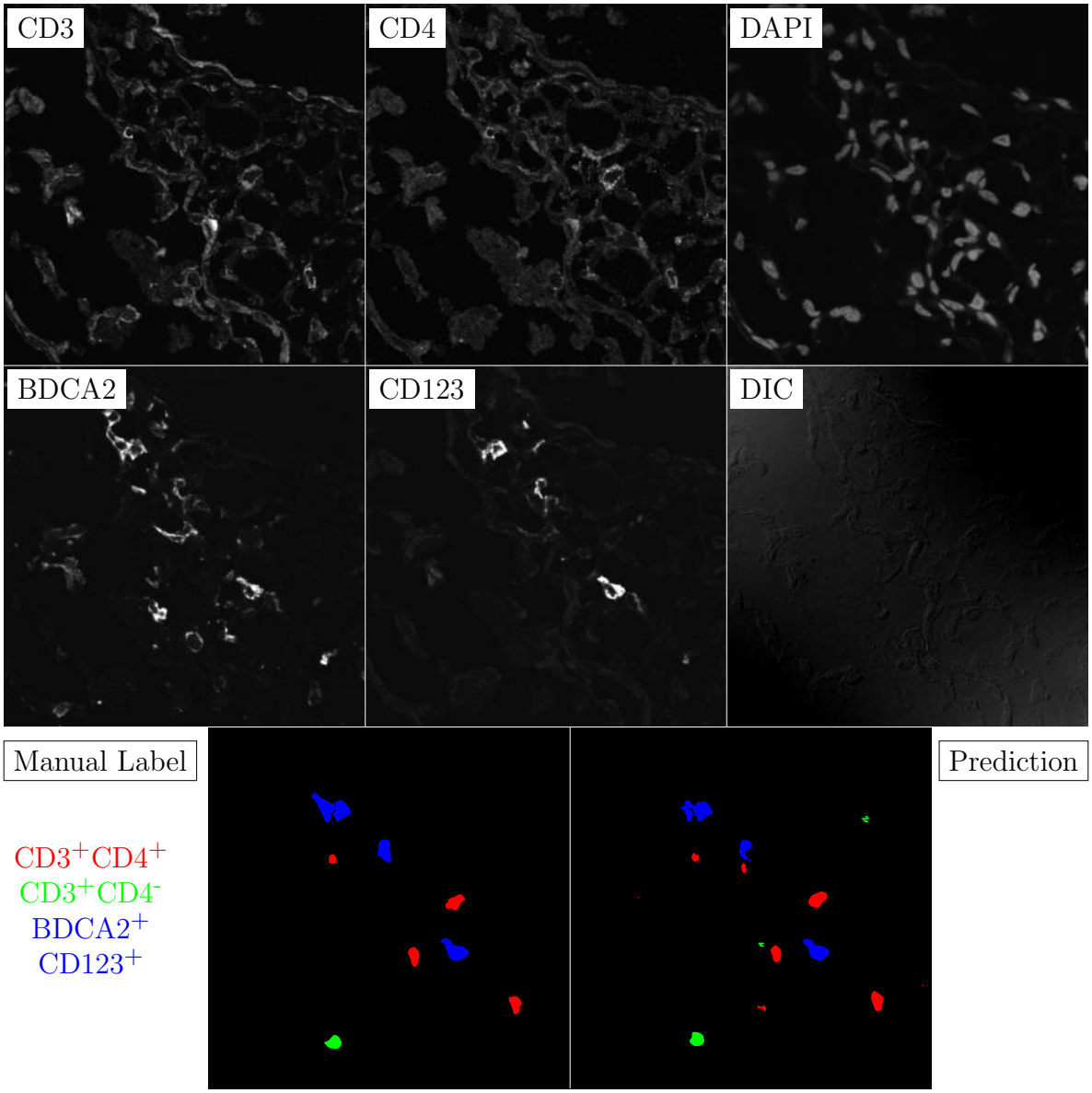

Figure G.217: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

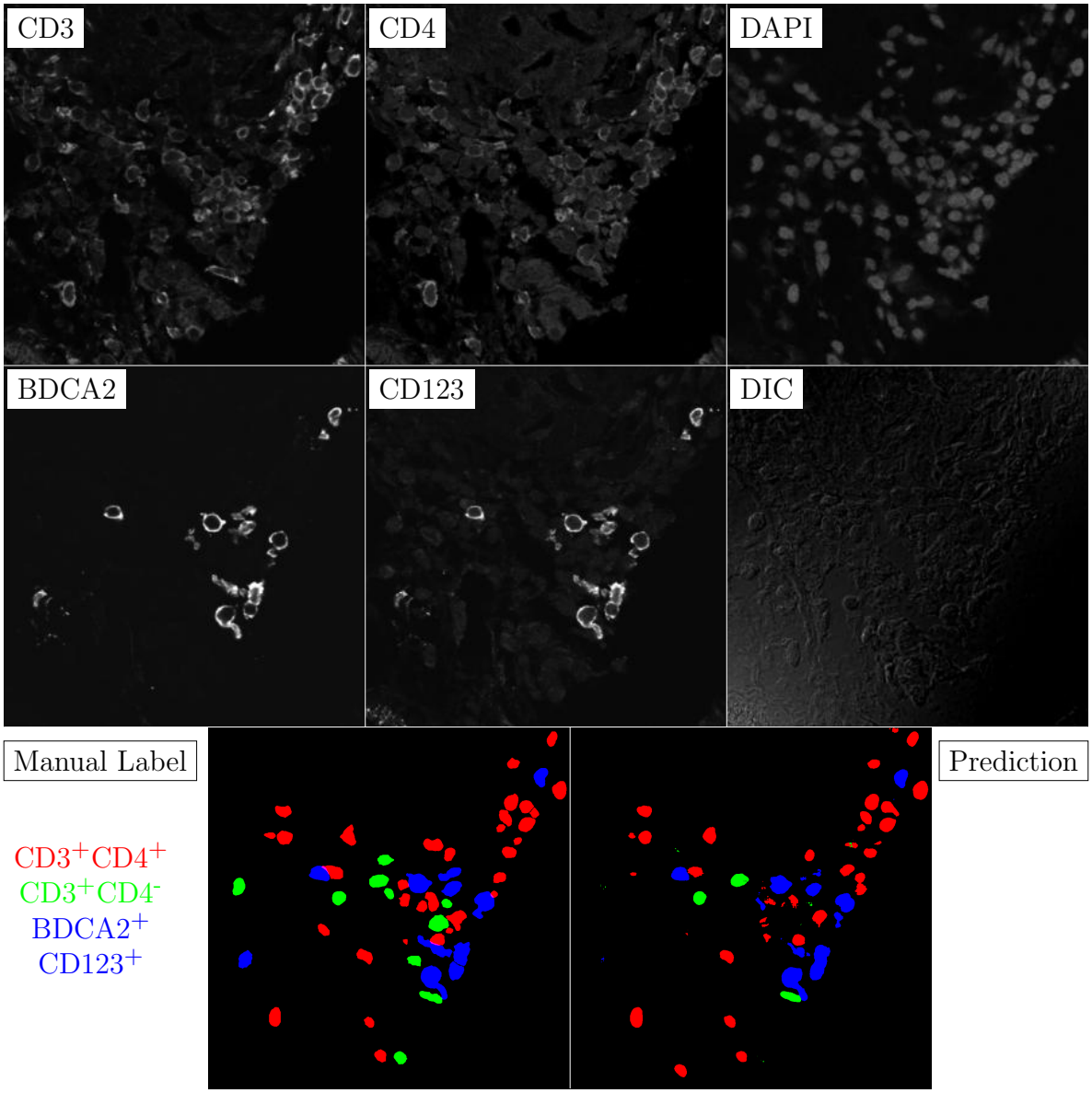

Figure G.218: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

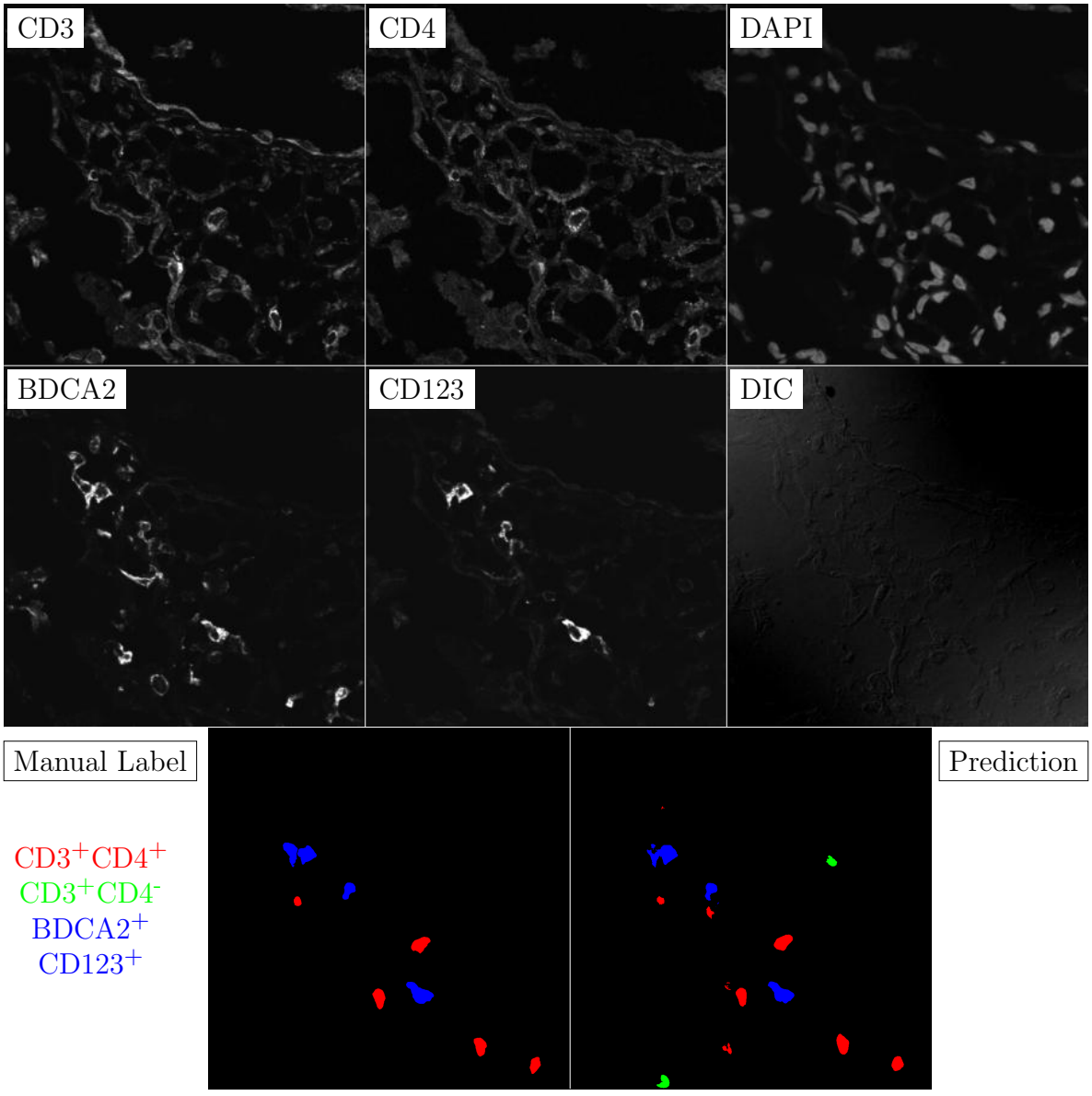

Figure G.219: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

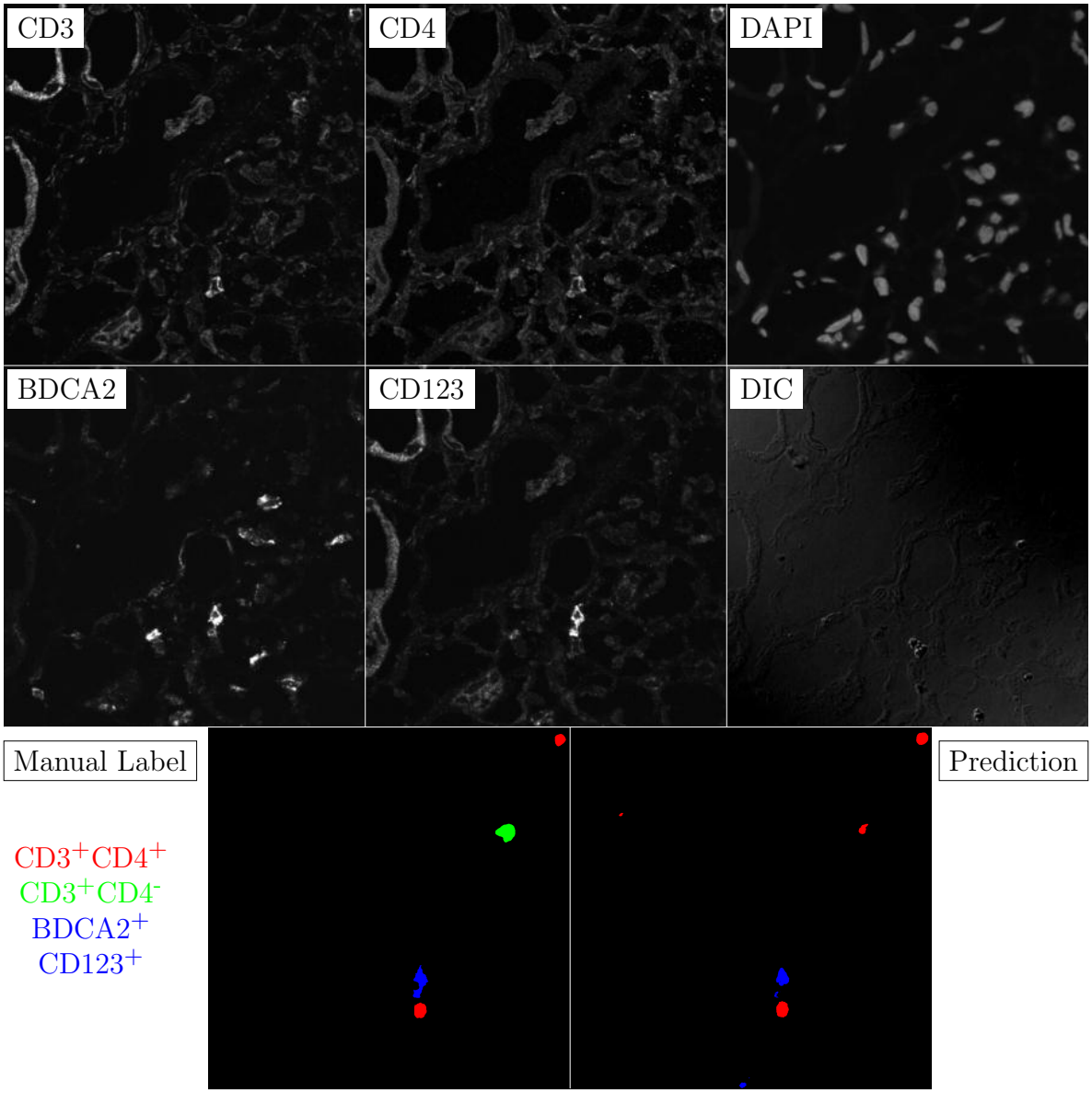

Figure G.220: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

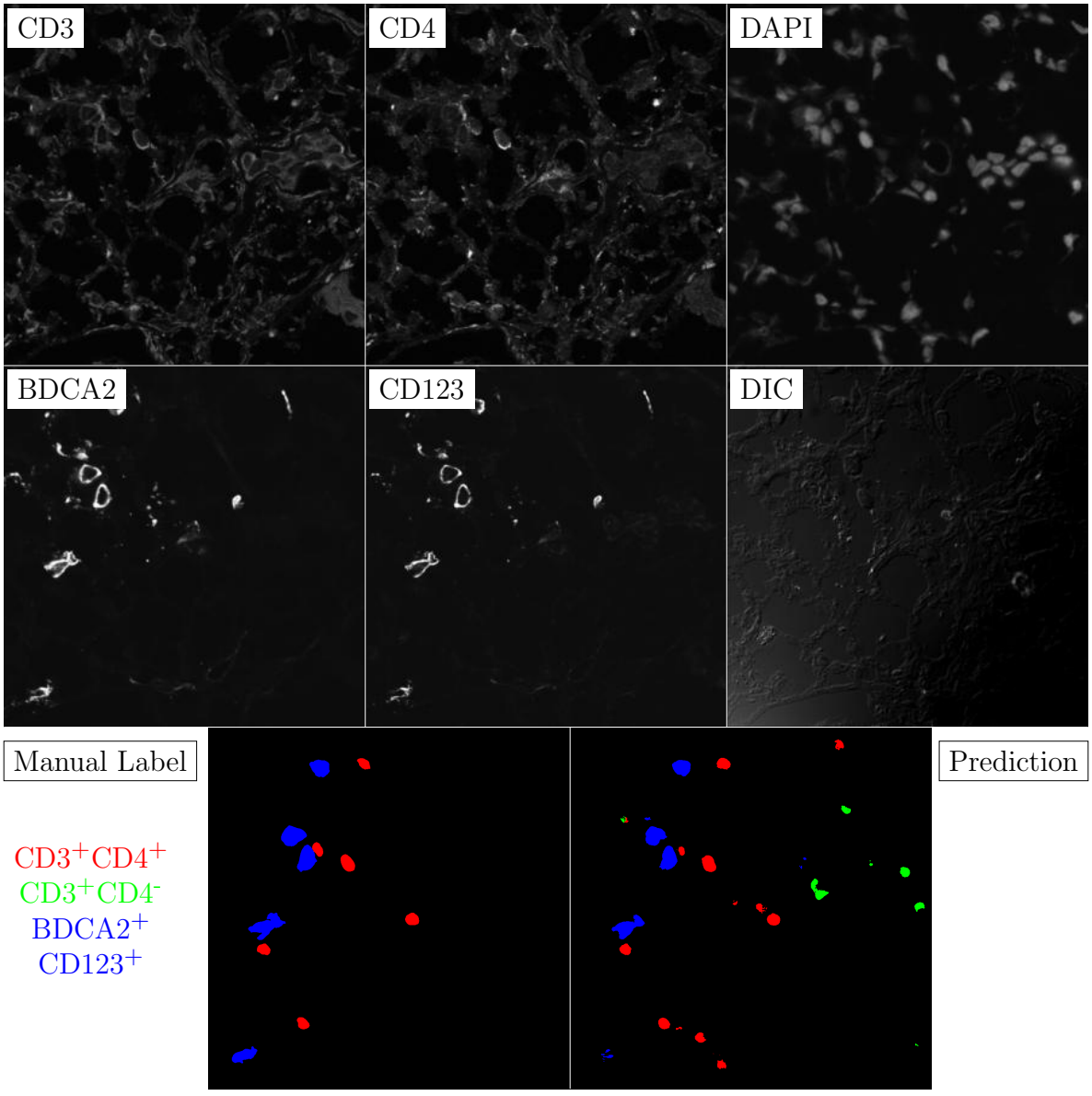

Figure G.221: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

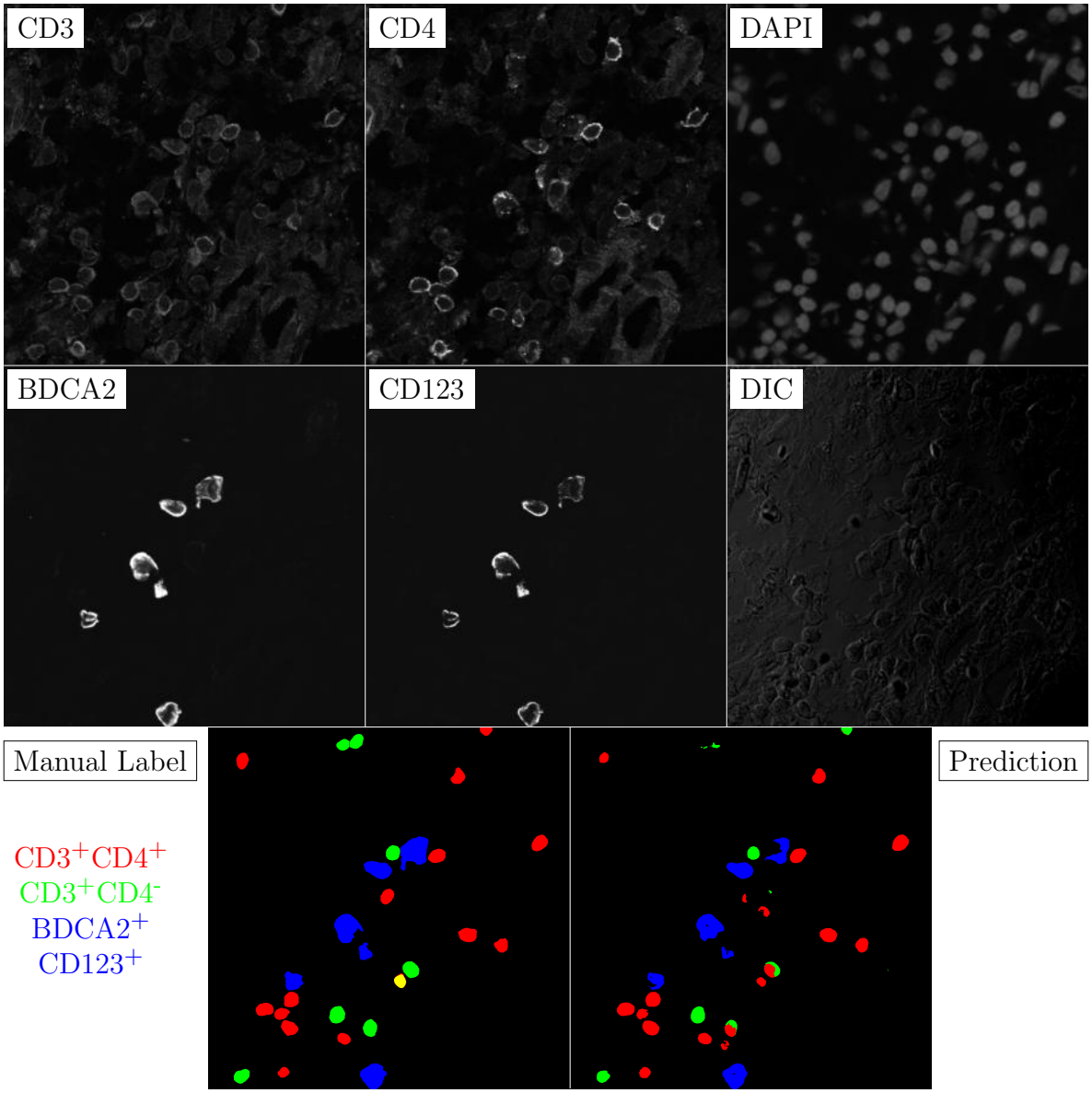

Figure G.222: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

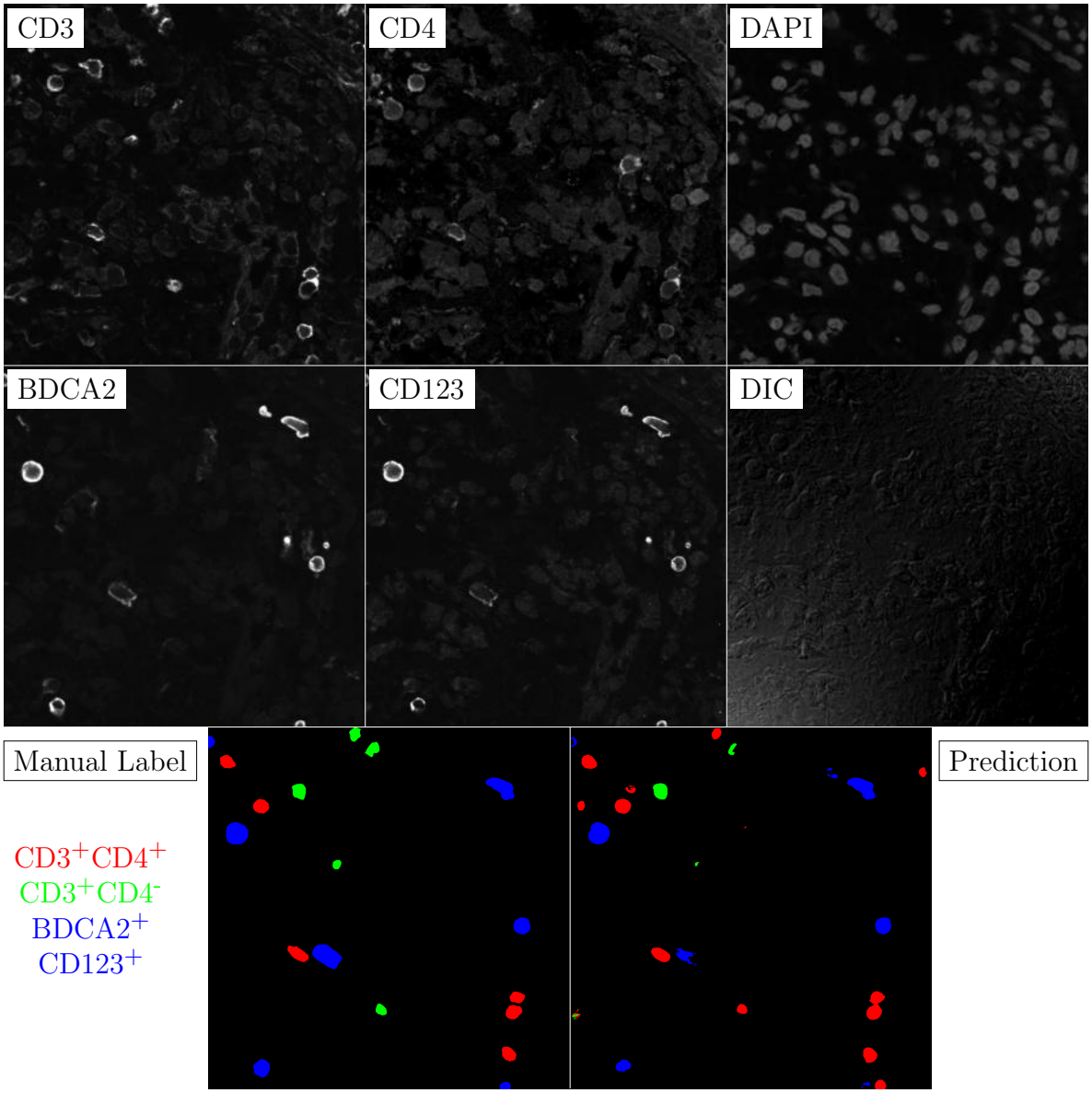

Figure G.223: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

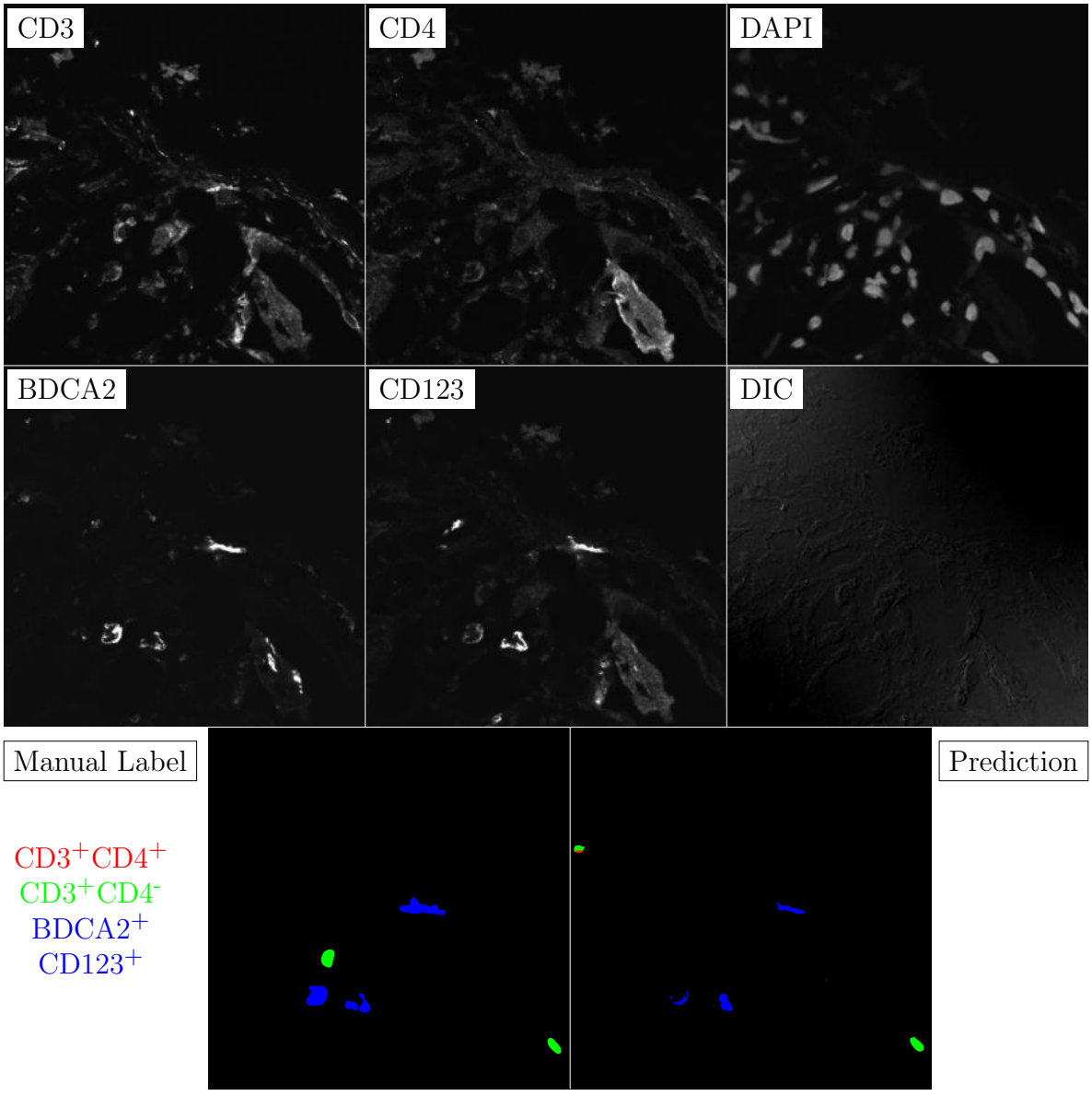

Figure G.224: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

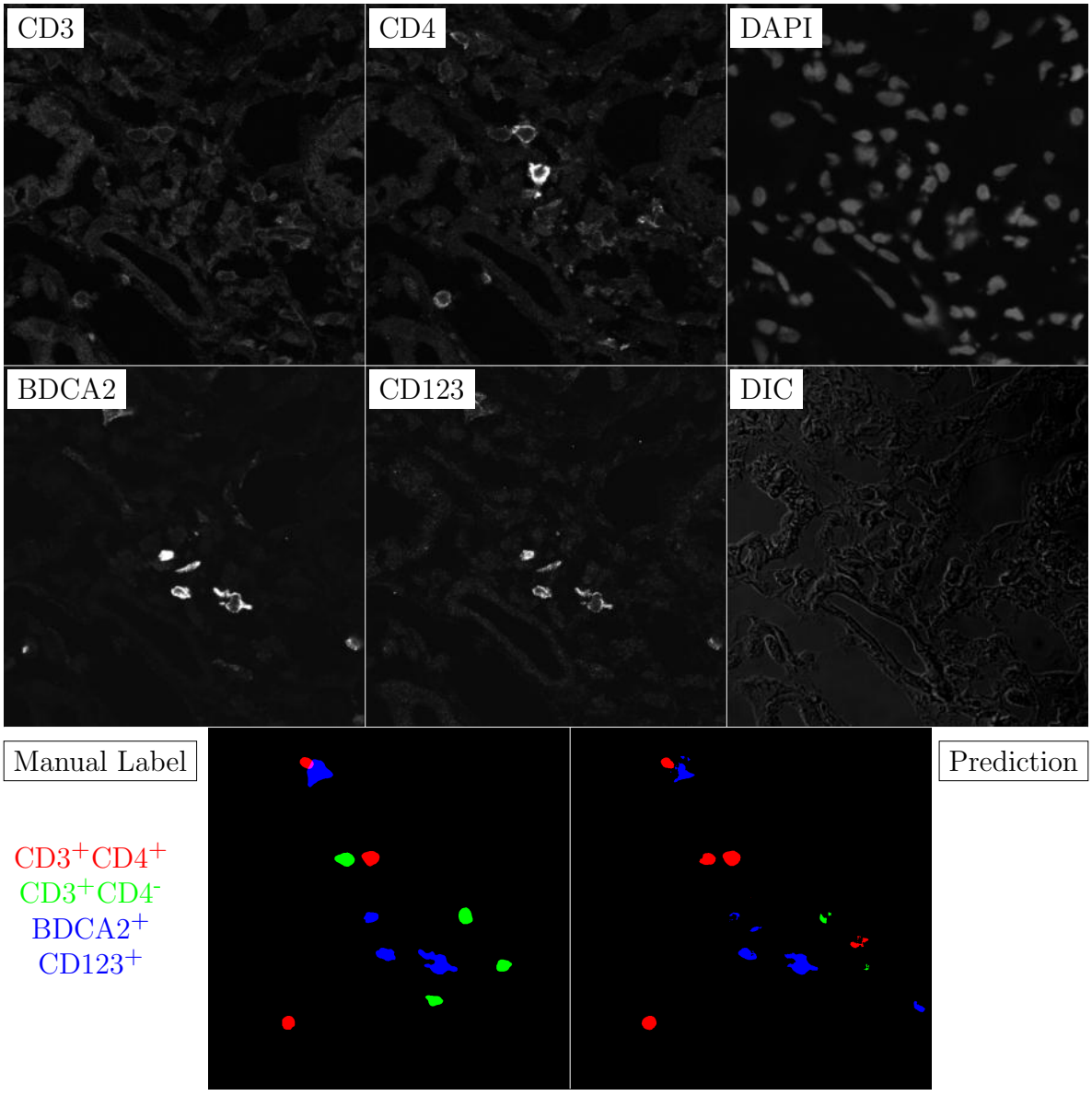

Figure G.225: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

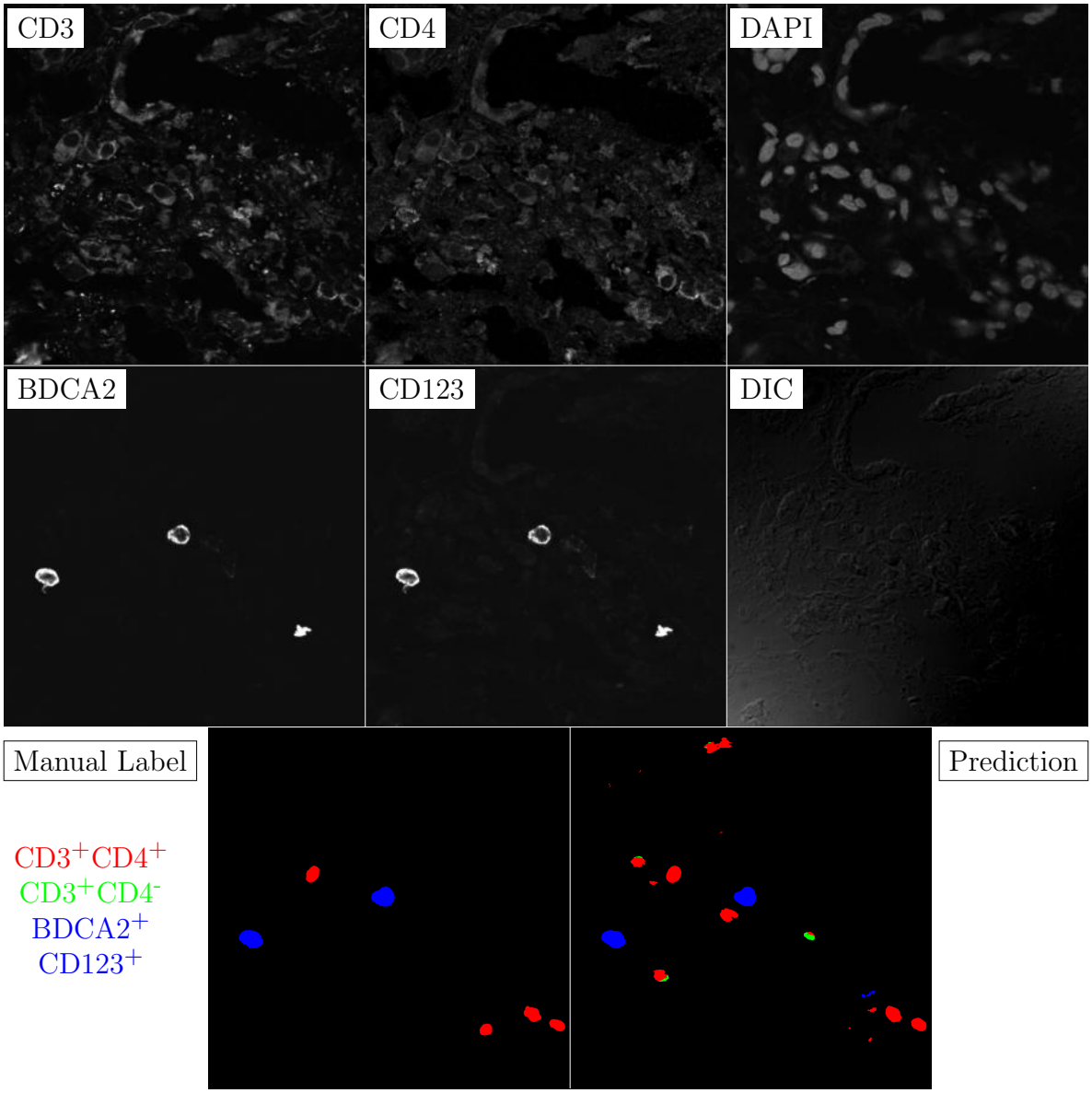

Figure G.226: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

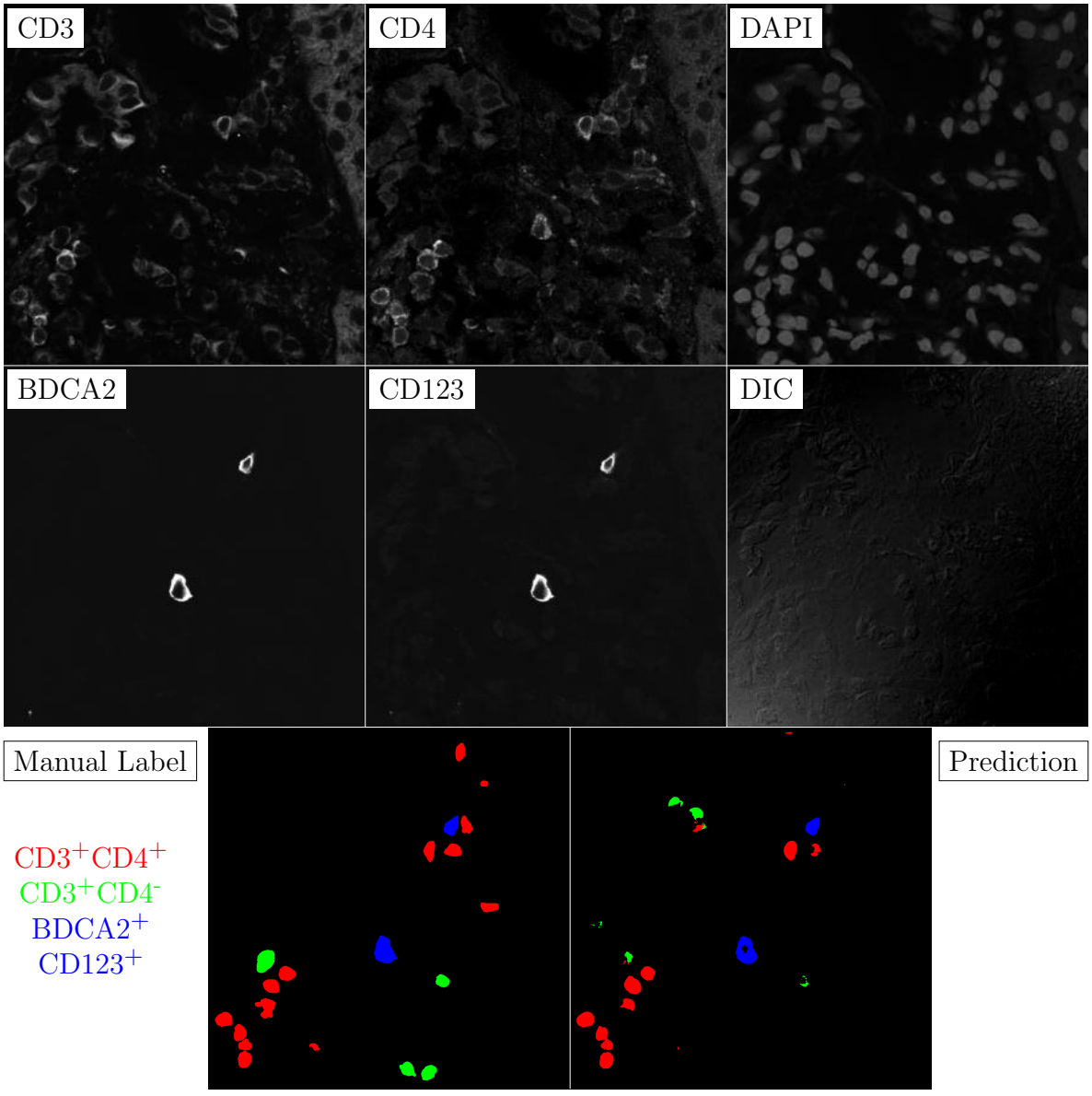

Figure G.227: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

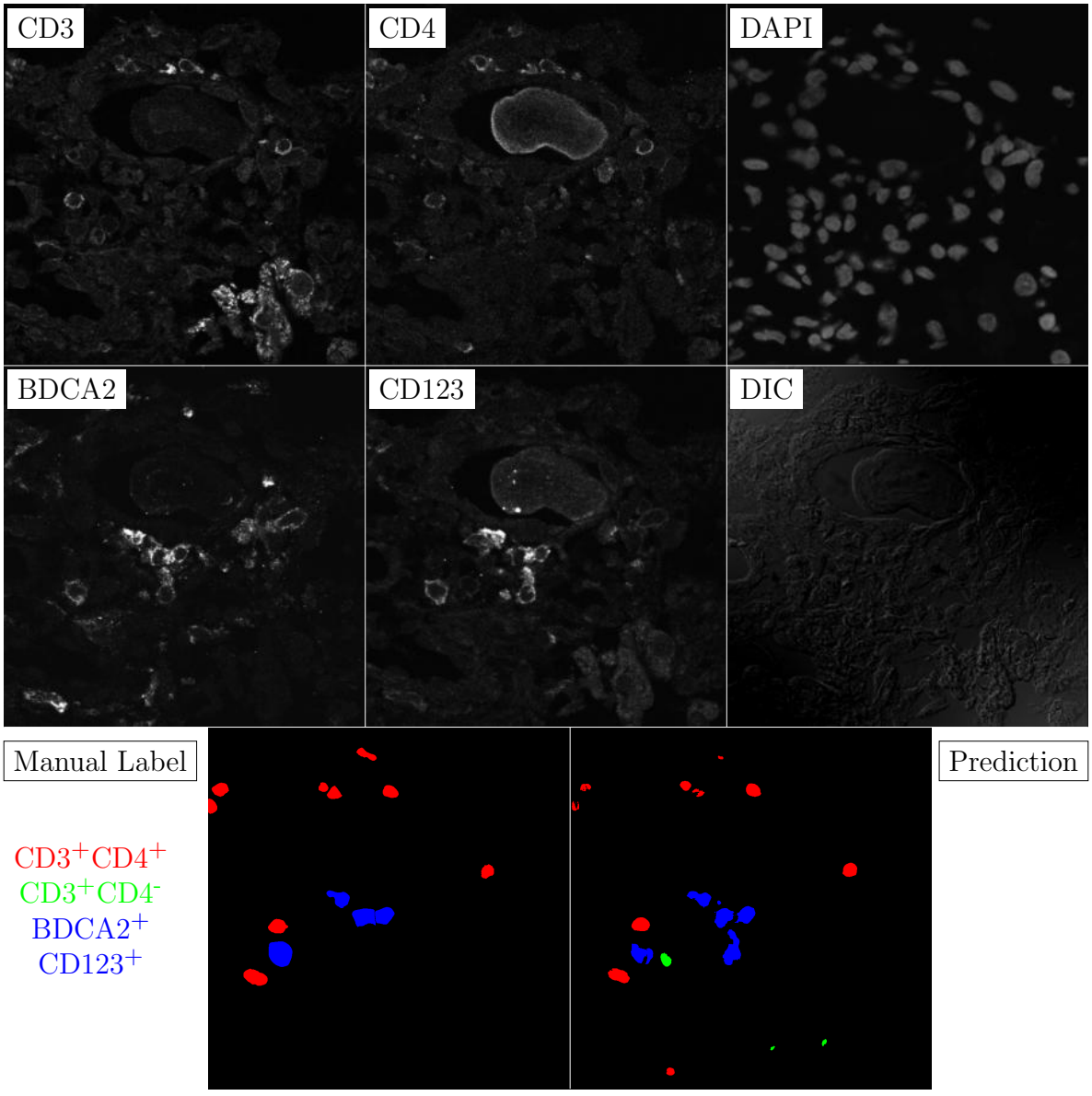

Figure G.228: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

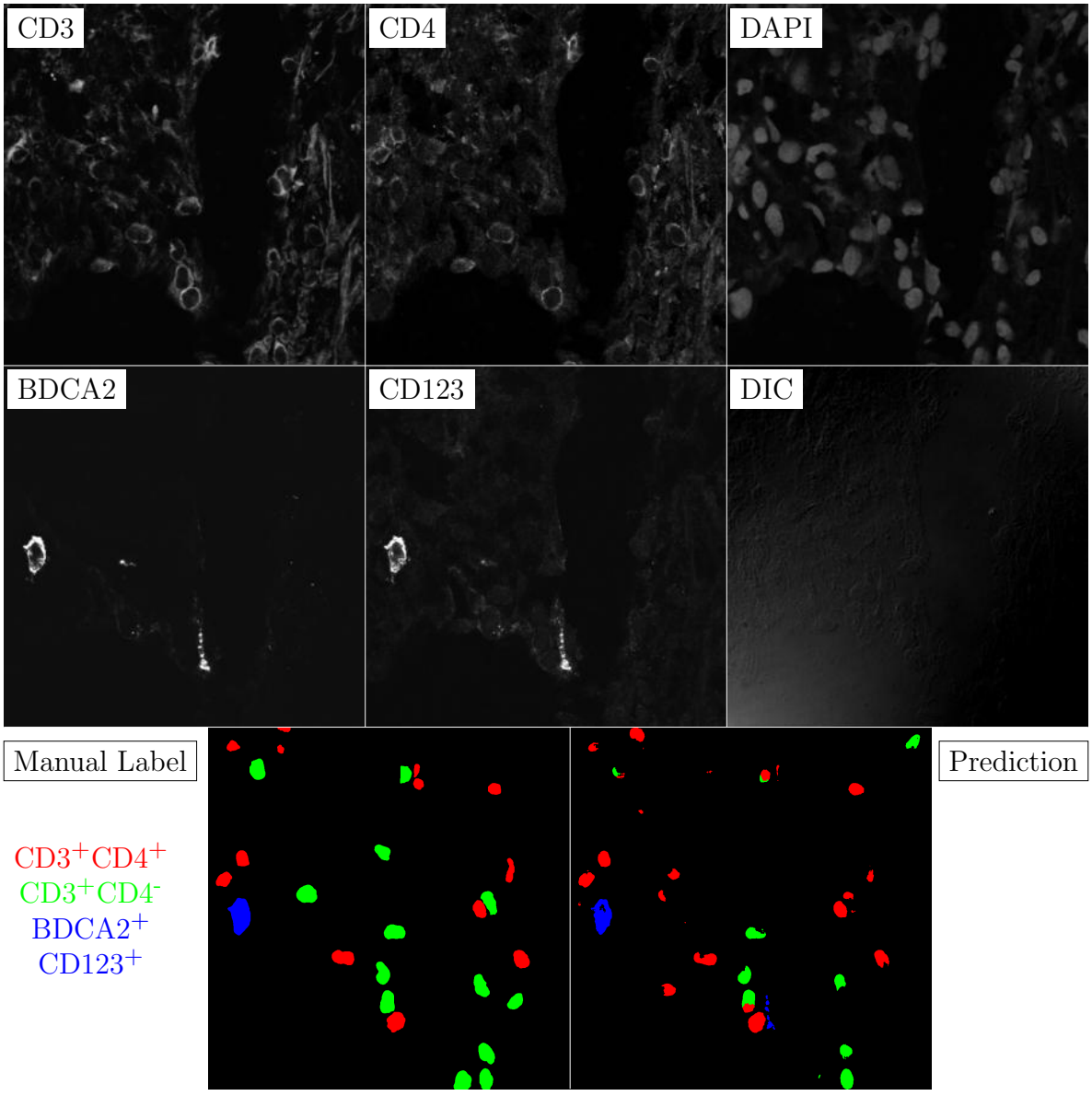

Figure G.229: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

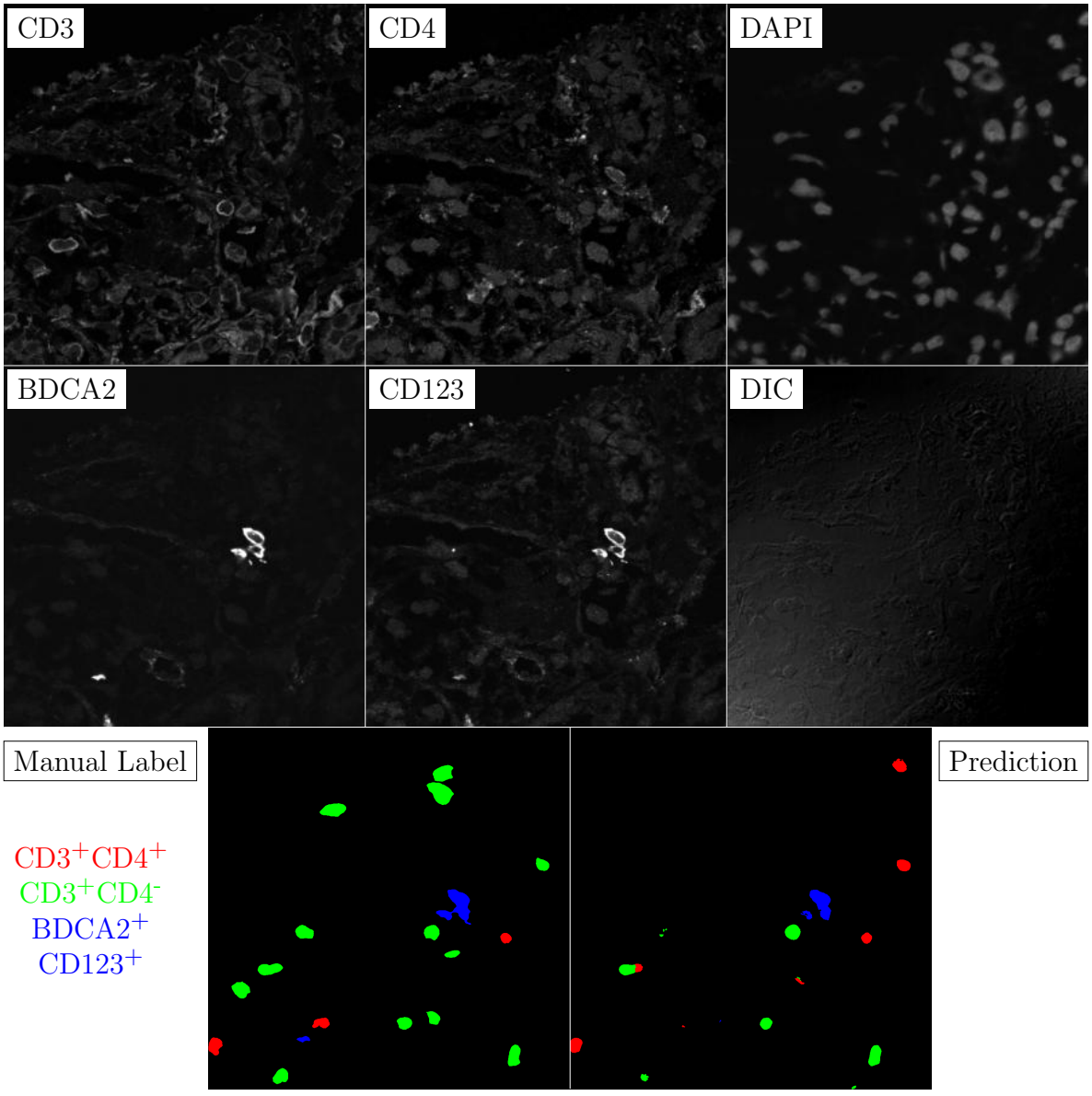

Figure G.230: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

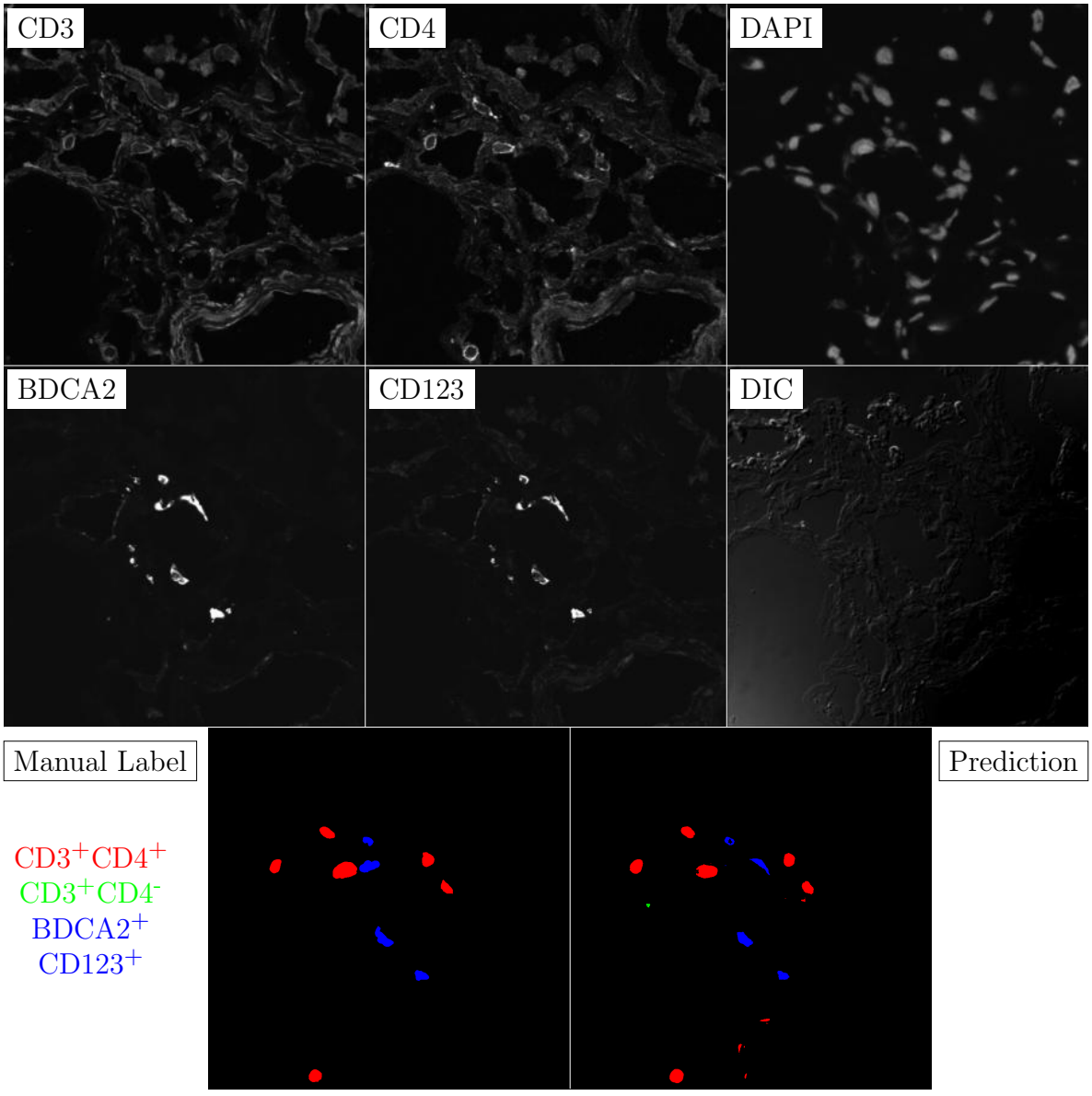

Figure G.231: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

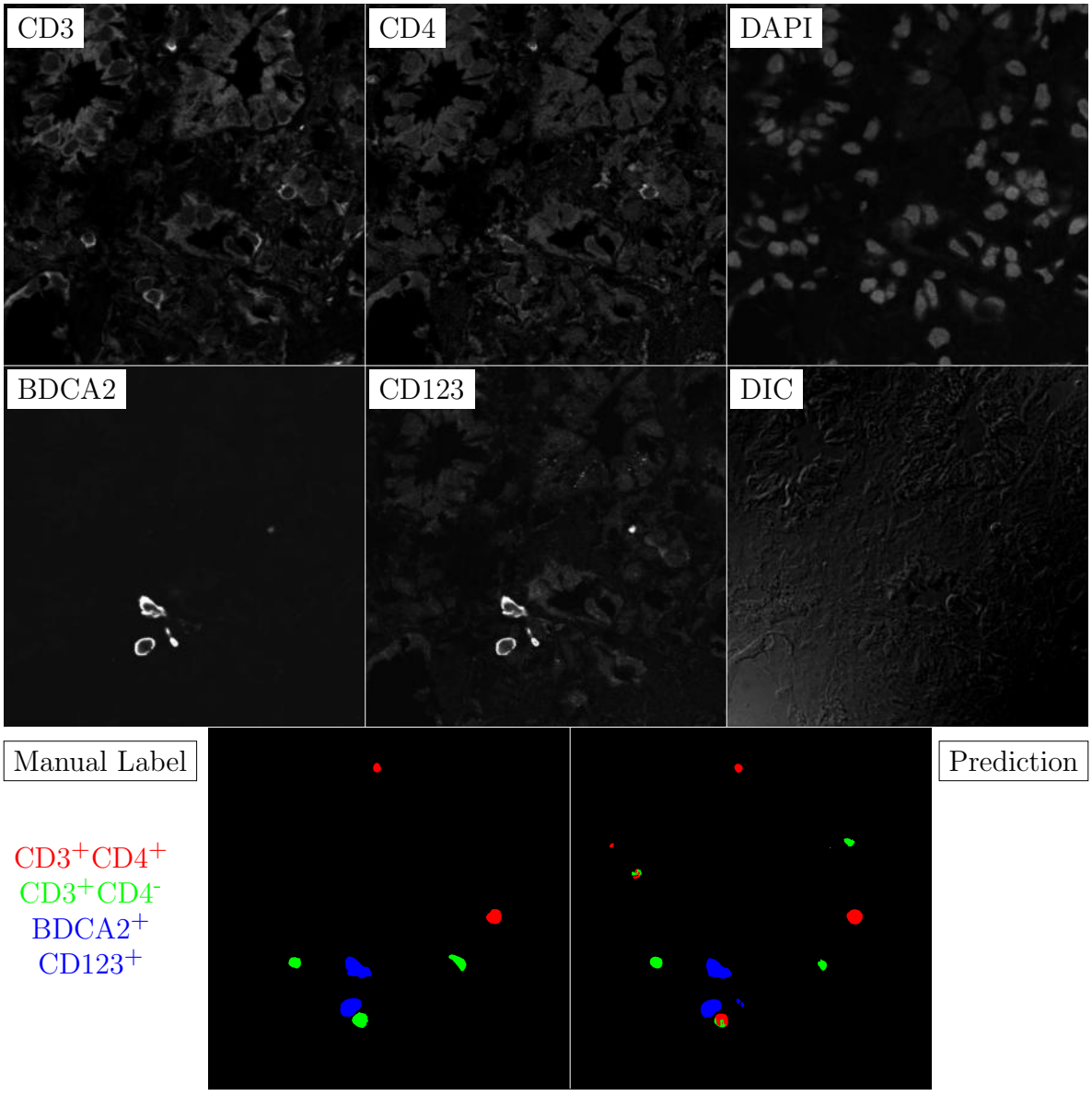

Figure G.232: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

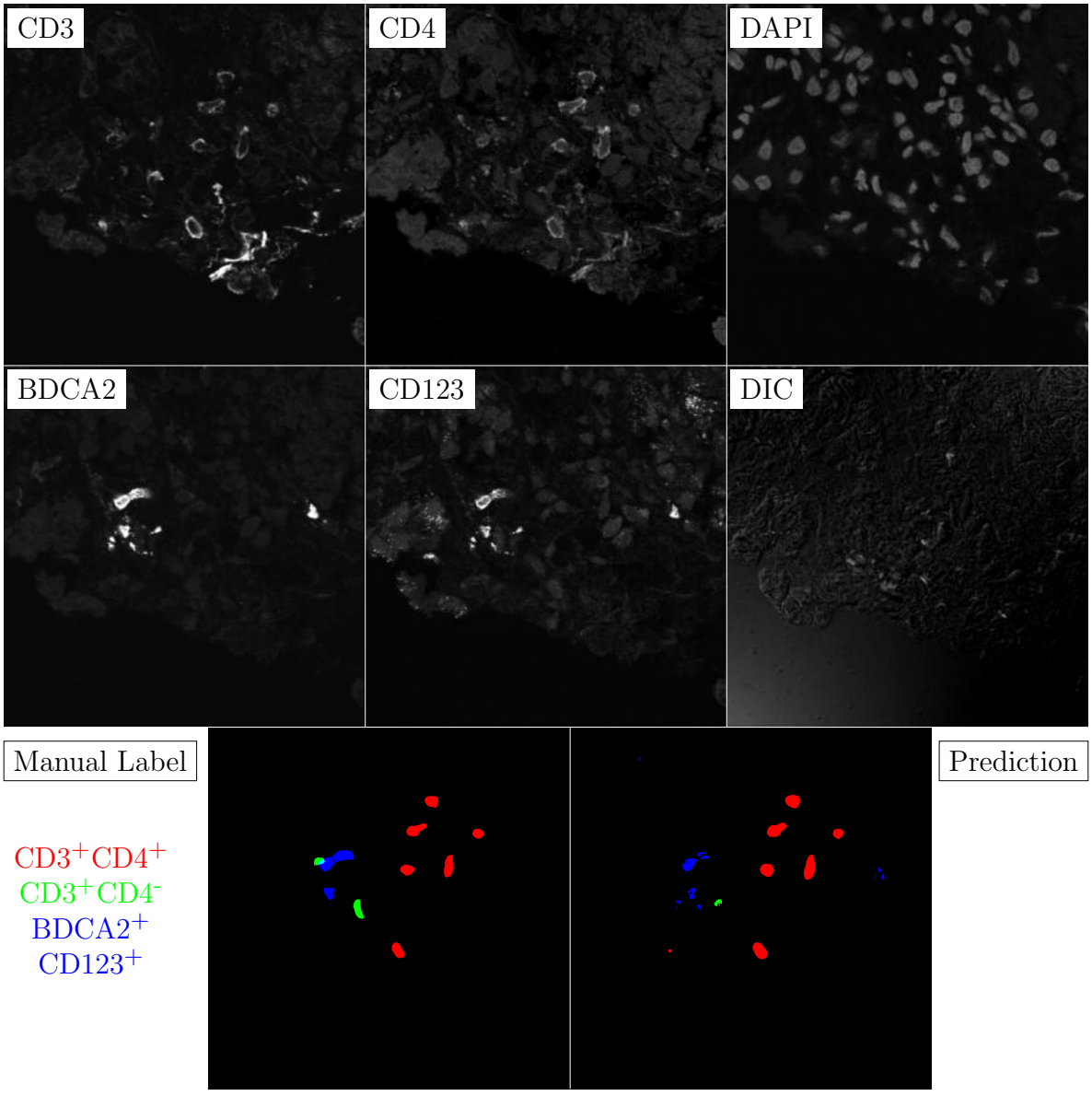

Figure G.233: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

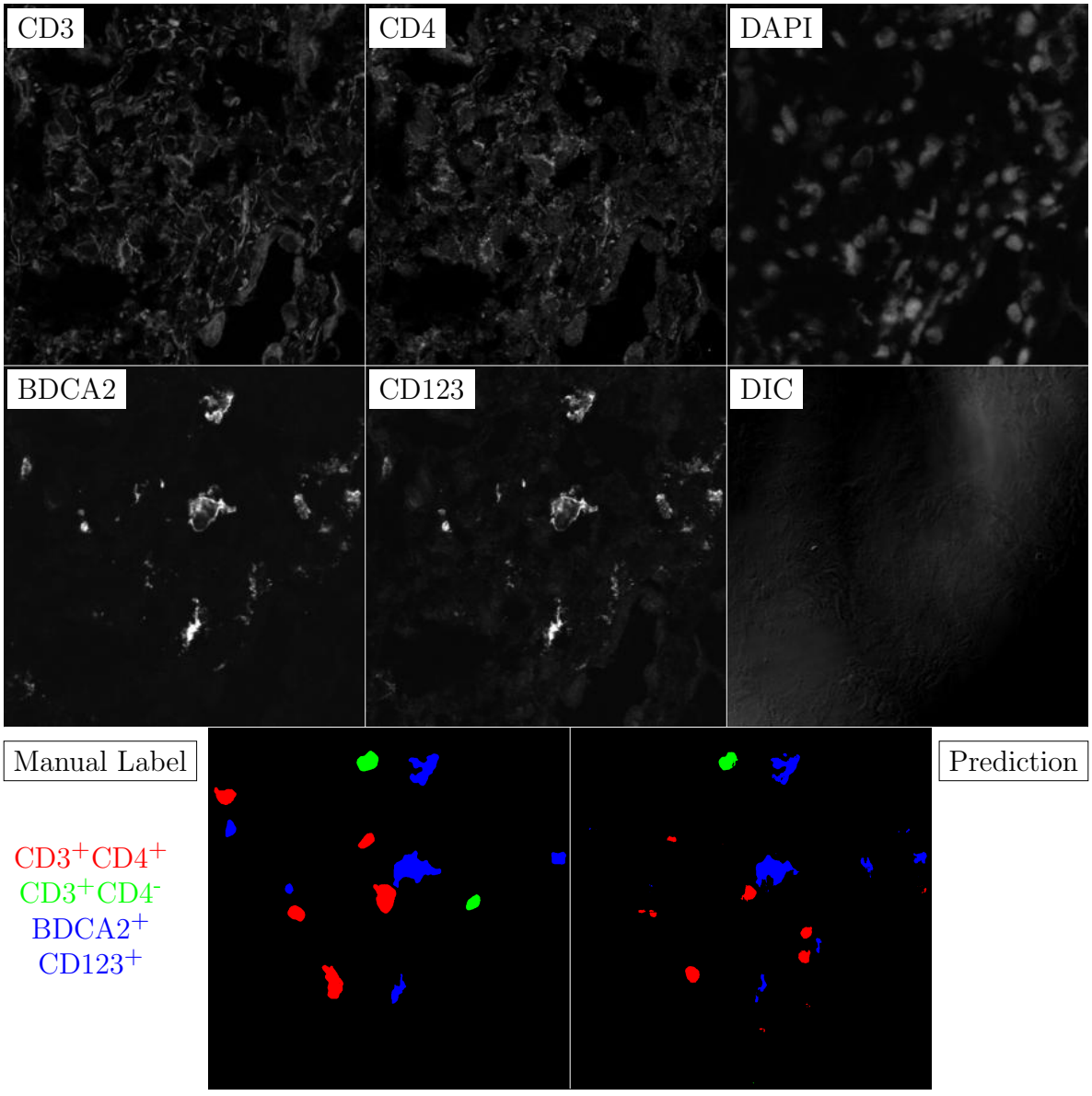

Figure G.234: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

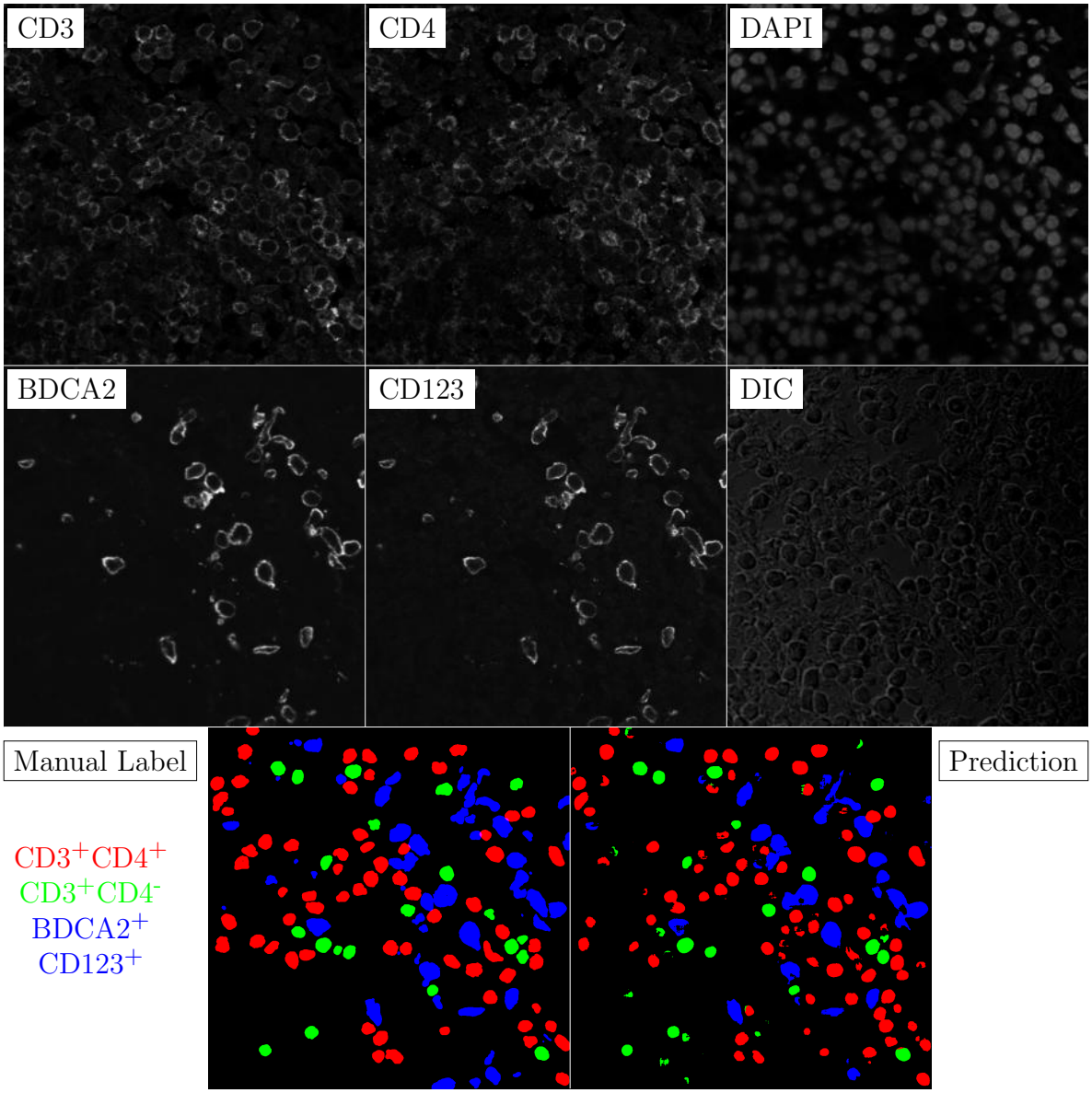

Figure G.235: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

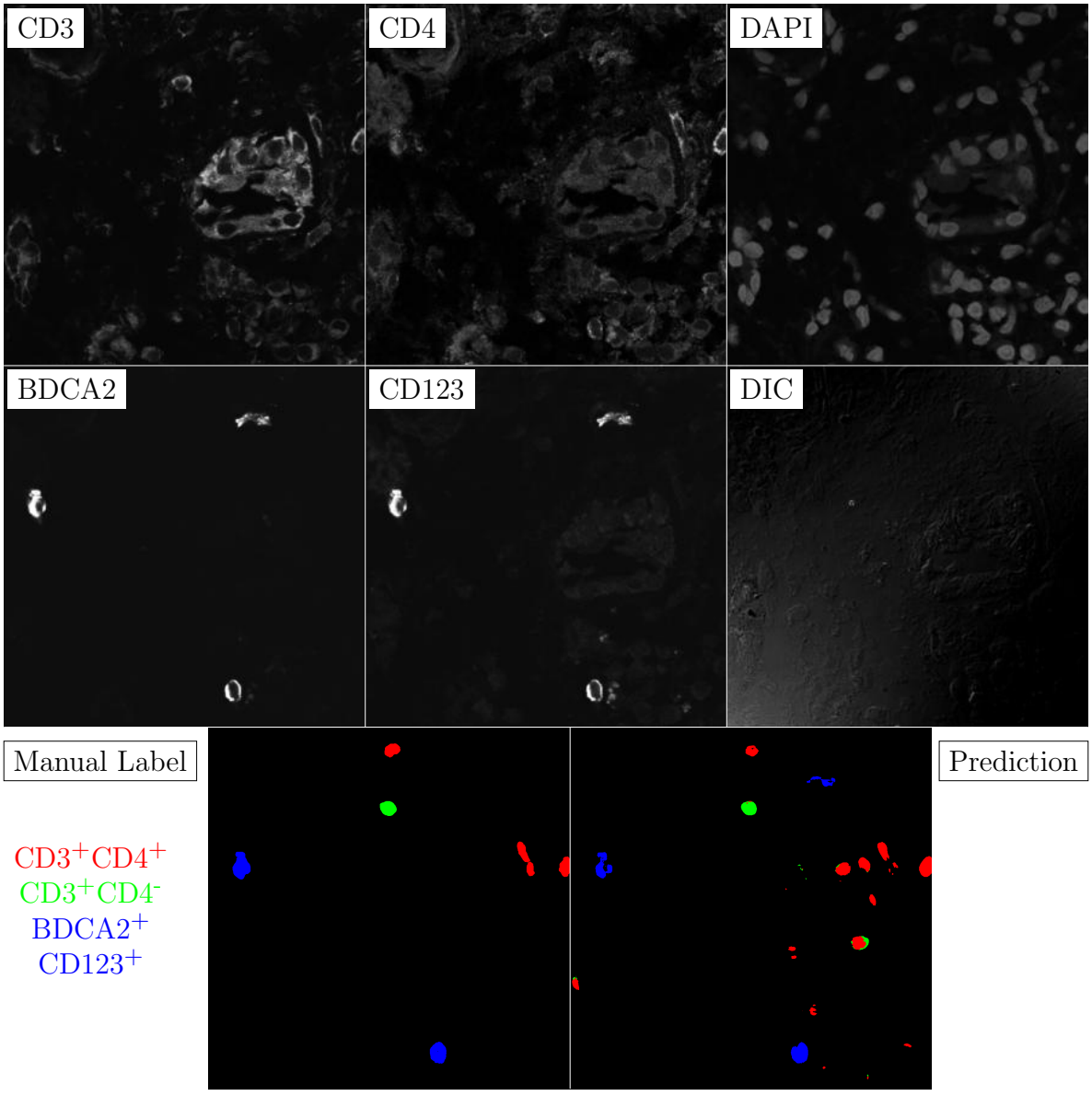

Figure G.236: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

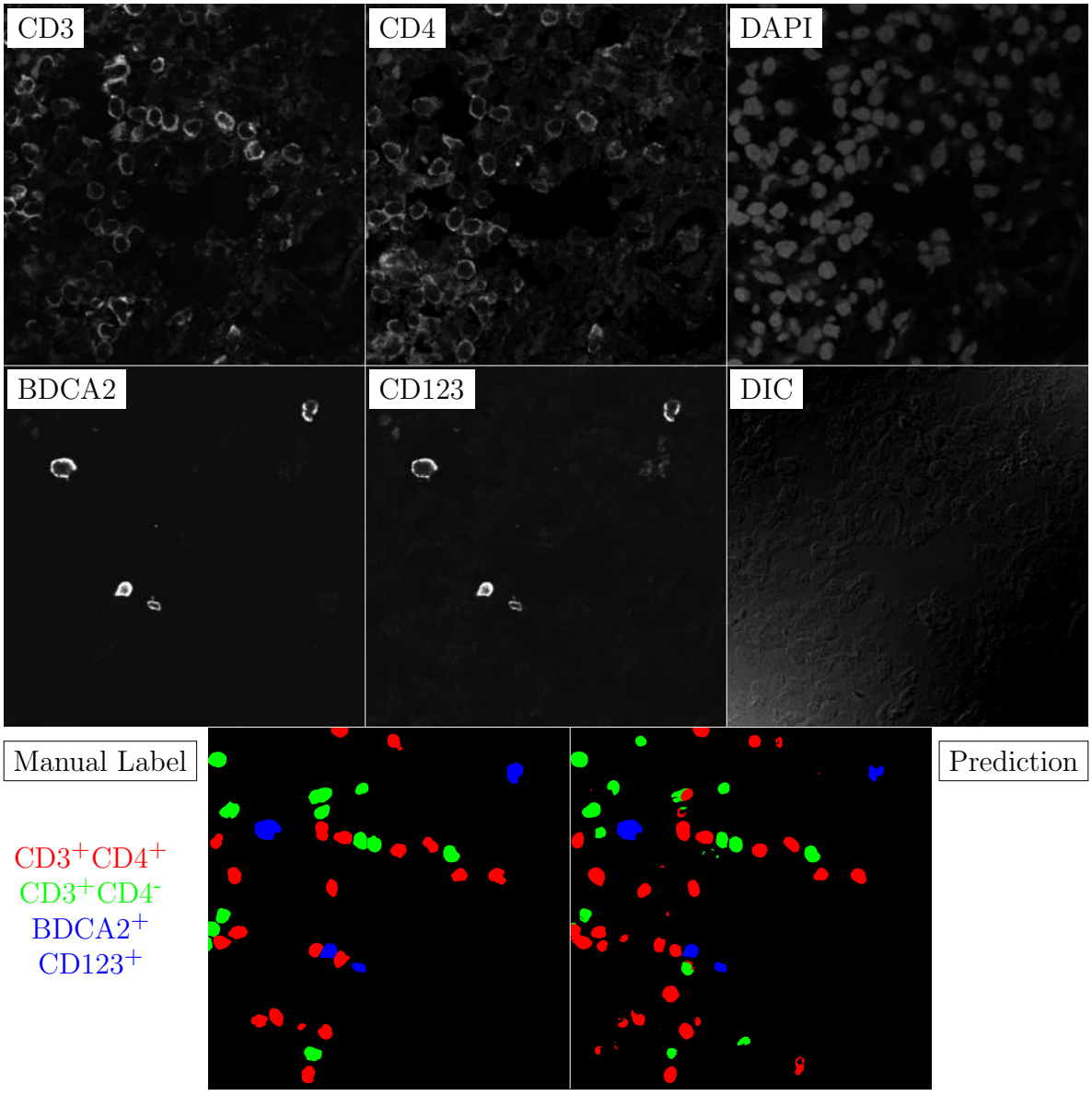

Figure G.237: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

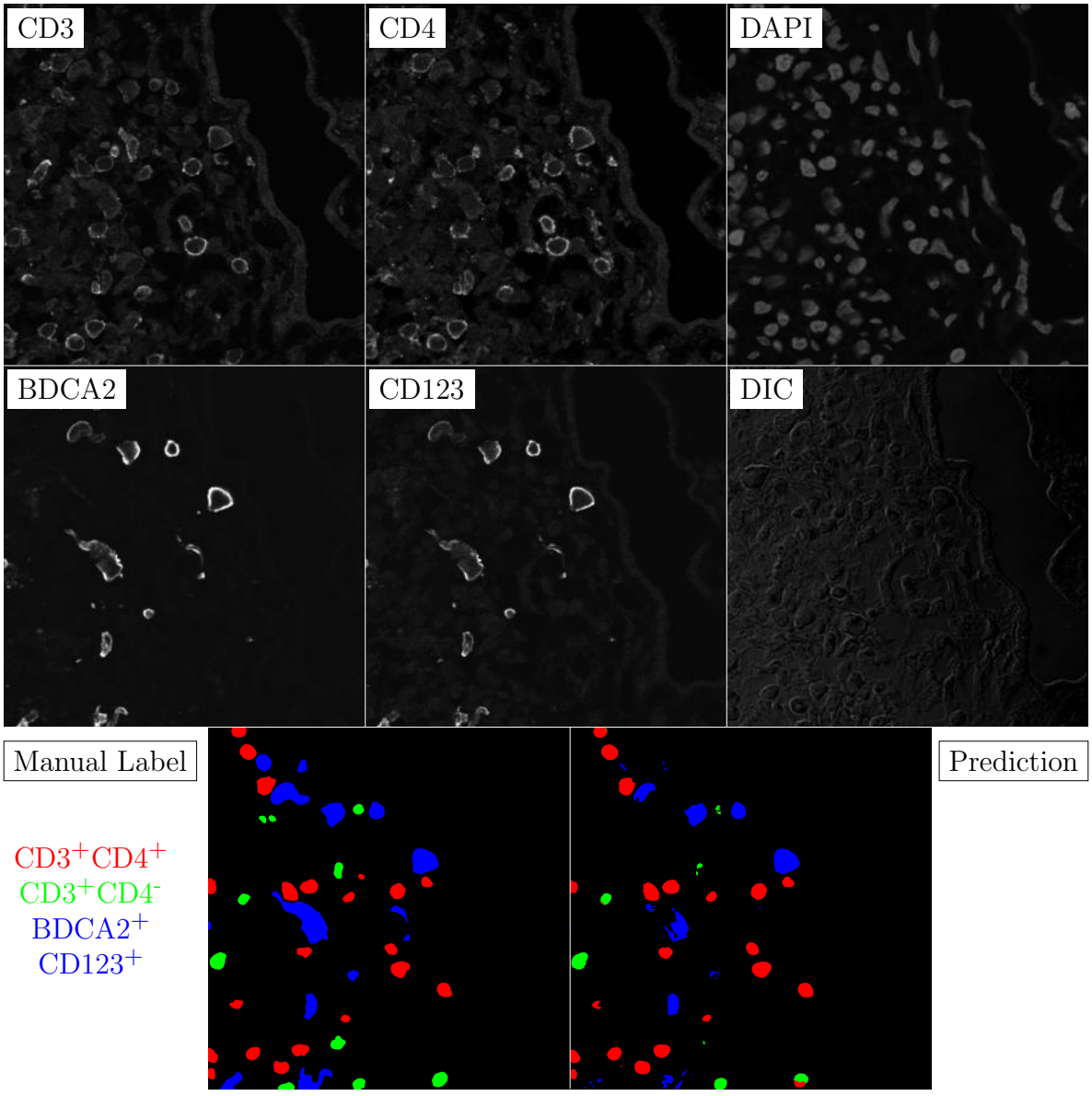

Figure G.238: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

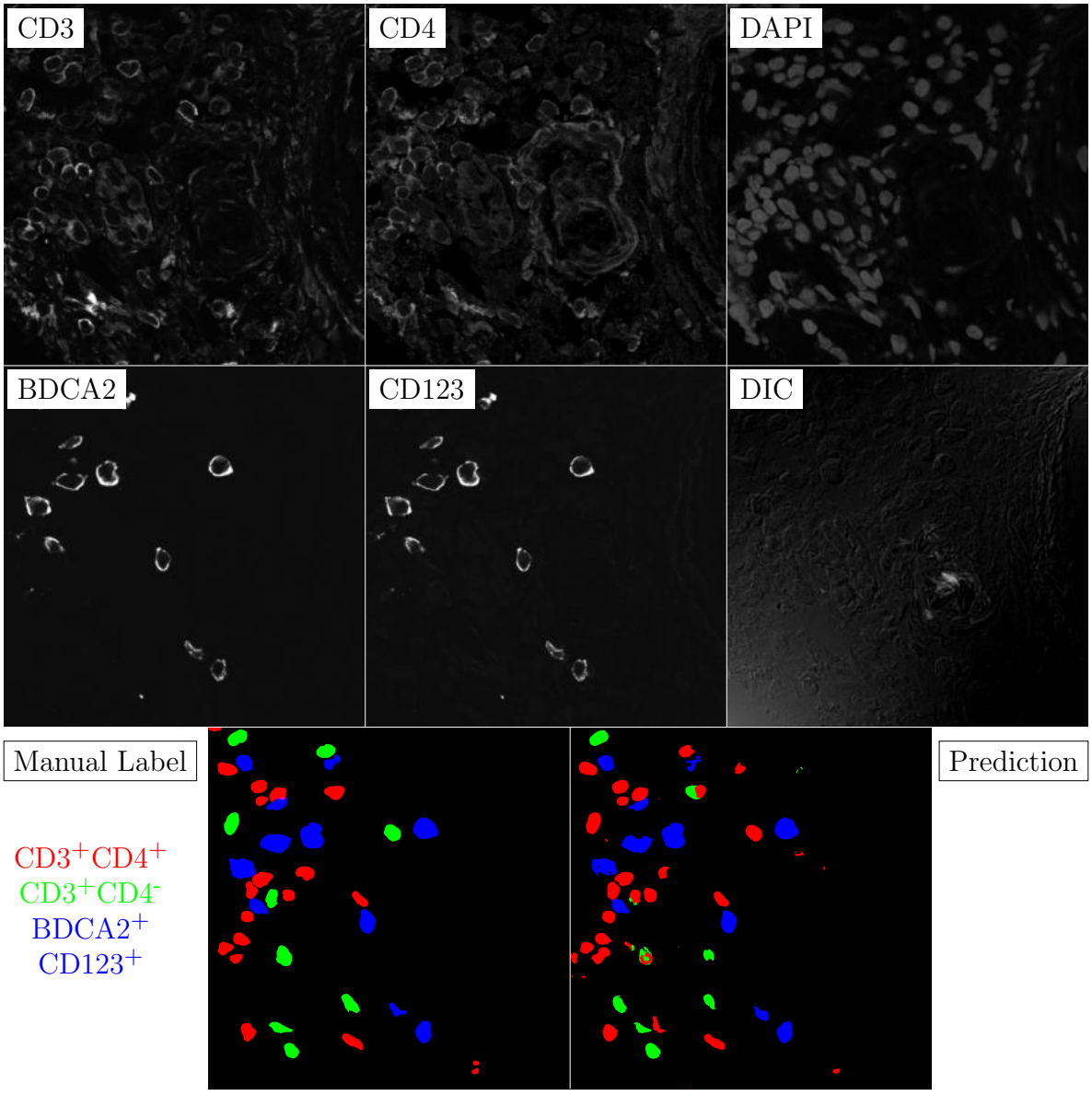

Figure G.239: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

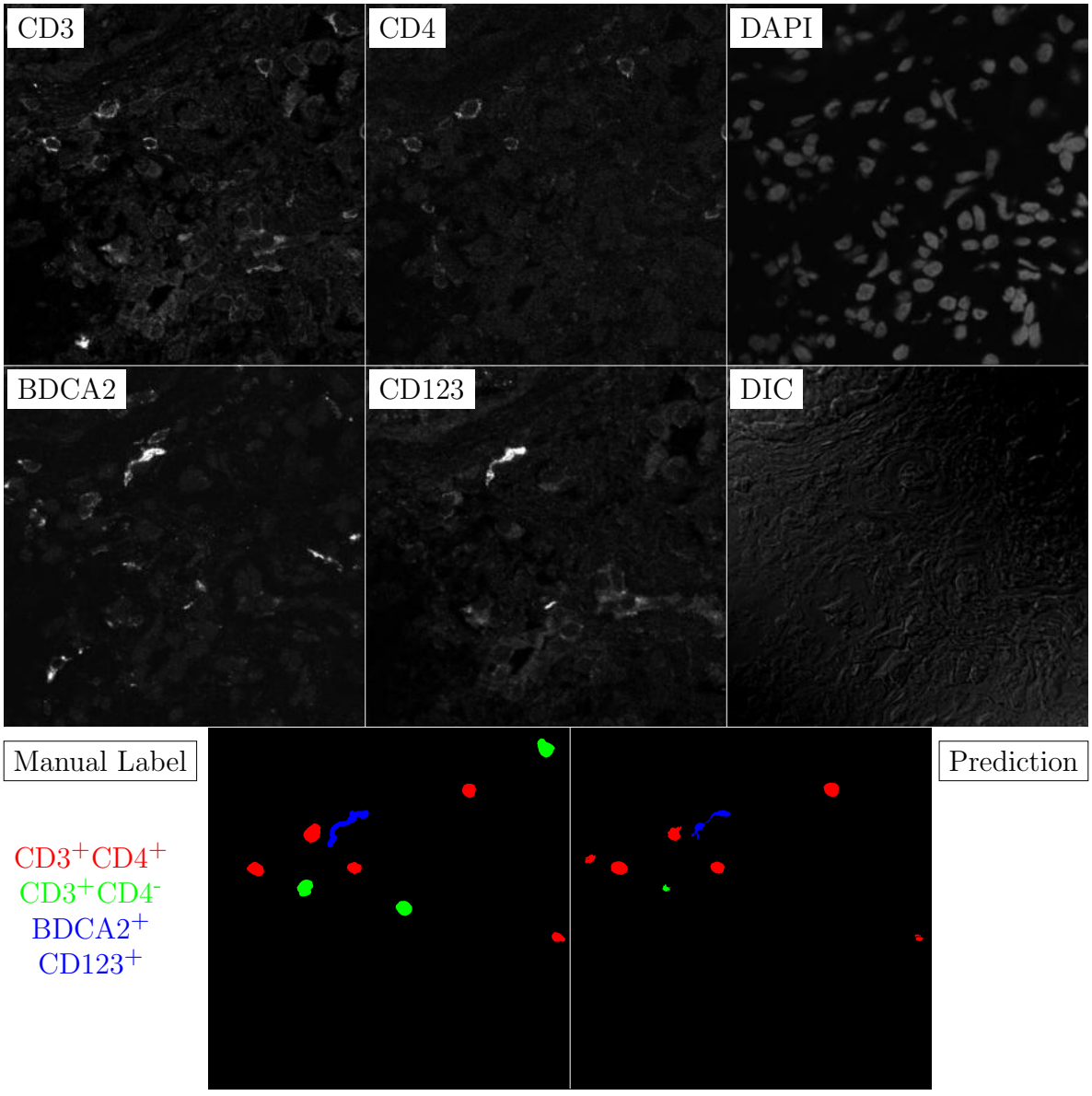

Figure G.240: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

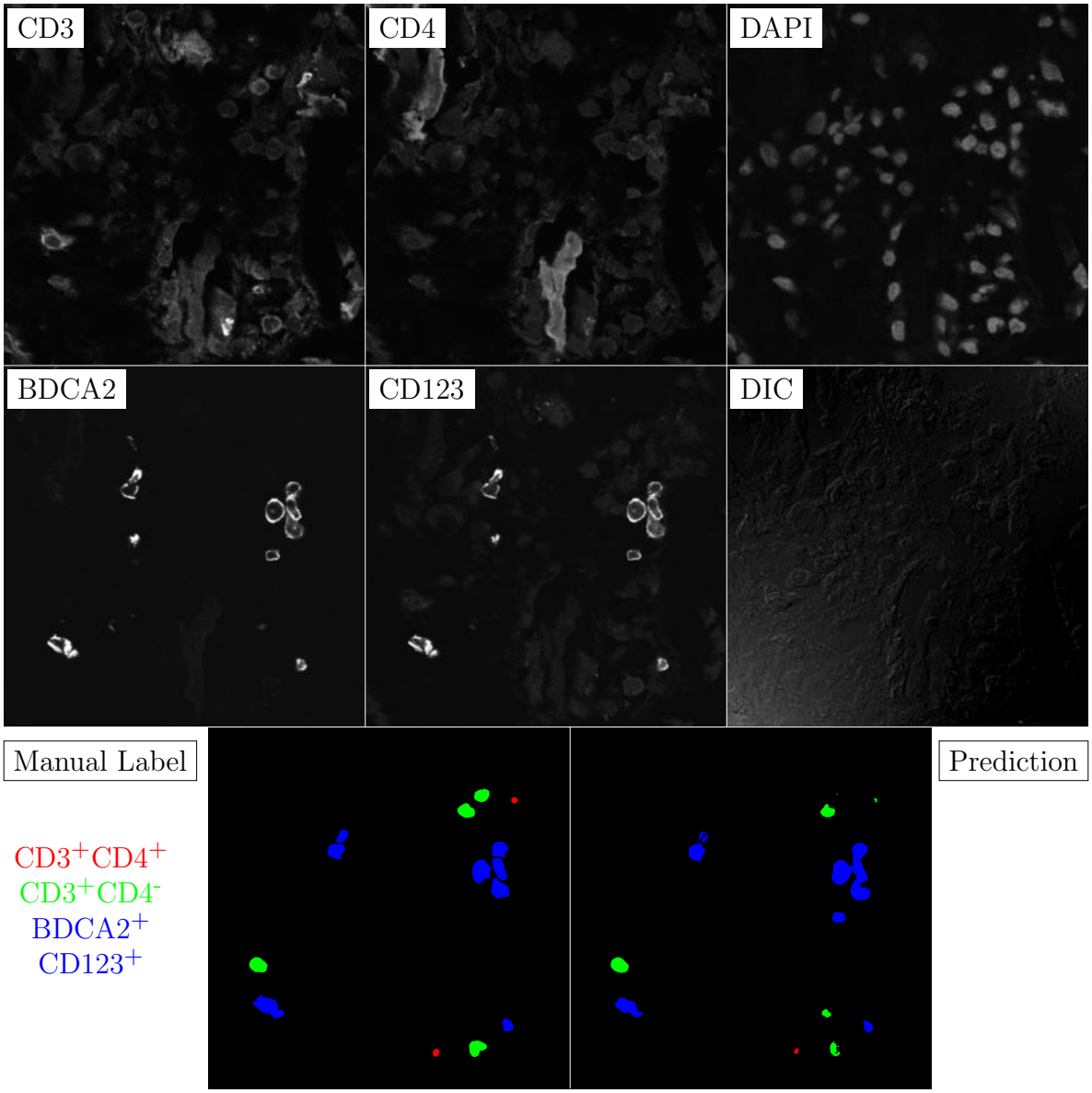

Figure G.241: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

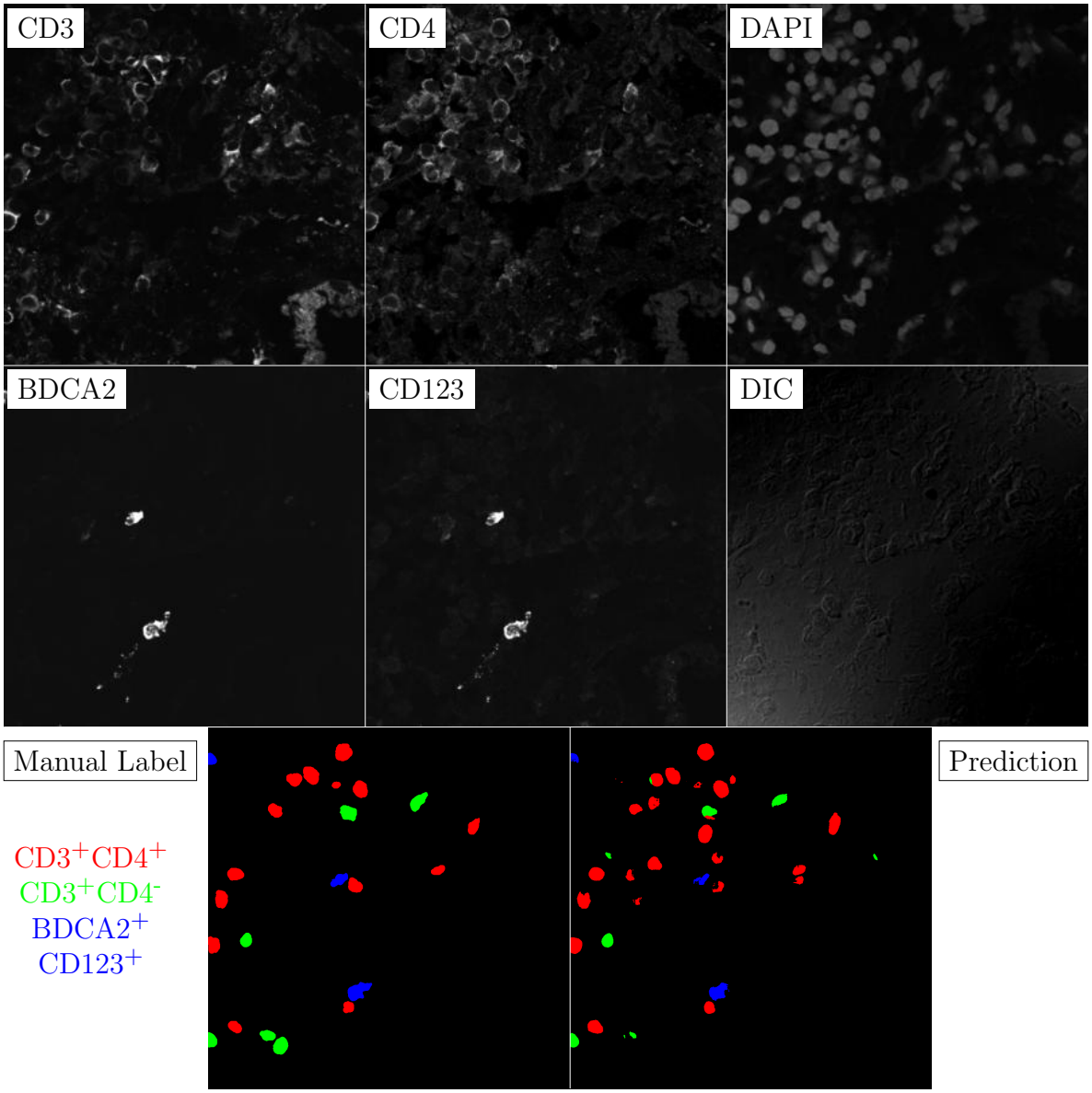

Figure G.242: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

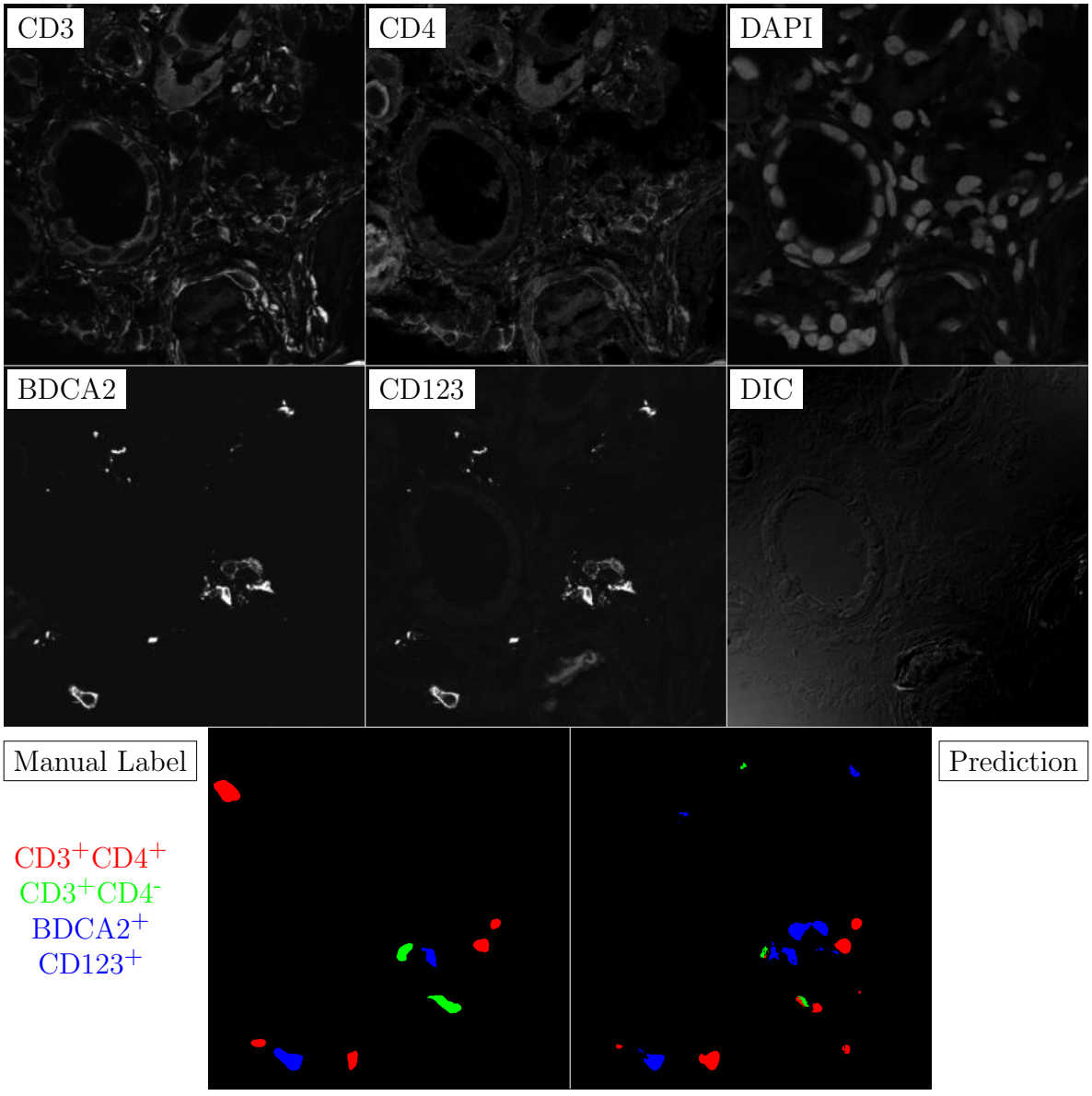

Figure G.243: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

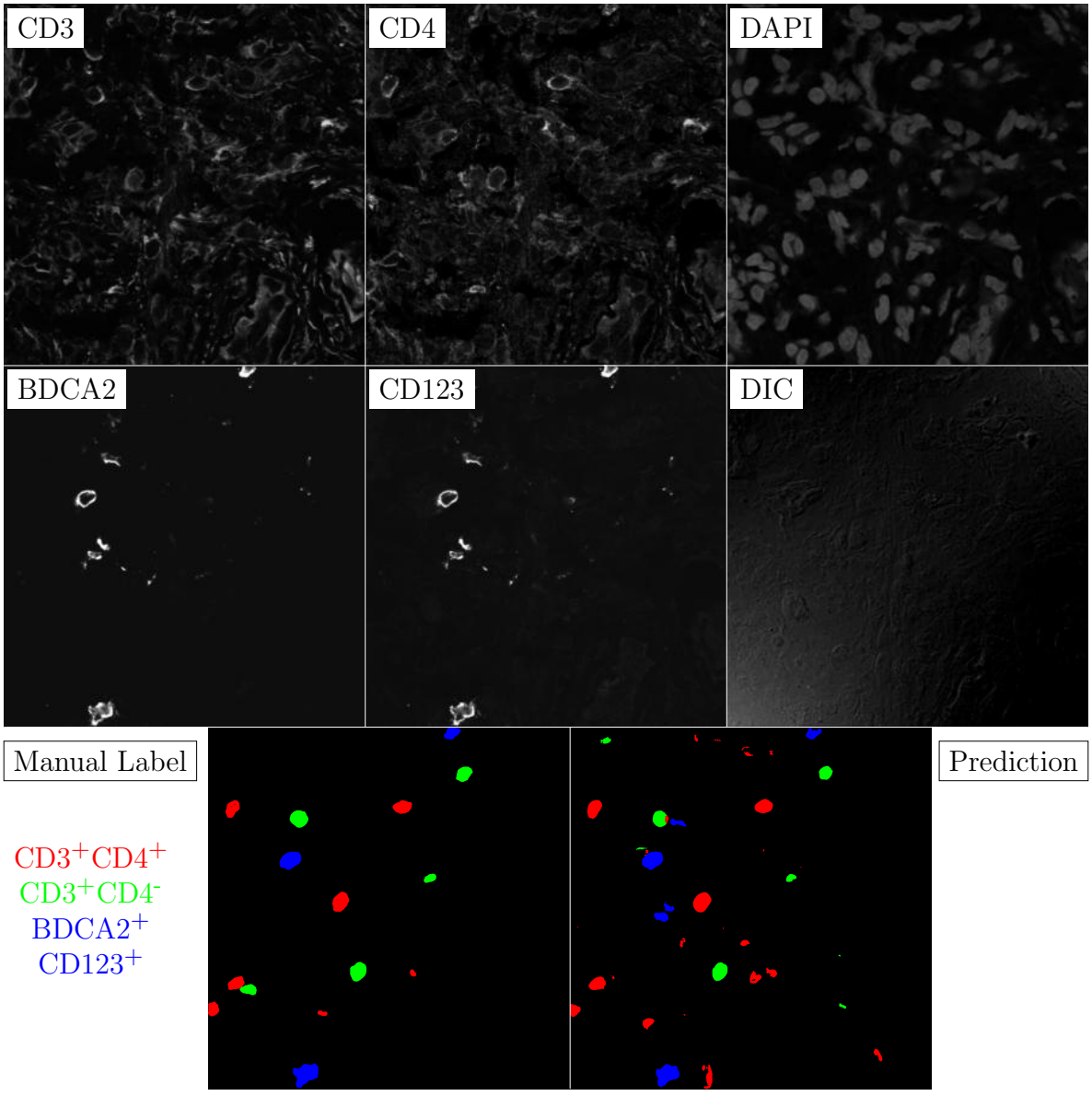

Figure G.244: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

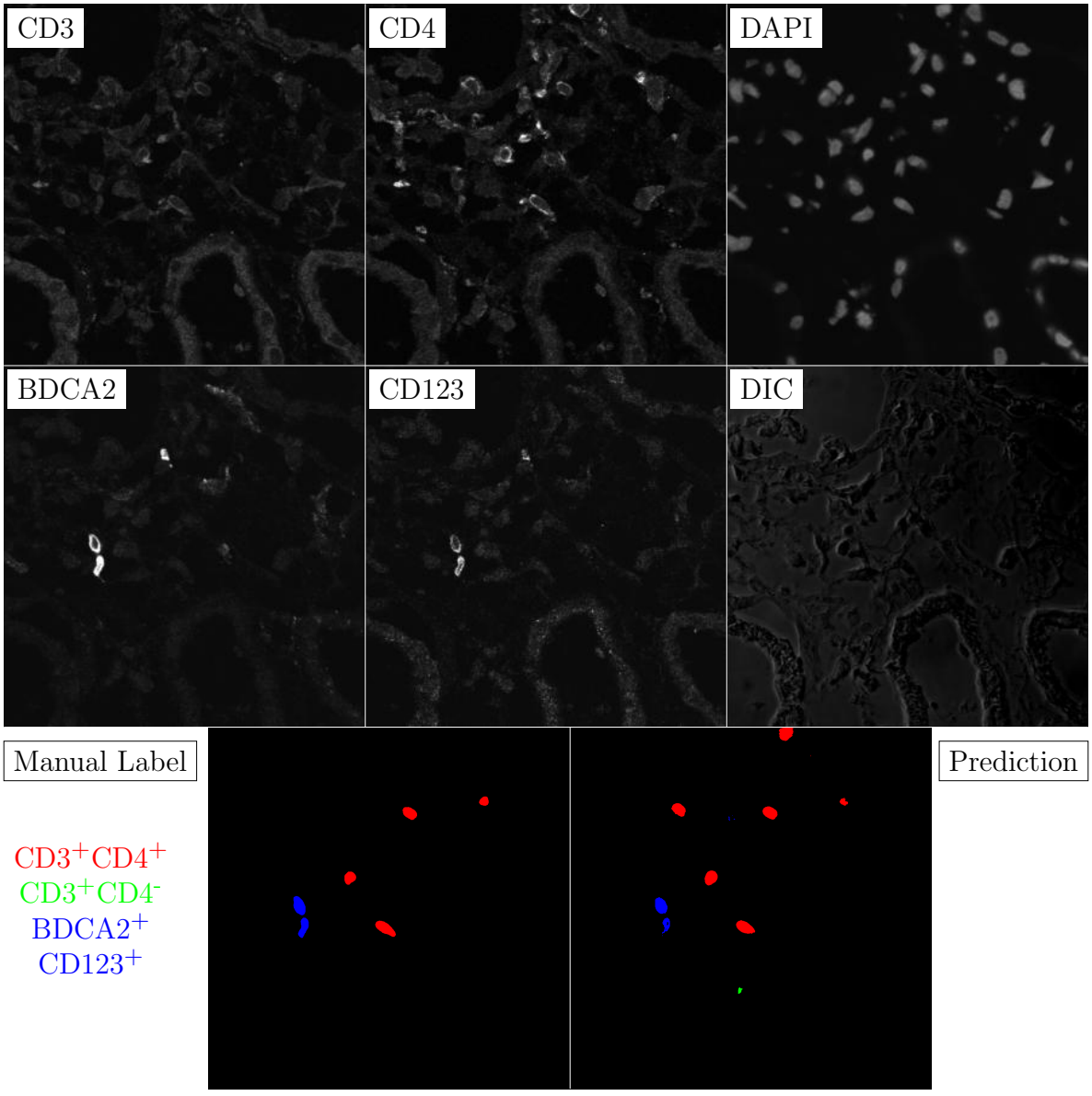

Figure G.245: **Human FF**. True ROI matrix size is 1024×1024. Depicted ROI matrix size is 300×300. for images and 1024×1024 for labels.

APPENDIX H

PARAFFIN PERFORMANCE EXAMPLES

This section presents the full dataset of 59 tiled biopsies from the paraffin dataset as described in Chapter 6. Labels for the image channels are presented on one page and image channels are presented on the following page in a separate figure. These figures are perhaps best viewed electronically in a pdf viewer such as the free software Adobe Acrobat reader in the side by side page setting. White borders painted on the edges of cells to highlight the separation between cells in close proximity. The border highlighting algorithm is not perfect and the borders are translated slightly from there proper positions.

Label ROIs are embedded in document with DEFLATE png compression at full resolution with exception for two very large tiled biopsies which were downsampled with bilinear interpolation to such that the maximum x,y dimension of the image matrix was equal to 10,000. This was done because though the memory footprint of the compressed label images is very small, either the pdf standard or the LaTeX type-setting software does not support images beyond a matrix size beyond somewhere in the range of [10,000,20,000] image matrix size. All images for the biopsy channel were downsampled with bilinear interpolation such that the maximum x,y dimension of the image was equal to 300 pixels. This was done to keep the memory footprint of this pdf document at a reasonable level while also avoiding the compression artifacts of high level jpeg compression. Images were transformed from native bit depth of 12 by rescaling minimum and maximum values in the individual channel image to a pixel intensity range of [0,255]. Pixel size is $0.14\,\mu m \times 0.14\,\mu m$. Bit depth is 8 in embedded images.

Images were rotated and tiled on page using a python software script to maximize page usage while keeping the image x,y dimension ratio constant.

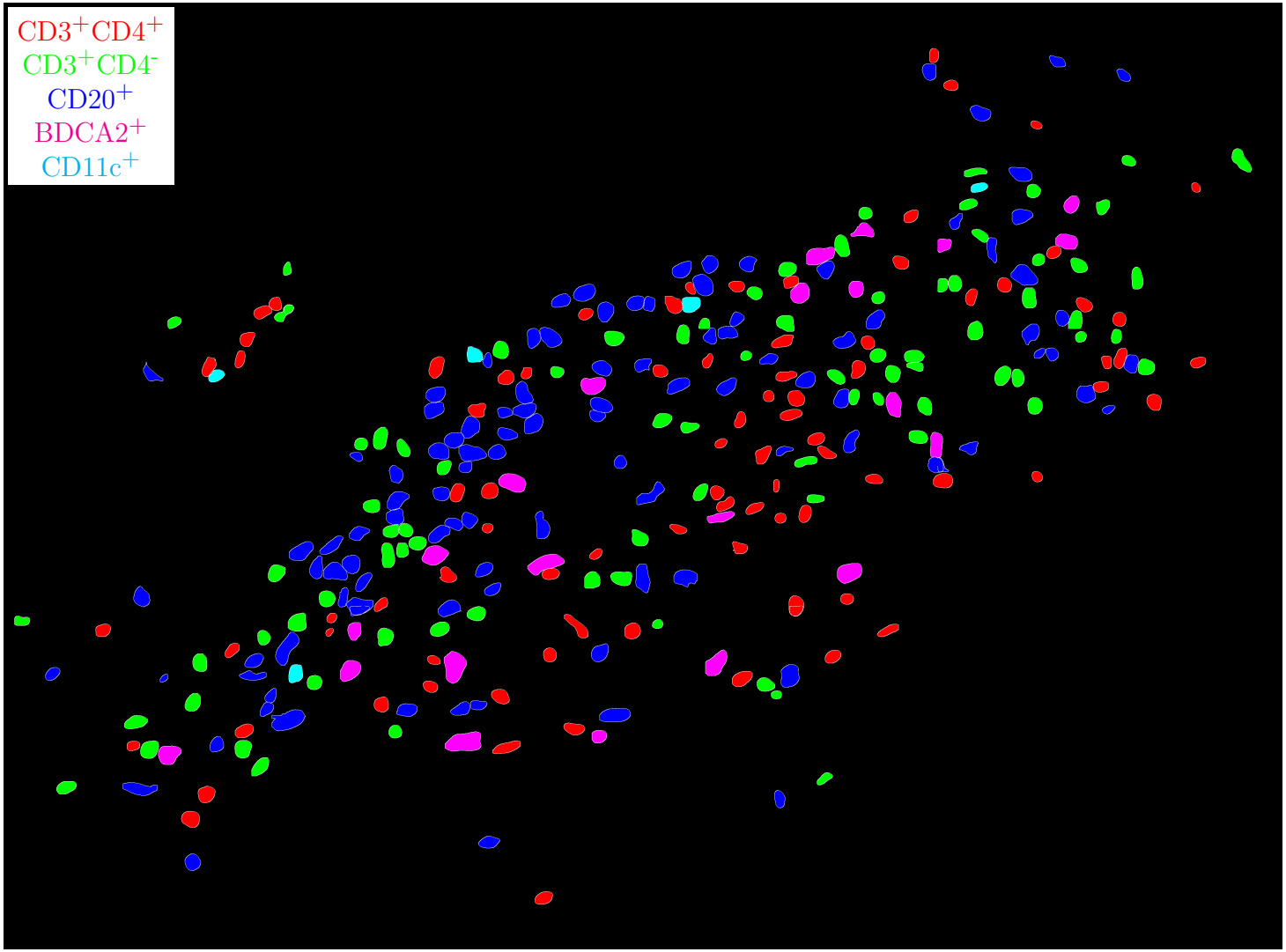

Figure H.1: **Label for H.2**. True label matrix size is 2820×3796. Depicted label matrix size is 2820×3796.

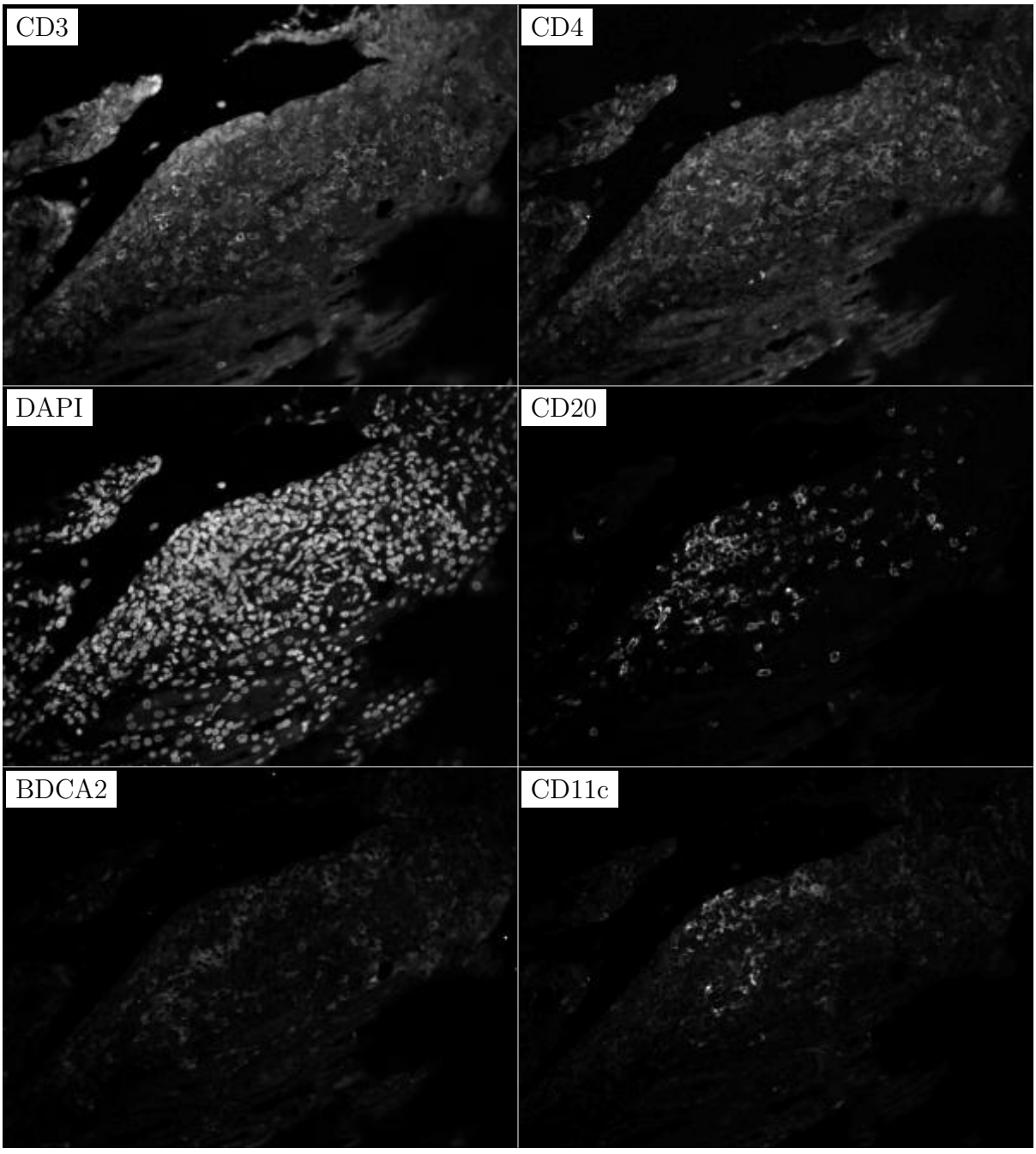

Figure H.2: **ROI for H.1**. True image channel matrix size is 2820×3796. Depicted image channel matrix size is 222×300.

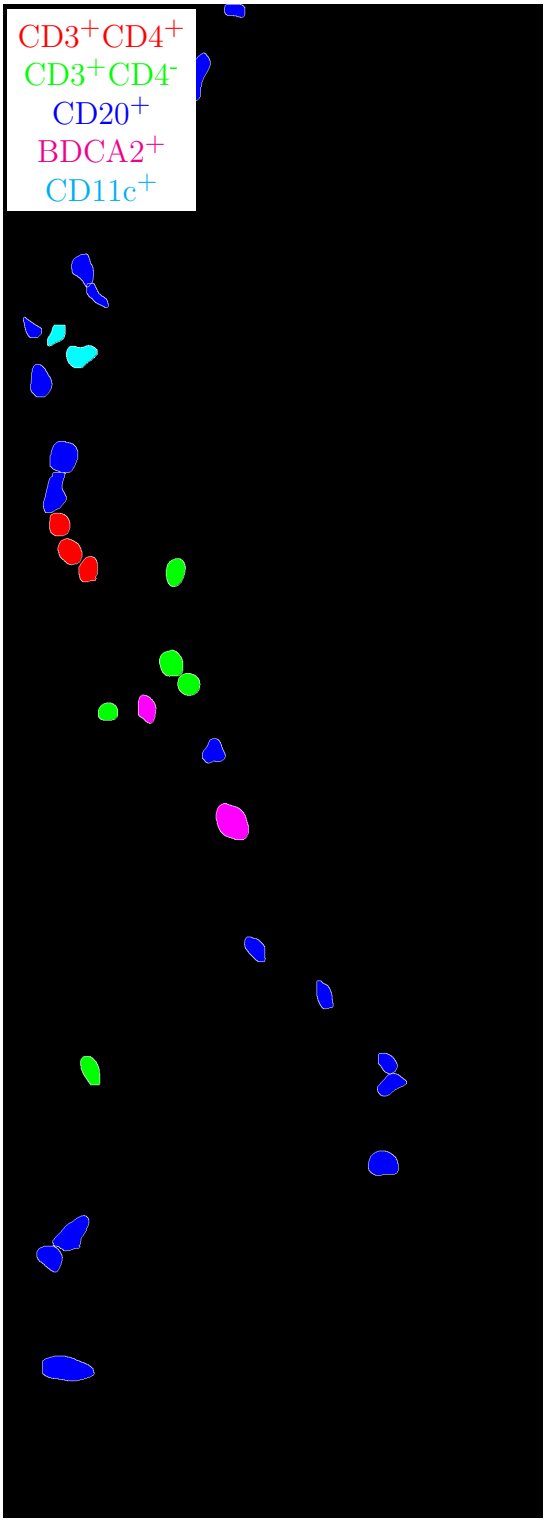

Figure H.3: **Label for H.4**. True label matrix size is 2865 × 1028. Depicted label matrix size is 2865 × 1028.

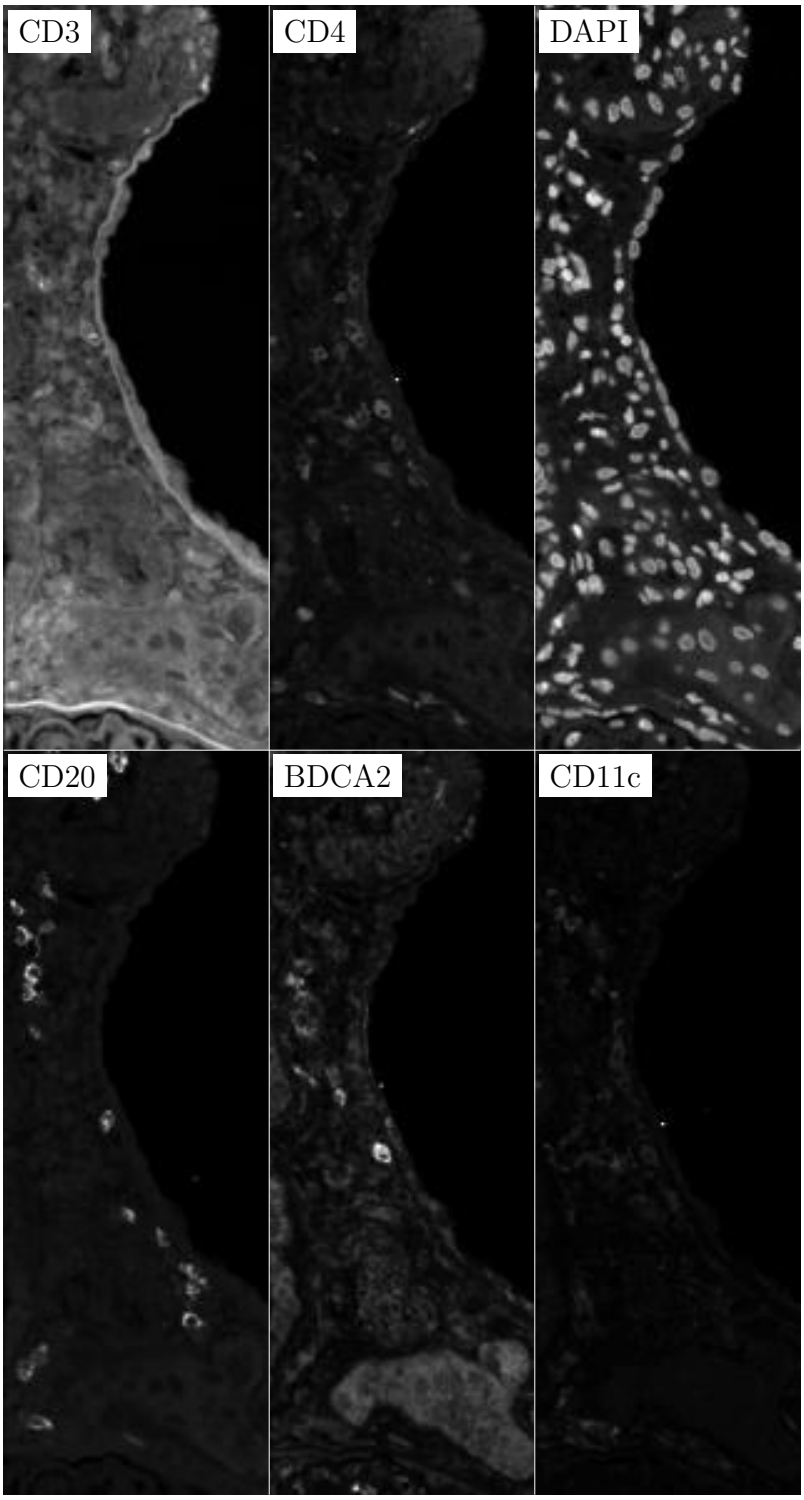

Figure H.4: **ROI for H.3**. True image channel matrix size is 2865×1028. Depicted image channel matrix size is 300×107.

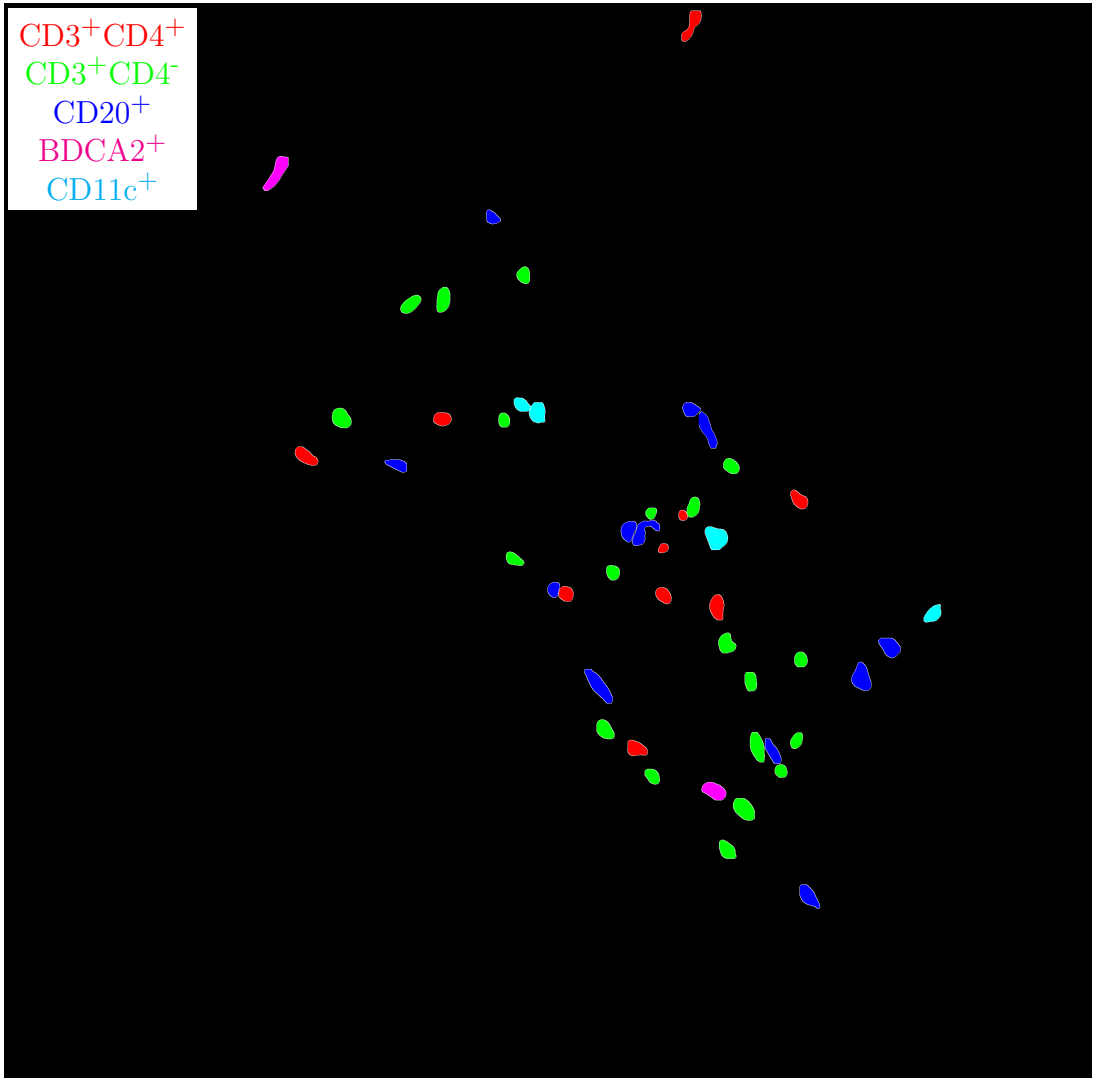

Figure H.5: **Label for H.6**. True label matrix size is 2865×2894. Depicted label matrix size is 2865×2894.

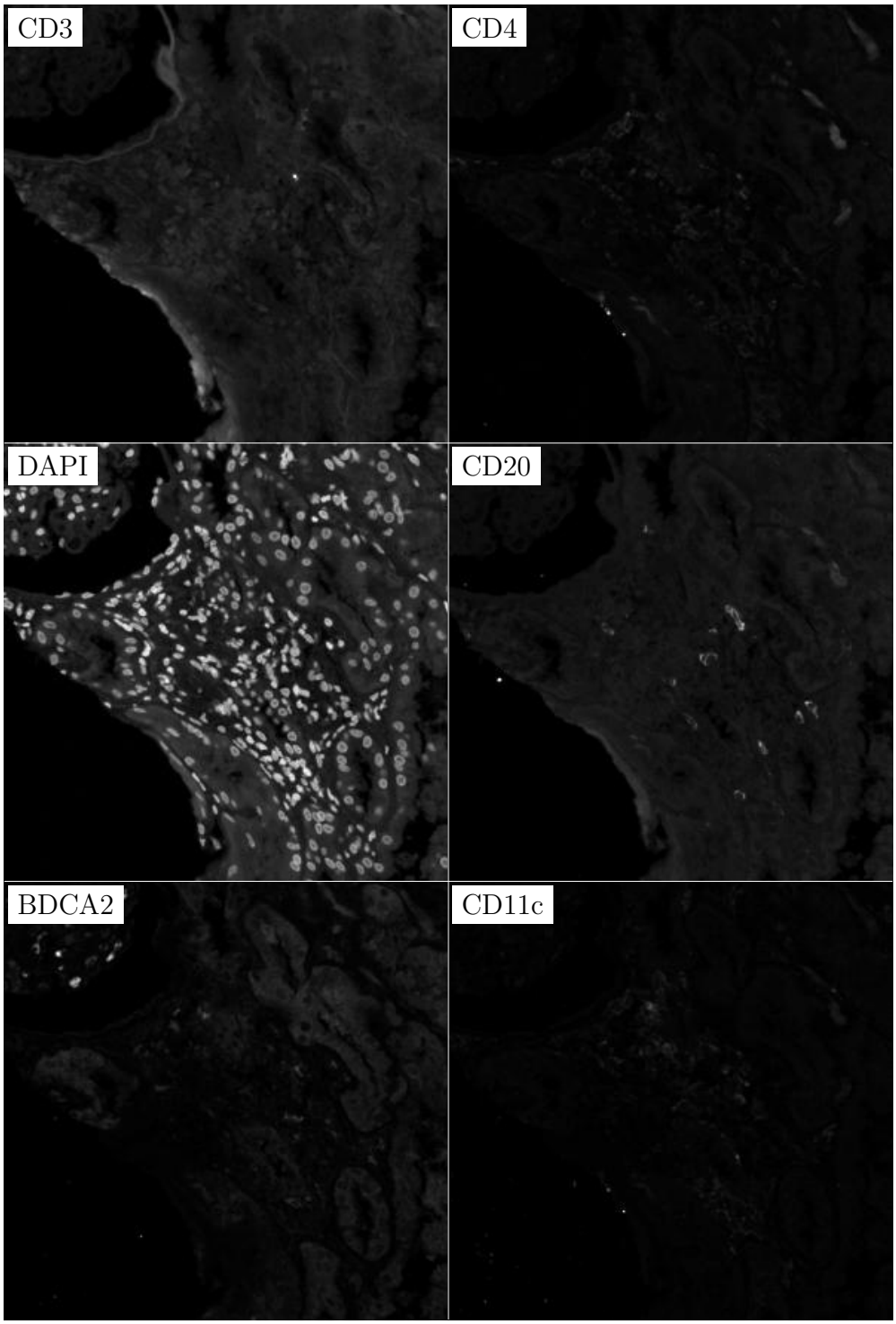

Figure H.6: **ROI for H.5**. True image channel matrix size is 2865×2894. Depicted image channel matrix size is 296×300.

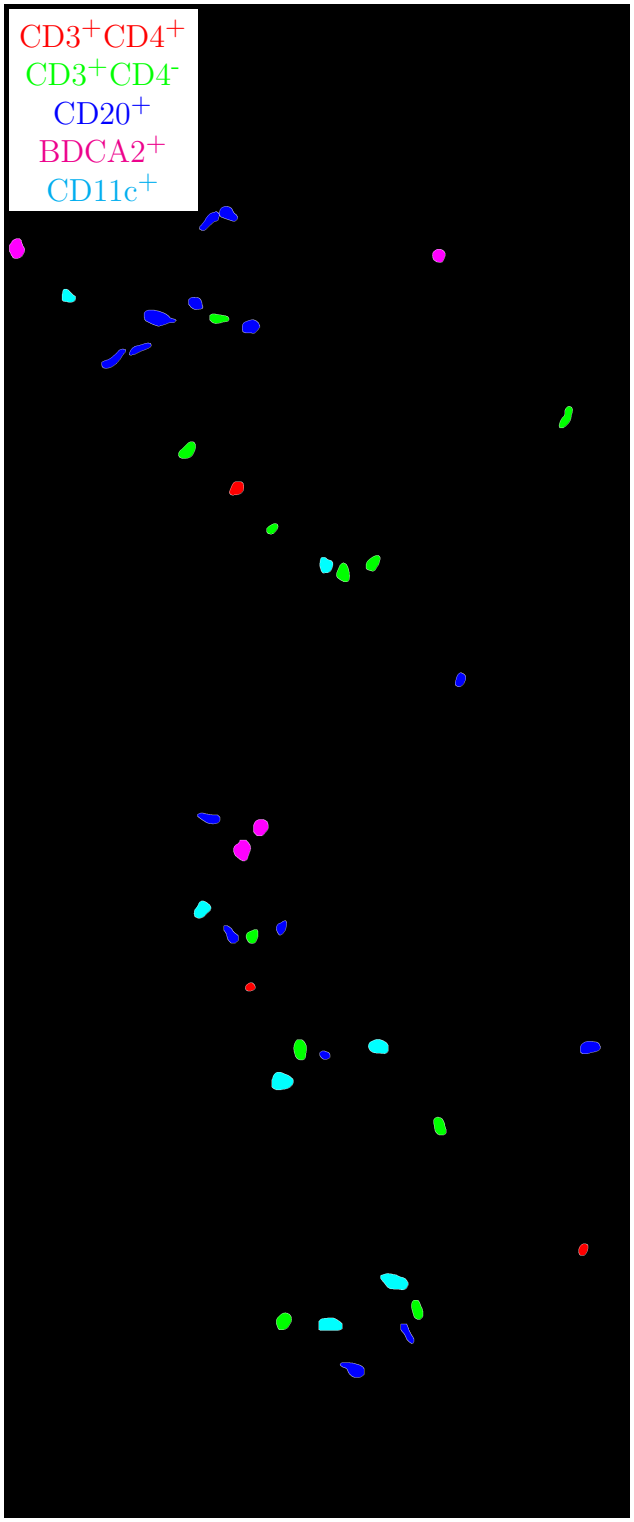

Figure H.7: **Label for H.8**. True label matrix size is 4702 × 1946. Depicted label matrix size is 4702 × 1946.

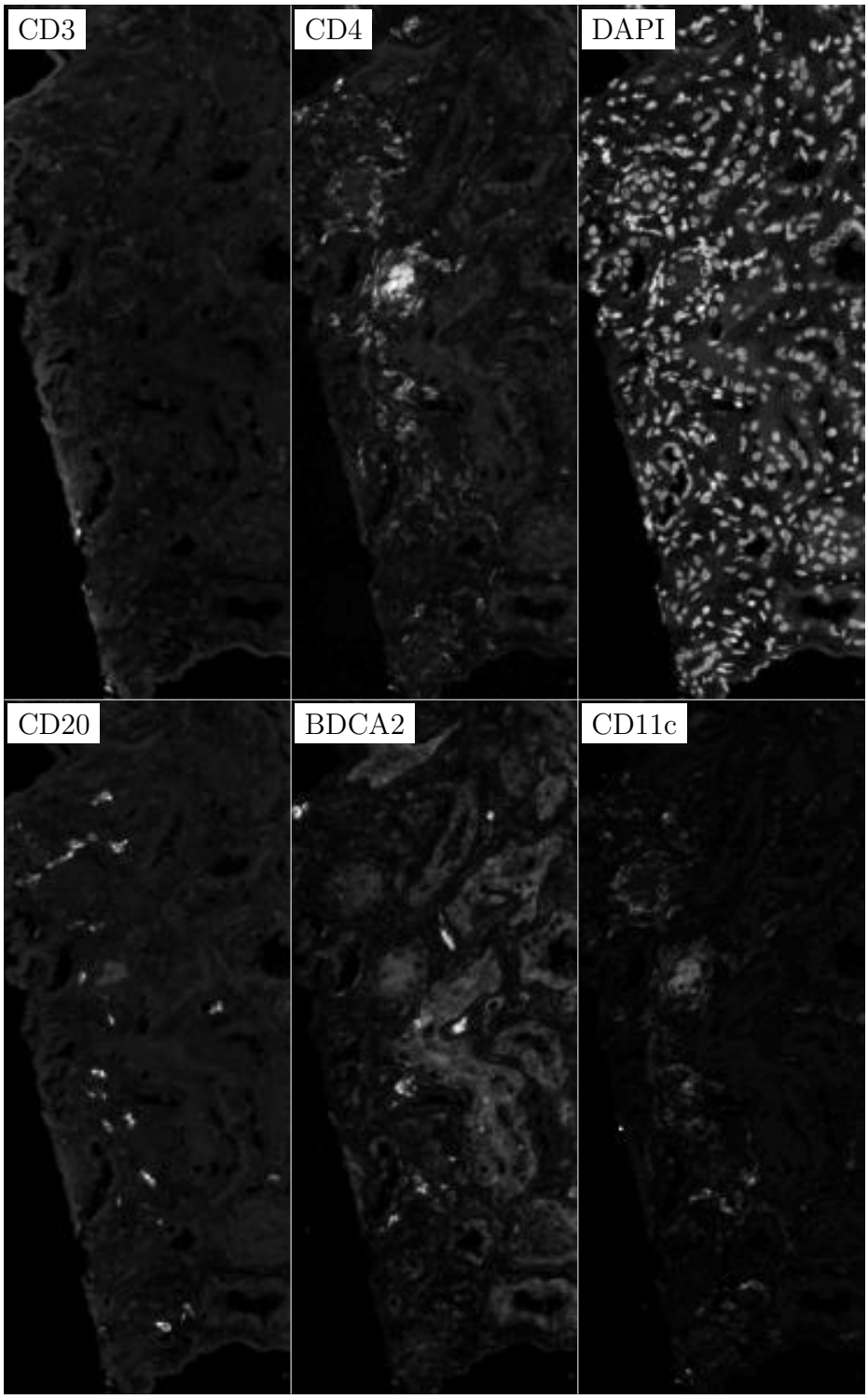

Figure H.8: **ROI for H.7**. True image channel matrix size is 4702×1946. Depicted image channel matrix size is 300×124.

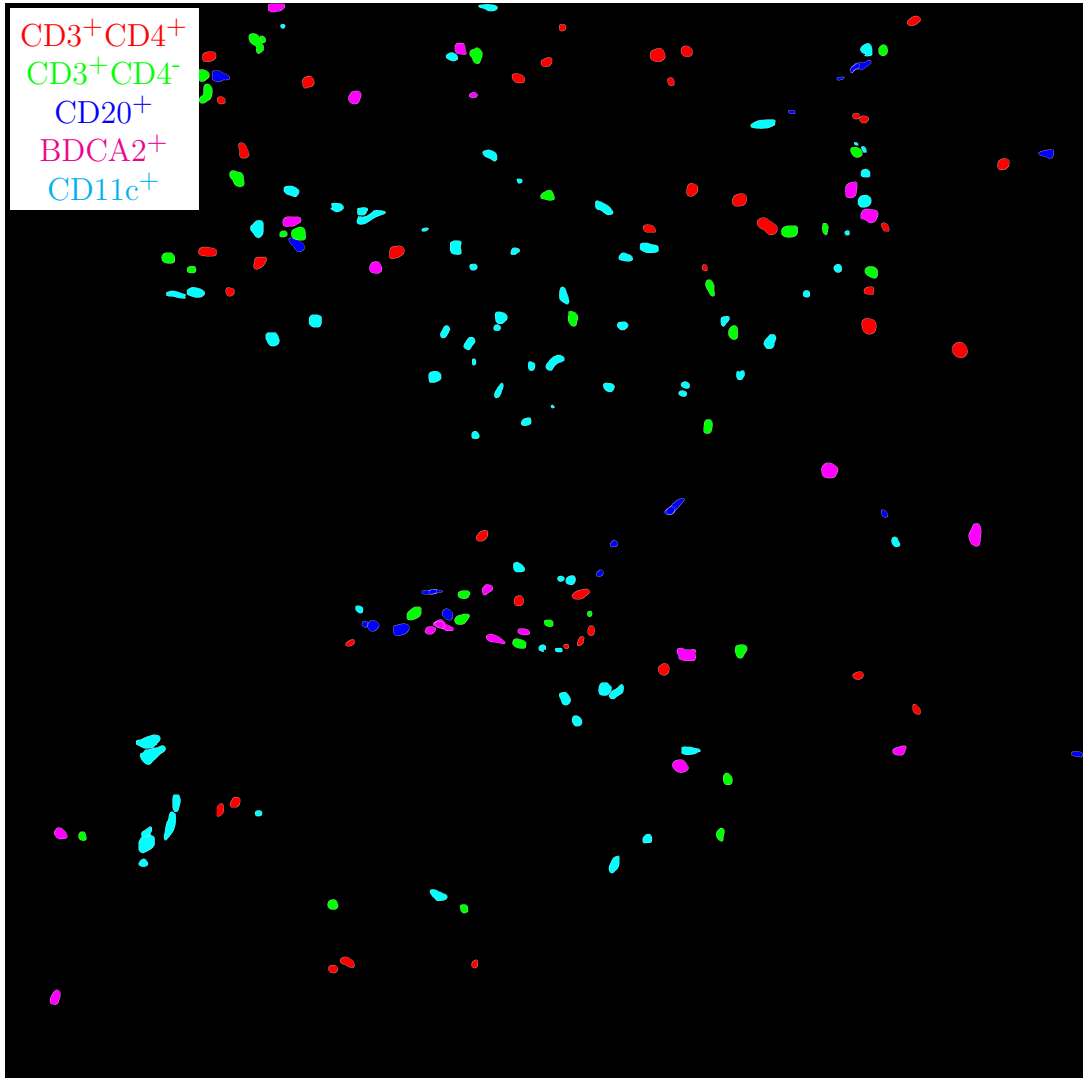

Figure H.9: **Label for H.10**. True label matrix size is 4699 × 4707. Depicted label matrix size is 4699 × 4707.

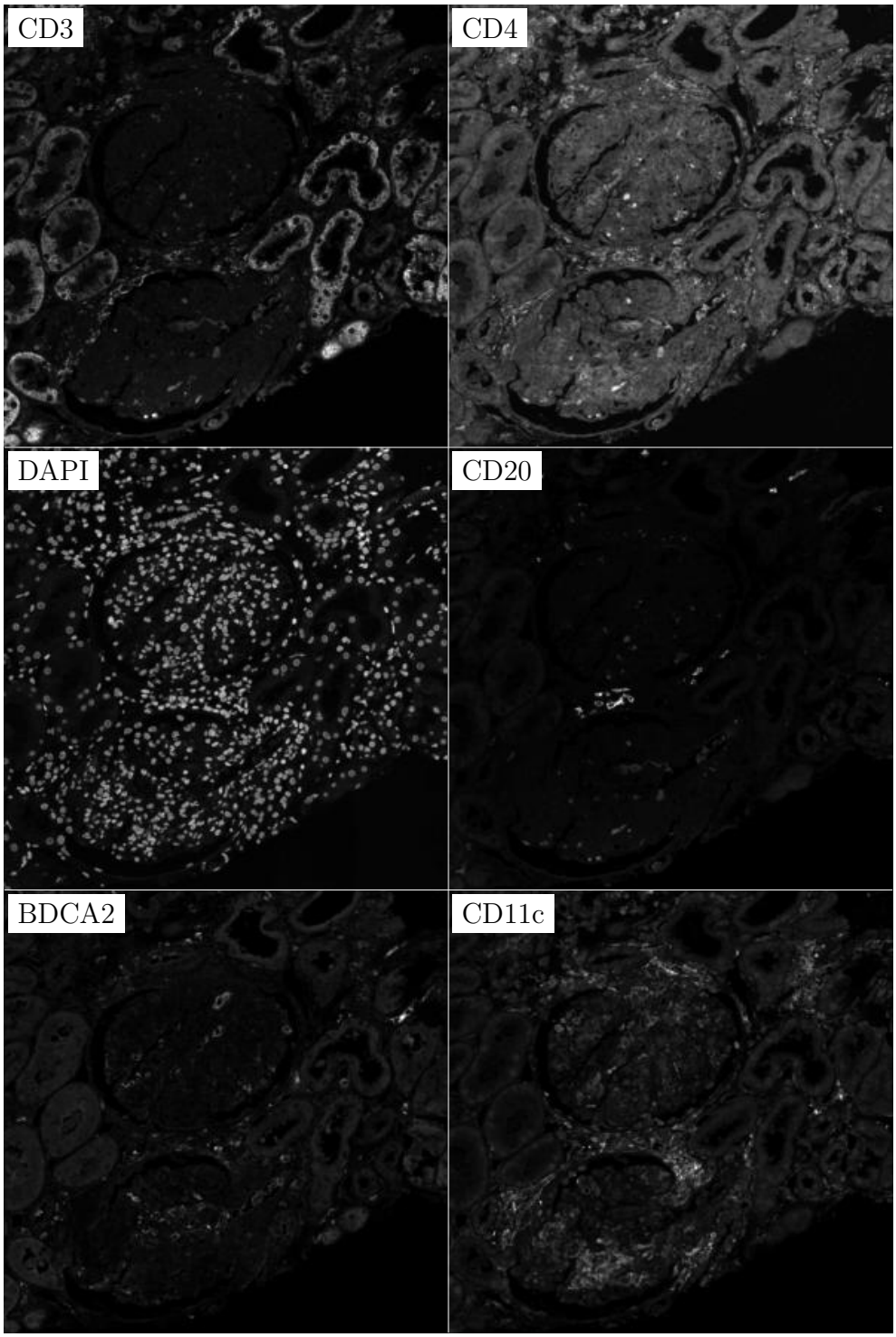

Figure H.10: **ROI for H.9**. True image channel matrix size is 4699×4707. Depicted image channel matrix size is 299×300.

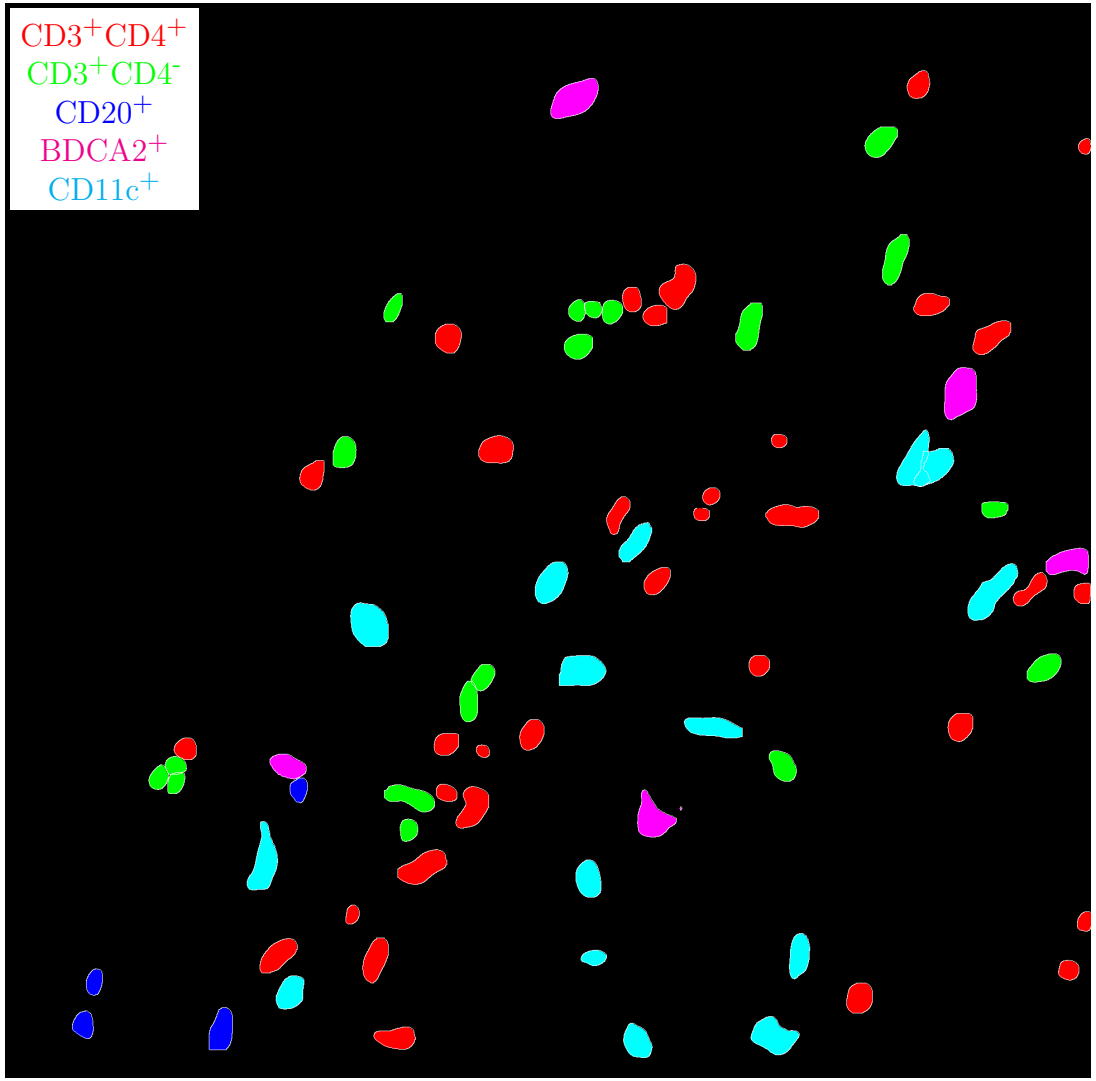

Figure H.11: **Label for H.12**. True label matrix size is 1930 × 1945. Depicted label matrix size is 1930 × 1945.

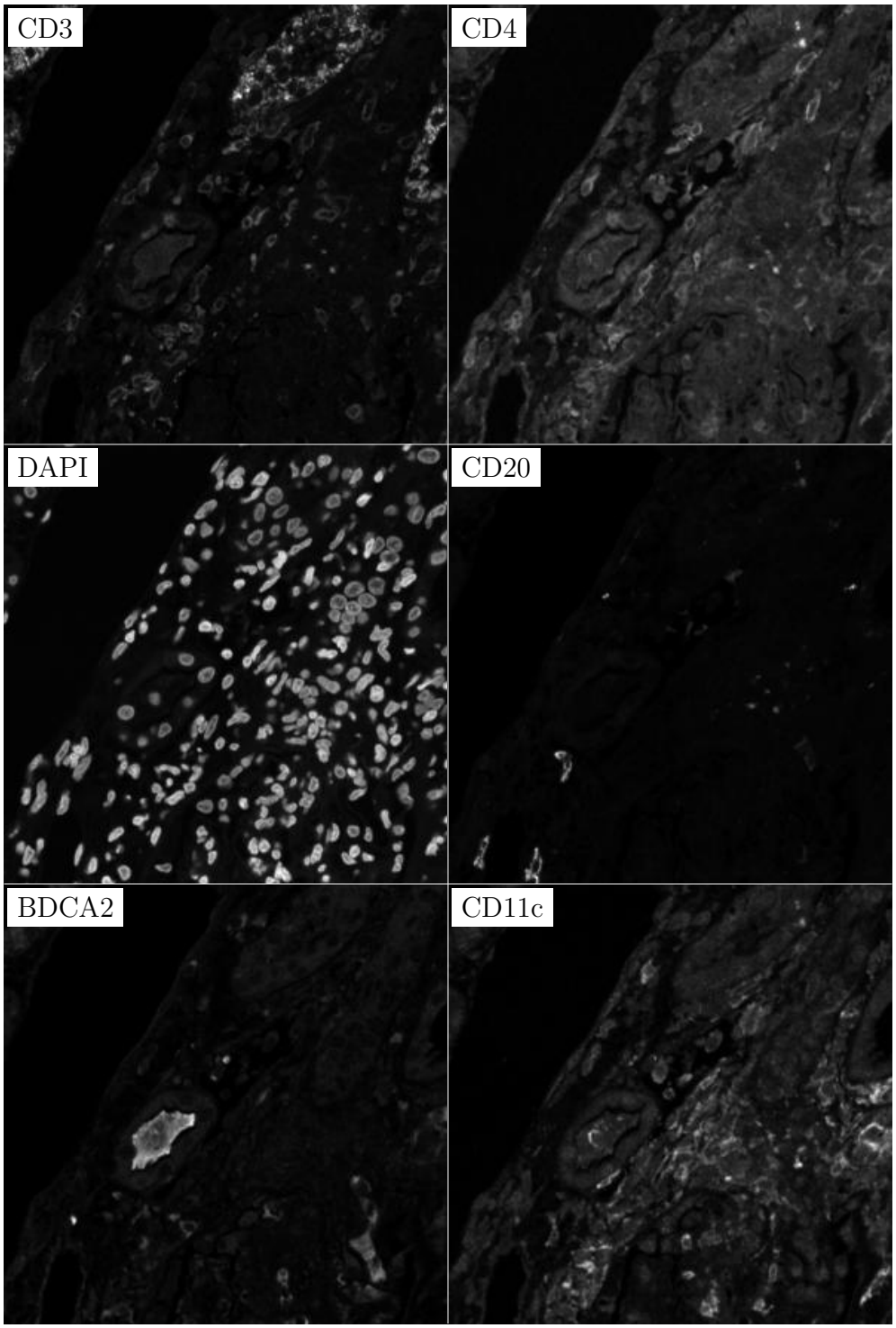

Figure H.12: **ROI for H.11**. True image channel matrix size is 1930 × 1945. Depicted image channel matrix size is 297 × 300.

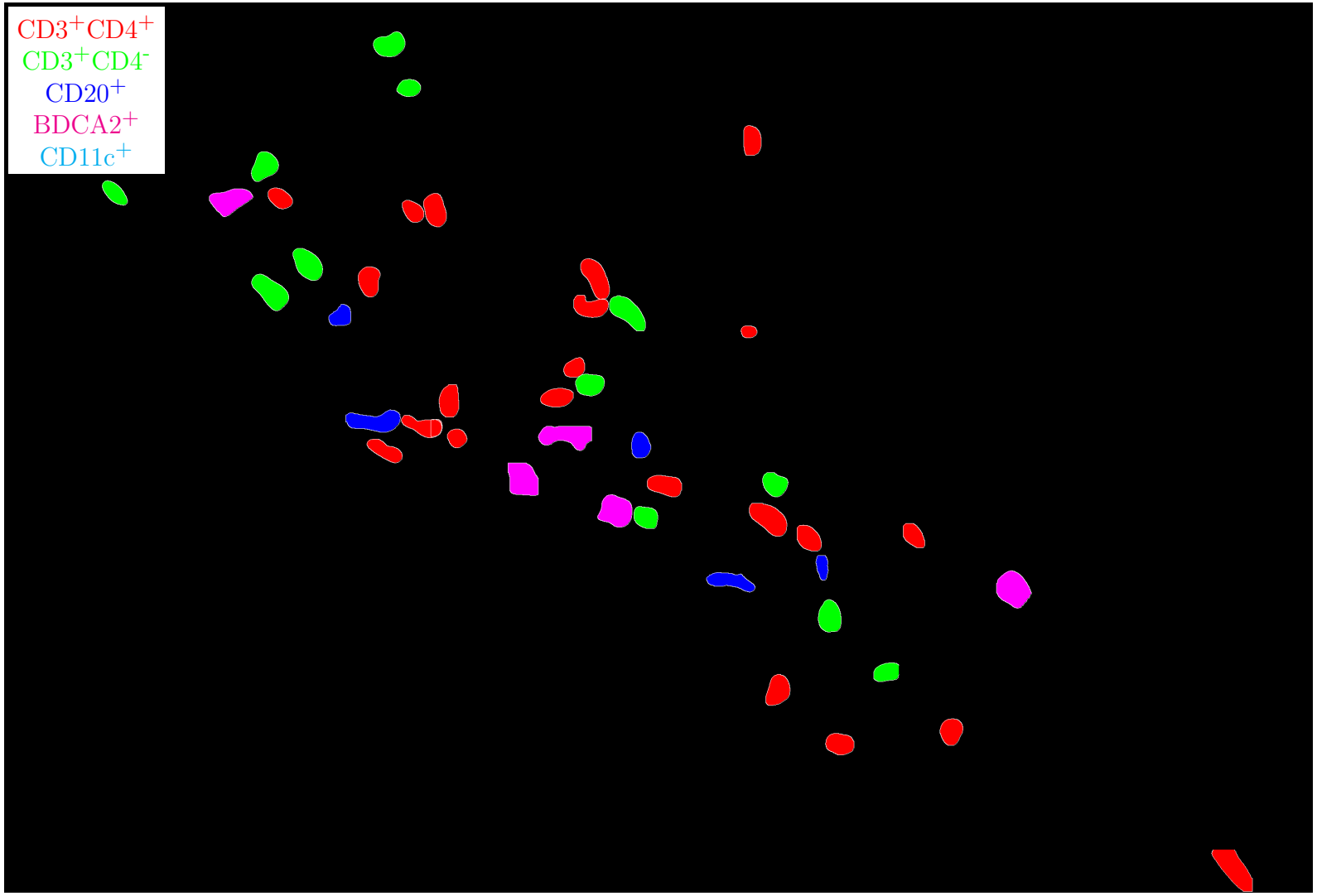

Figure H.13: **Label for H.14**. True label matrix size is 1942 × 2848. Depicted label matrix size is 1942 × 2848.

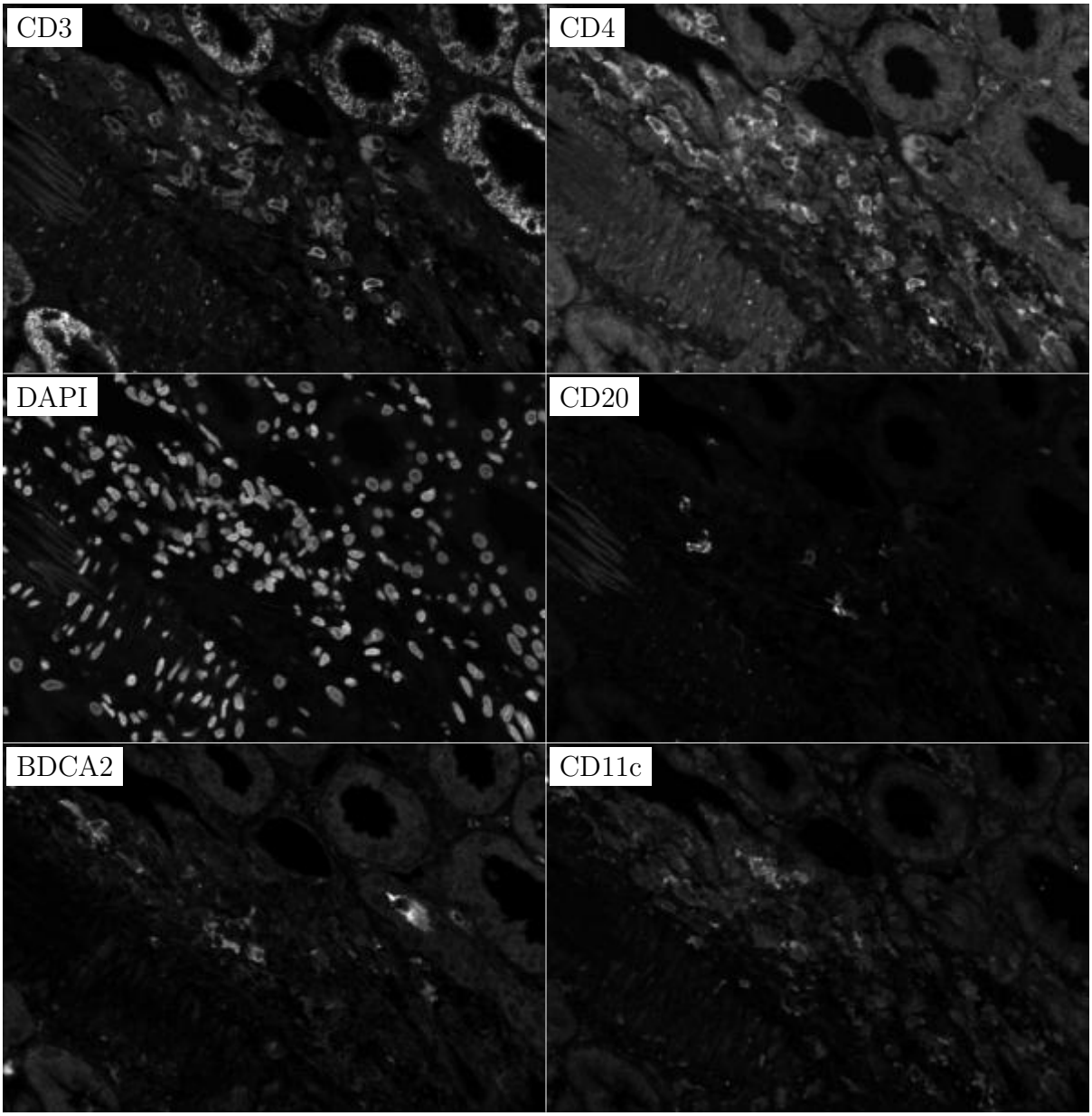

Figure H.14: **ROI for H.13**. True image channel matrix size is 1942×2848. Depicted image channel matrix size is 204×300.

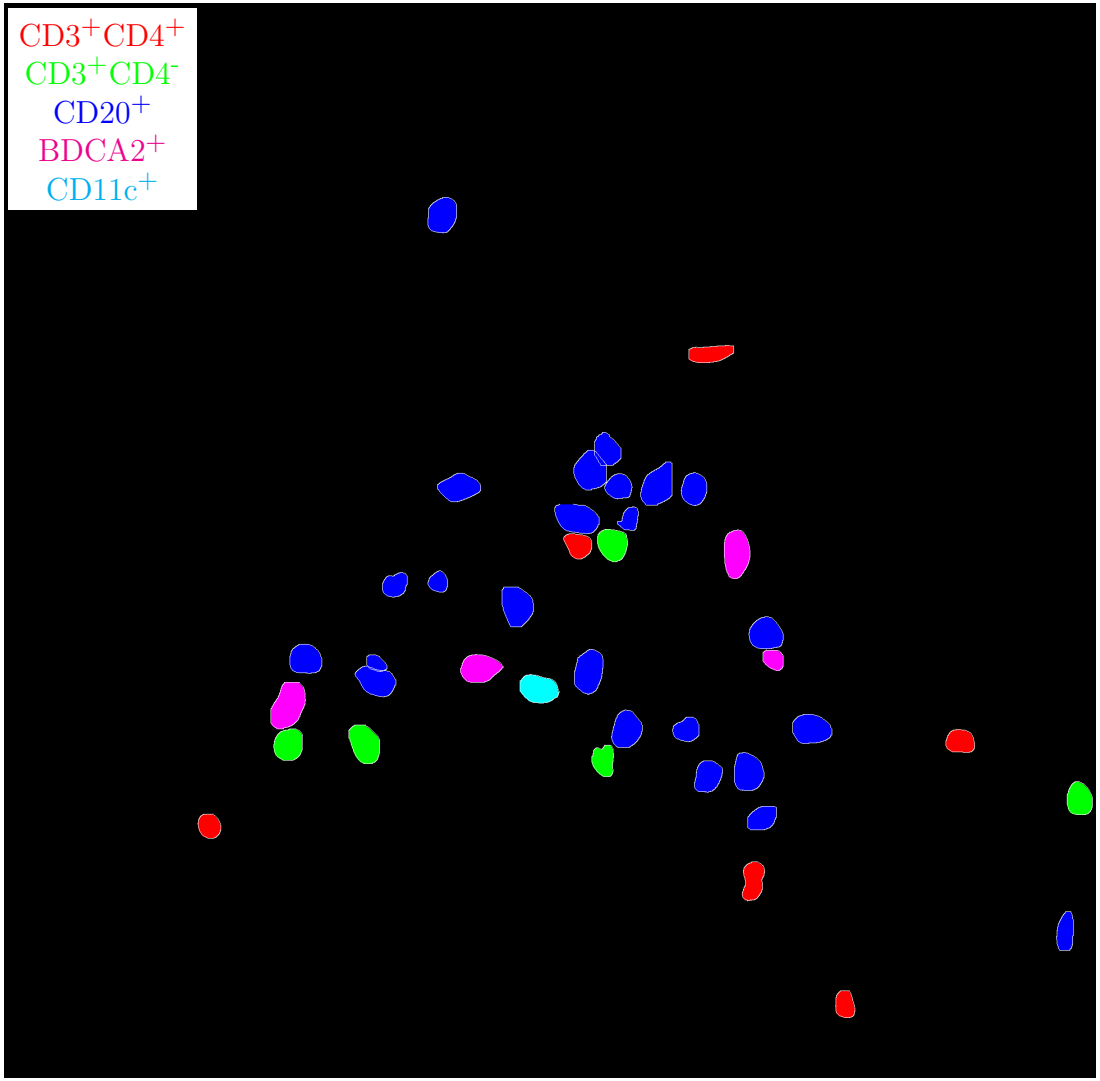

Figure H.15: **Label for H.16**. True label matrix size is 1917 × 1946. Depicted label matrix size is 1917 × 1946.

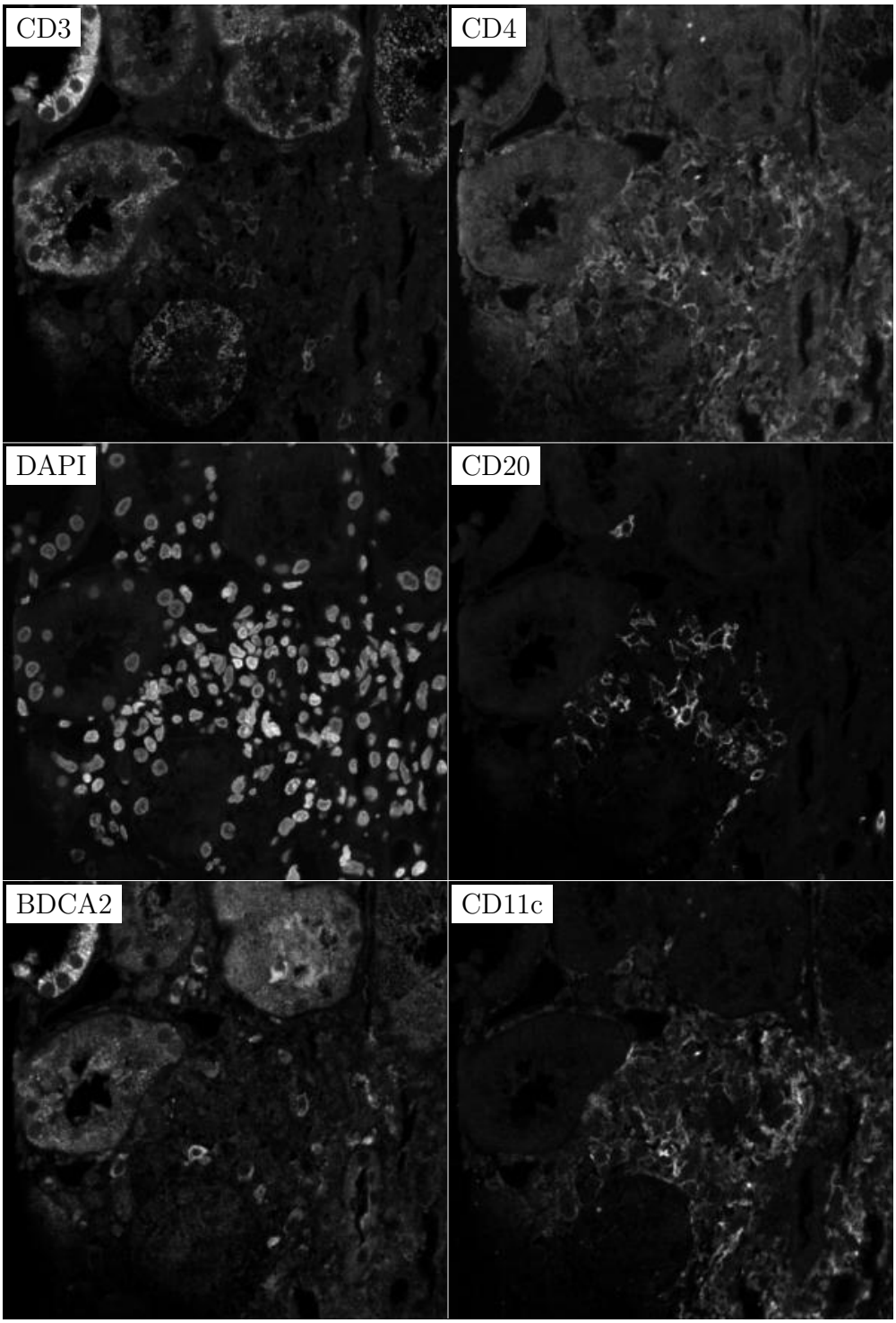

Figure H.16: **ROI for H.15**. True image channel matrix size is 1917 × 1946. Depicted image channel matrix size is 295 × 300.

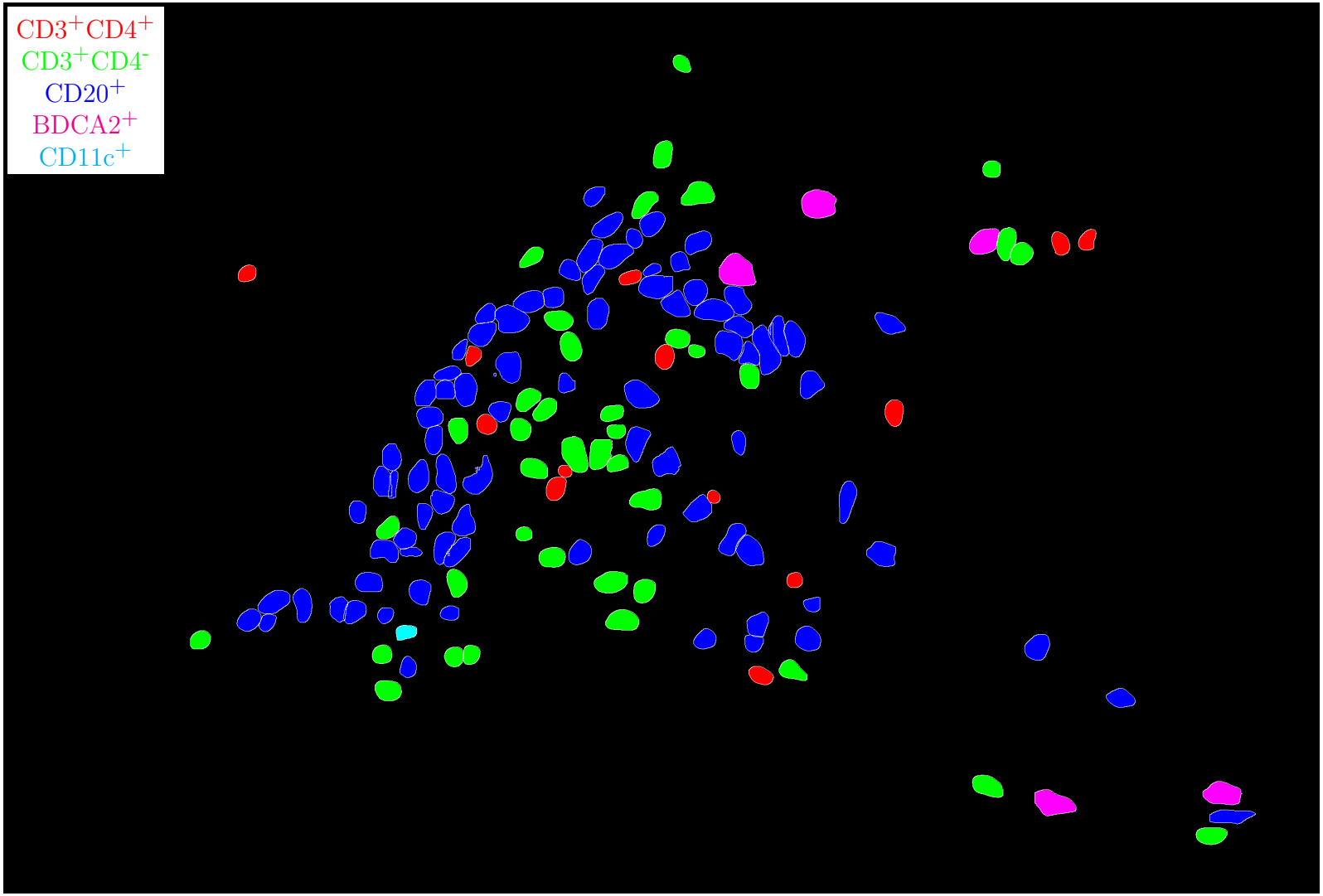

Figure H.17: **Label for H.18**. True label matrix size is 1944 × 2867. Depicted label matrix size is 1944 × 2867.

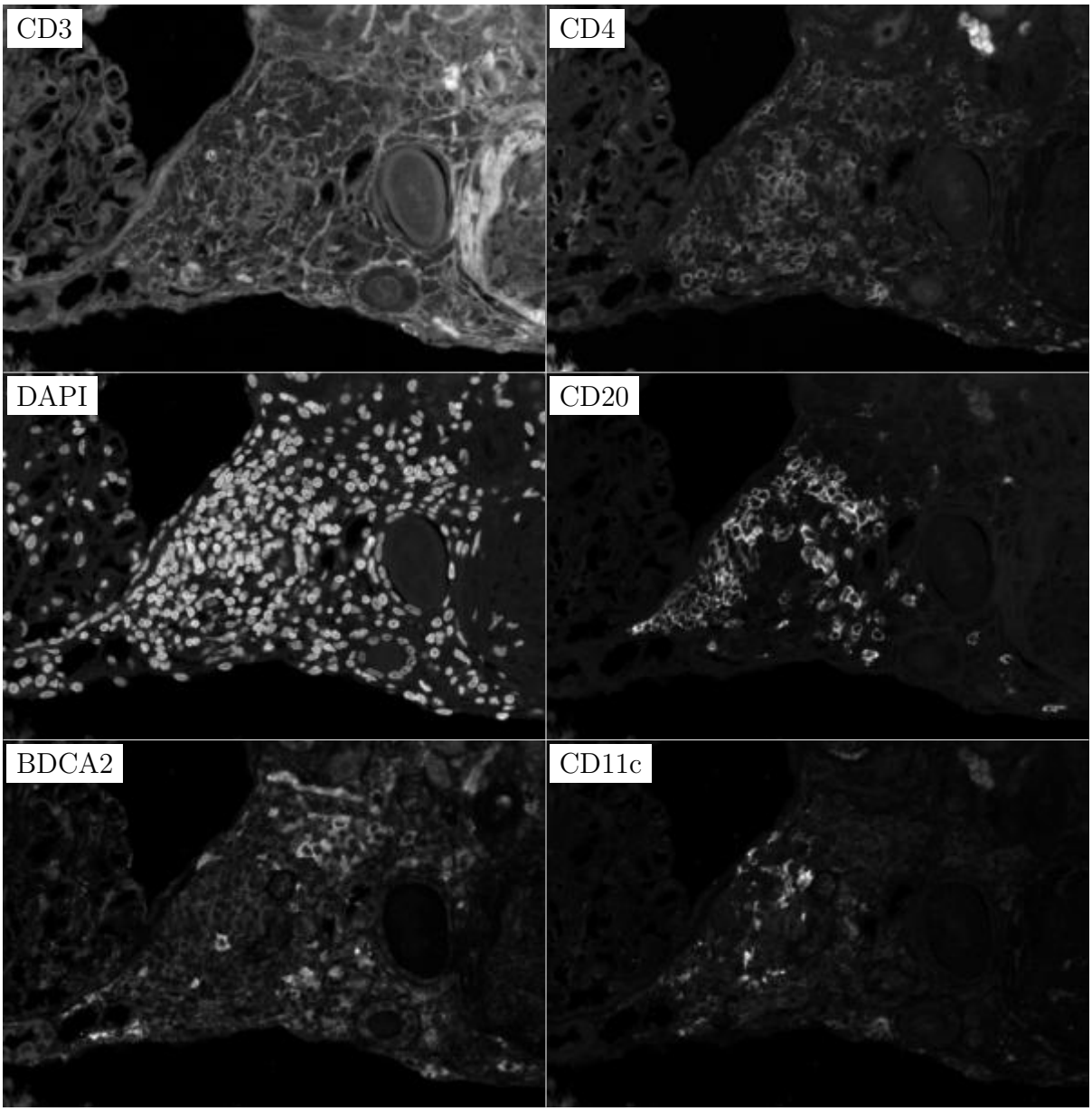

Figure H.18: **ROI for H.17**. True image channel matrix size is 1944×2867. Depicted image channel matrix size is 203×300.

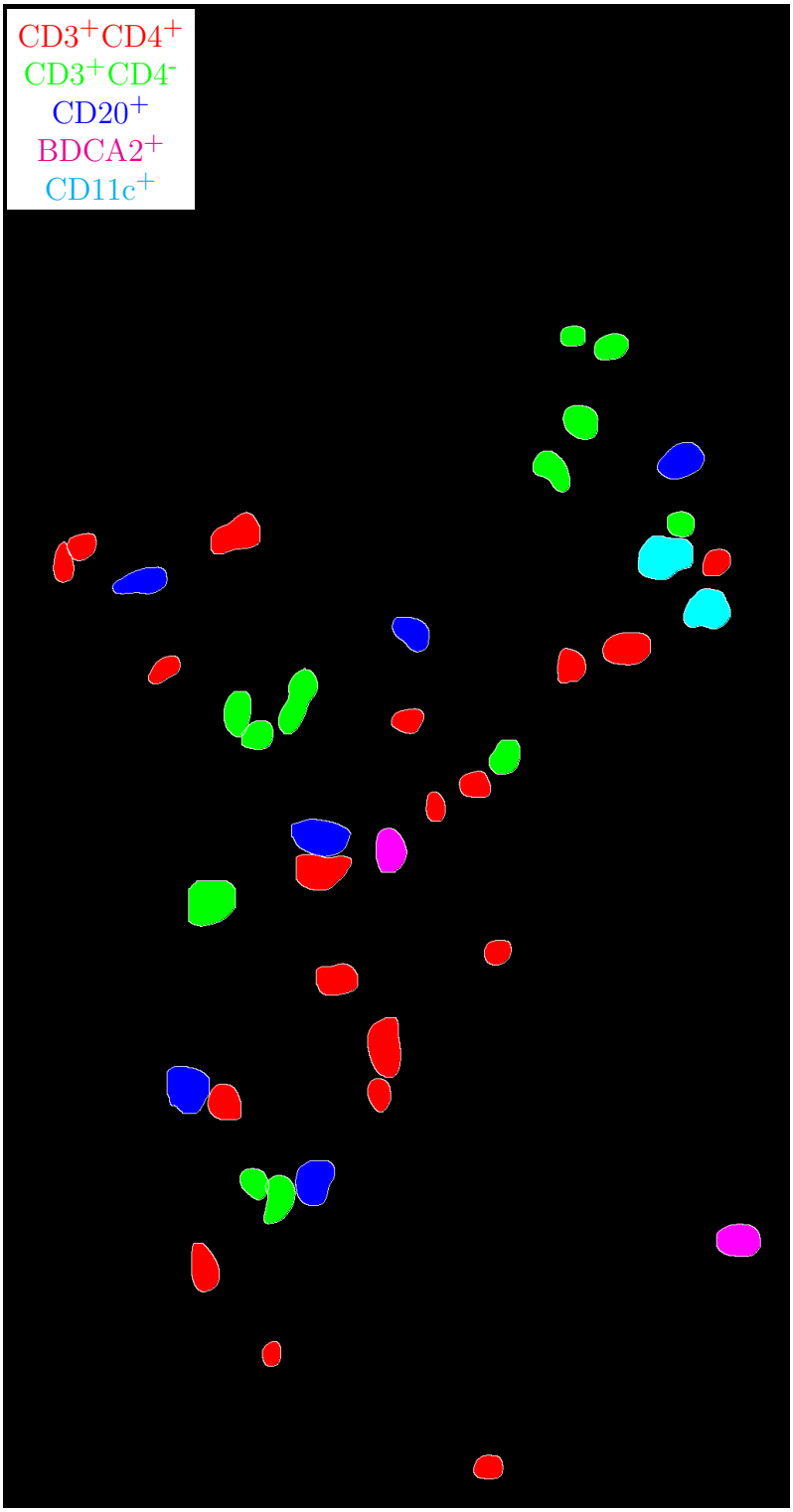

Figure H.19: **Label for H.20**. True label matrix size is 1946×1024. Depicted label matrix size is 1946×1024.

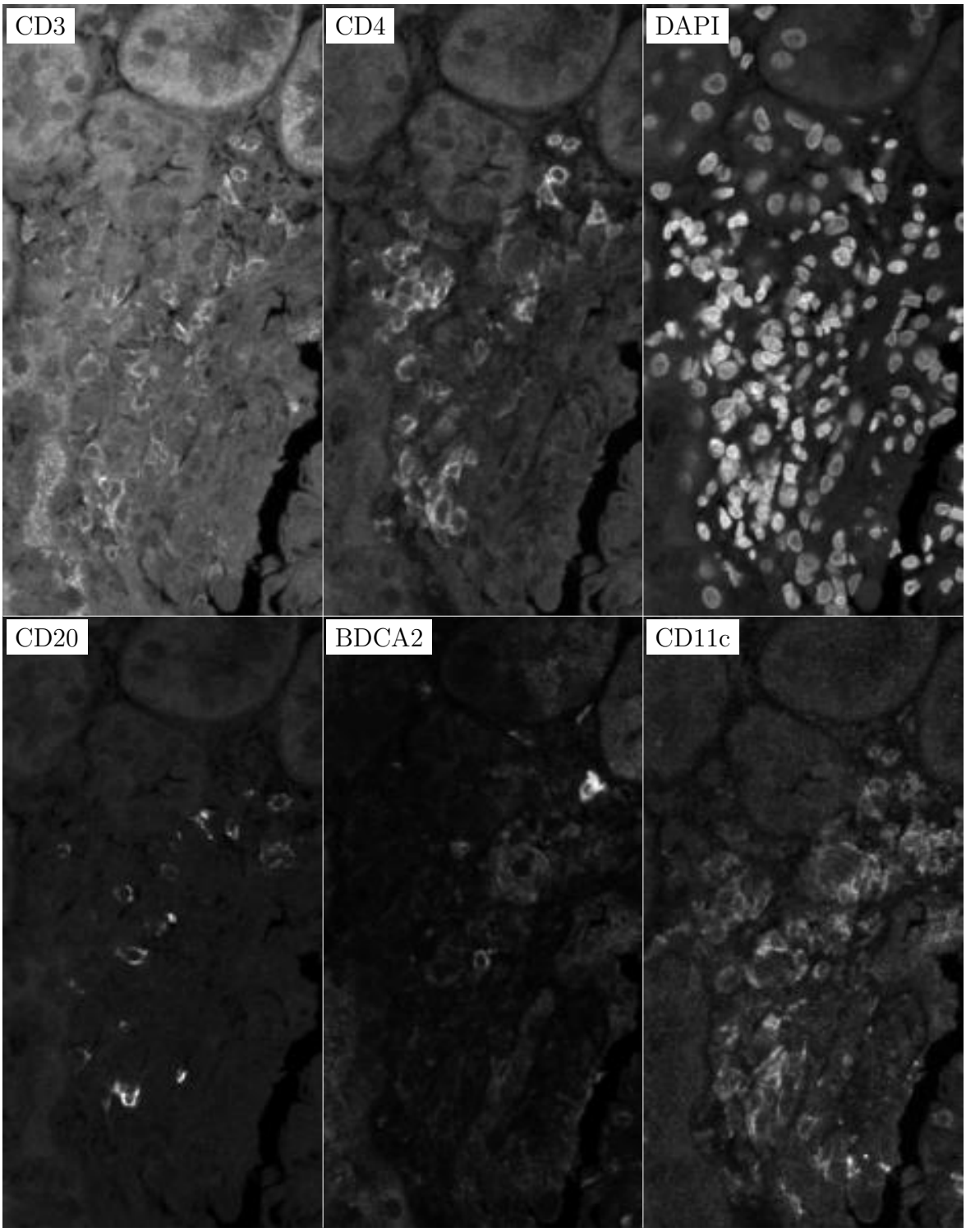

Figure H.20: **ROI for H.19**. True image channel matrix size is 1946 × 1024. Depicted image channel matrix size is 300 × 157.

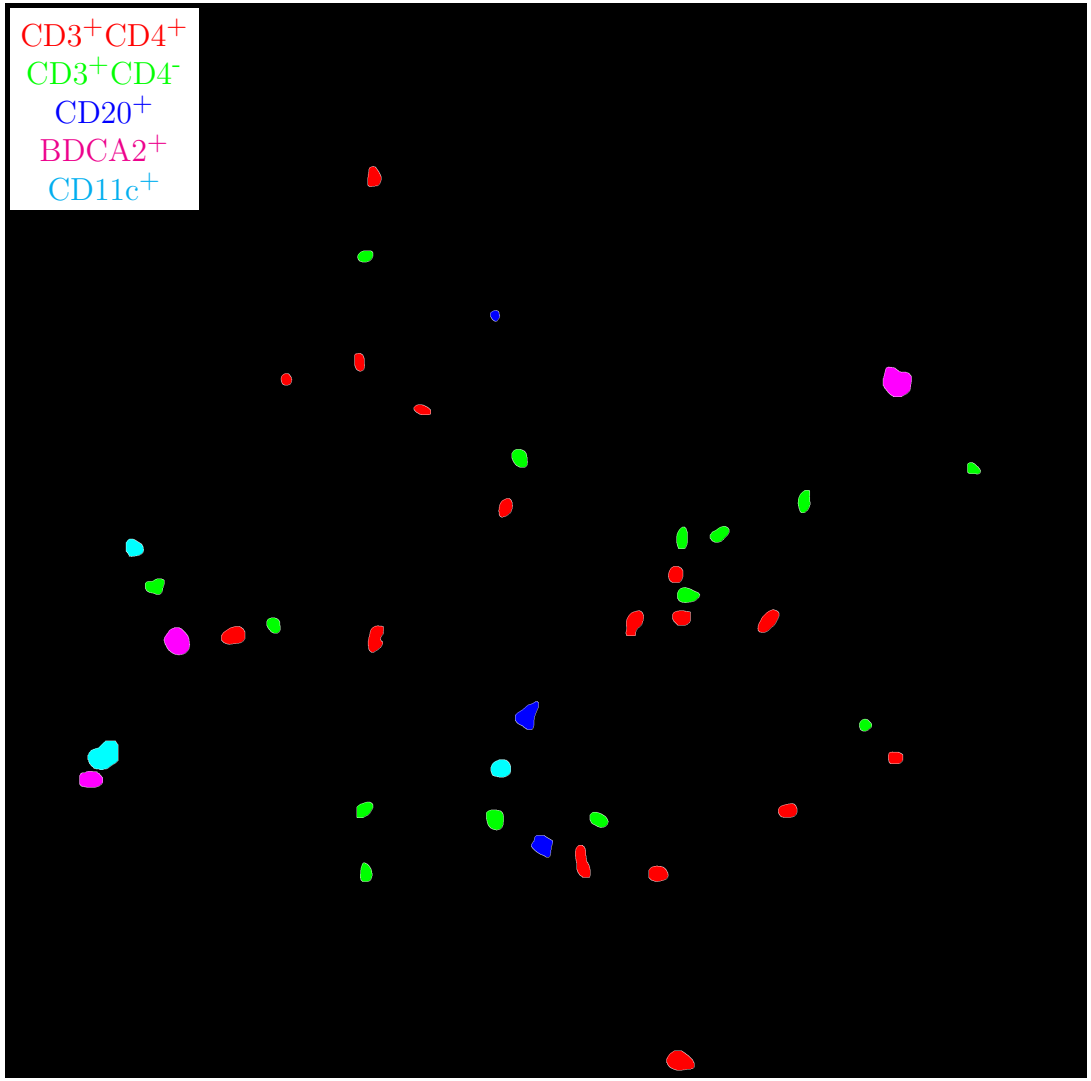

Figure H.21: **Label for H.22**. True label matrix size is 2865 × 2883. Depicted label matrix size is 2865 × 2883.

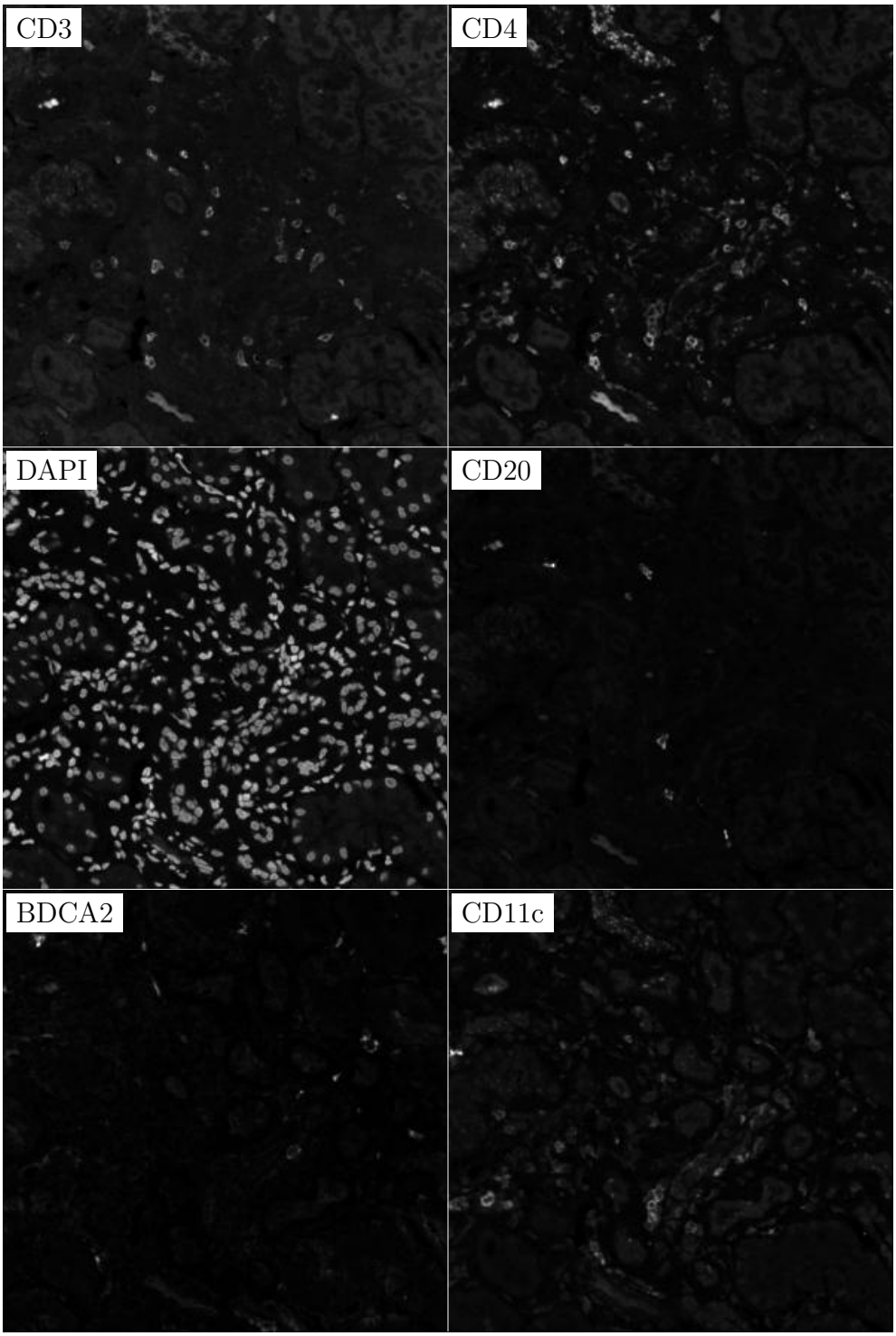

Figure H.22: **ROI for H.21**. True image channel matrix size is 2865 × 2883. Depicted image channel matrix size is 298 × 300.

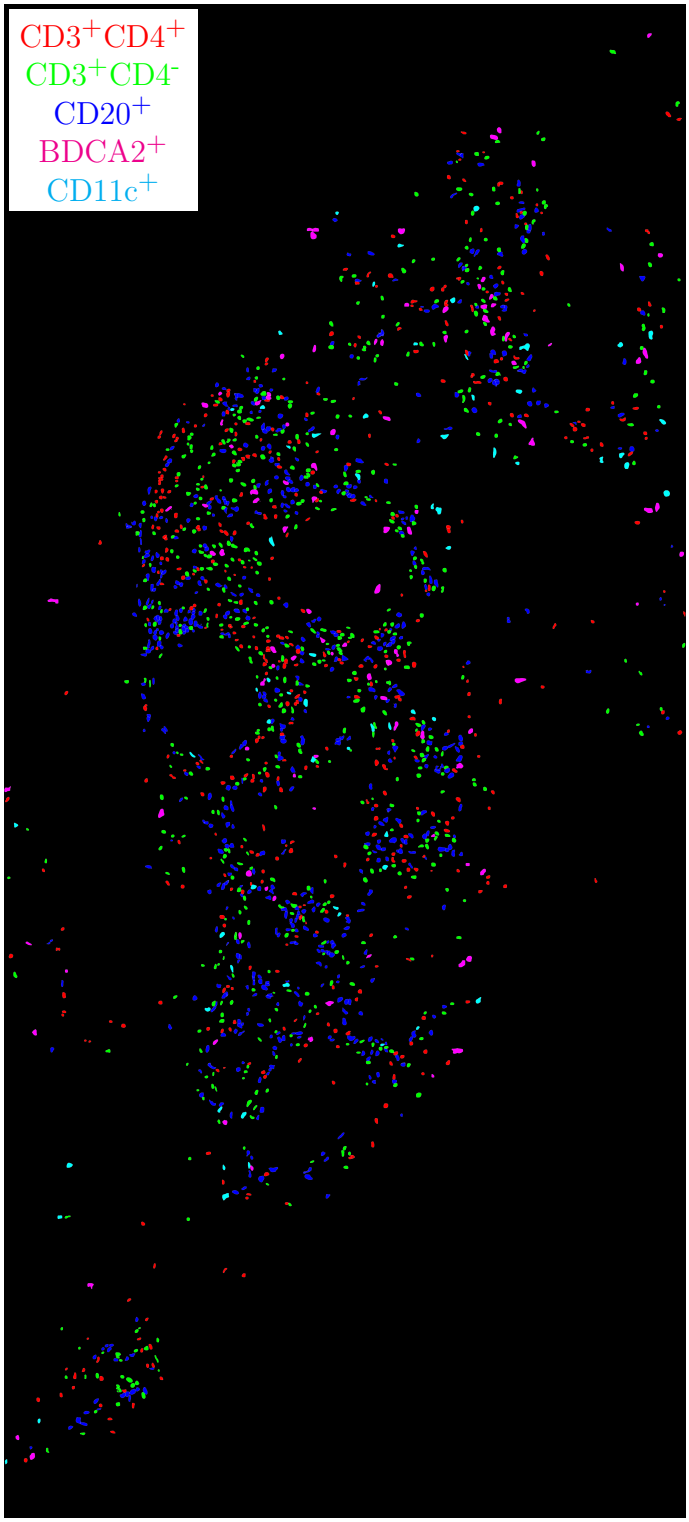

Figure H.23: **Label for H.24**. True label matrix size is 18488×8364. Depicted label matrix size is 10000×4524.

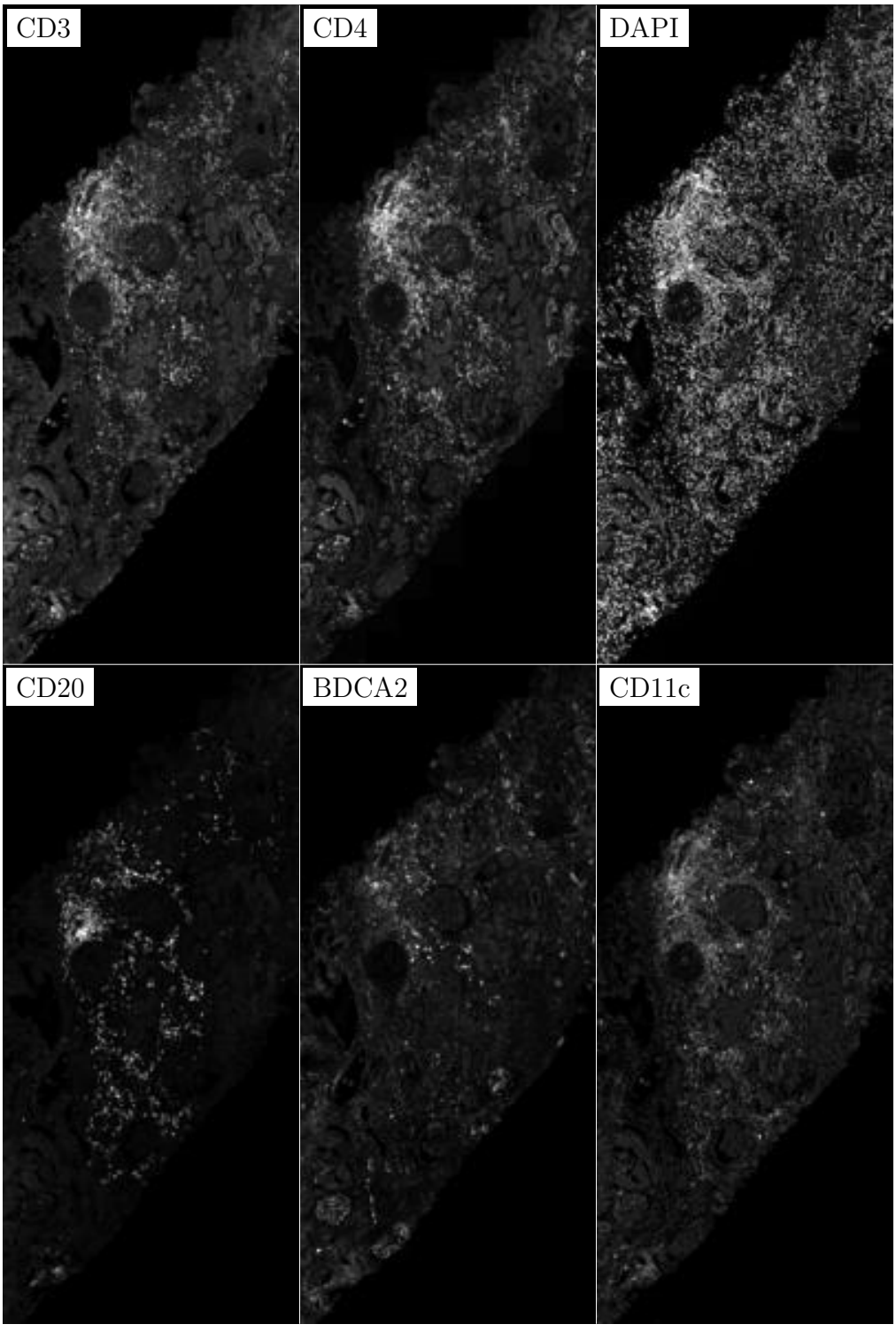

Figure H.24: **ROI for H.23**. True image channel matrix size is 18488 × 8364. Depicted image channel matrix size is 300 × 135.

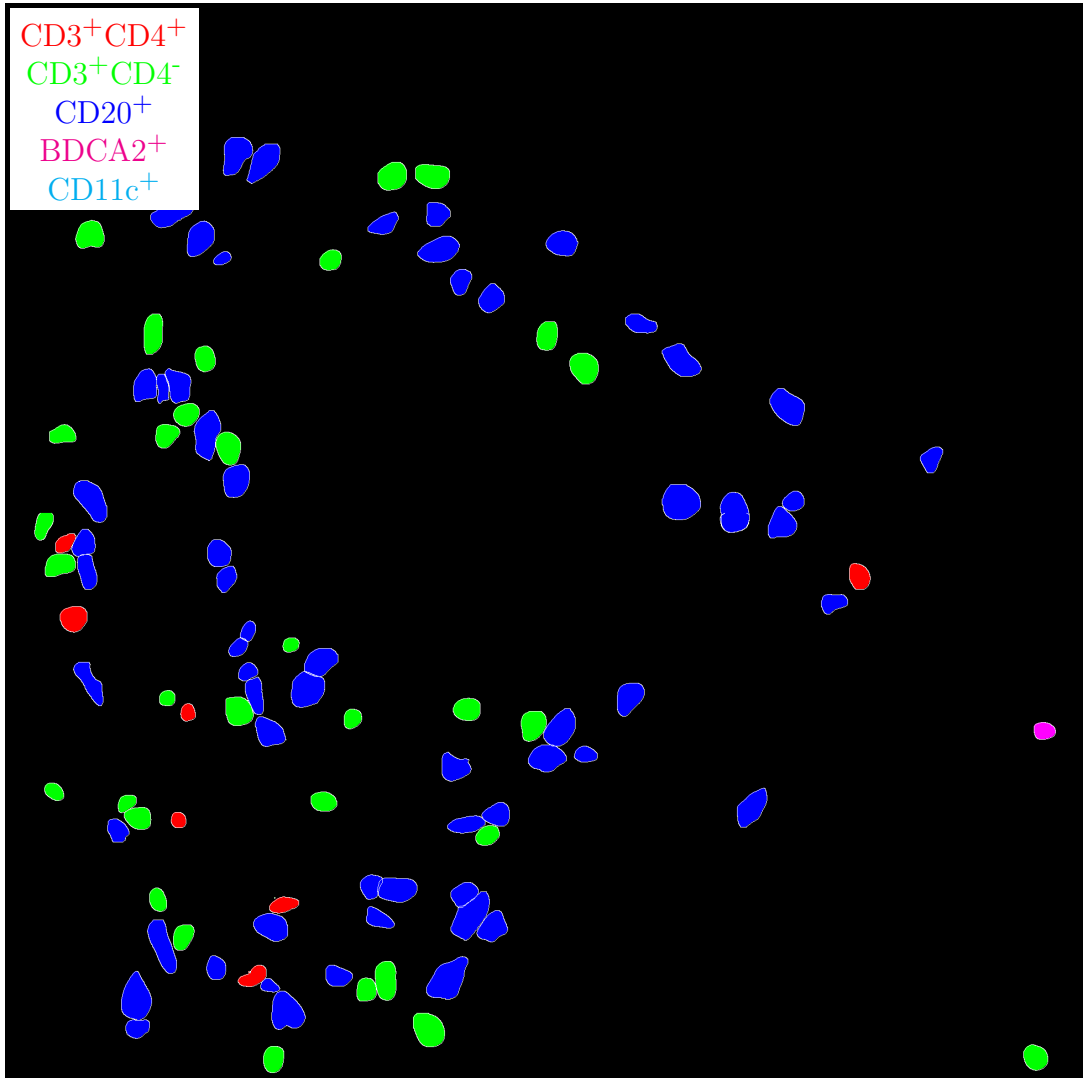

Figure H.25: **Label for H.26**. True label matrix size is 1948 × 1950. Depicted label matrix size is 1948 × 1950.

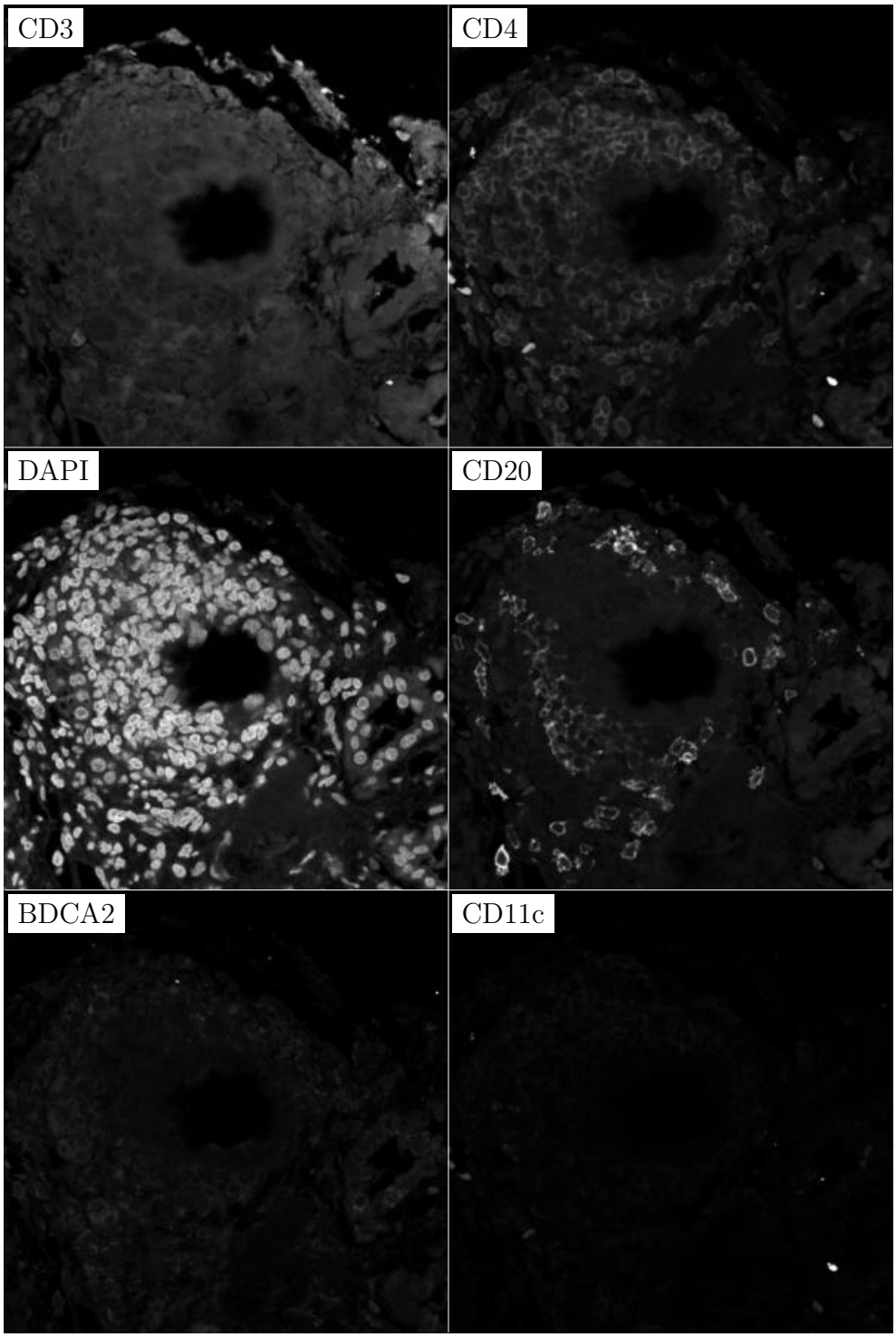

Figure H.26: **ROI for H.25**. True image channel matrix size is 1948 × 1950. Depicted image channel matrix size is 299 × 300.

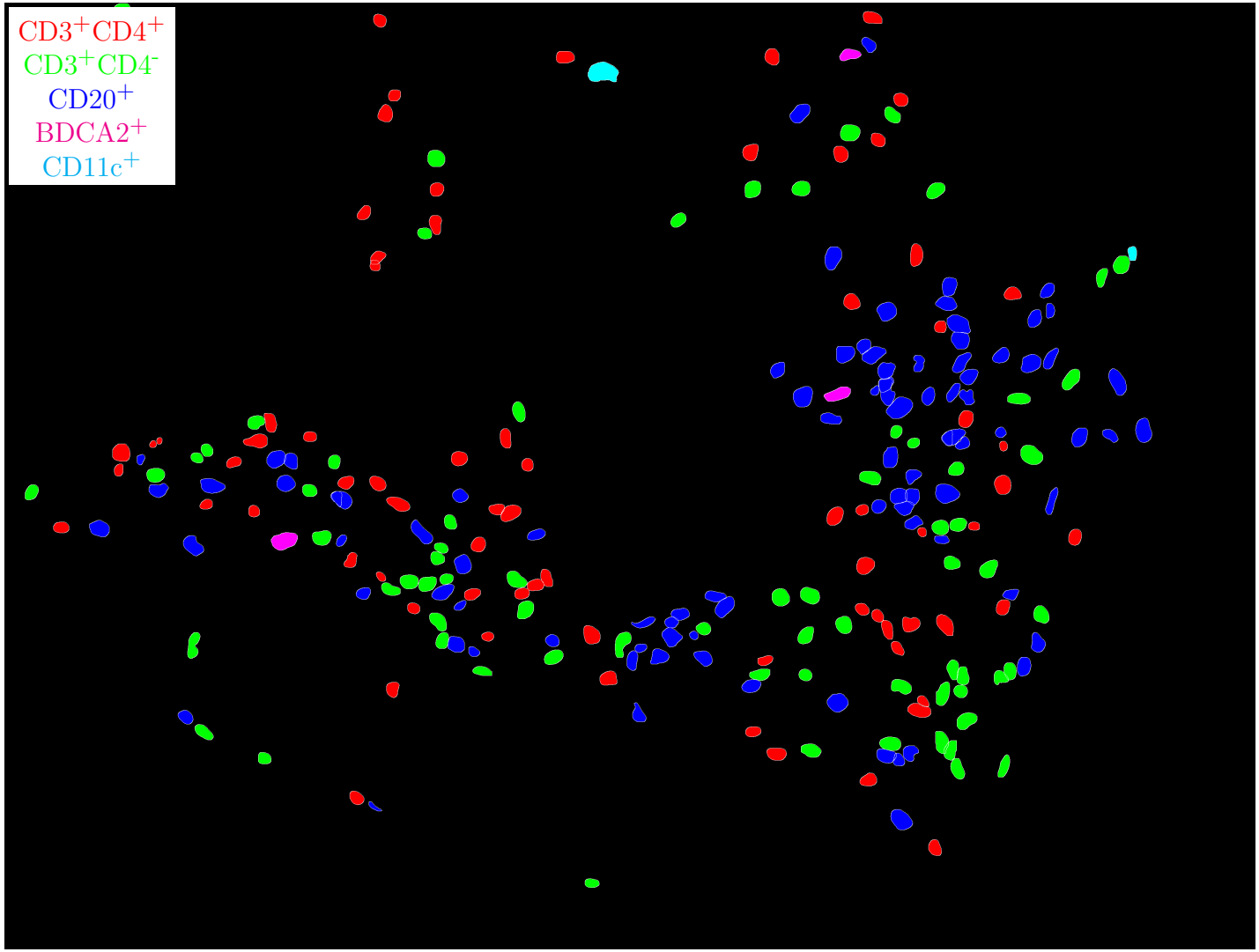

Figure H.27: **Label for H.28**. True label matrix size is 2872 × 3787. Depicted label matrix size is 2872 × 3787.

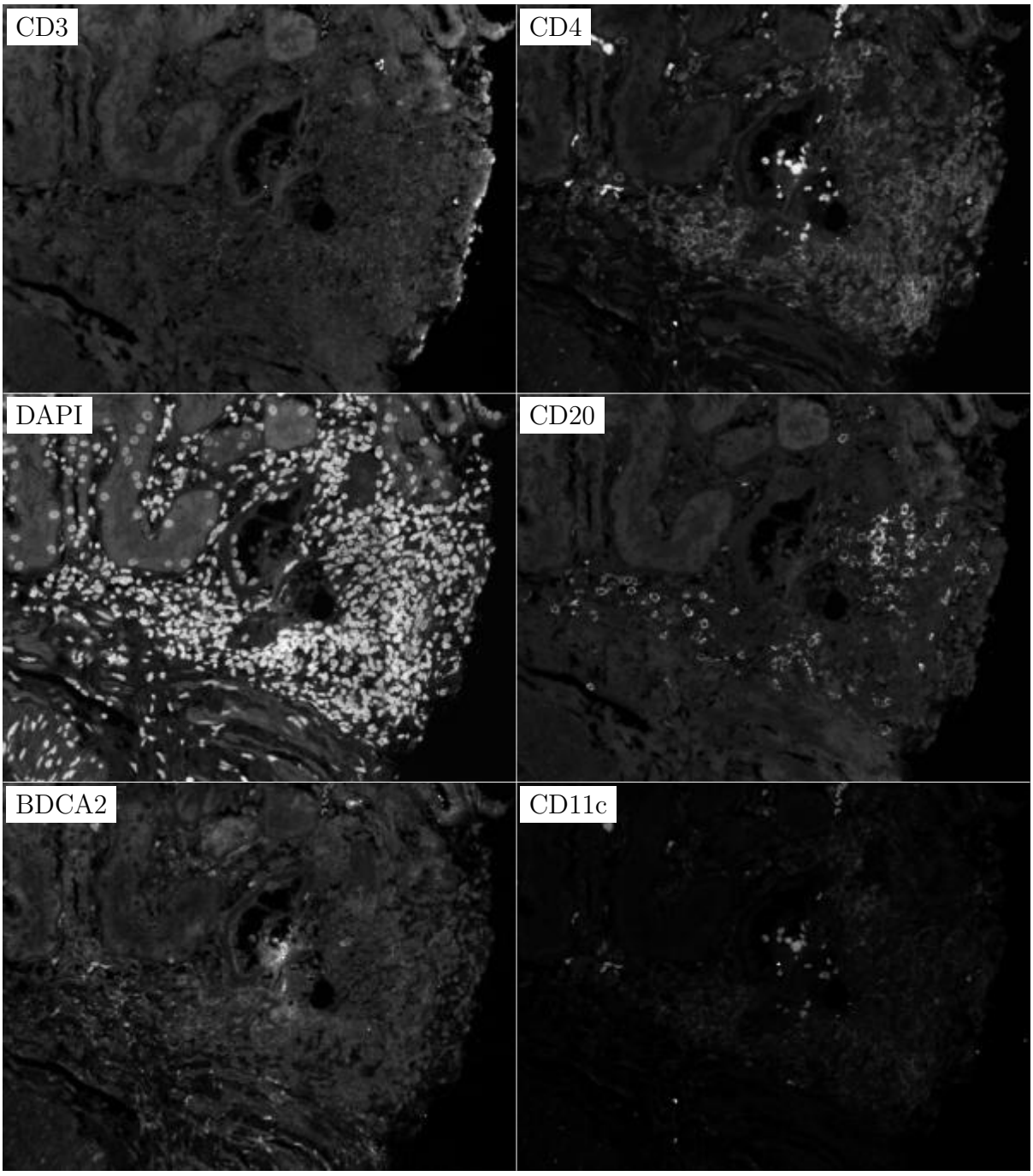

Figure H.28: **ROI for H.27**. True image channel matrix size is 2872×3787. Depicted image channel matrix size is 227×300.

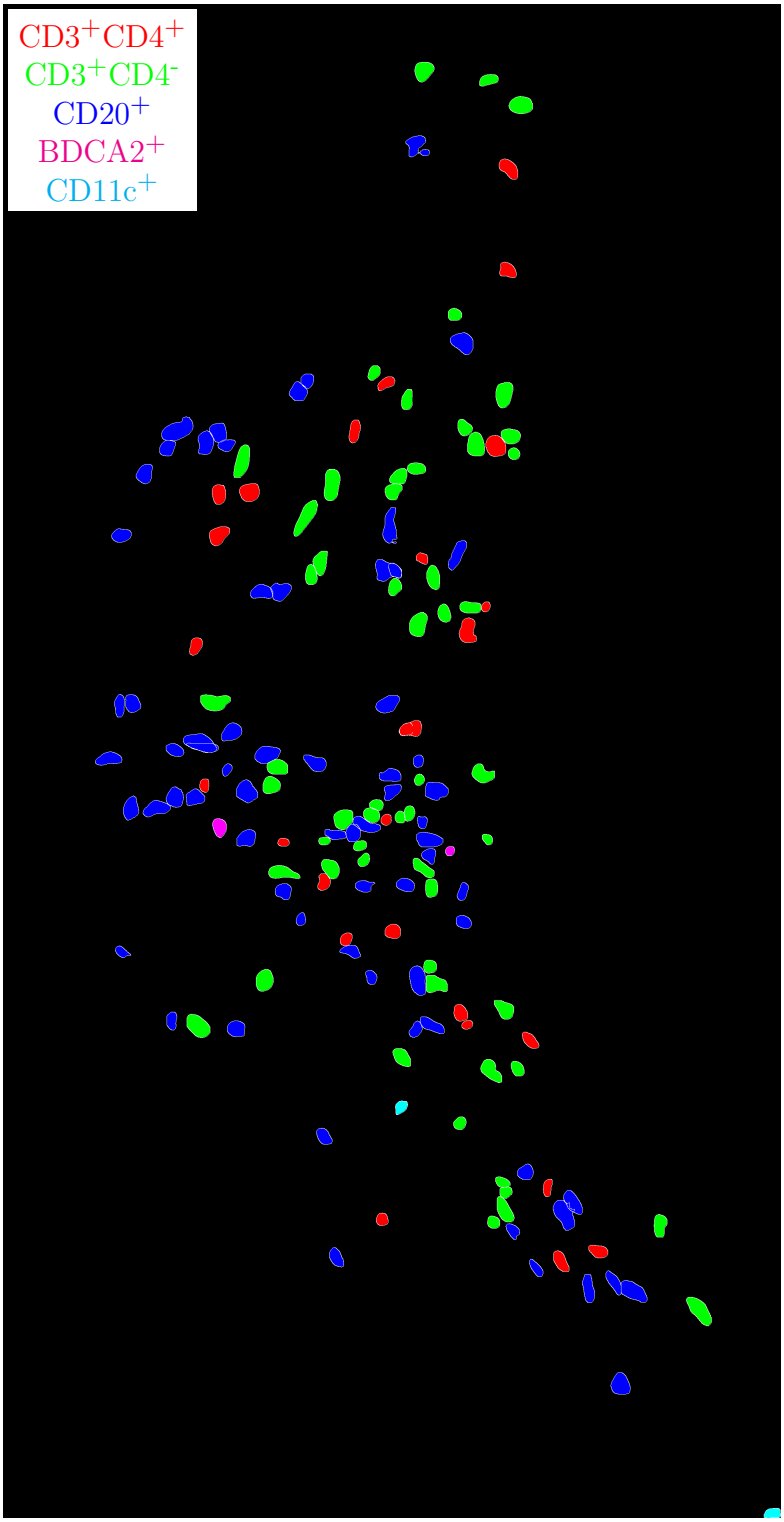

Figure H.29: **Label for H.30**. True label matrix size is 3787×1944. Depicted label matrix size is 3787×1944.

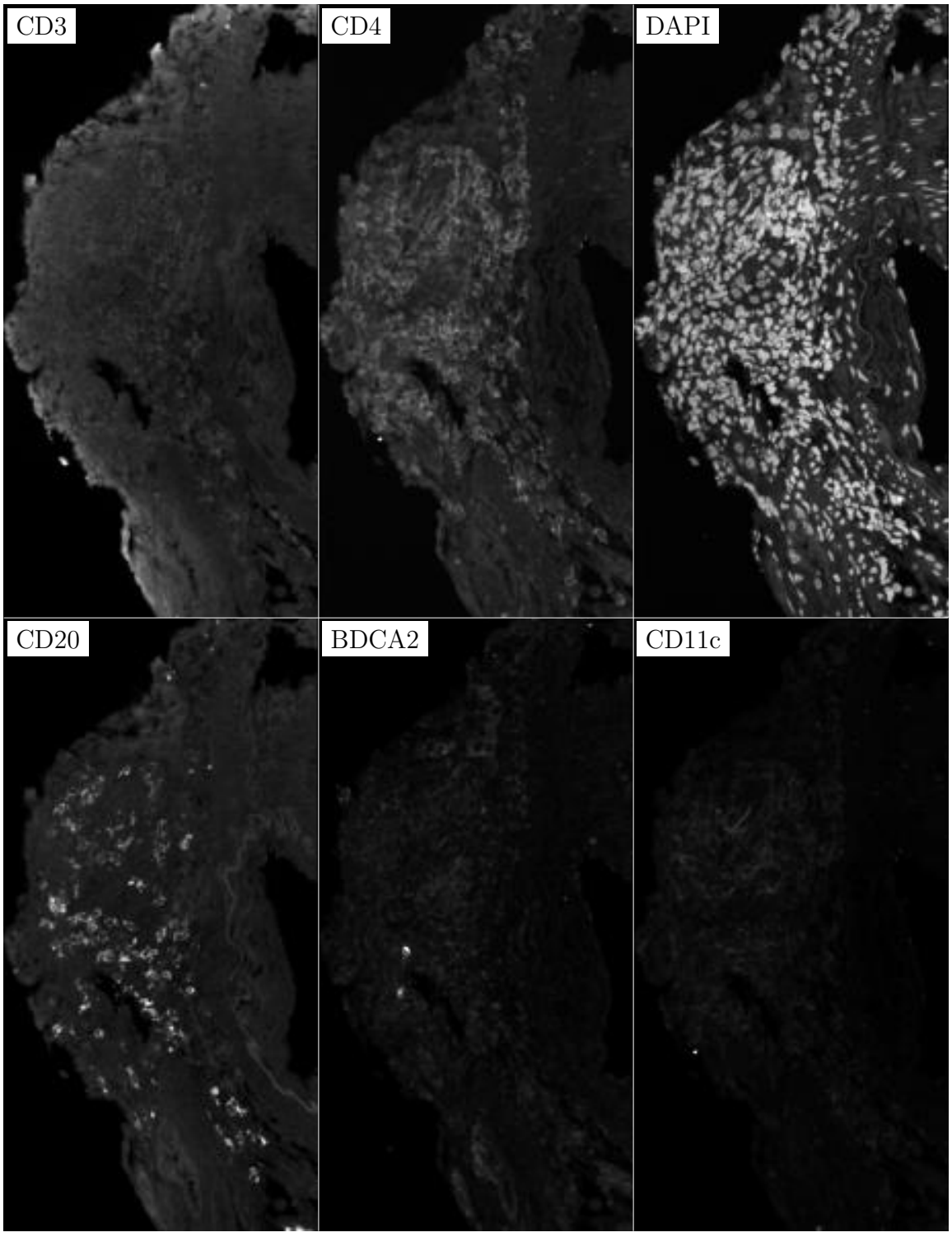

Figure H.30: **ROI for H.29**. True image channel matrix size is 3787 × 1944. Depicted image channel matrix size is 300 × 154.

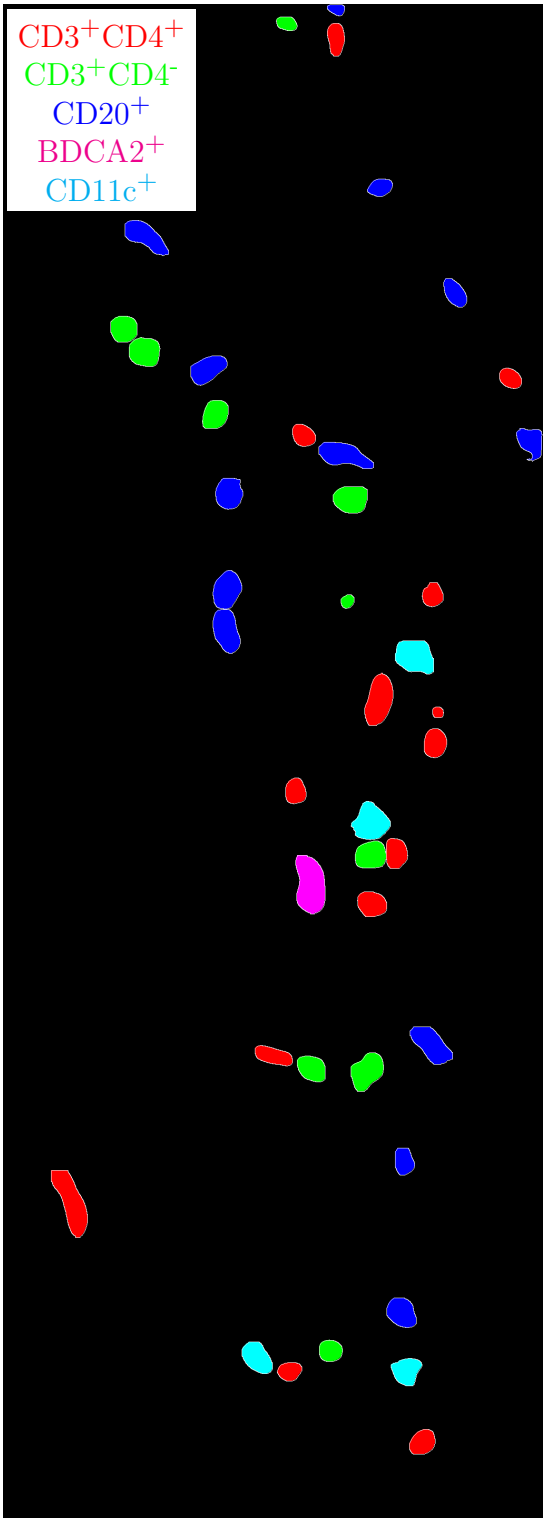

Figure H.31: **Label for H.32**. True label matrix size is 2859×1025. Depicted label matrix size is 2859×1025.

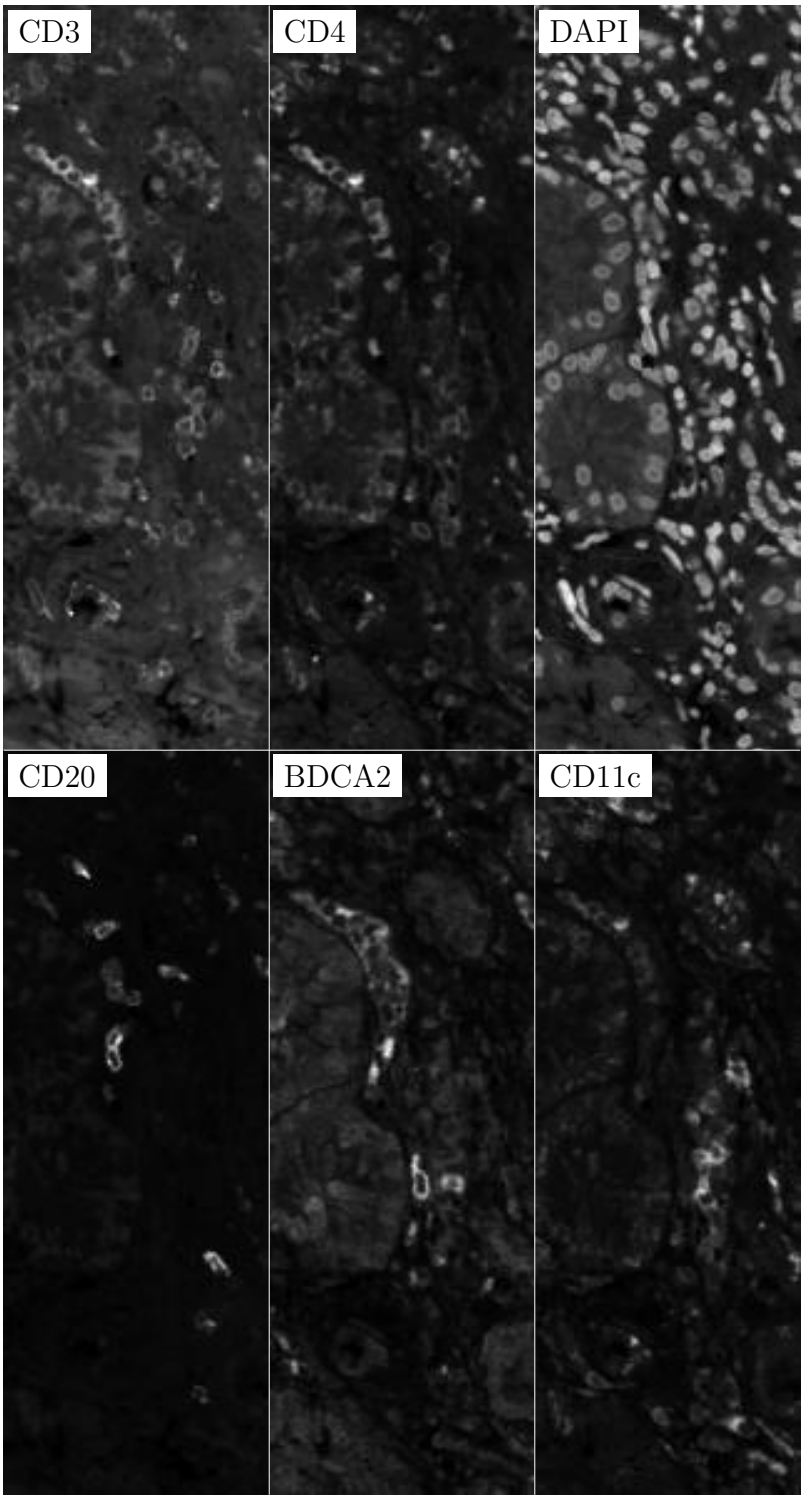

Figure H.32: **ROI for H.31**. True image channel matrix size is 2859 × 1025. Depicted image channel matrix size is 300 × 107.

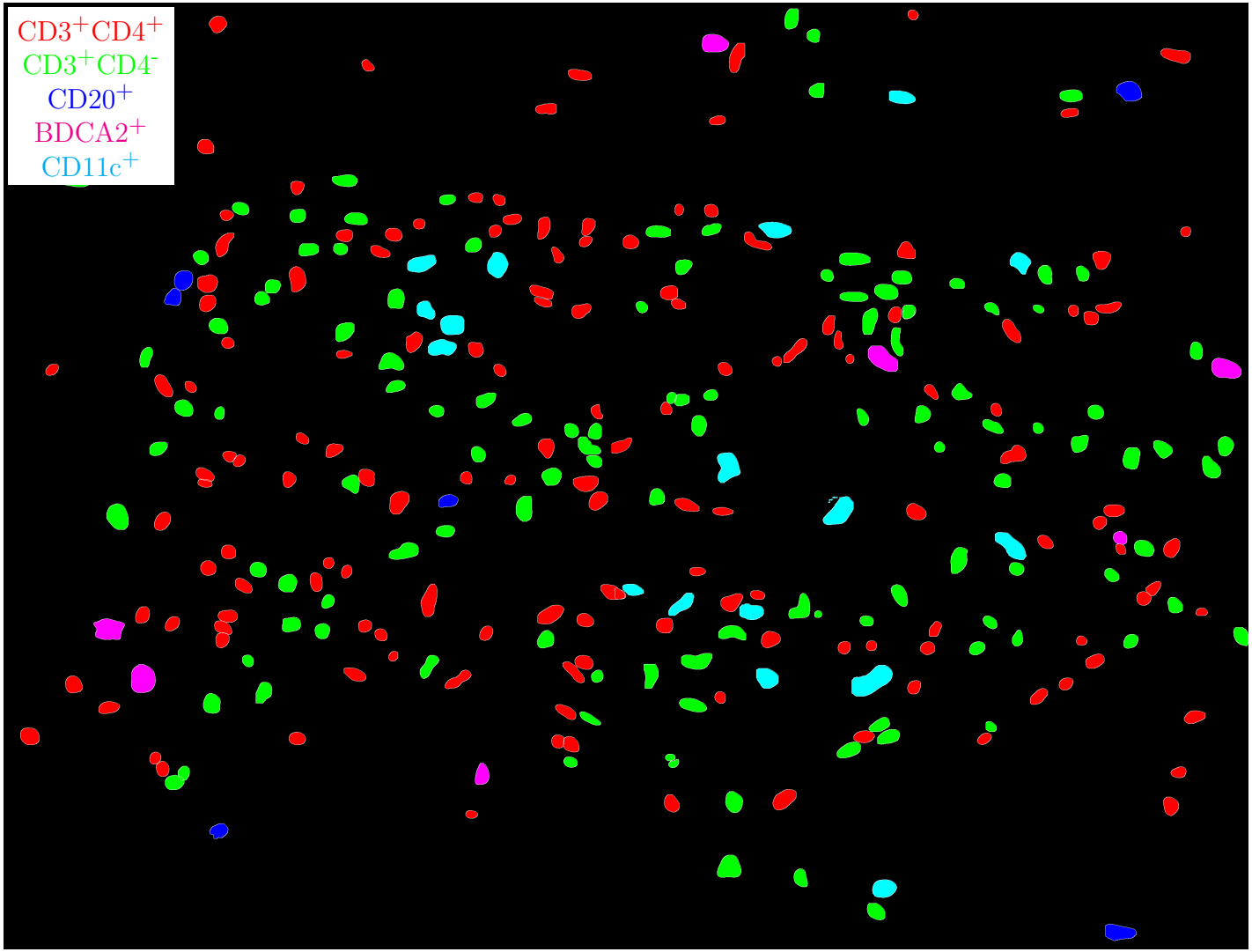

Figure H.33: **Label for H.34**. True label matrix size is 2886 × 3786. Depicted label matrix size is 2886 × 3786.

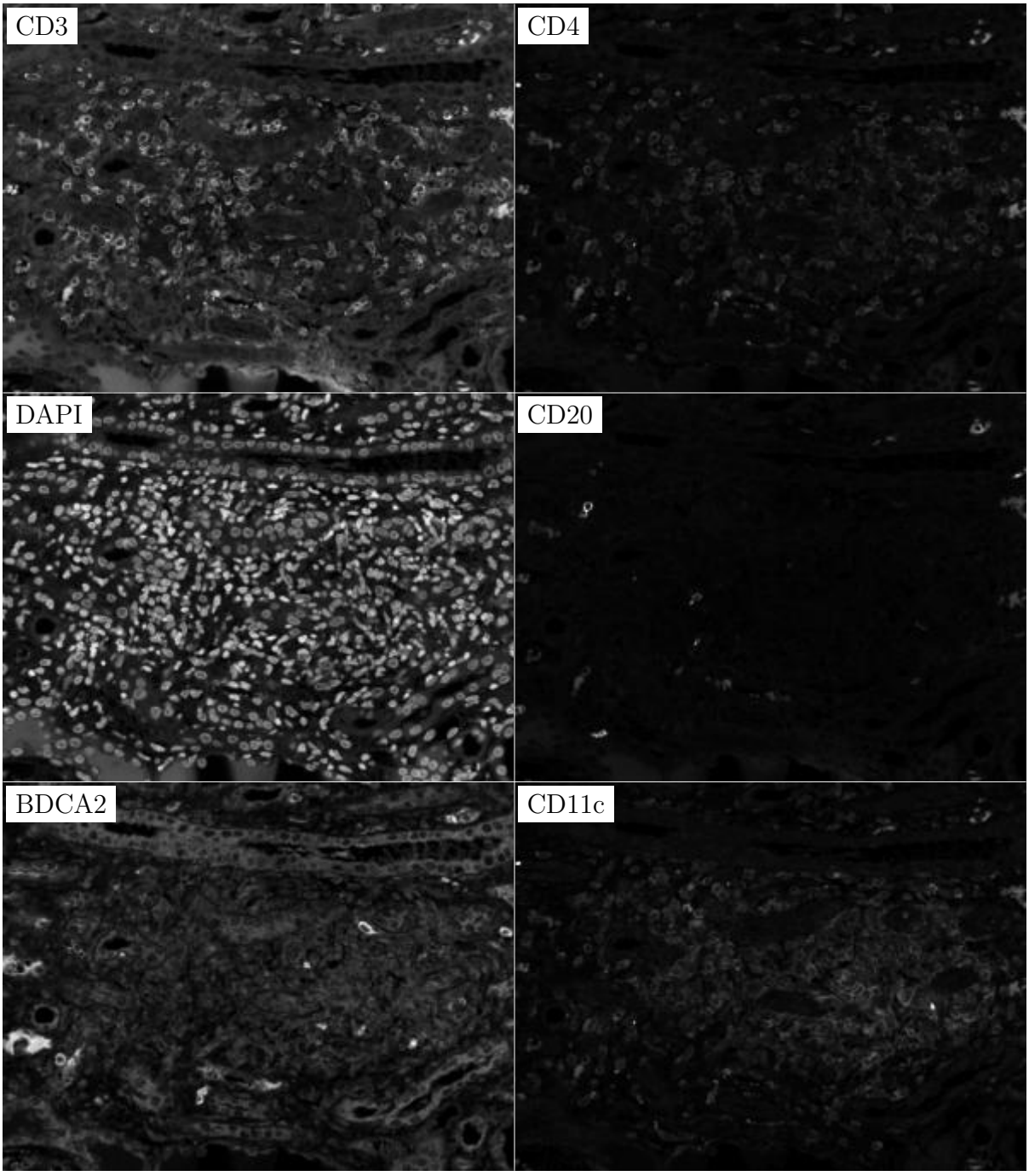

Figure H.34: **ROI for H.33**. True image channel matrix size is 2886 × 3786. Depicted image channel matrix size is 228 × 300.

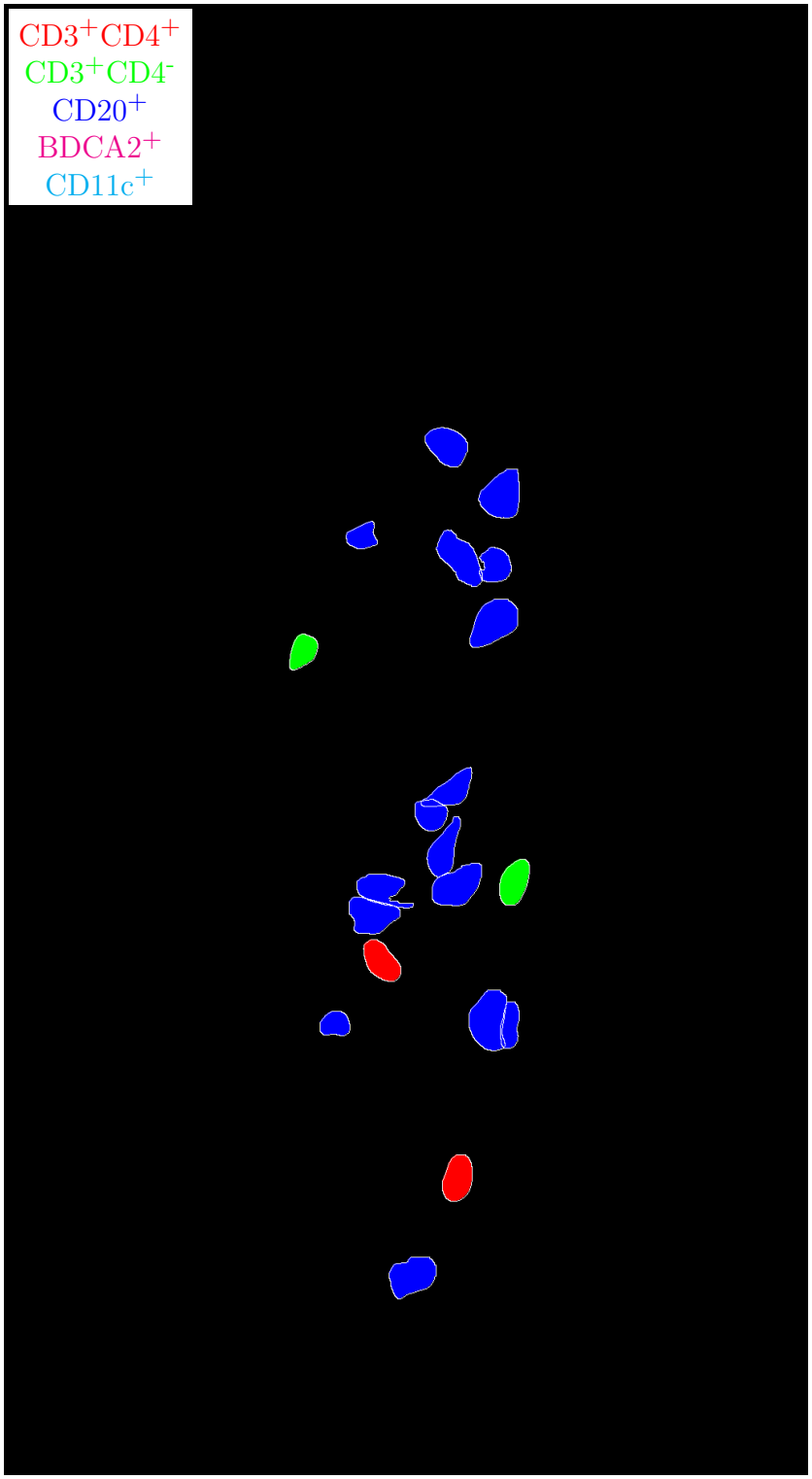

Figure H.35: **Label for H.36**. True label matrix size is 1867×1026. Depicted label matrix size is 1867×1026.

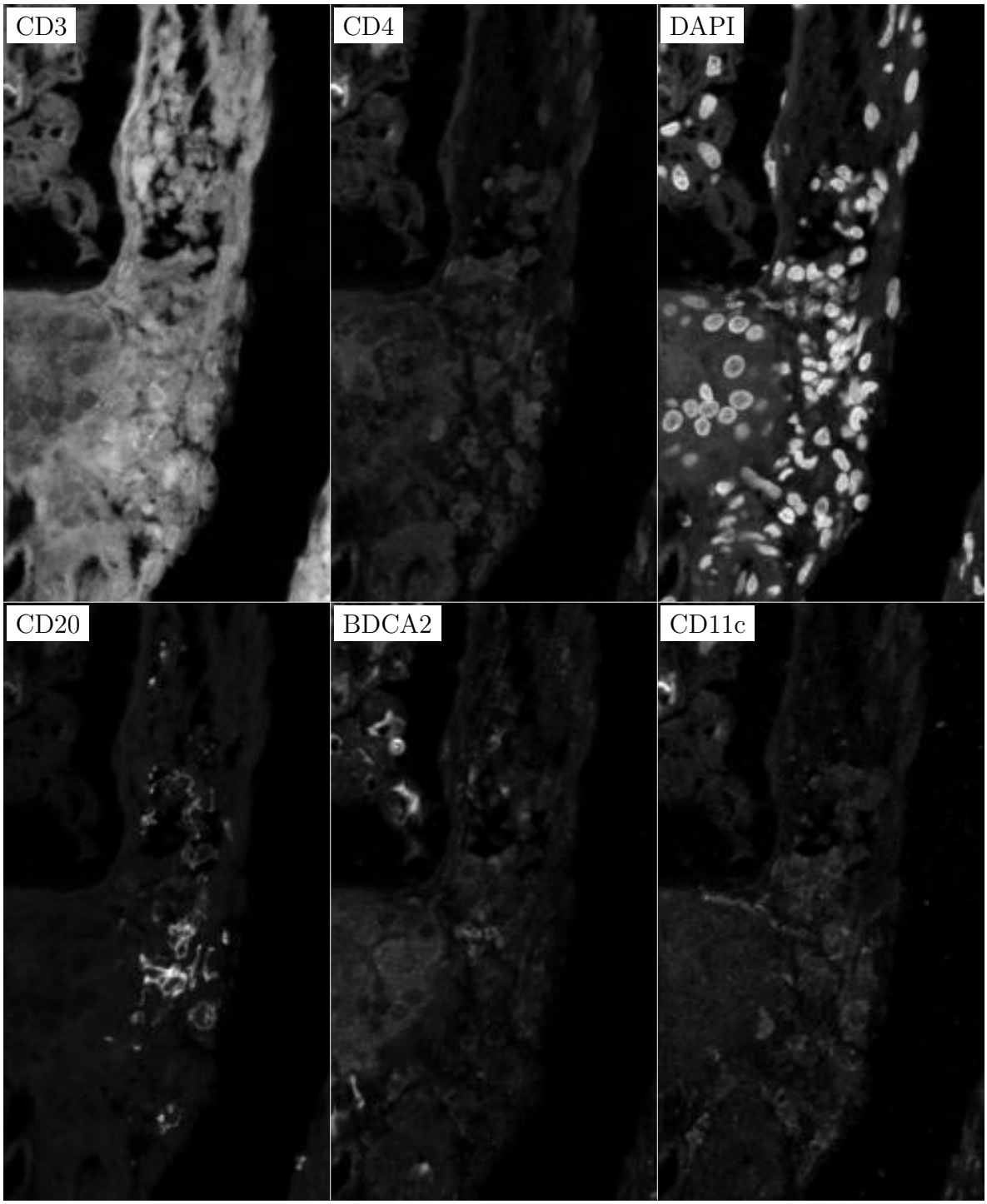

Figure H.36: **ROI for H.35**. True image channel matrix size is 1867×1026. Depicted image channel matrix size is 300×164.

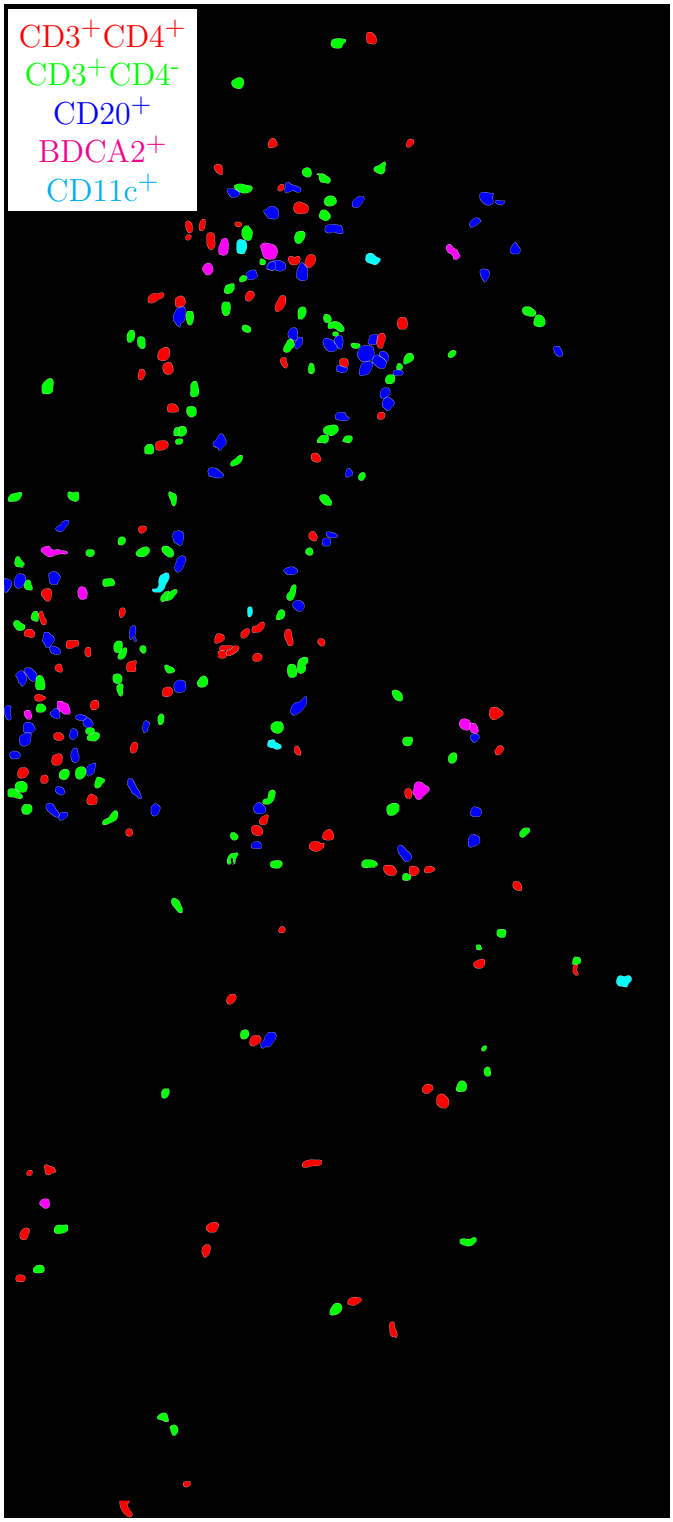

Figure H.37: **Label for H.38**. True label matrix size is 6543×2879. Depicted label matrix size is 6543×2879.

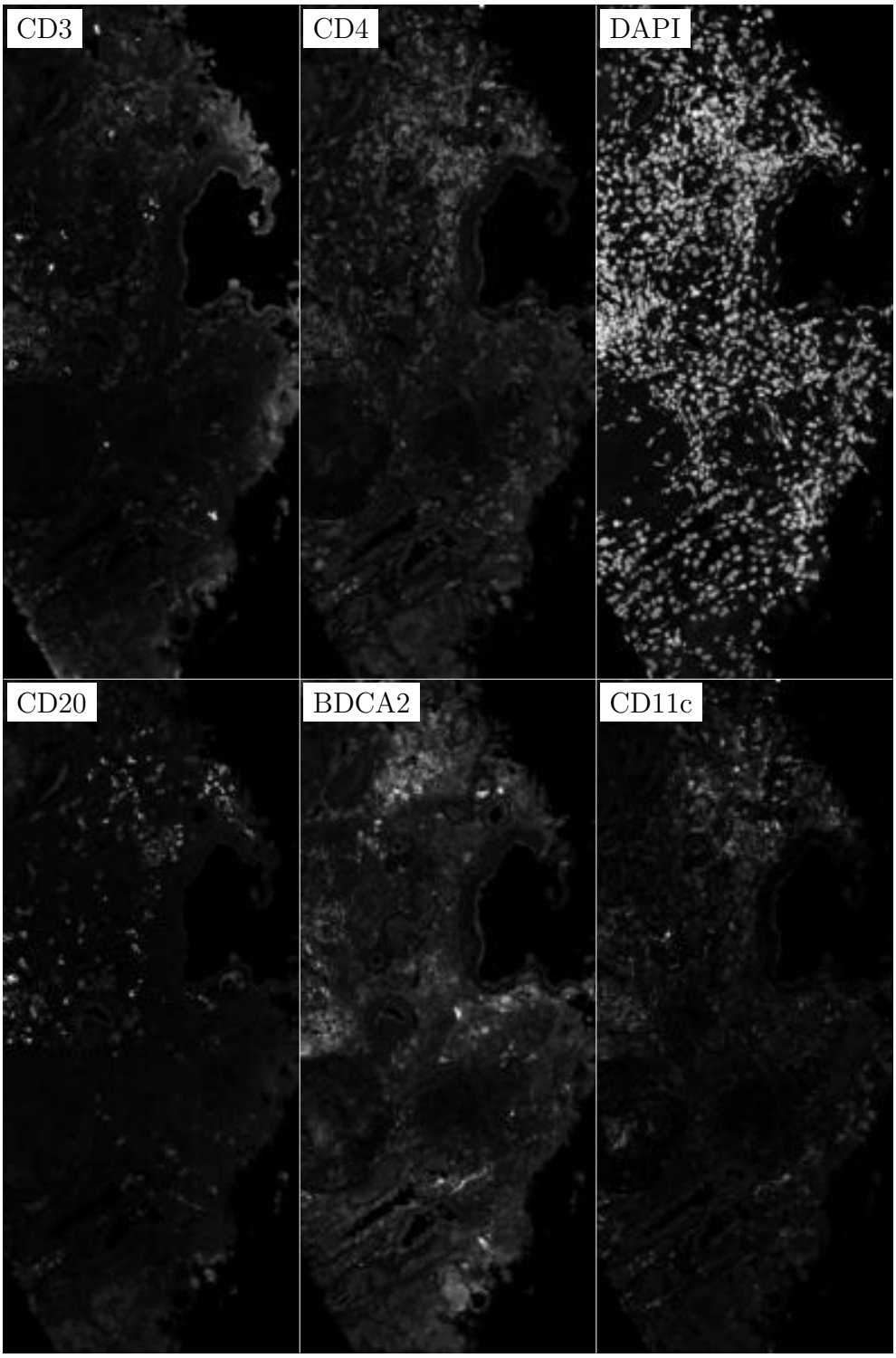

Figure H.38: **ROI for H.37**. True image channel matrix size is 6543 × 2879. Depicted image channel matrix size is 300 × 132.

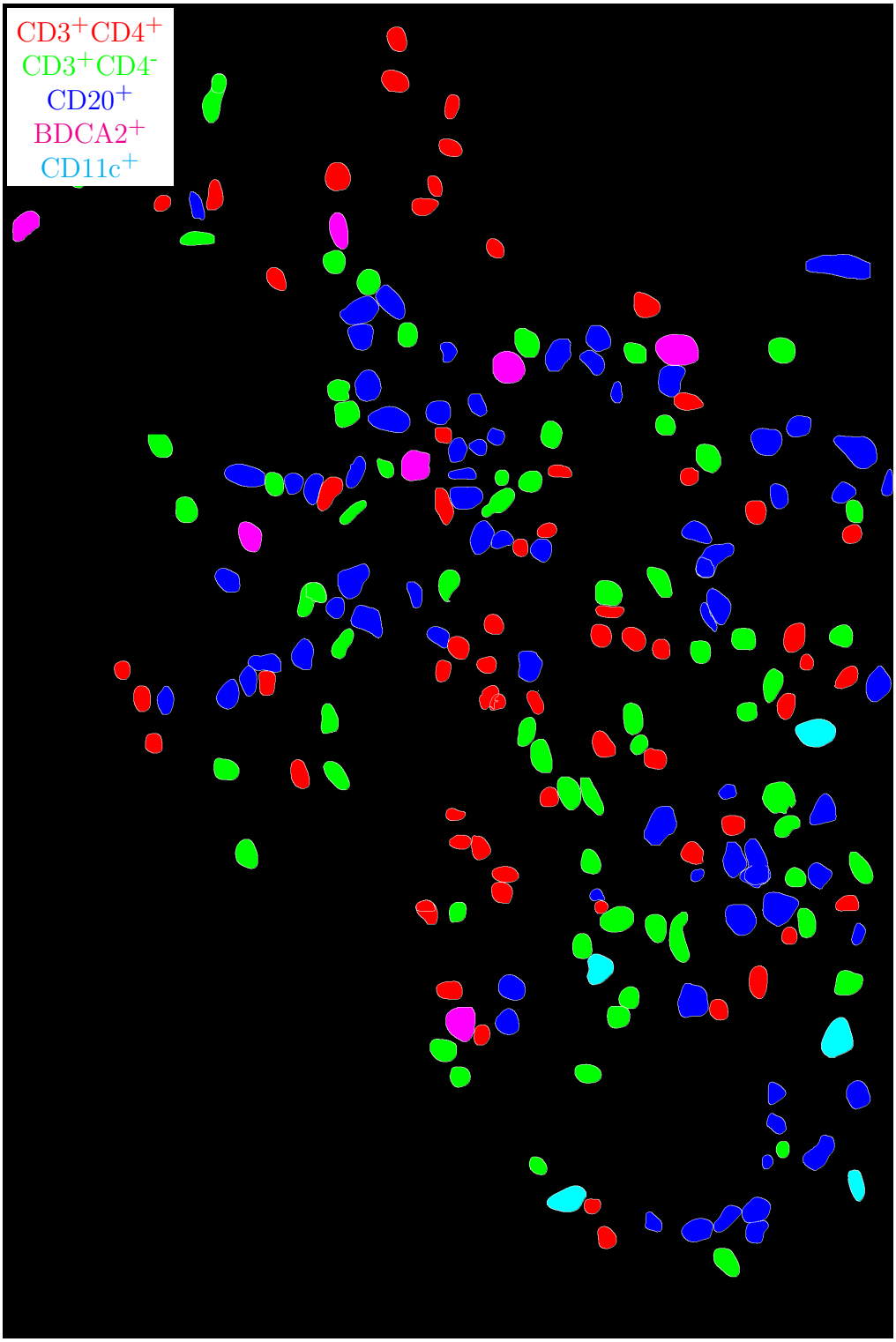

Figure H.39: **Label for H.40**. True label matrix size is 2863×1912. Depicted label matrix size is 2863×1912.

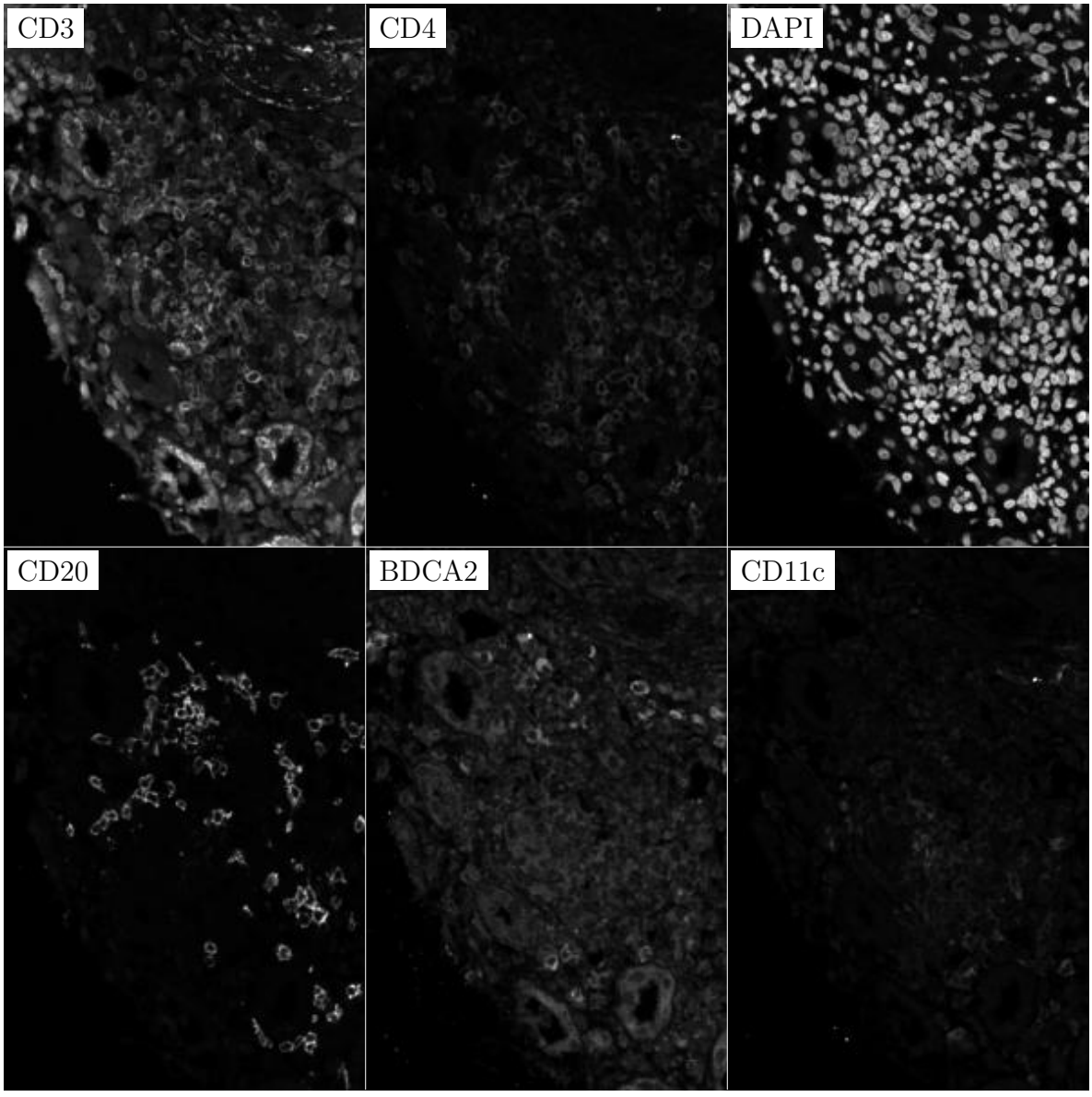

Figure H.40: **ROI for H.39**. True image channel matrix size is 2863 × 1912. Depicted image channel matrix size is 300 × 200.

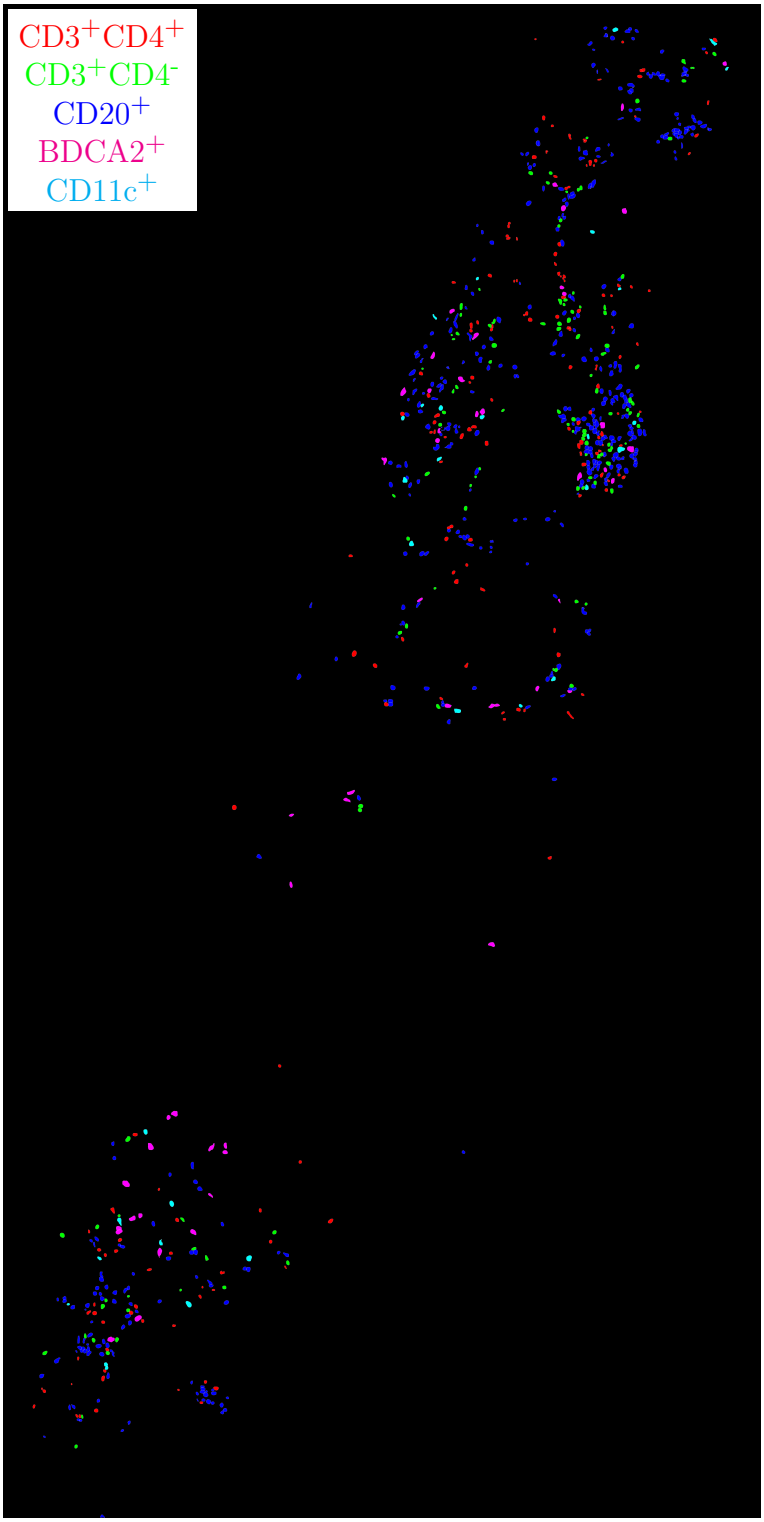

Figure H.41: **Label for H.42**. True label matrix size is 18484×9295. Depicted label matrix size is 10000×5028.

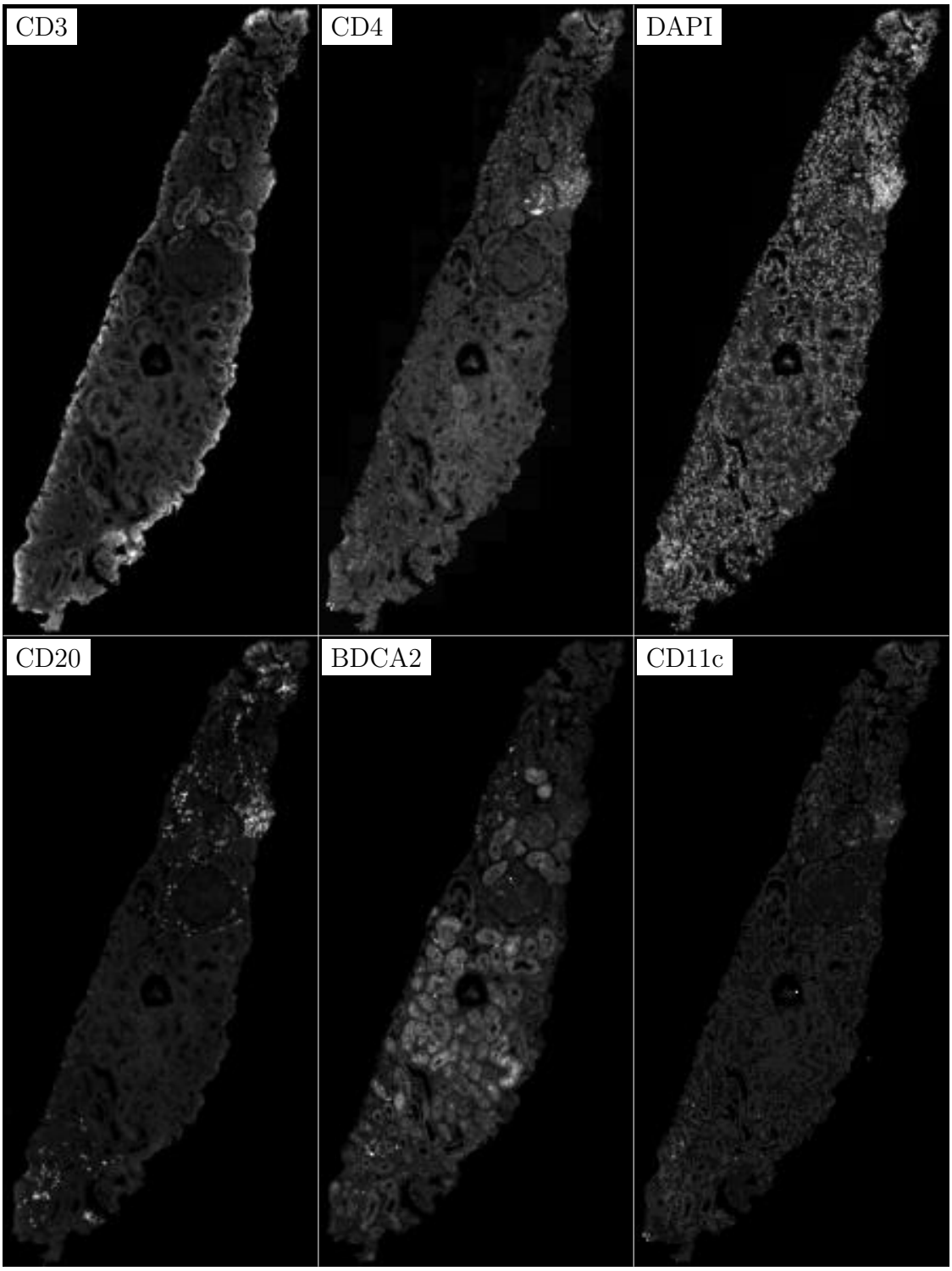

Figure H.42: **ROI for H.41**. True image channel matrix size is 18484 × 9295. Depicted image channel matrix size is 300 × 150.

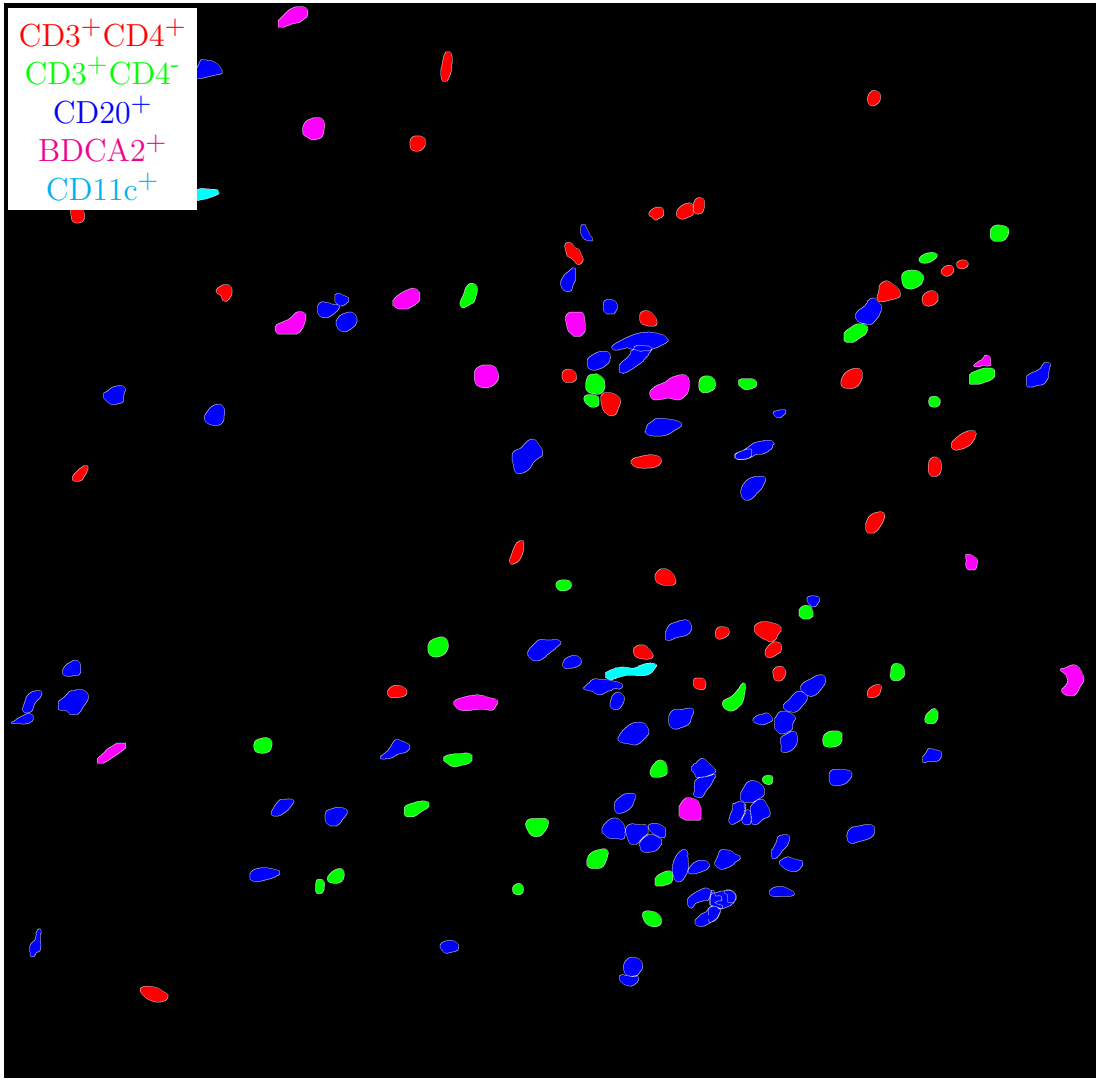

Figure H.43: **Label for H.44**. True label matrix size is 2826 × 2866. Depicted label matrix size is 2826 × 2866.

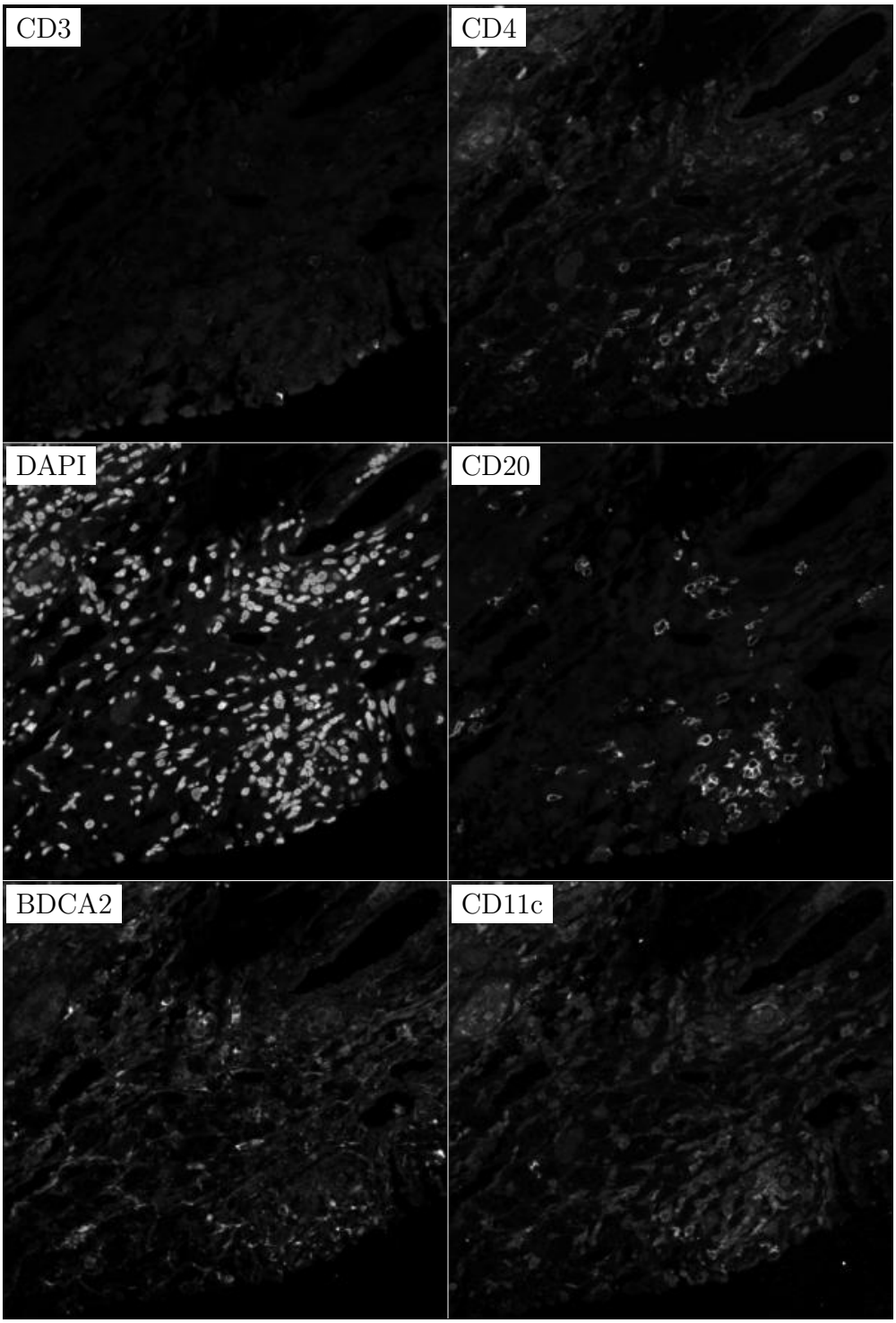

Figure H.44: **ROI for H.43**. True image channel matrix size is 2826 × 2866. Depicted image channel matrix size is 295 × 300.

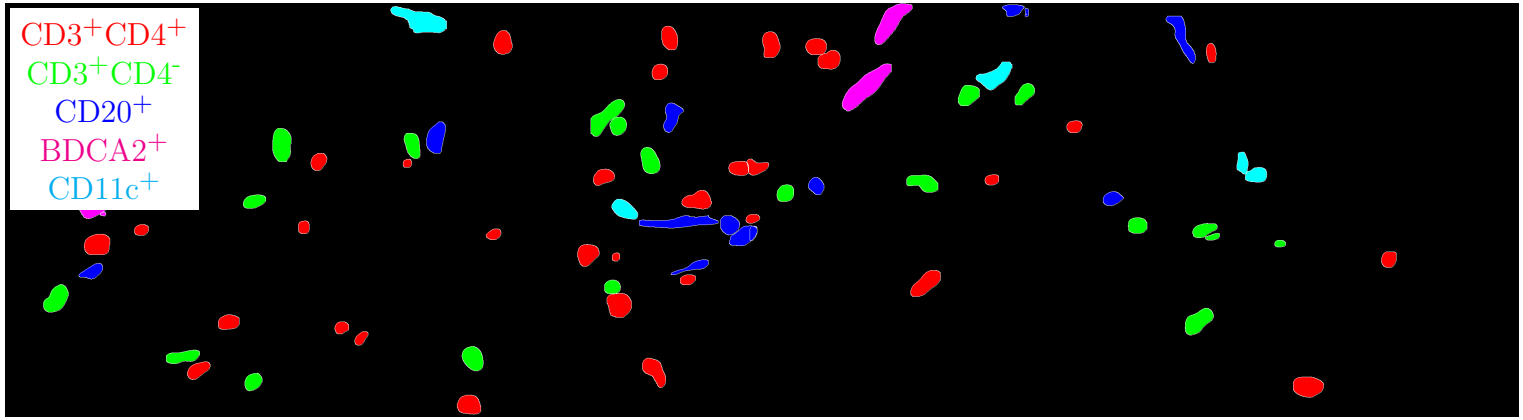

Figure H.45: **Label for H.46**. True label matrix size is 1029×3759. Depicted label matrix size is 1029×3759.

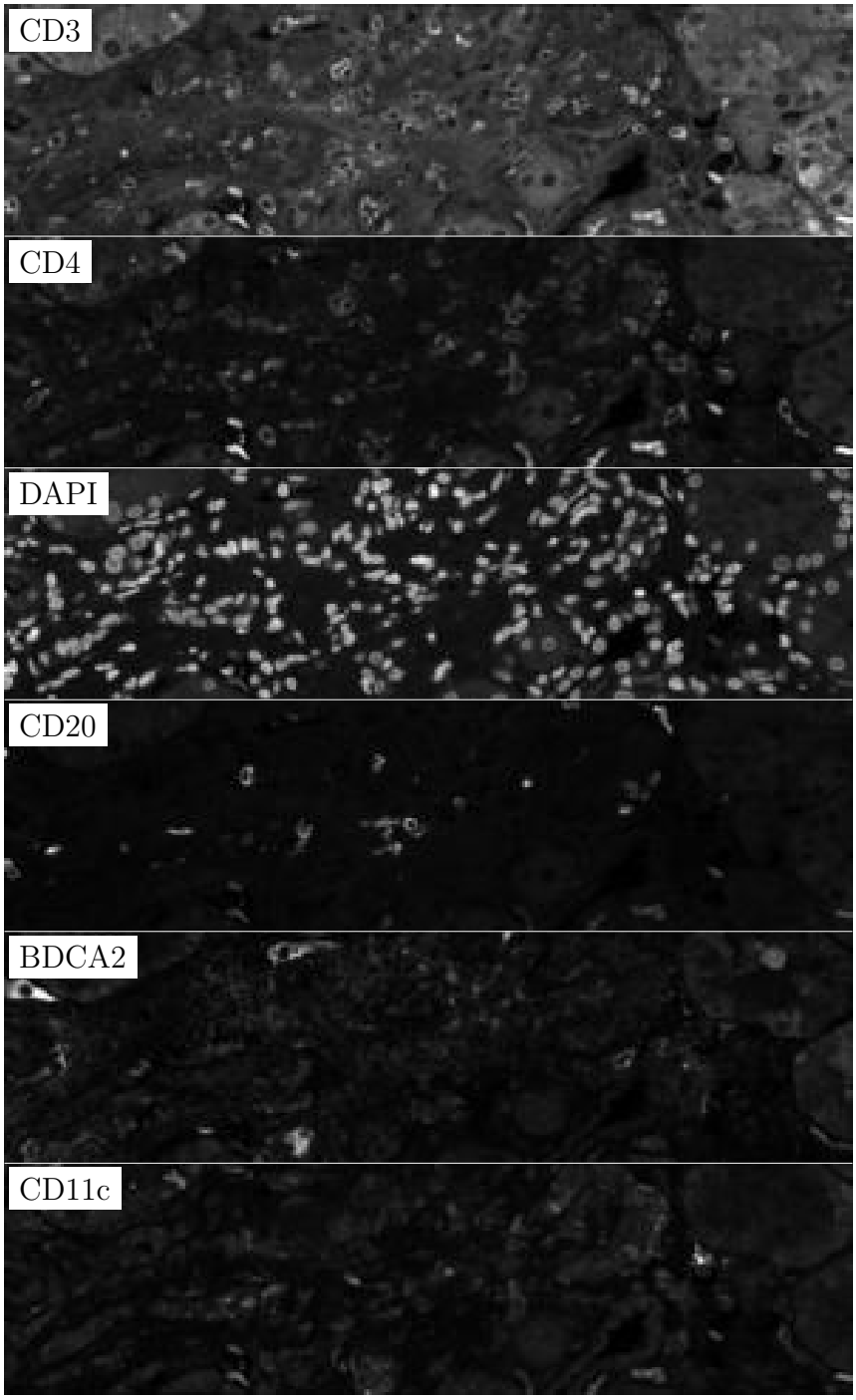

Figure H.46: **ROI for H.45**. True image channel matrix size is 1029×3759. Depicted image channel matrix size is 82×300.

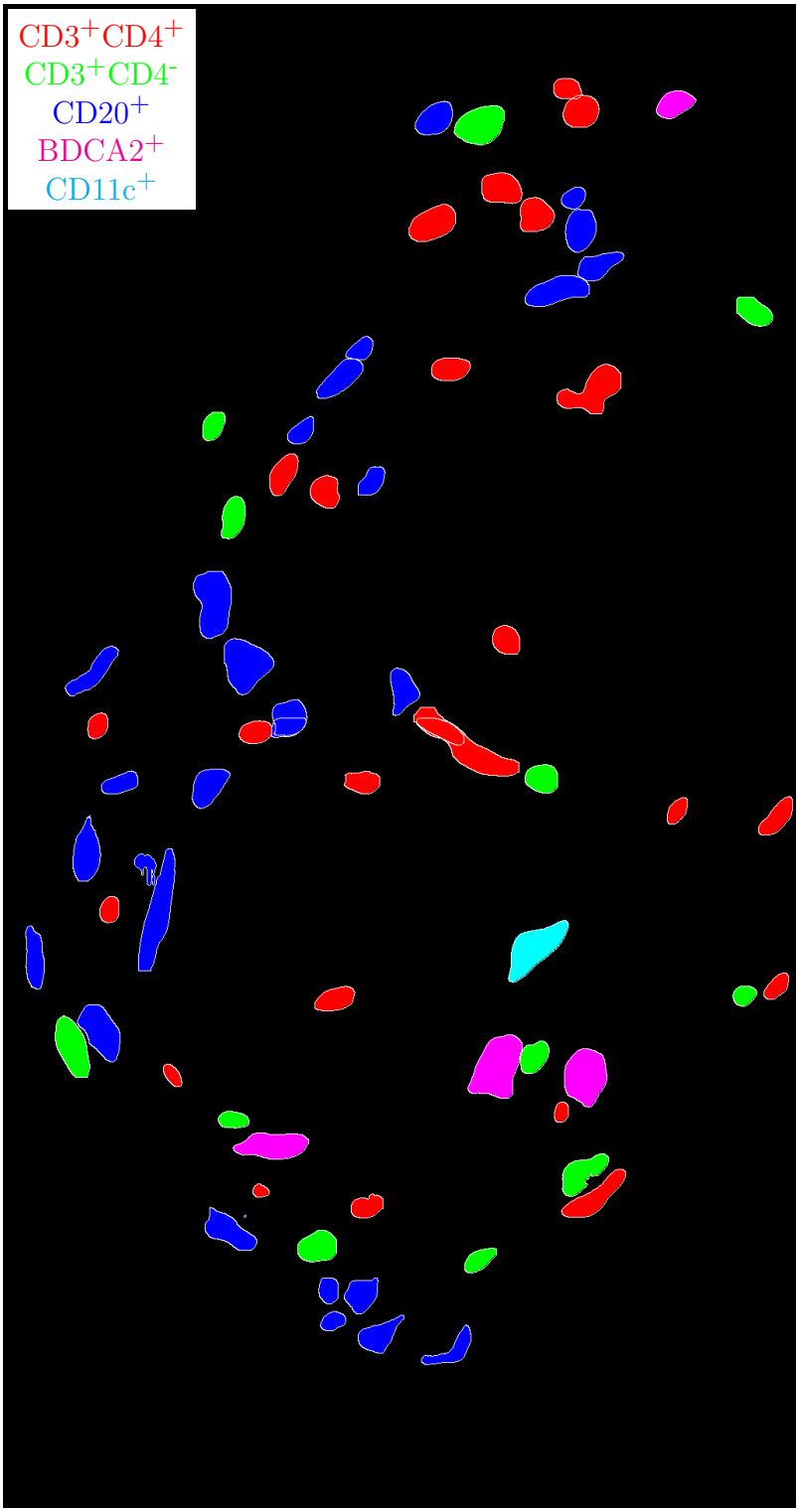

Figure H.47: **Label for H.48**. True label matrix size is 1947×1026. Depicted label matrix size is 1947×1026.

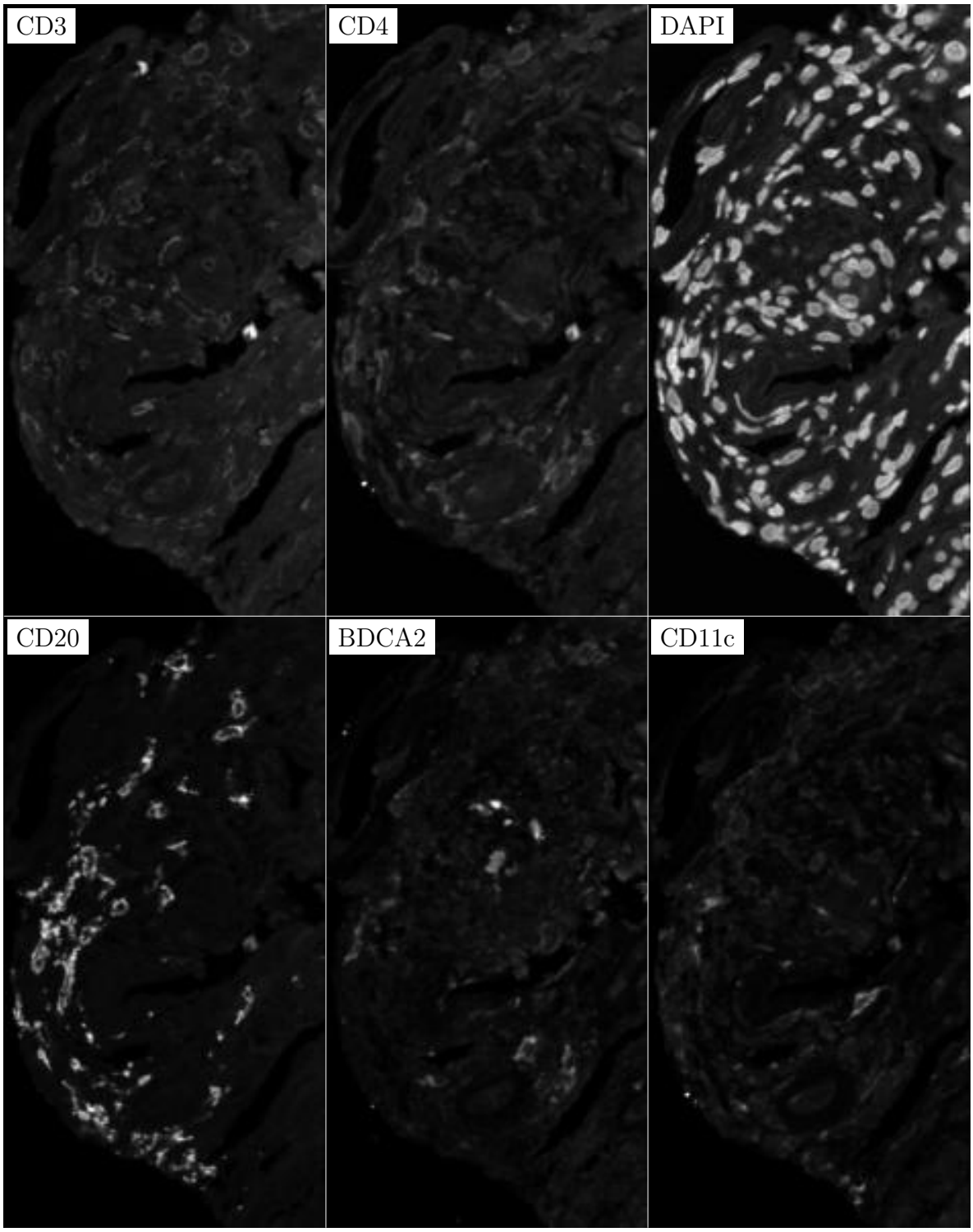

Figure H.48: **ROI for H.47**. True image channel matrix size is 1947 × 1026. Depicted image channel matrix size is 300 × 158.

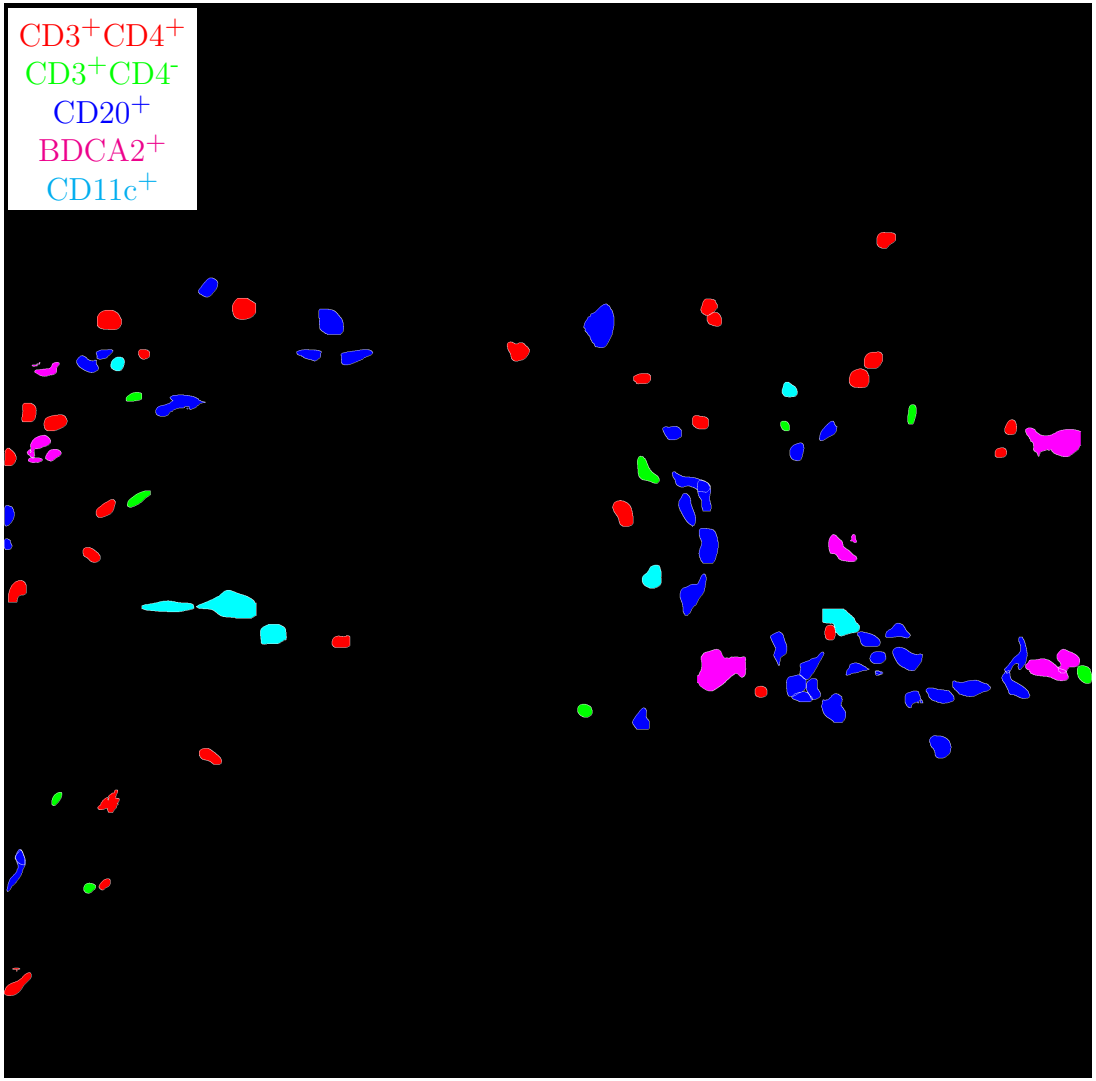

Figure H.49: **Label for H.50**. True label matrix size is 2868 × 2901. Depicted label matrix size is 2868 × 2901.

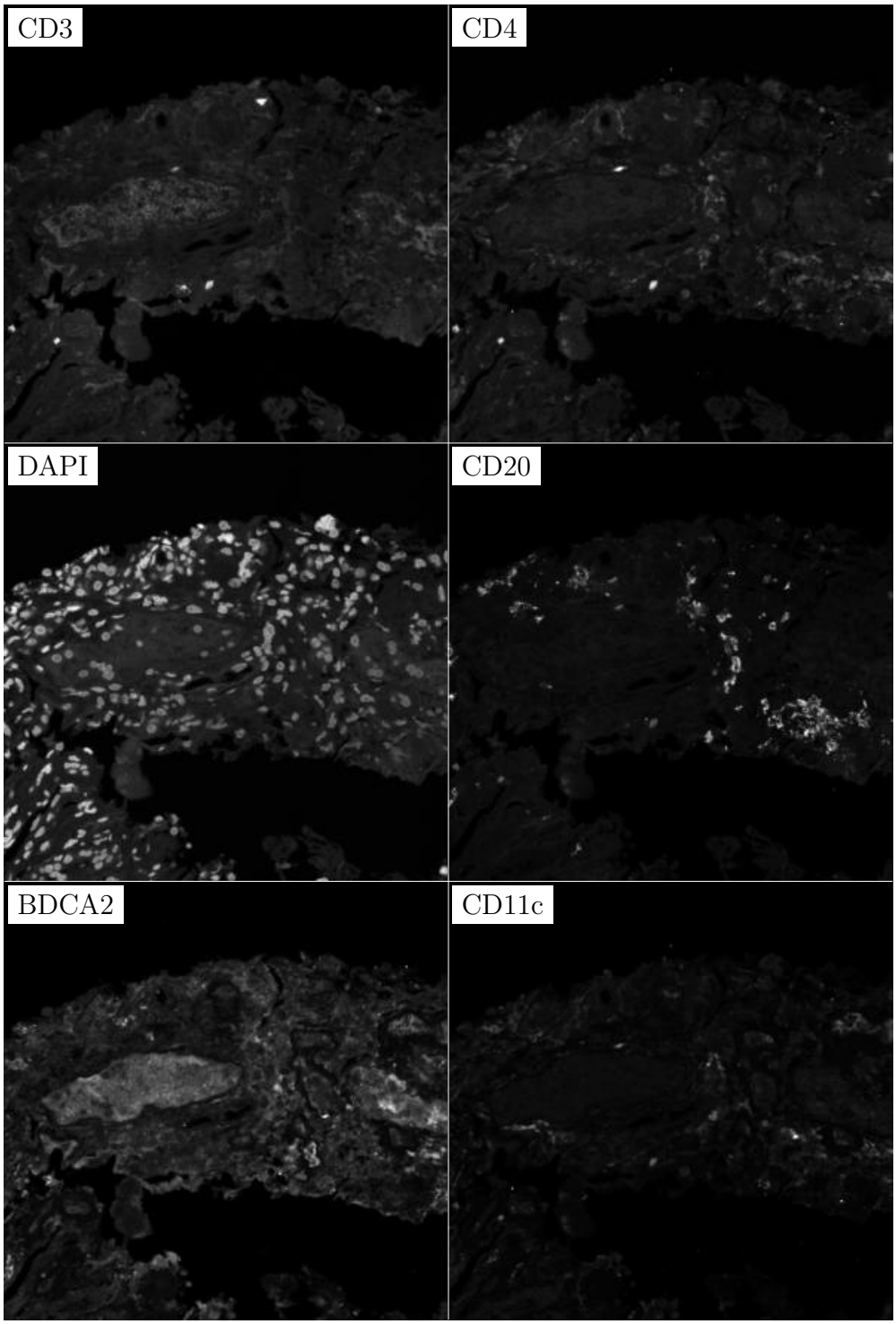

Figure H.50: **ROI for H.49**. True image channel matrix size is 2868 × 2901. Depicted image channel matrix size is 296 × 300.

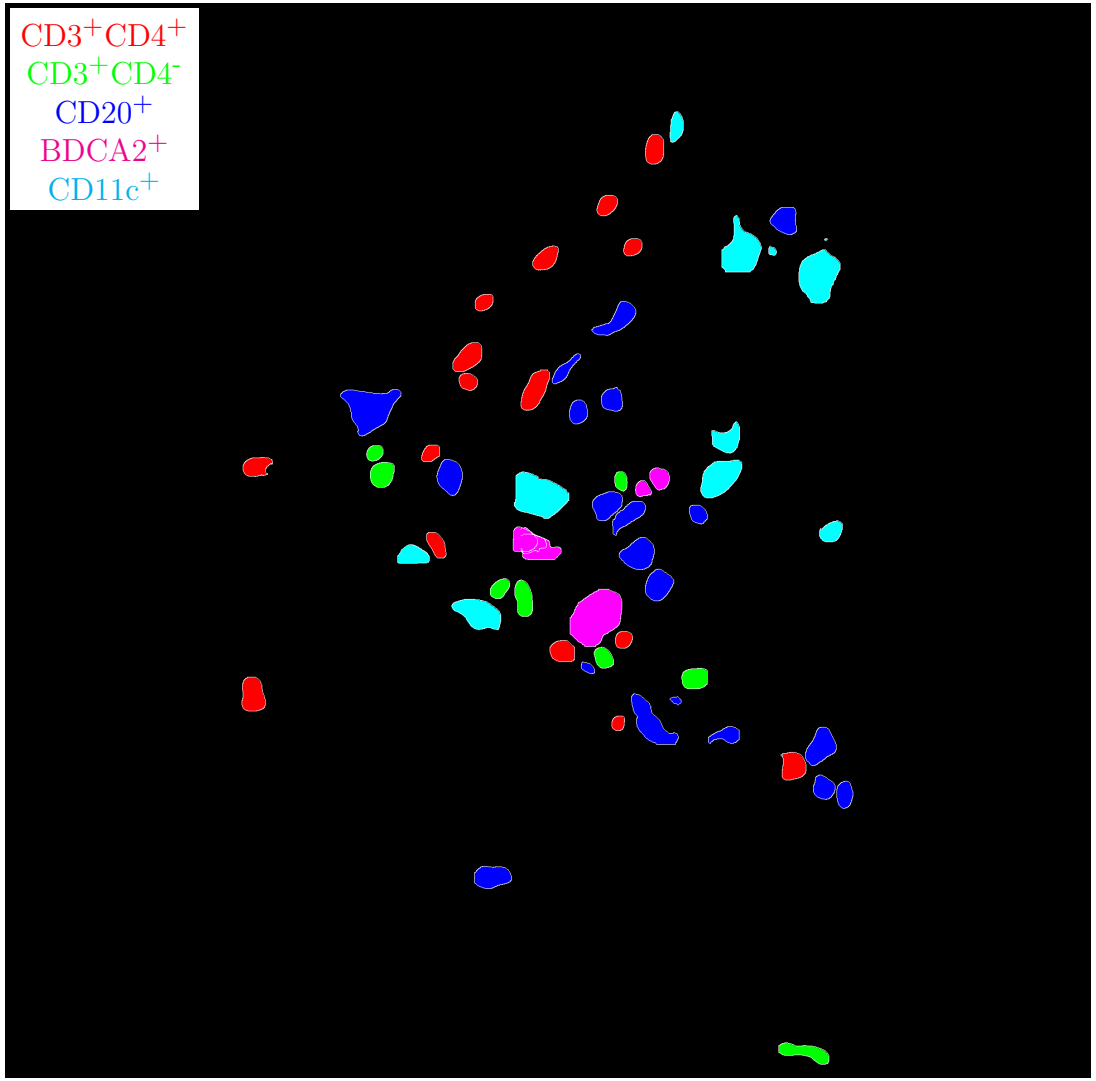

Figure H.51: **Label for H.52**. True label matrix size is 1932 × 1946. Depicted label matrix size is 1932 × 1946.

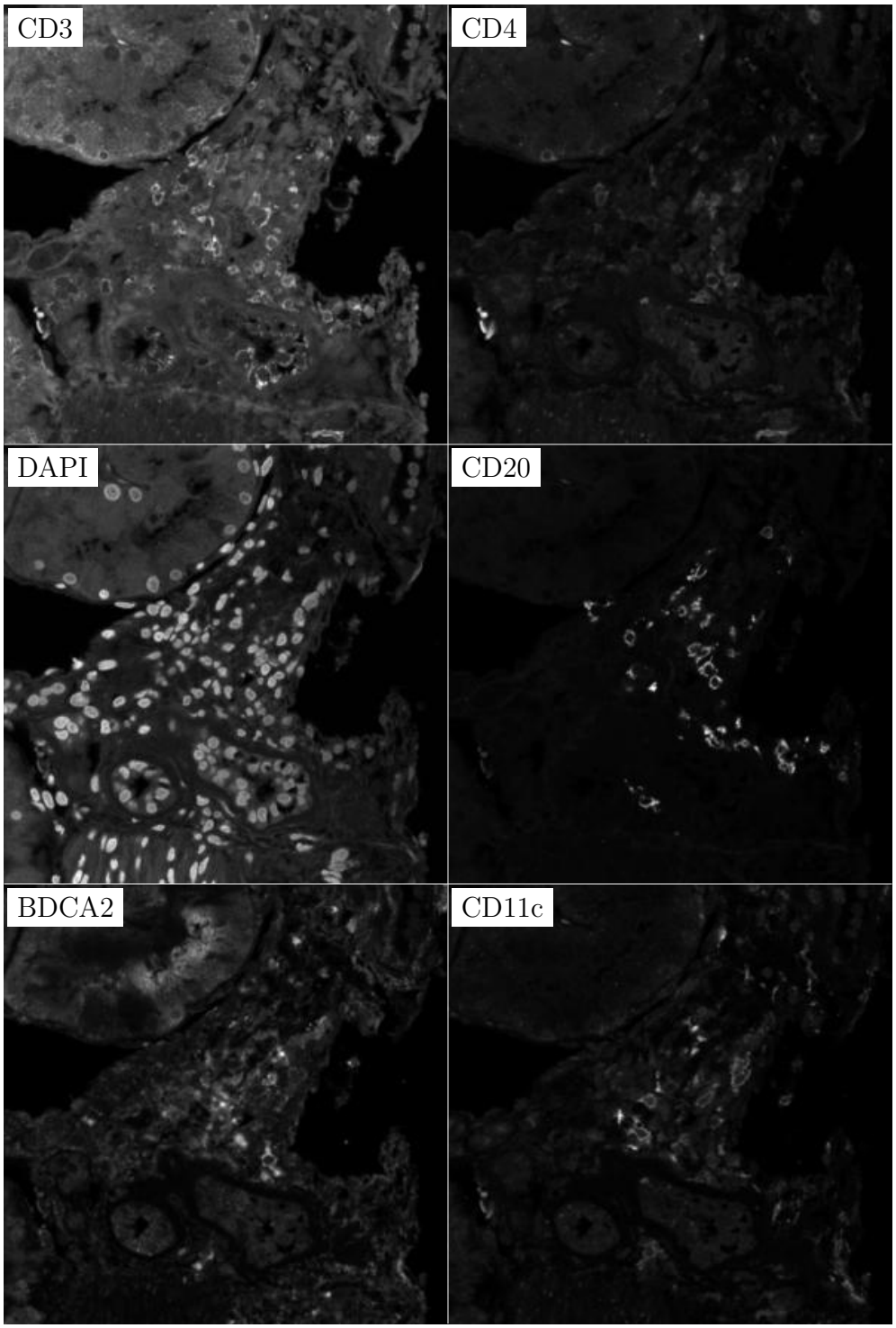

Figure H.52: **ROI for H.51**. True image channel matrix size is 1932 × 1946. Depicted image channel matrix size is 297 × 300.

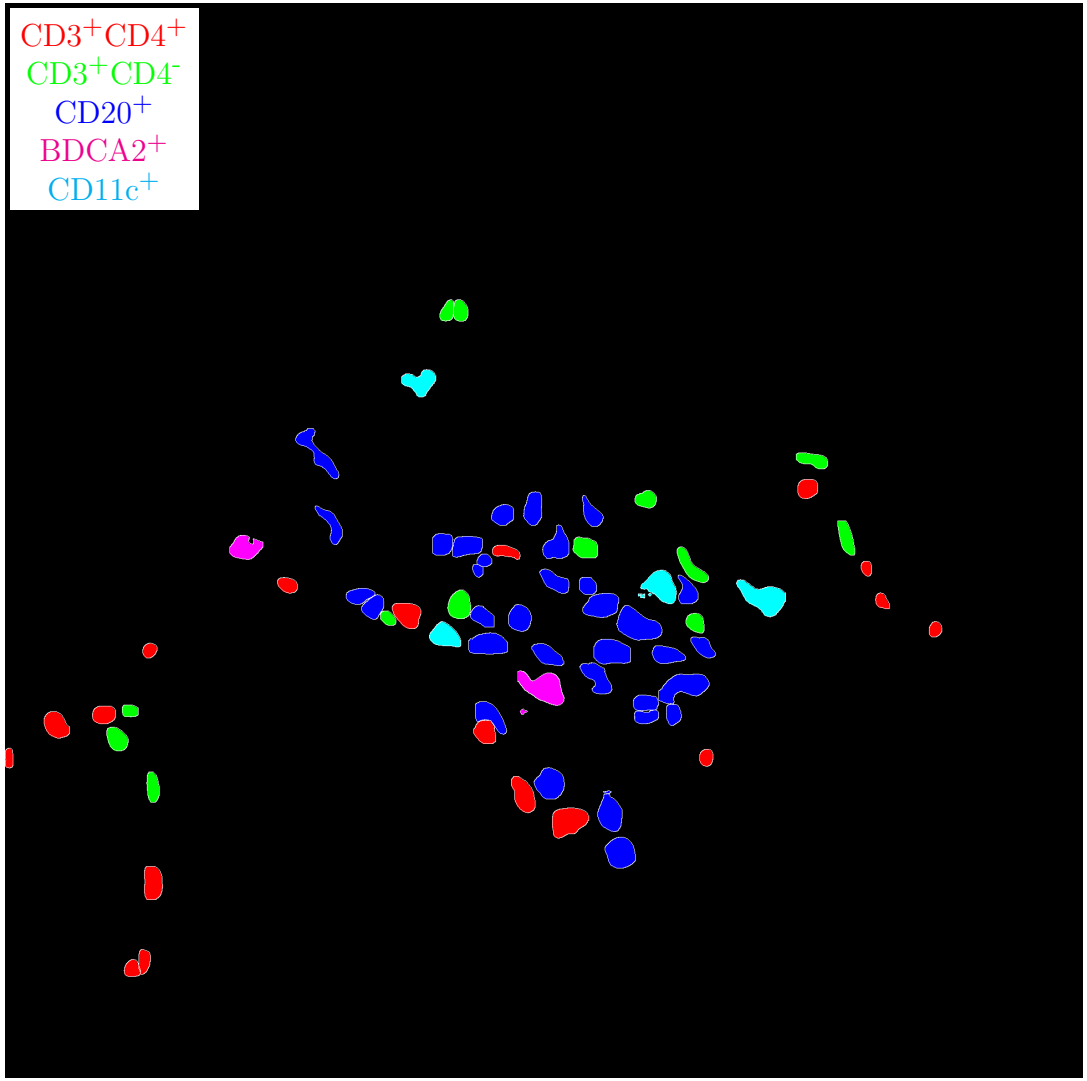

Figure H.53: **Label for H.54**. True label matrix size is 1944 × 1945. Depicted label matrix size is 1944 × 1945.

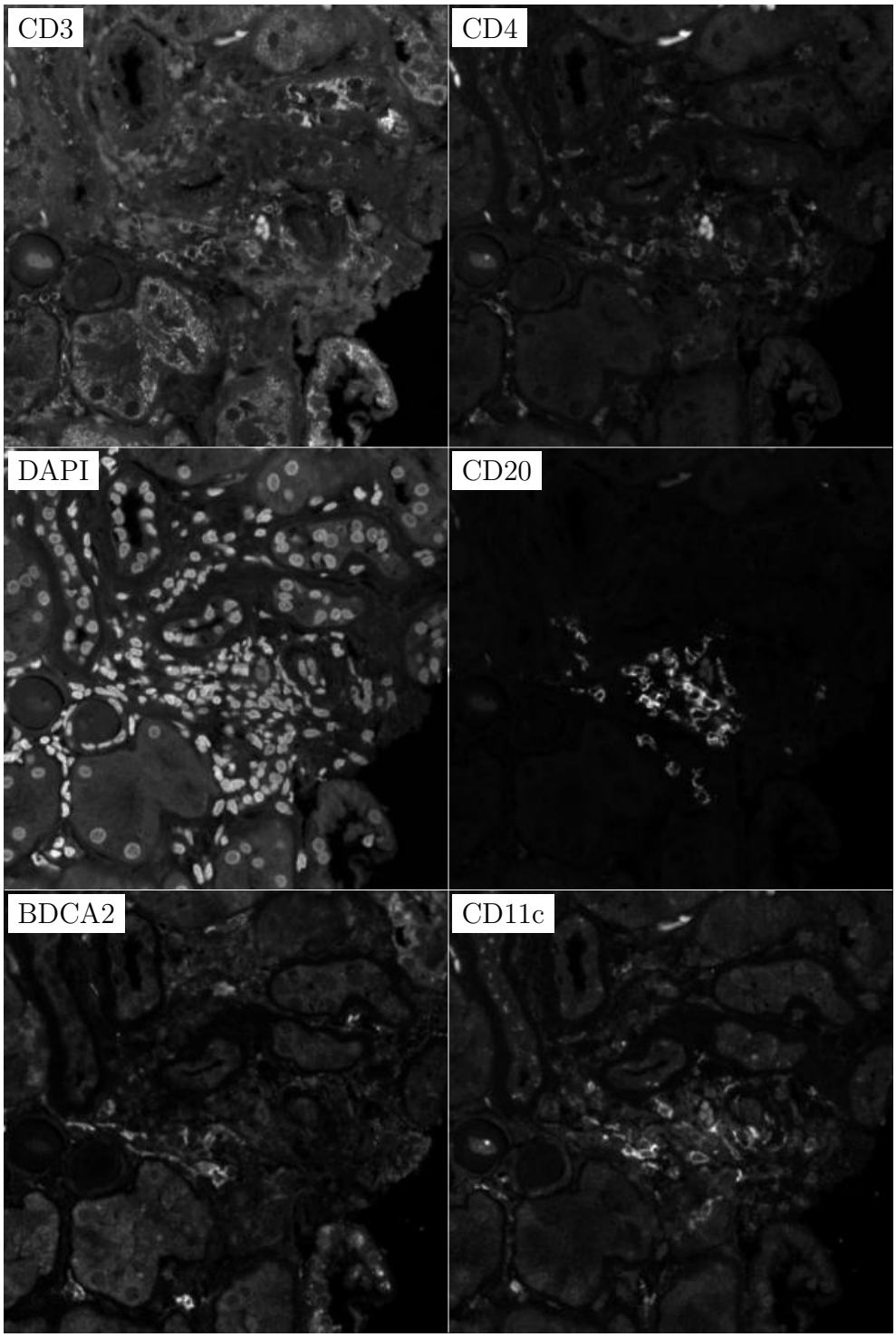

Figure H.54: **ROI for H.53**. True image channel matrix size is 1944 × 1945. Depicted image channel matrix size is 299 × 300.

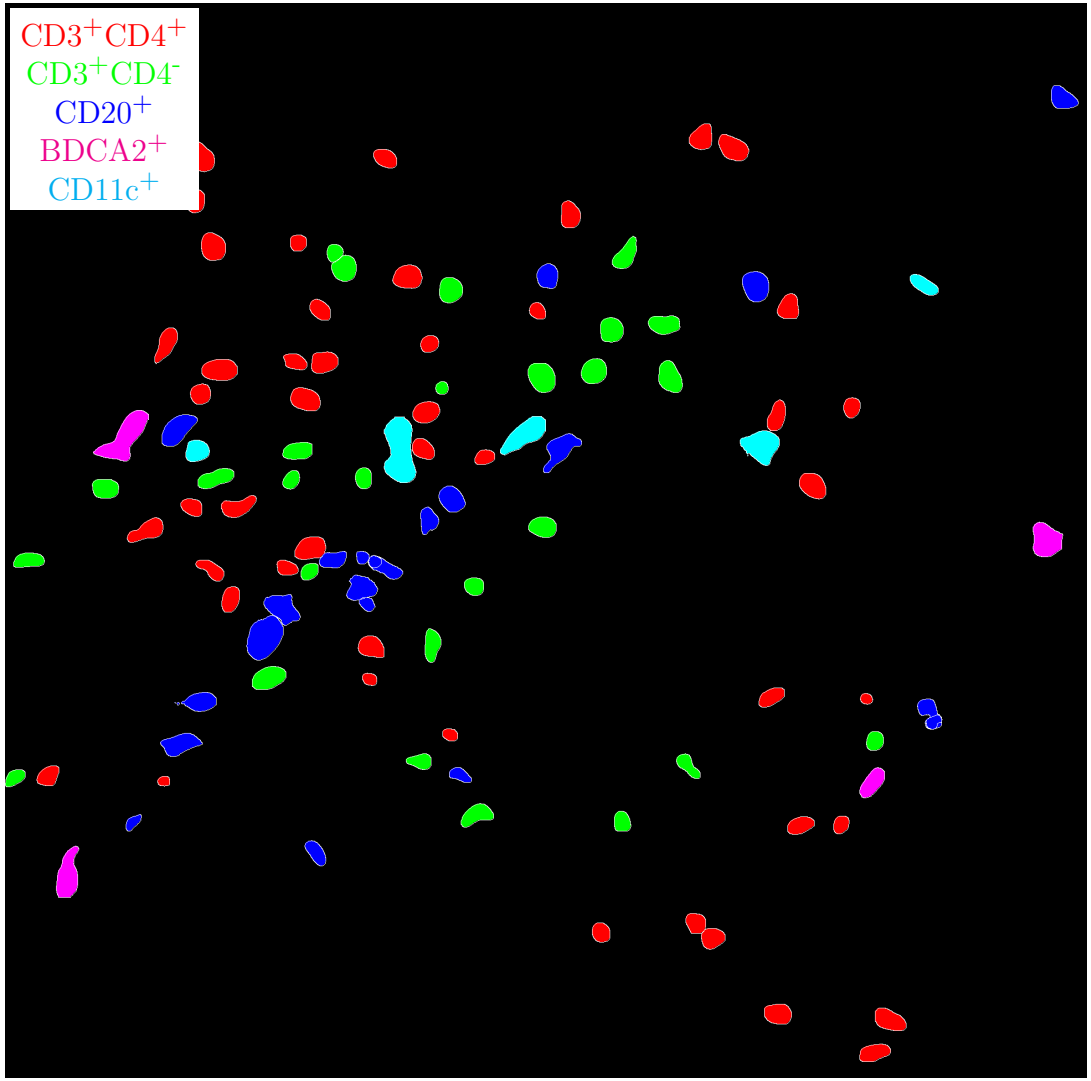

Figure H.55: **Label for H.56**. True label matrix size is 1944 × 1954. Depicted label matrix size is 1944 × 1954.

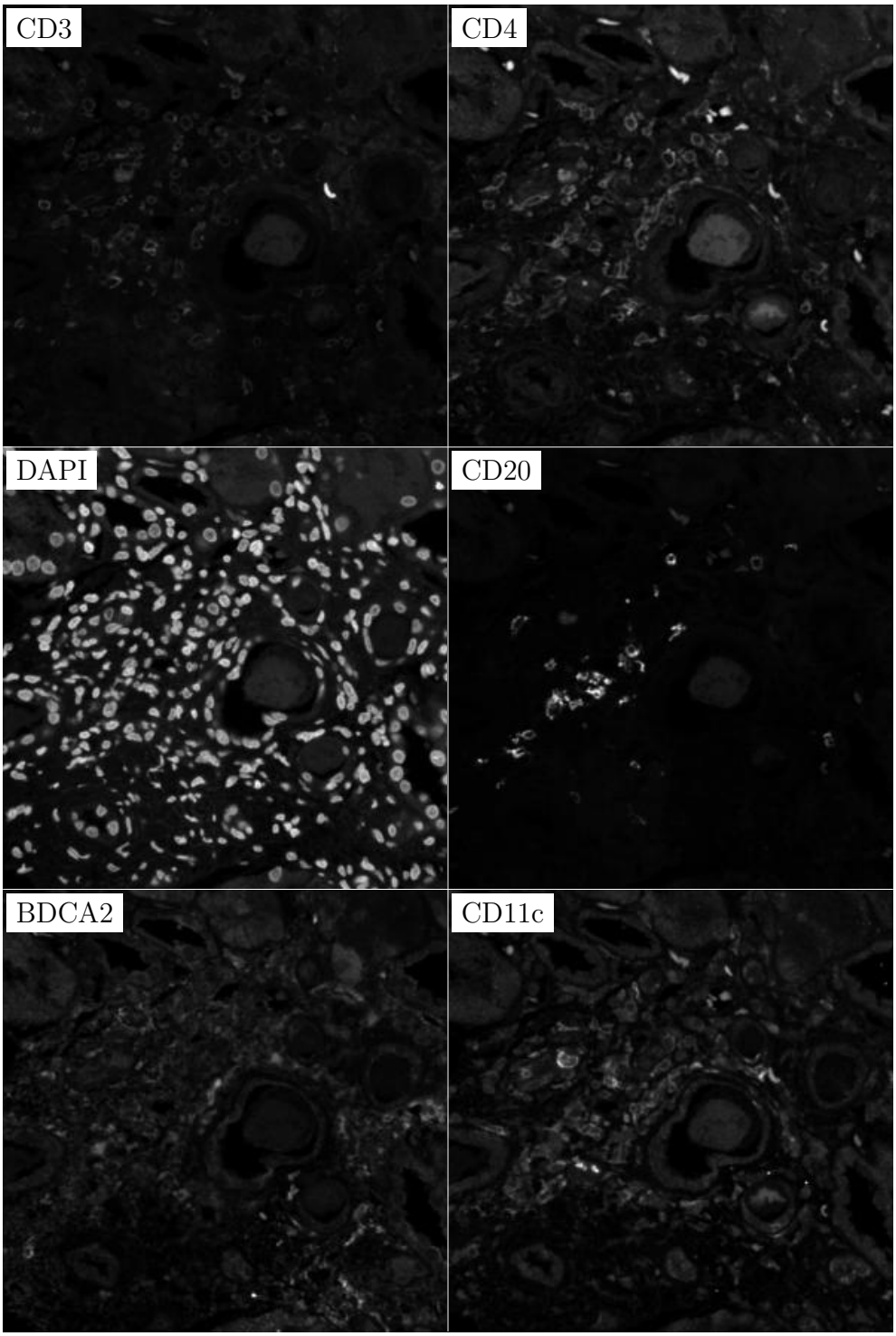

Figure H.56: **ROI for H.55**. True image channel matrix size is 1944 × 1954. Depicted image channel matrix size is 298 × 300.

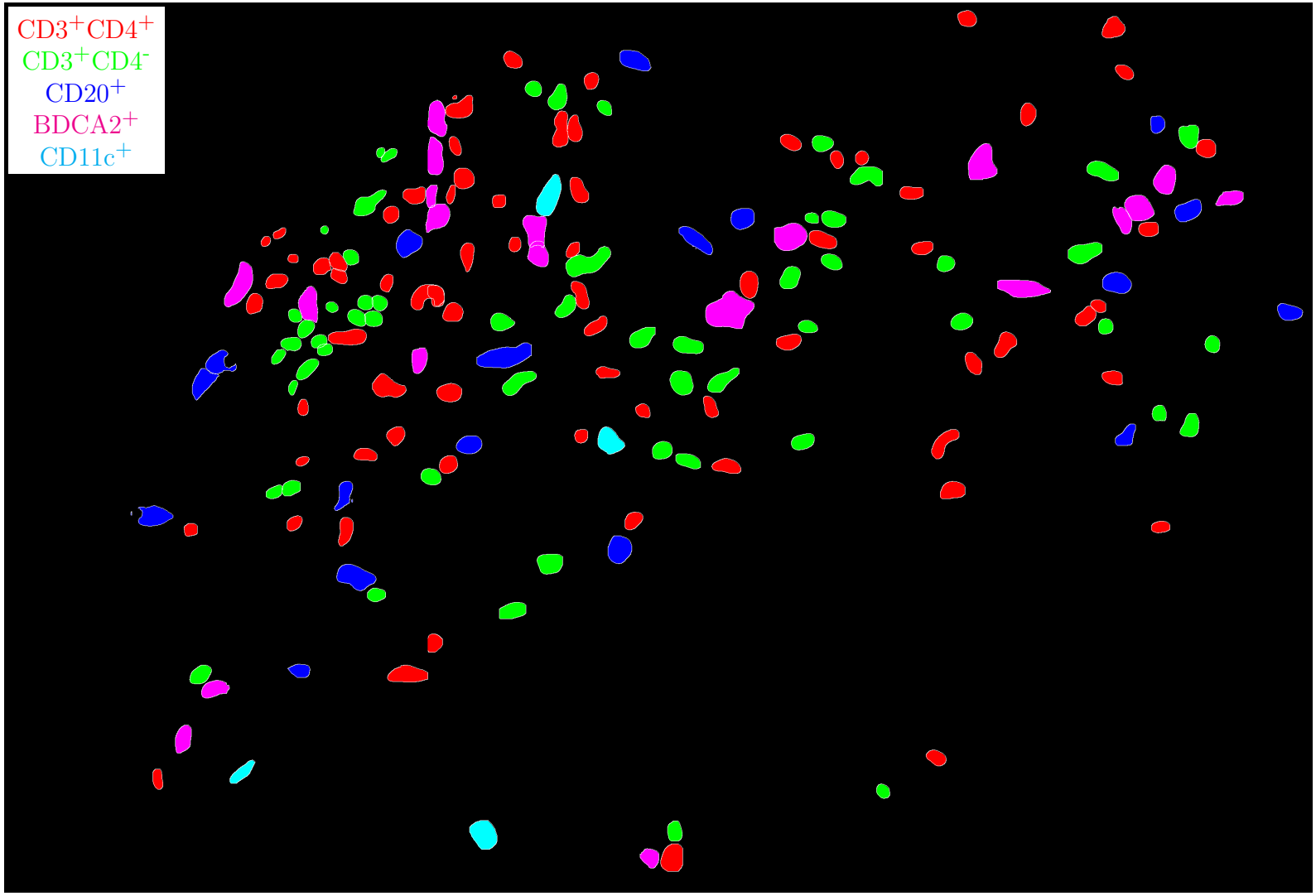

Figure H.57: **Label for H.58**. True label matrix size is 1942 × 2855. Depicted label matrix size is 1942 × 2855.

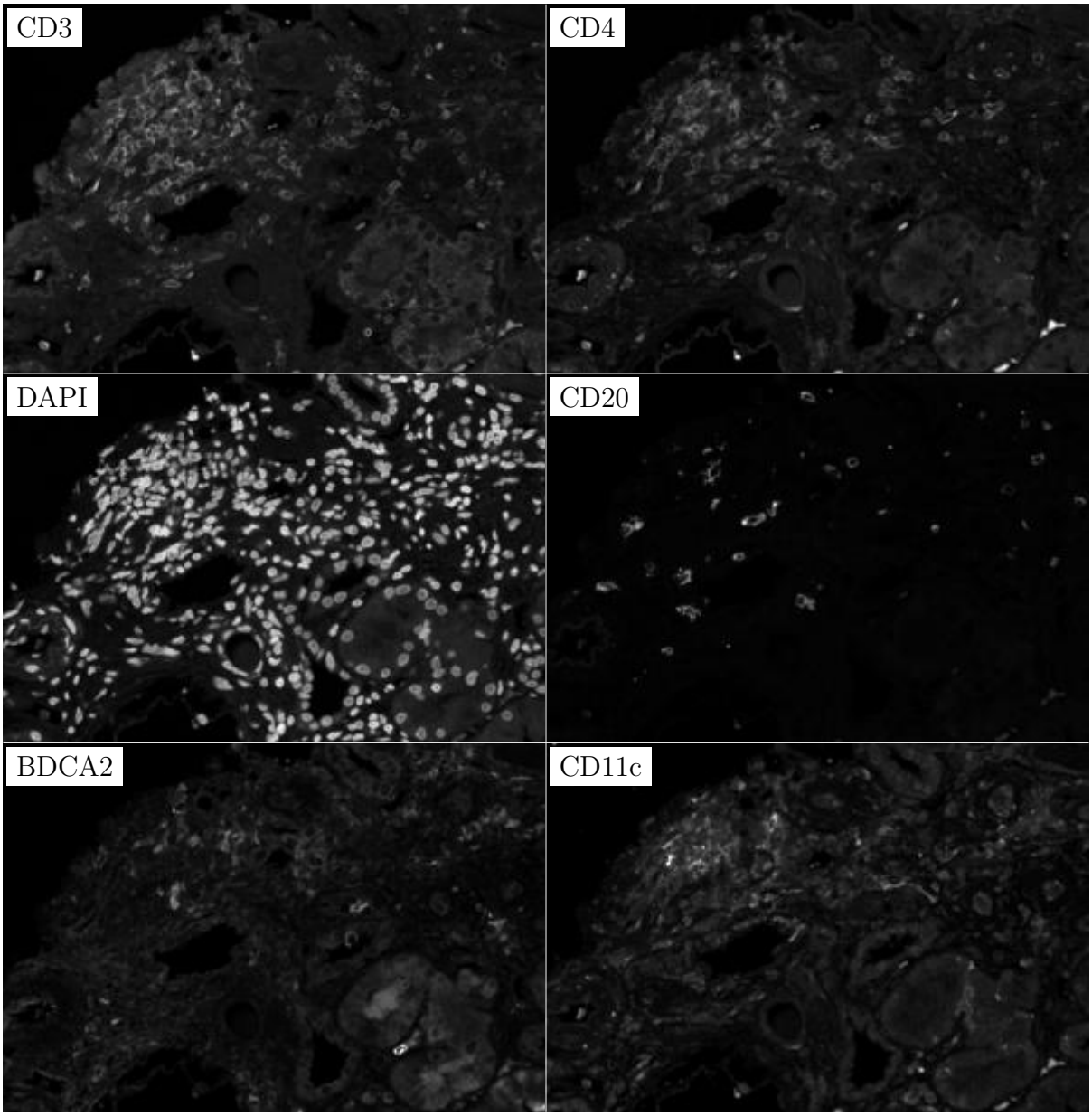

Figure H.58: **ROI for H.57**. True image channel matrix size is 1942 × 2855. Depicted image channel matrix size is 204 × 300.

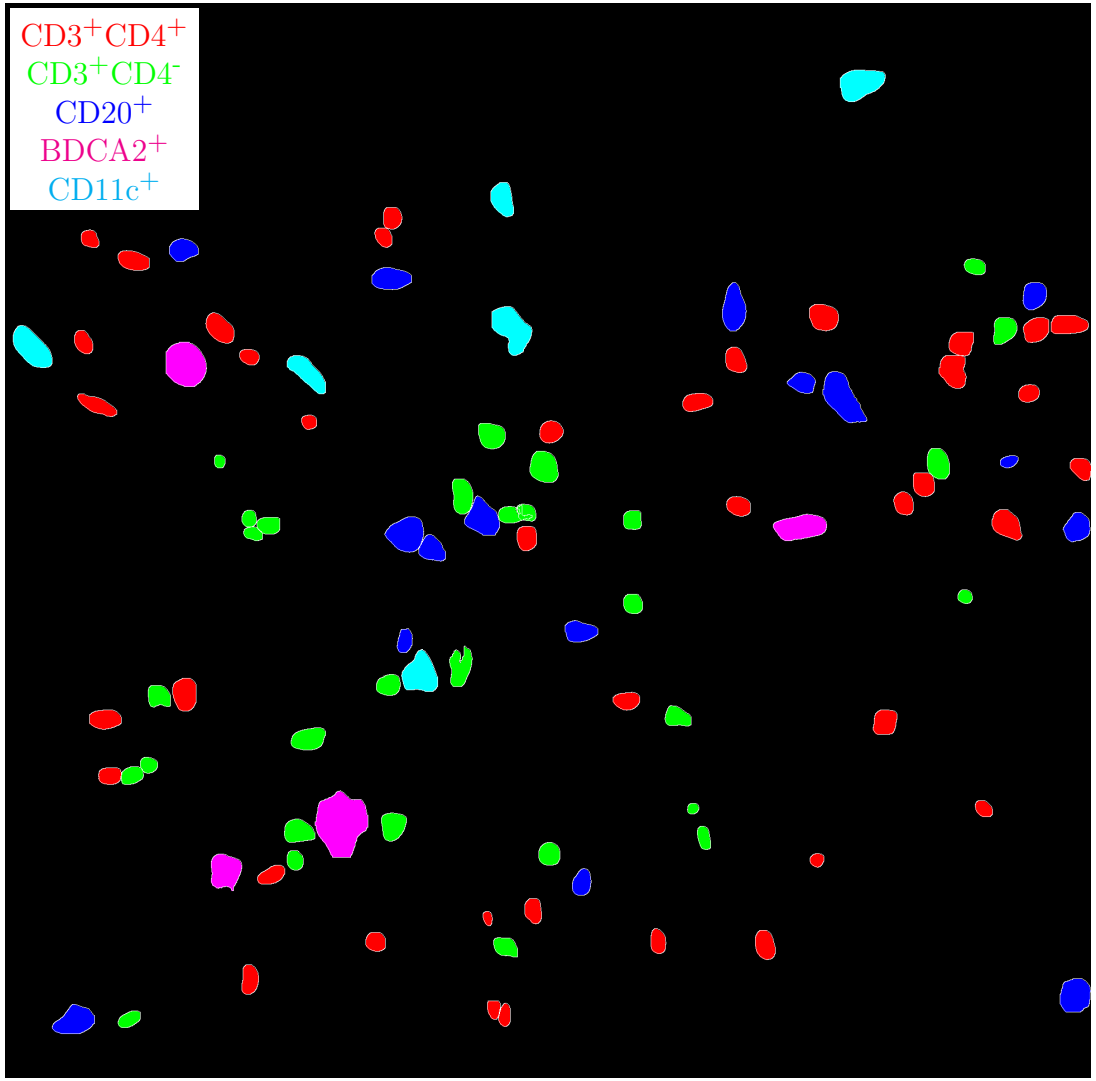

Figure H.59: **Label for H.60**. True label matrix size is 1932 × 1946. Depicted label matrix size is 1932 × 1946.

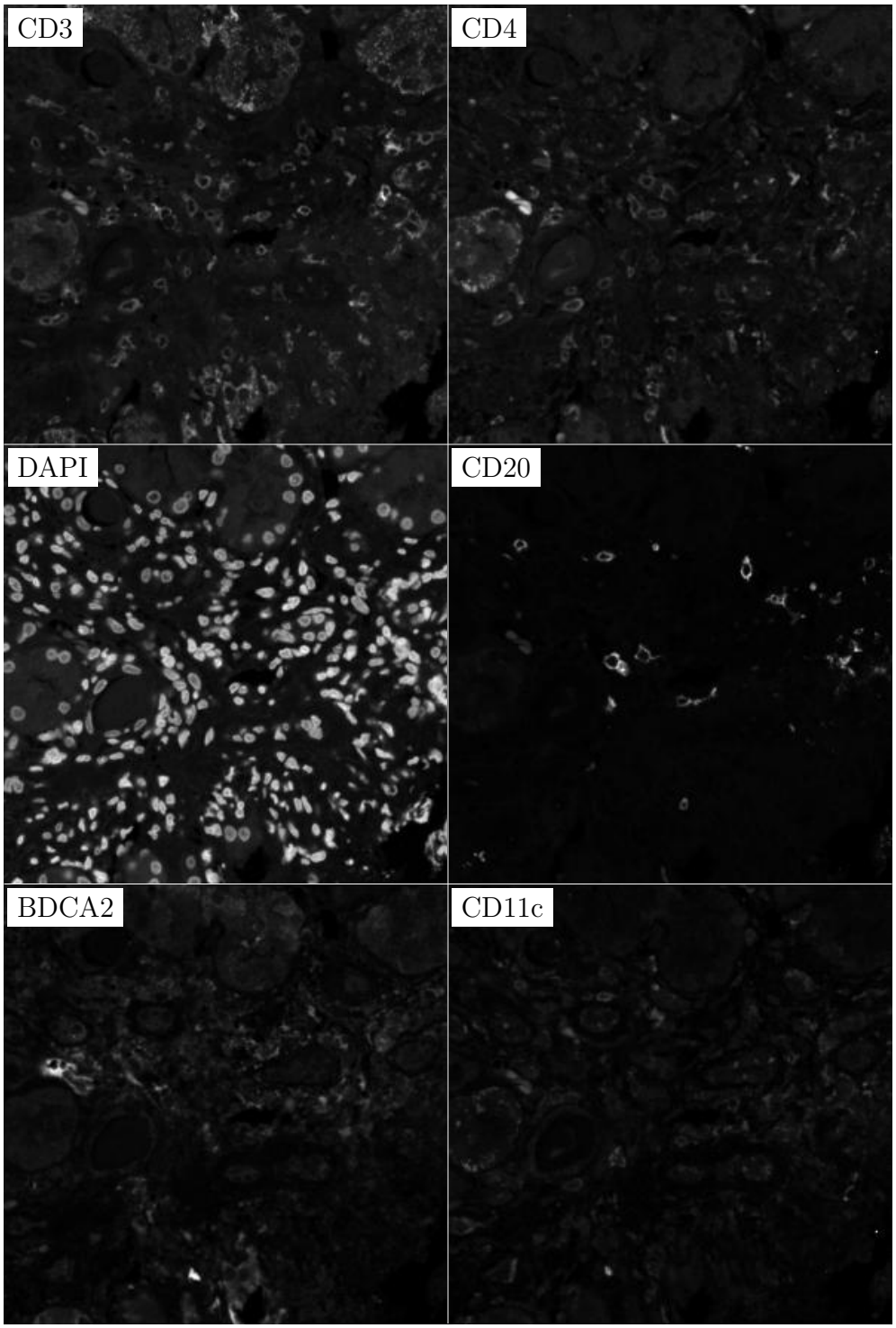

Figure H.60: **ROI for H.59**. True image channel matrix size is 1932 × 1946. Depicted image channel matrix size is 297 × 300.

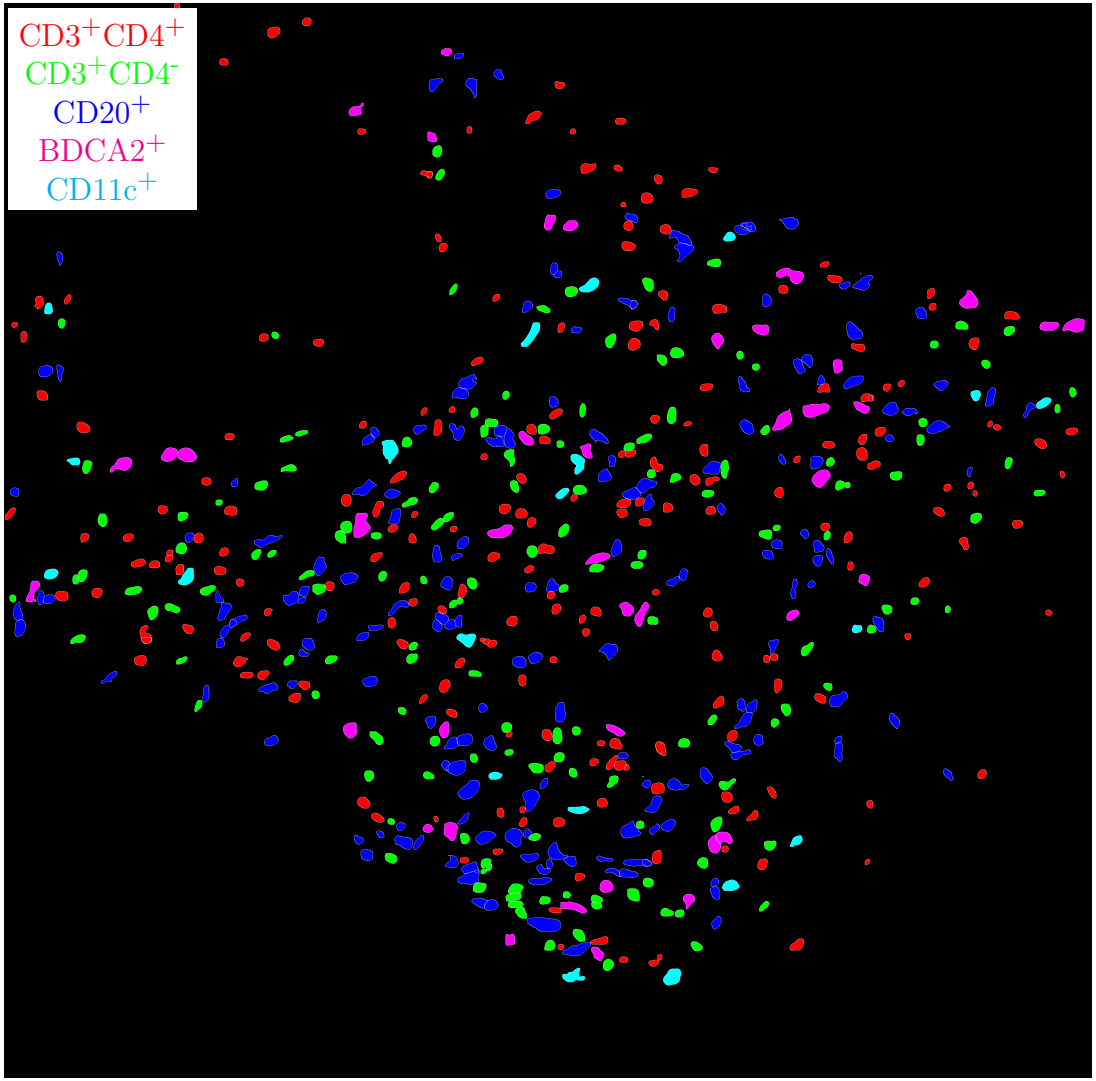

Figure H.61: **Label for H.62**. True label matrix size is 4699 × 4751. Depicted label matrix size is 4699 × 4751.

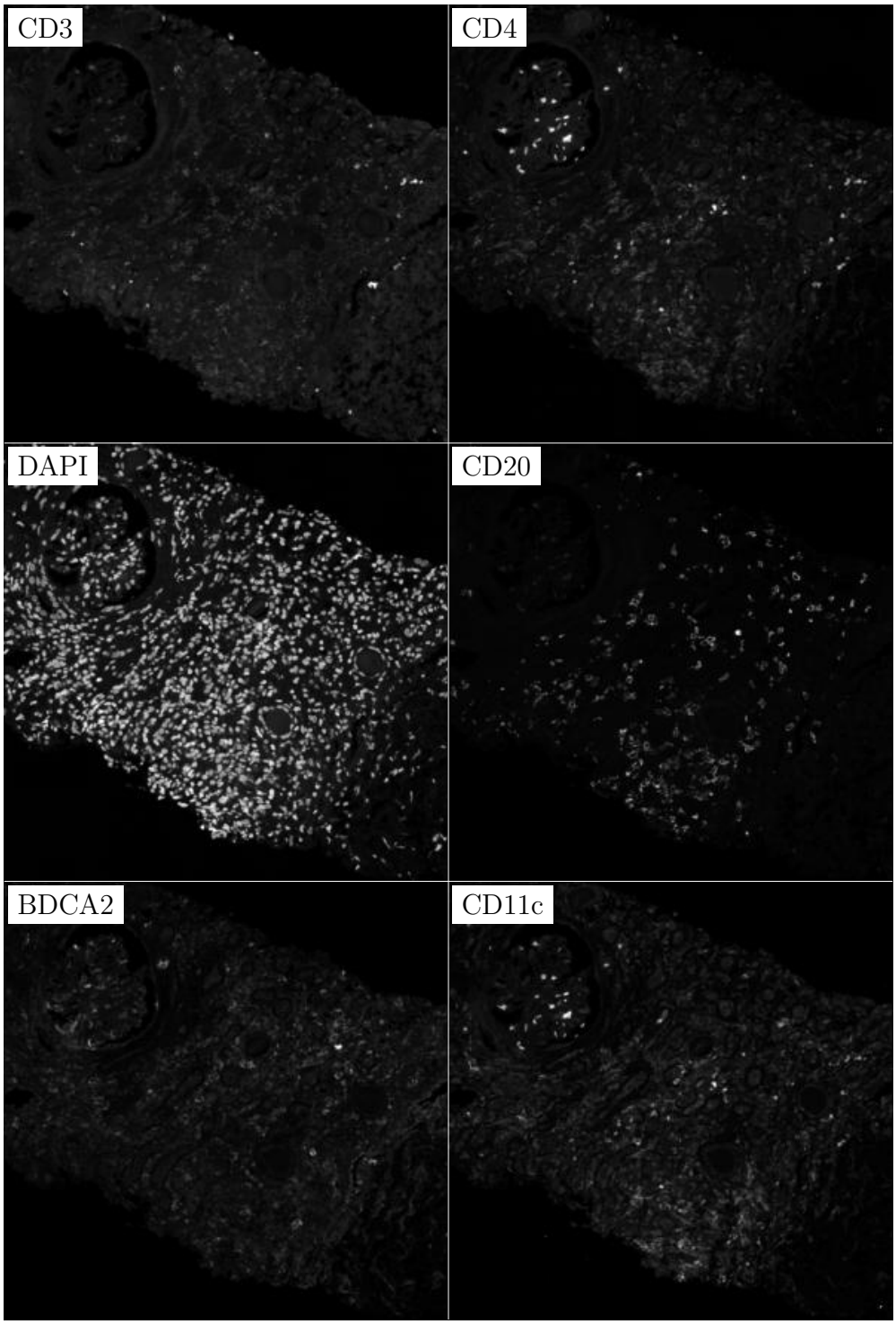

Figure H.62: **ROI for H.61**. True image channel matrix size is 4699 × 4751. Depicted image channel matrix size is 296 × 300.

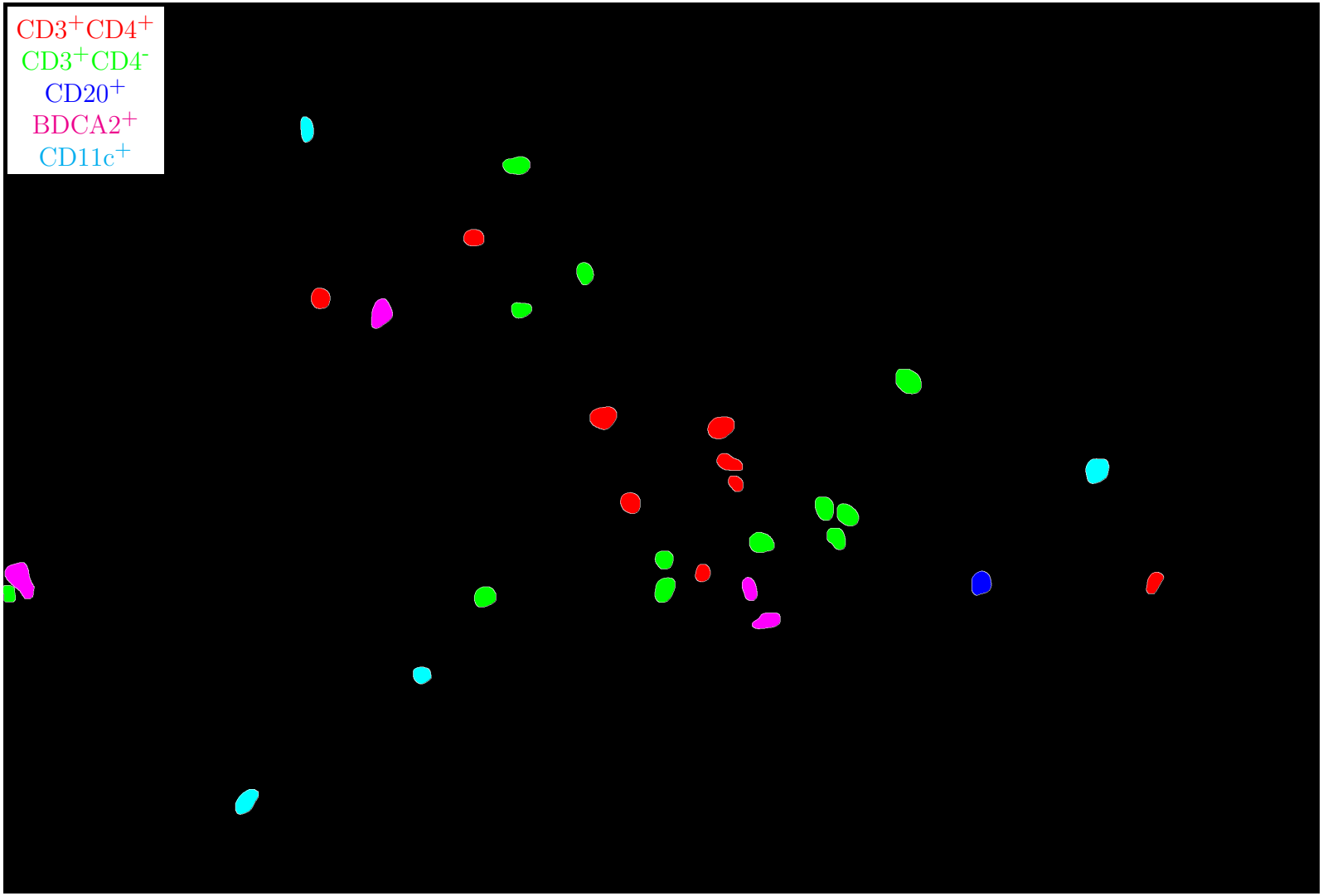

Figure H.63: **Label for H.64**. True label matrix size is 1945 × 2866. Depicted label matrix size is 1945 × 2866.

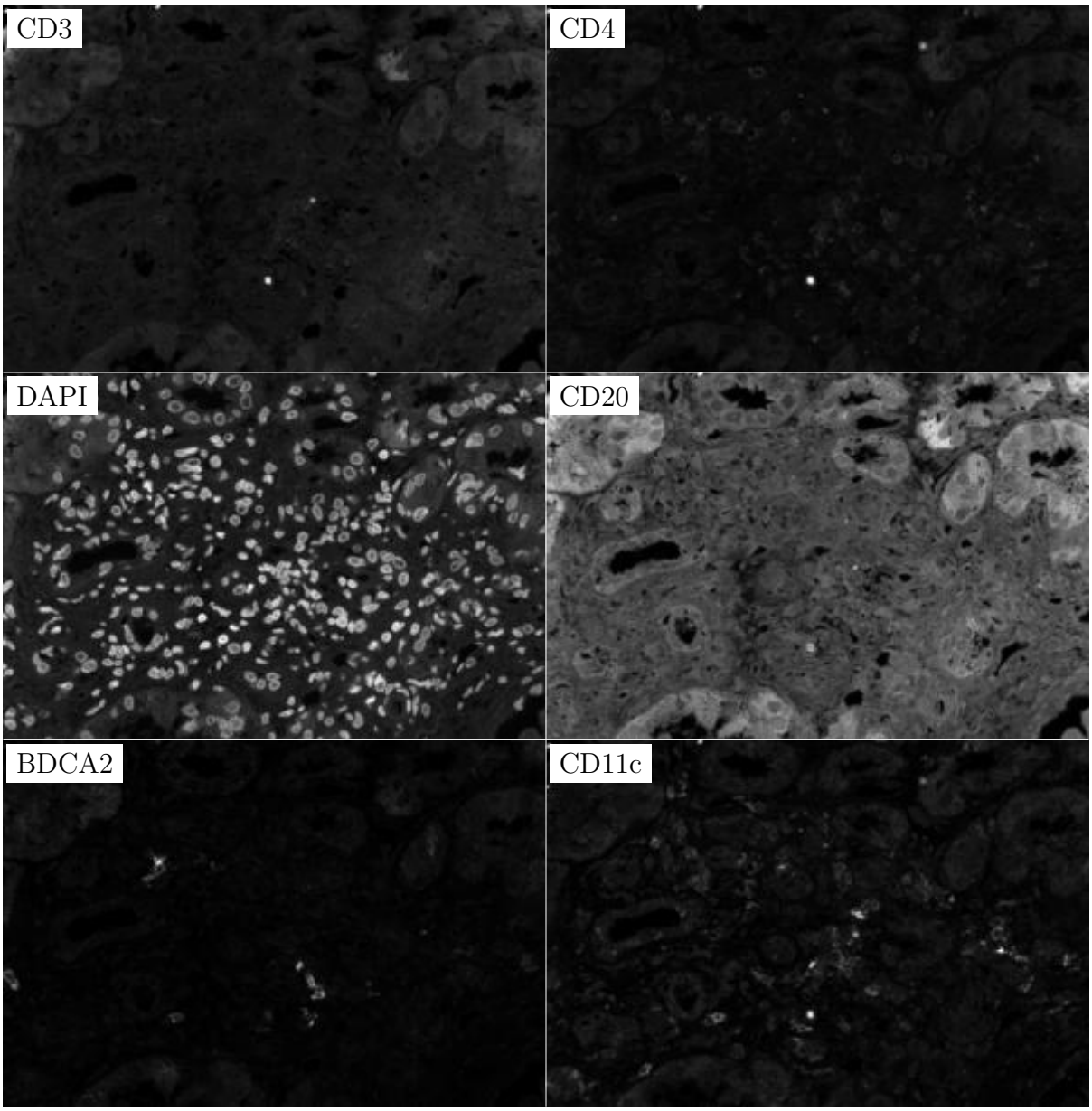

Figure H.64: **ROI for H.63**. True image channel matrix size is 1945 × 2866. Depicted image channel matrix size is 203 × 300.

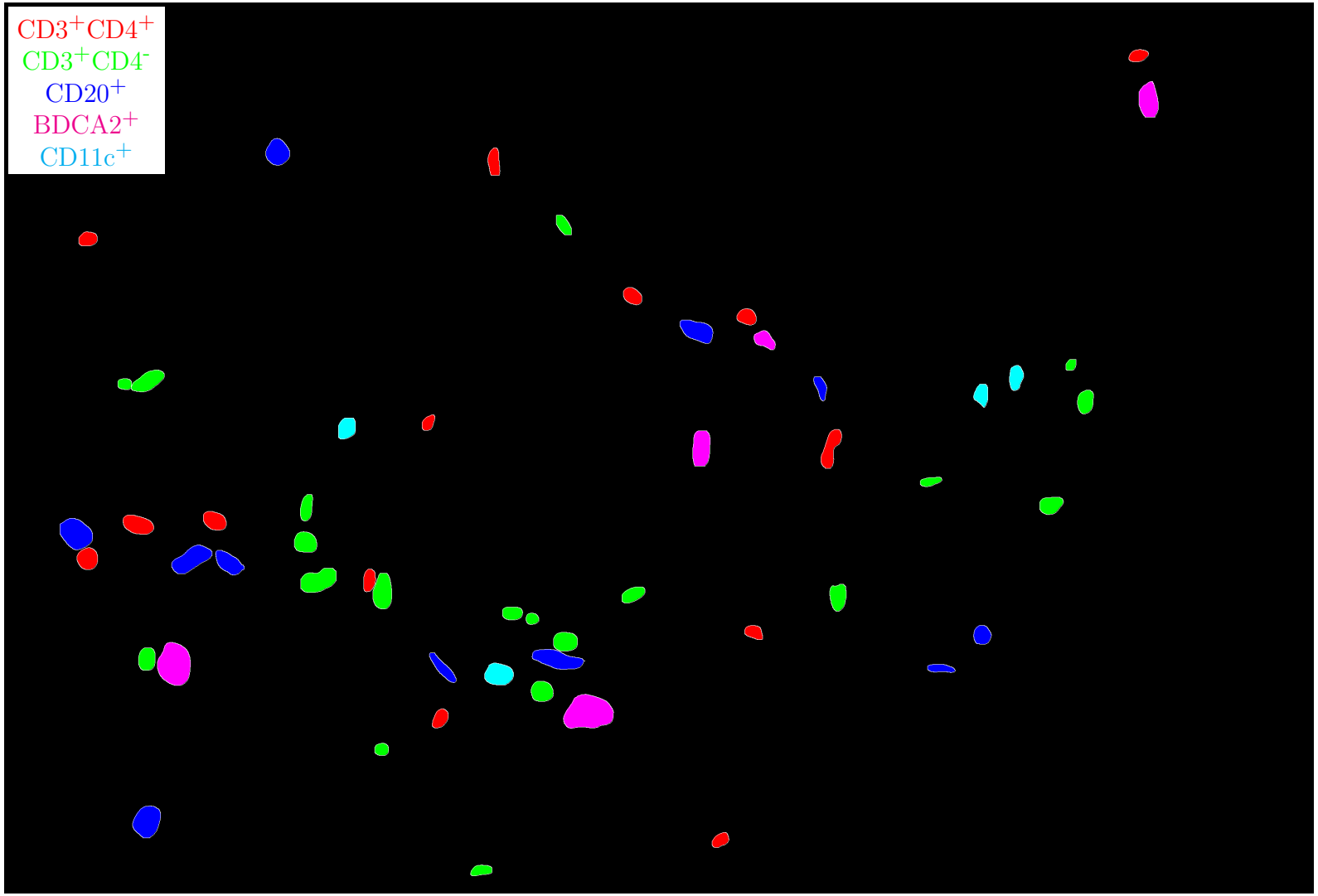

Figure H.65: **Label for H.66**. True label matrix size is 1946 × 2850. Depicted label matrix size is 1946 × 2850.

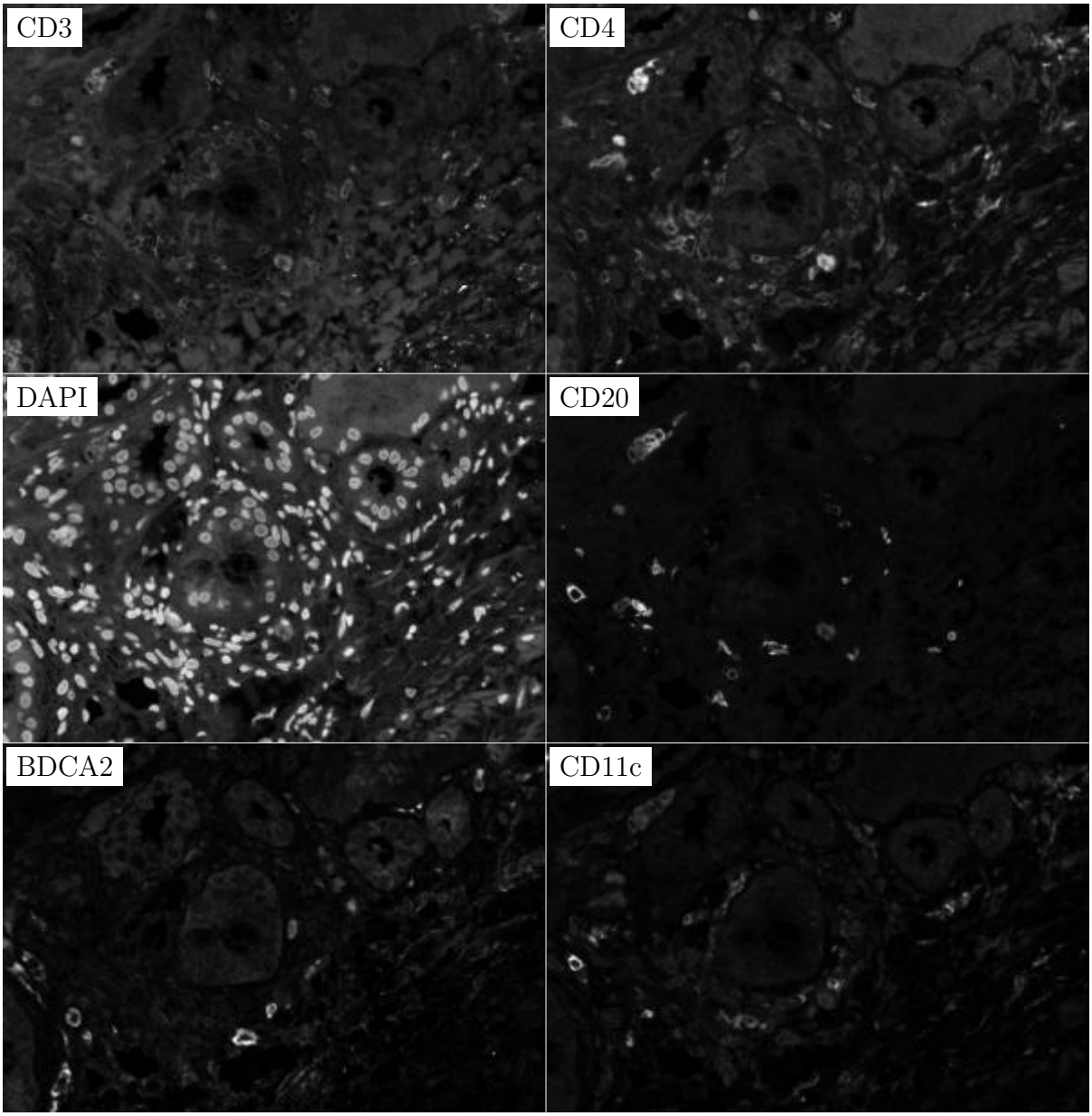

Figure H.66: **ROI for H.65**. True image channel matrix size is 1946 × 2850. Depicted image channel matrix size is 204 × 300.

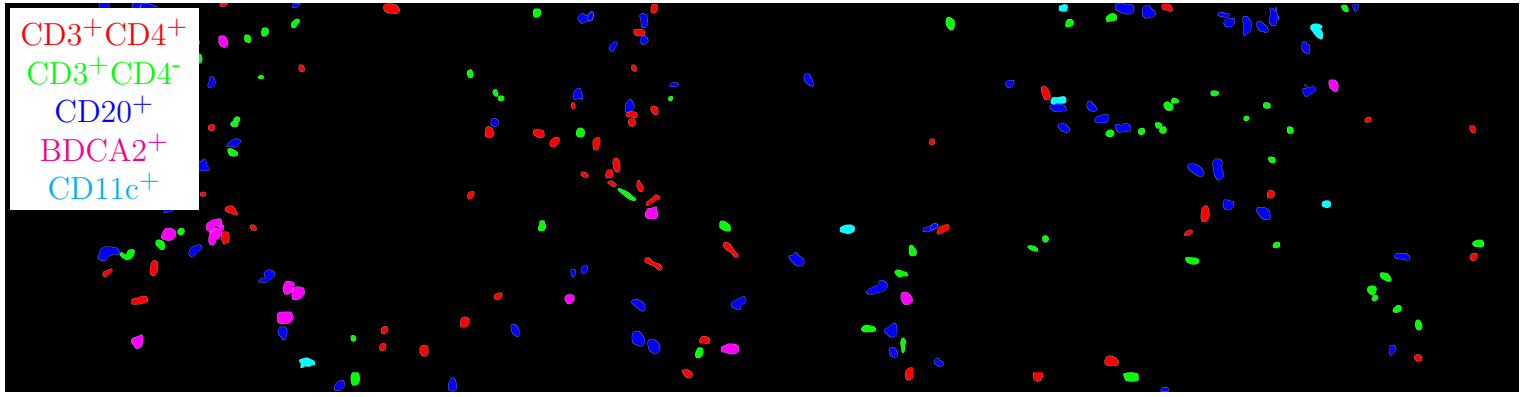

Figure H.67: **Label for H.68**. True label matrix size is 1947 × 7564. Depicted label matrix size is 1947 × 7564.

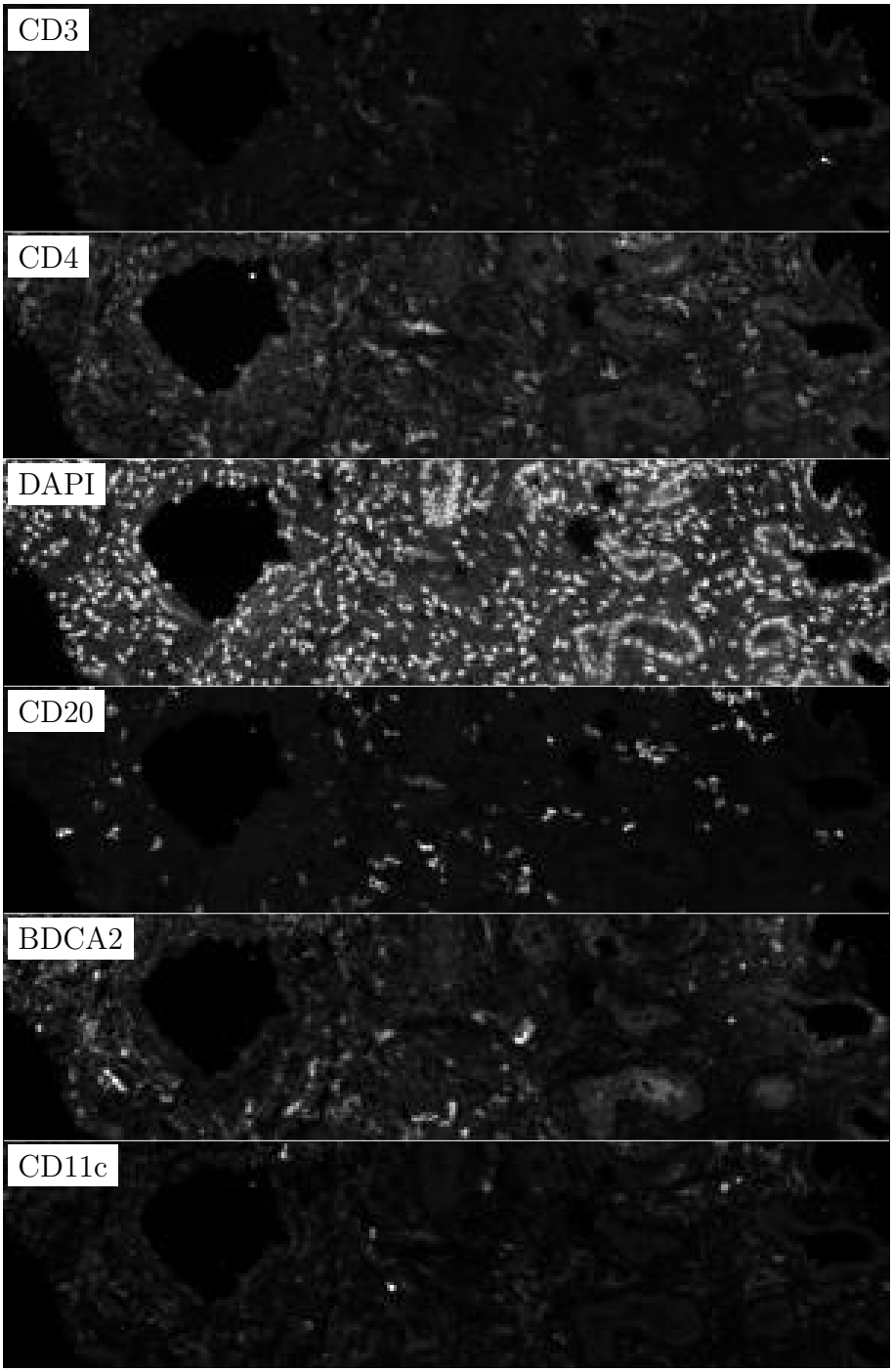

Figure H.68: **ROI for H.67**. True image channel matrix size is 1947×7564. Depicted image channel matrix size is 77×300.

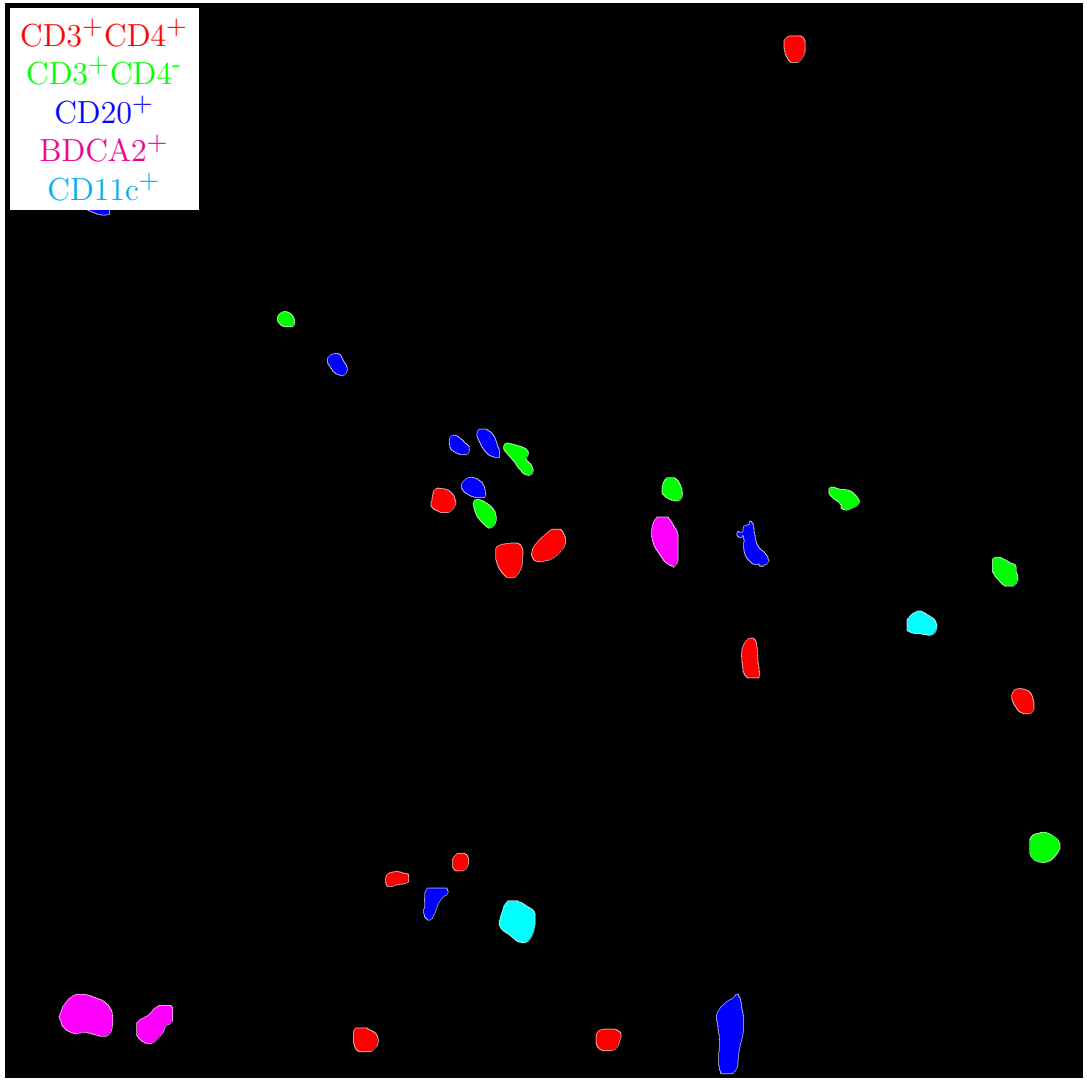

Figure H.69: **Label for H.70**. True label matrix size is 1944 × 1945. Depicted label matrix size is 1944 × 1945.

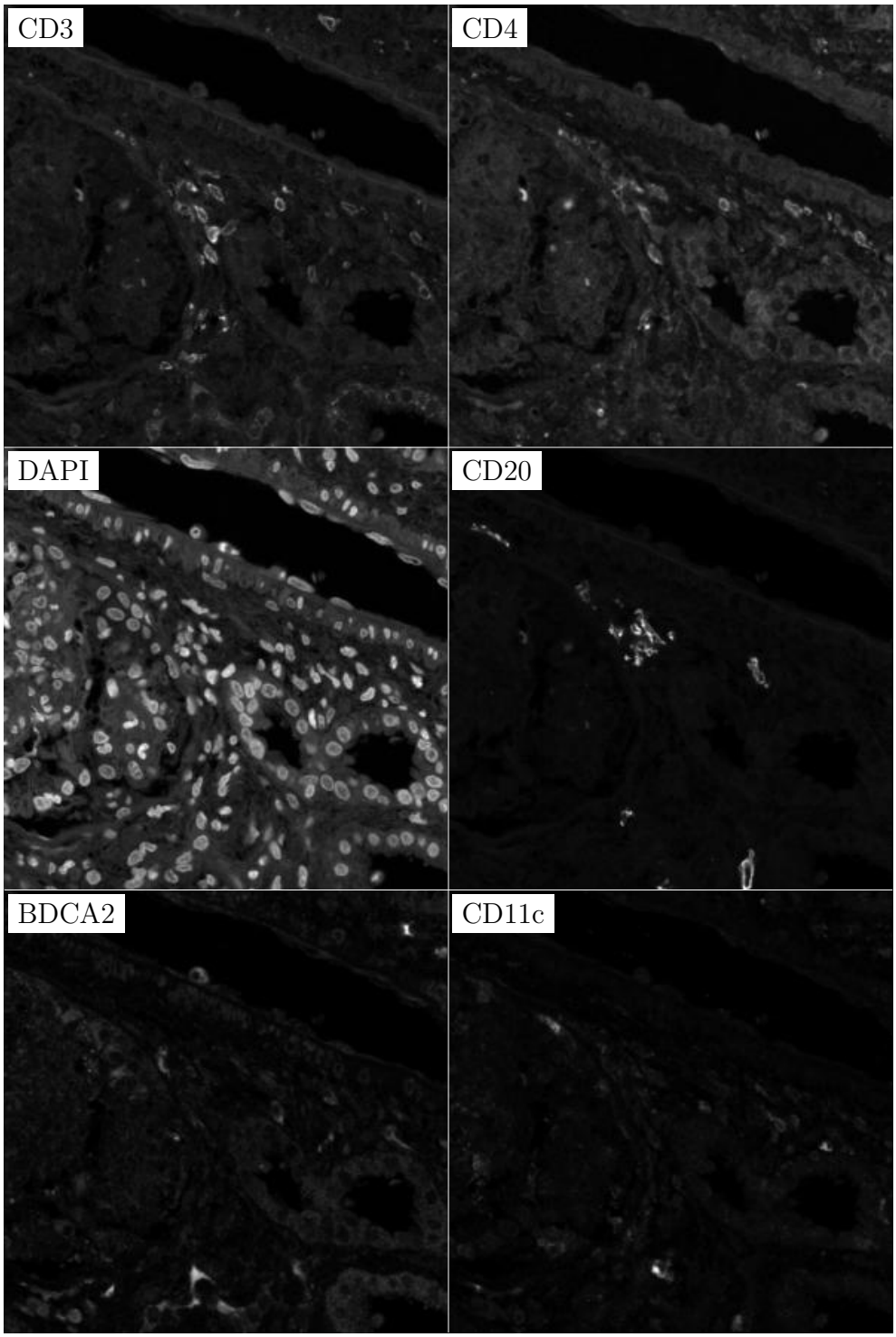

Figure H.70: **ROI for H.69**. True image channel matrix size is 1944 × 1945. Depicted image channel matrix size is 299 × 300.

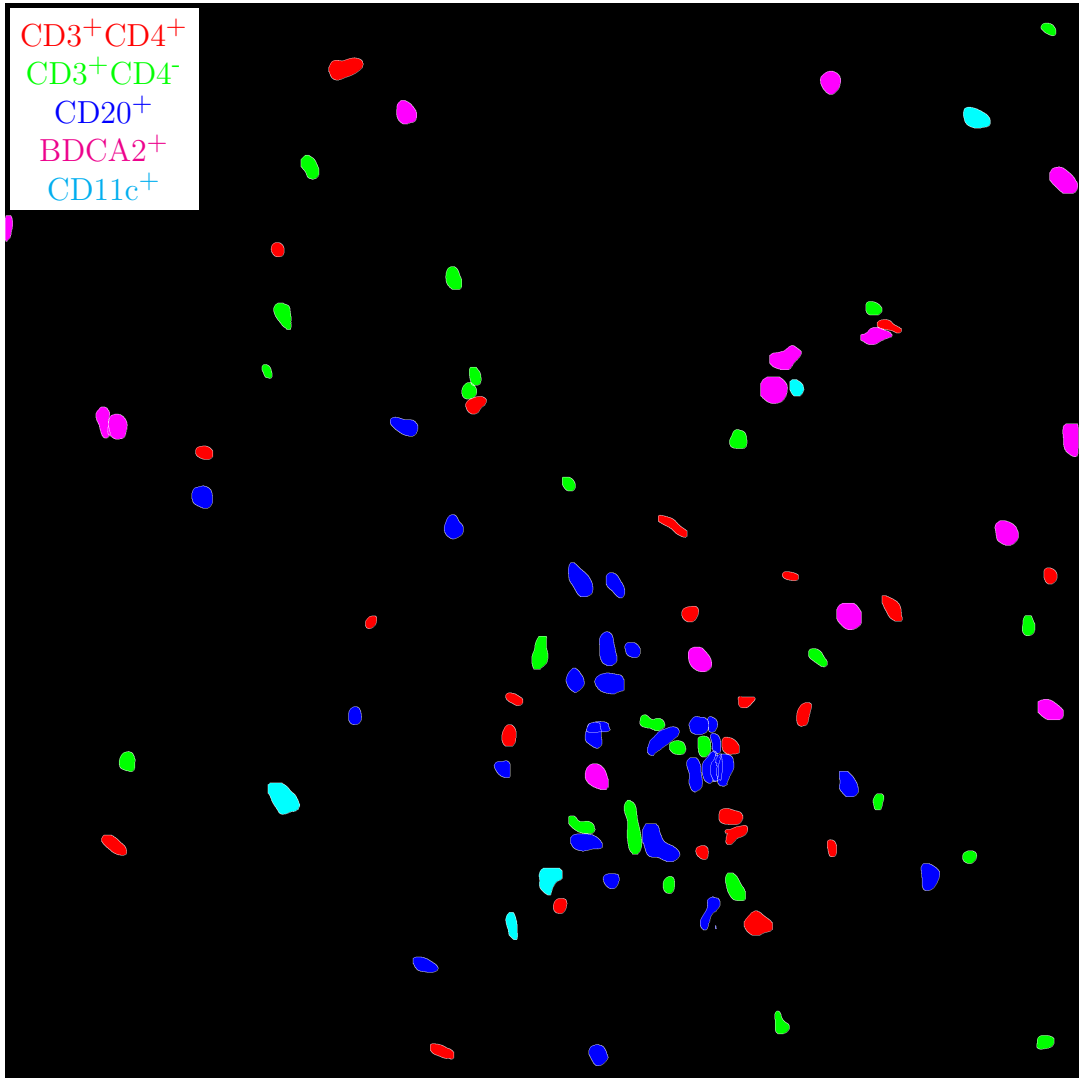

Figure H.71: **Label for H.72**. True label matrix size is 2865 × 2865. Depicted label matrix size is 2865 × 2865.

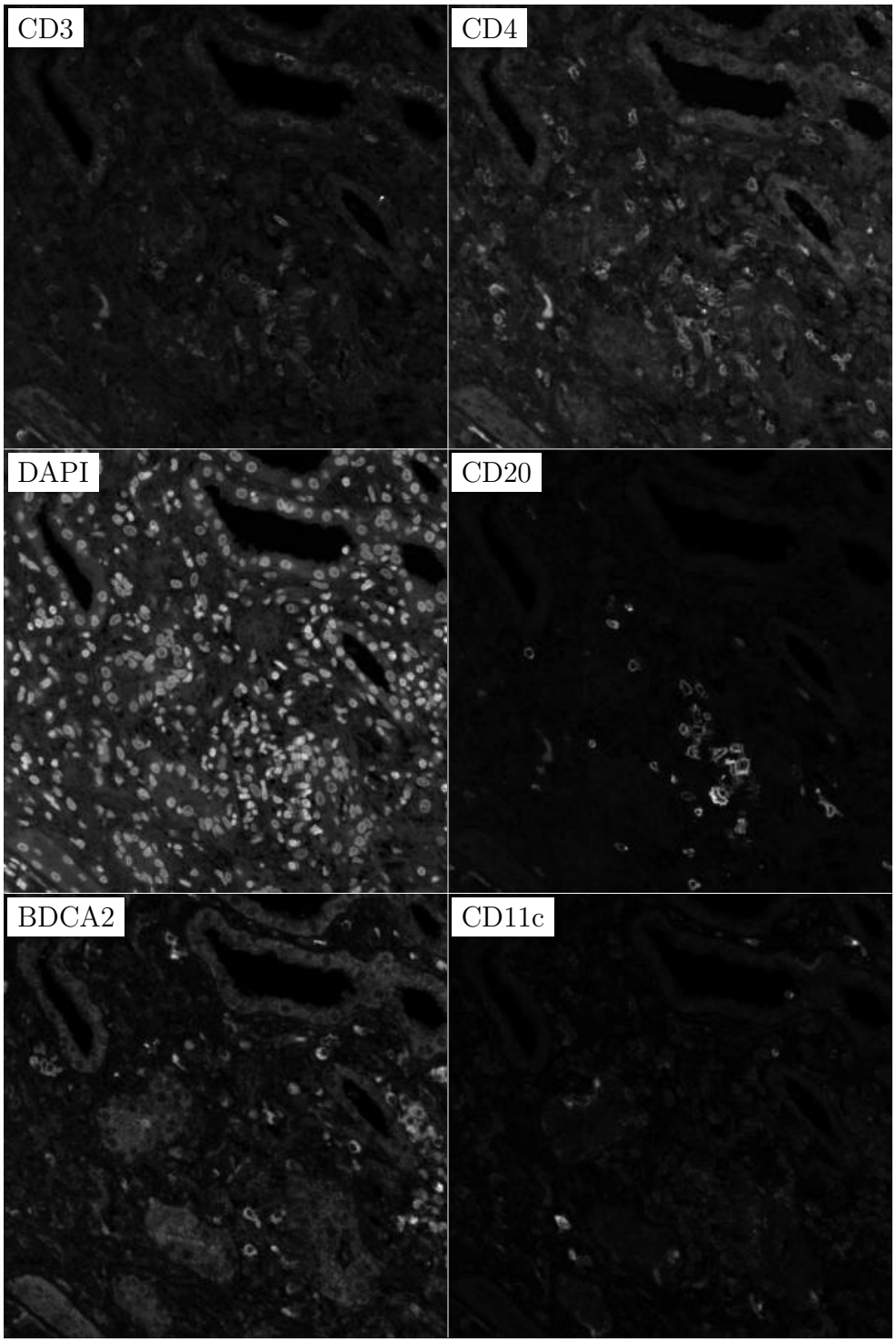

Figure H.72: **ROI for H.71**. True image channel matrix size is 2865 × 2865. Depicted image channel matrix size is 300 × 300.

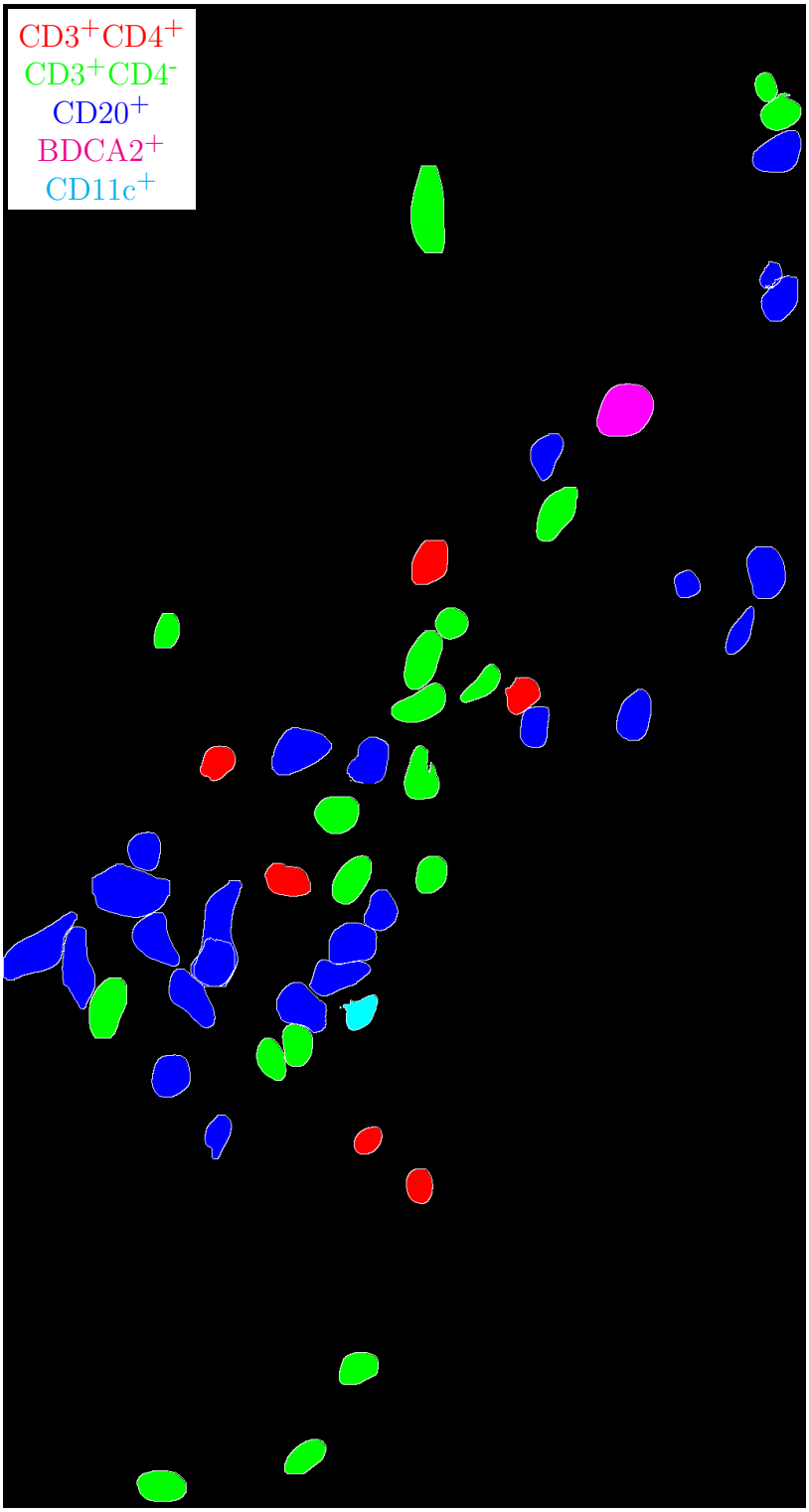

Figure H.73: **Label for H.74**. True label matrix size is 1919×1028. Depicted label matrix size is 1919×1028.

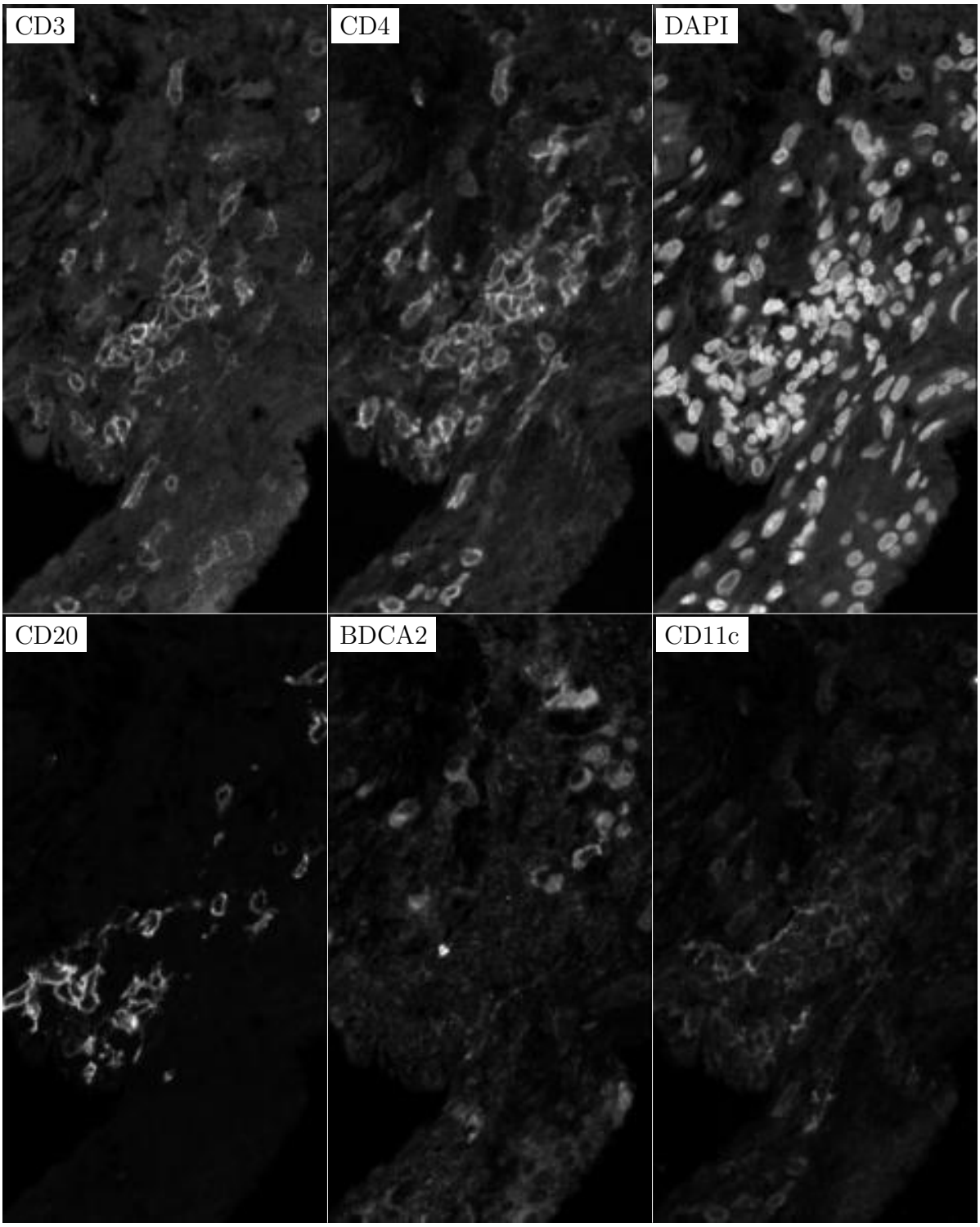

Figure H.74: **ROI for H.73**. True image channel matrix size is 1919 × 1028. Depicted image channel matrix size is 300 × 160.

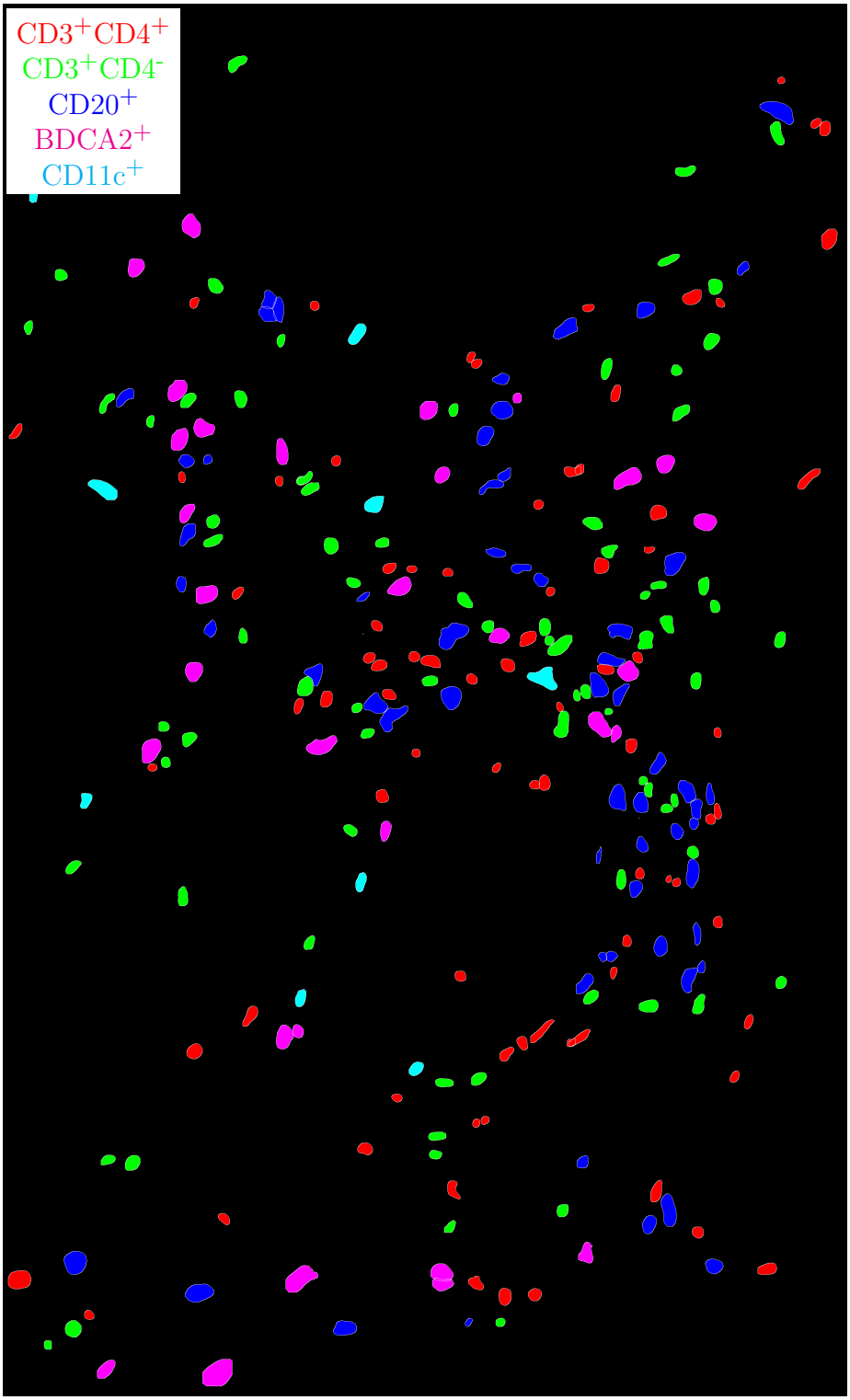

Figure H.75: **Label for H.76**. True label matrix size is 4707×2866. Depicted label matrix size is 4707×2866.

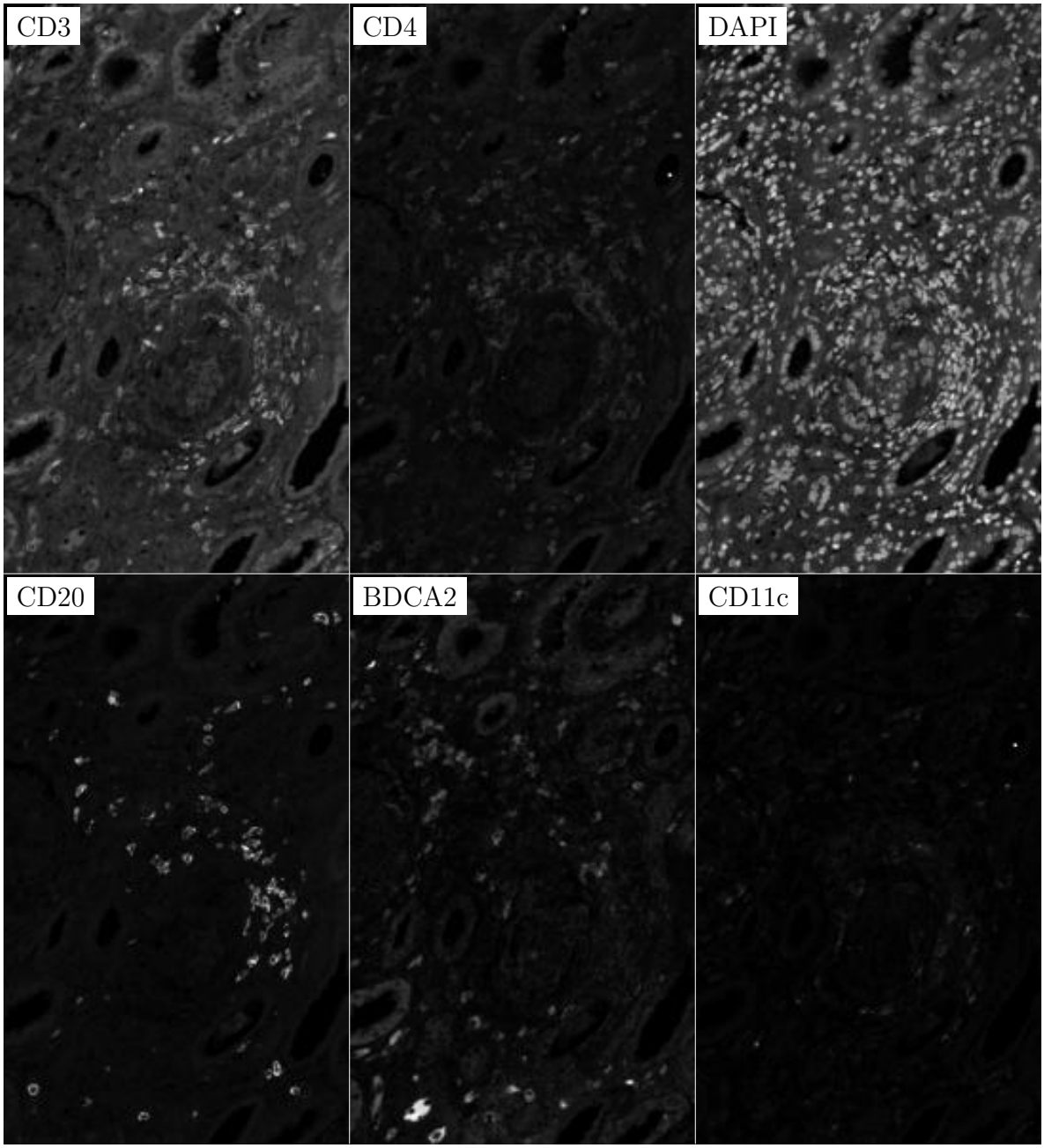

Figure H.76: **ROI for H.75**. True image channel matrix size is 4707 × 2866. Depicted image channel matrix size is 300 × 182.

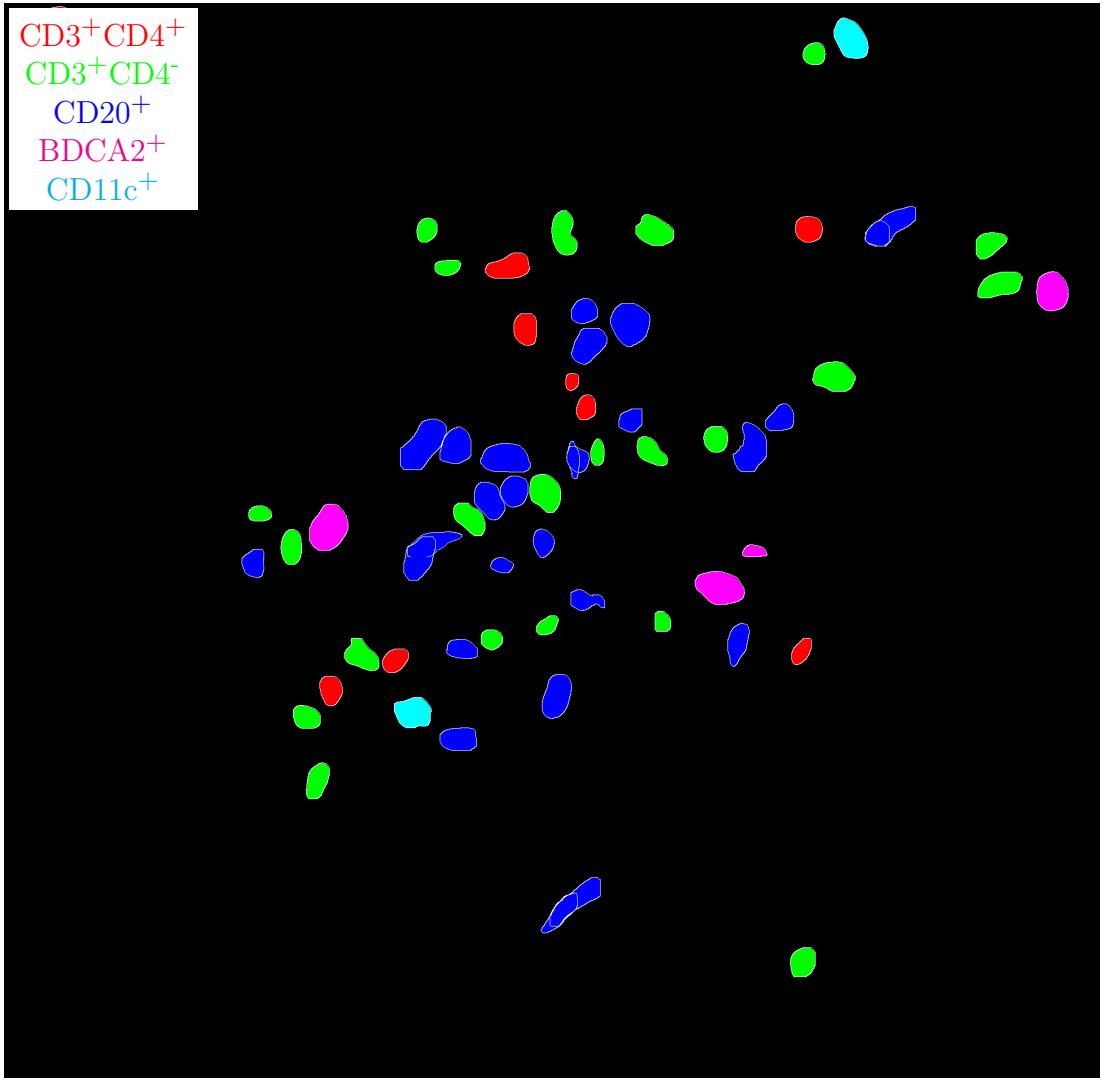

Figure H.77: **Label for H.78**. True label matrix size is 1908 × 1942. Depicted label matrix size is 1908 × 1942.

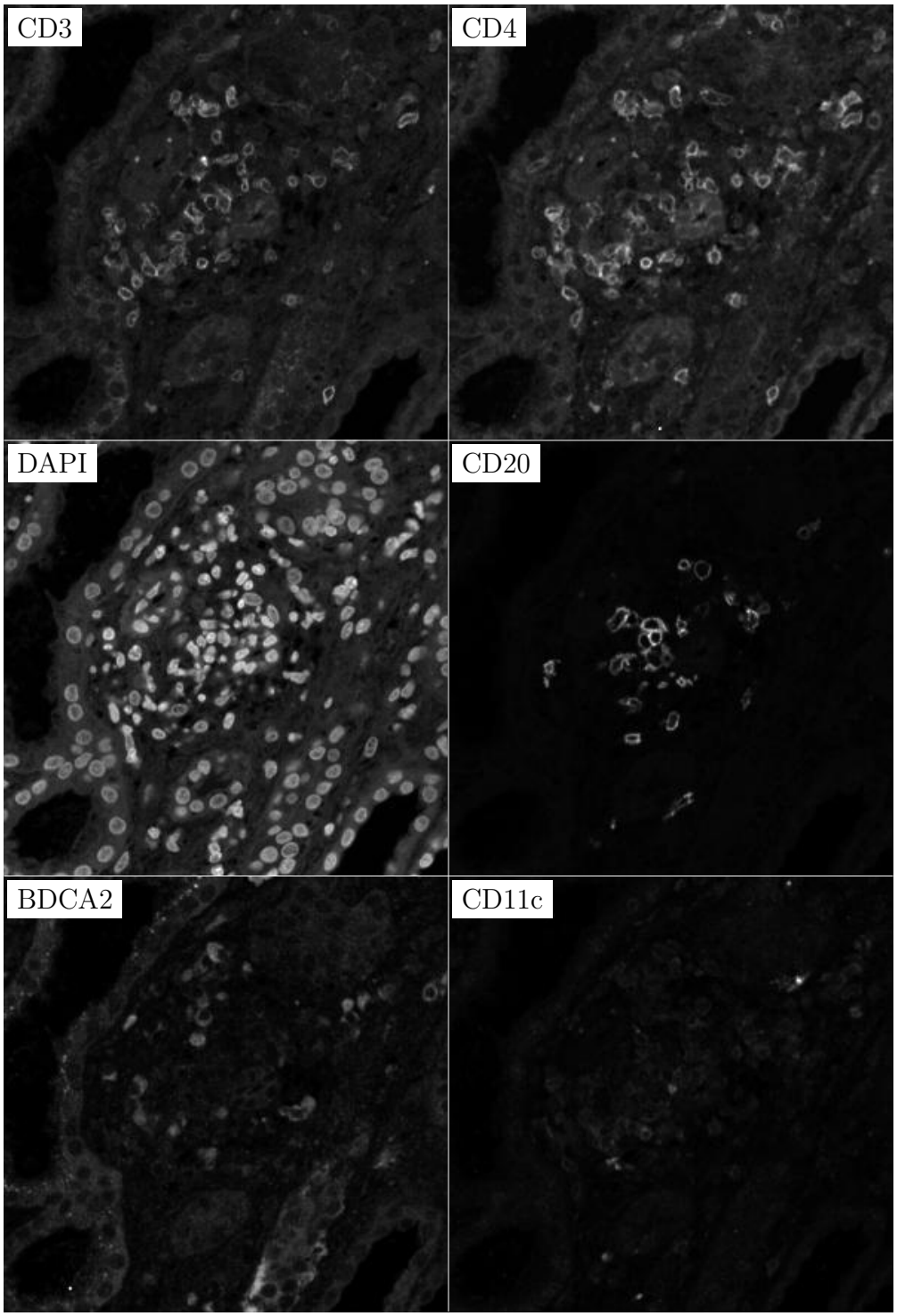

Figure H.78: **ROI for H.77**. True image channel matrix size is 1908 × 1942. Depicted image channel matrix size is 294 × 300.

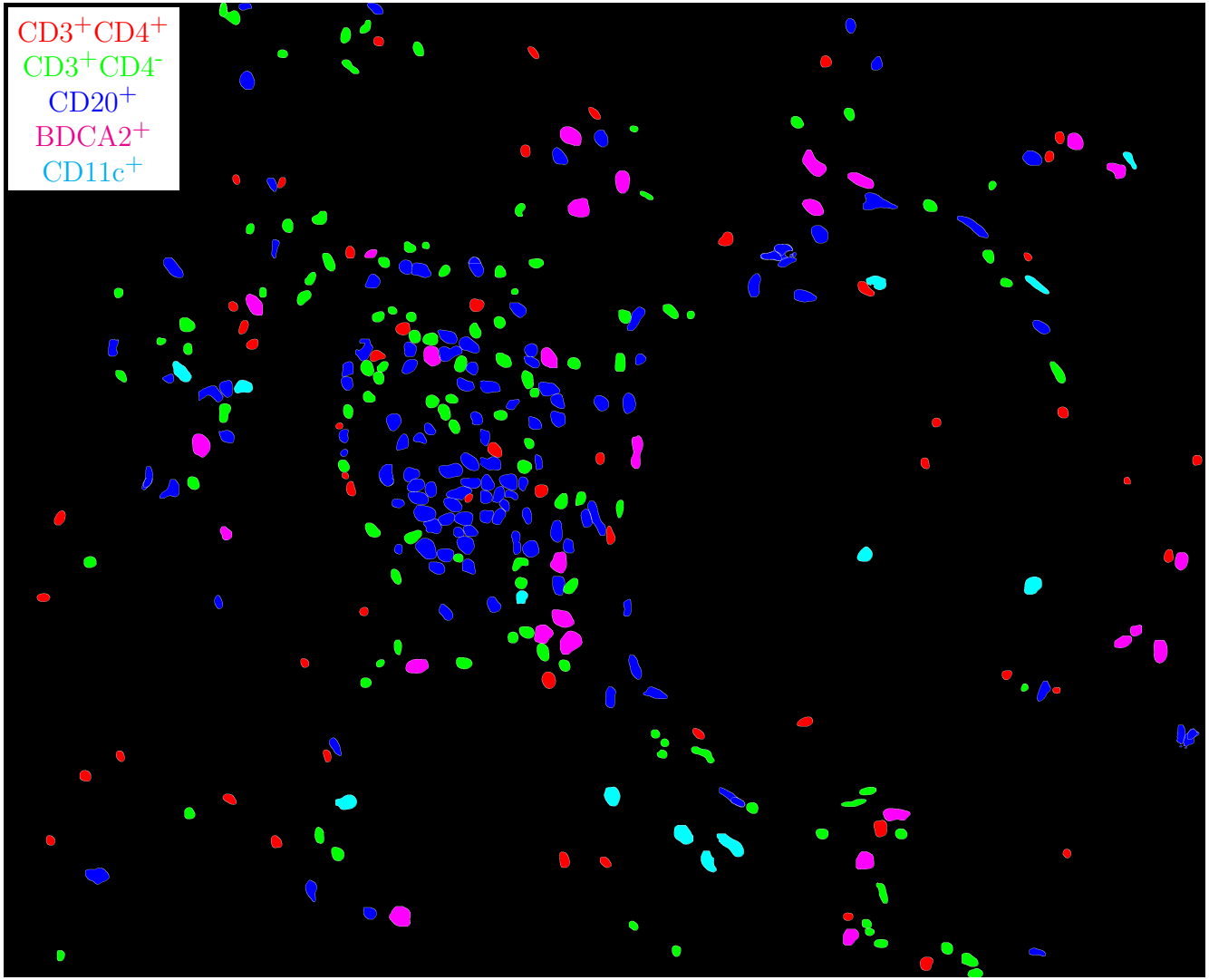

Figure H.79: **Label for H.80**. True label matrix size is 3789 × 4661. Depicted label matrix size is 3789 × 4661.

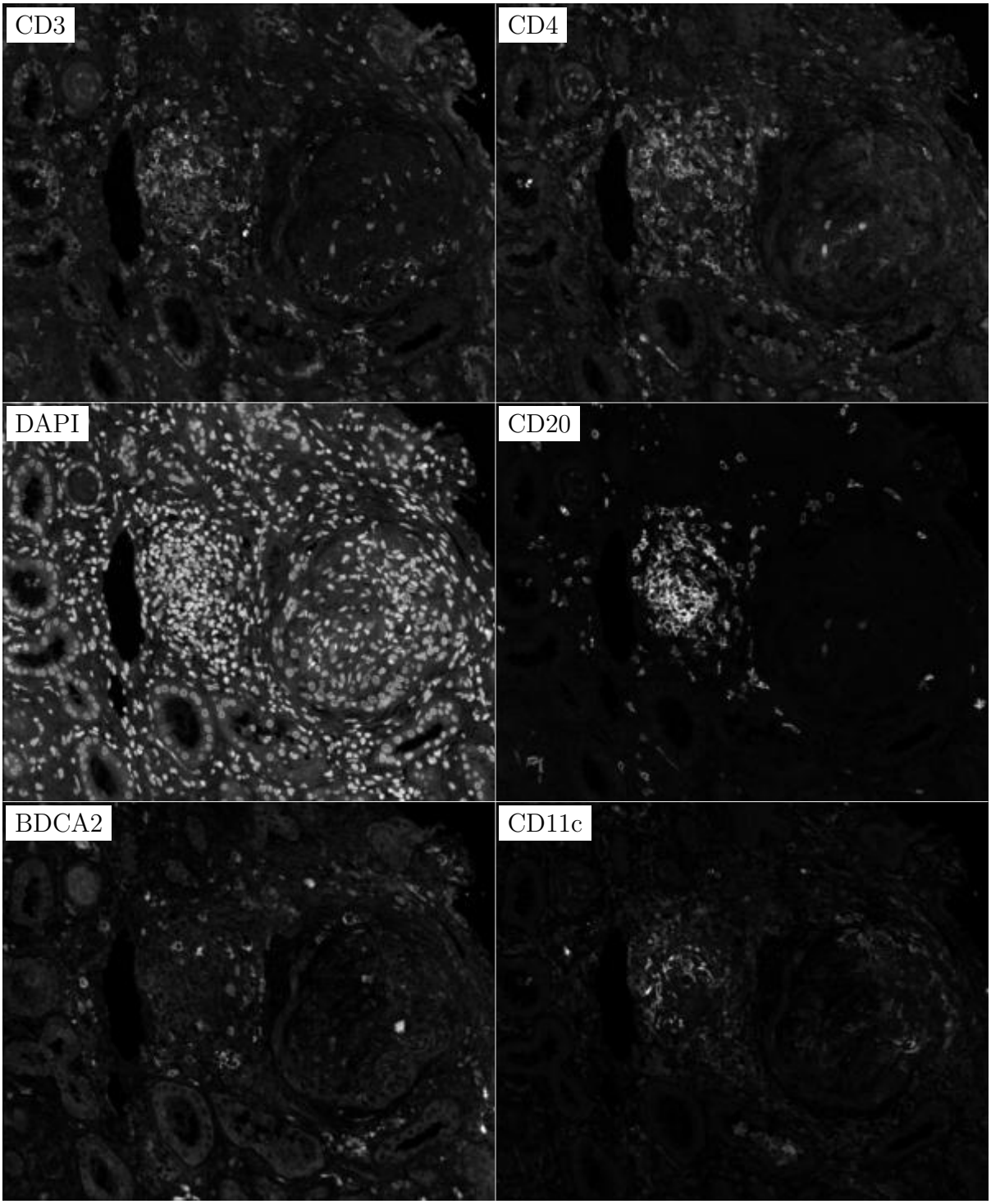

Figure H.80: **ROI for H.79**. True image channel matrix size is 3789×4661. Depicted image channel matrix size is 243×300.

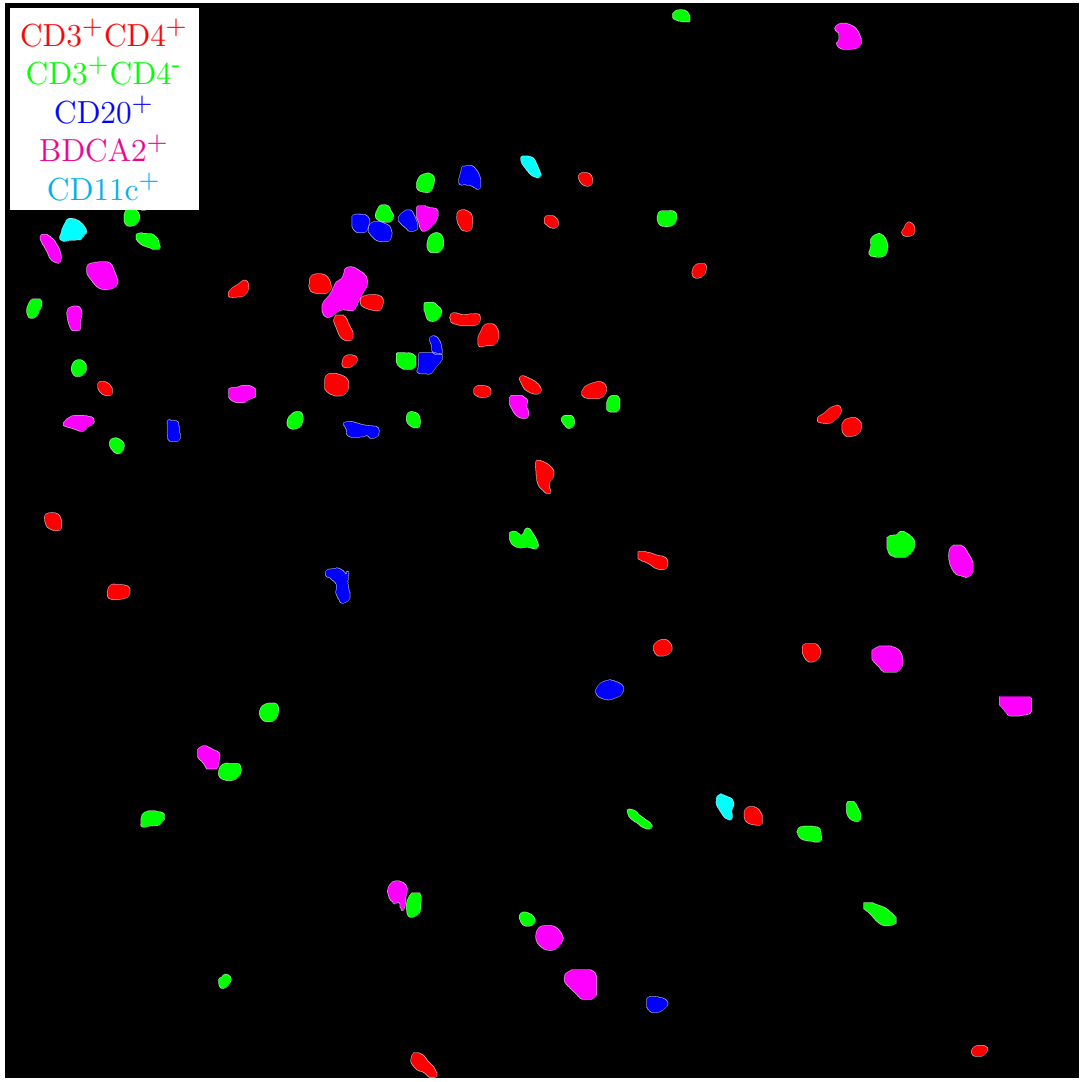

Figure H.81: **Label for H.82**. True label matrix size is 2865 × 2865. Depicted label matrix size is 2865 × 2865.

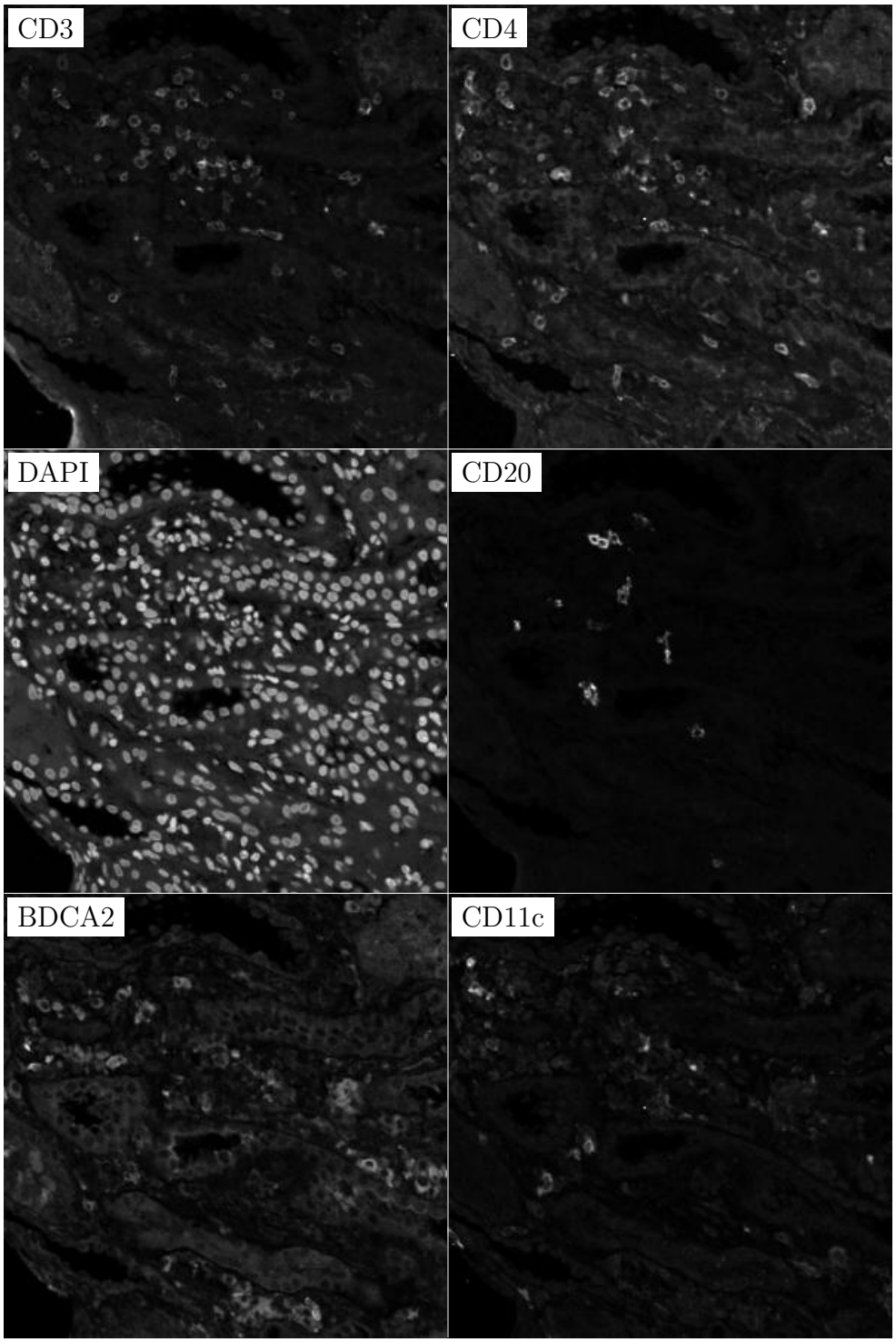

Figure H.82: **ROI for H.81**. True image channel matrix size is 2865 × 2865. Depicted image channel matrix size is 300 × 300.

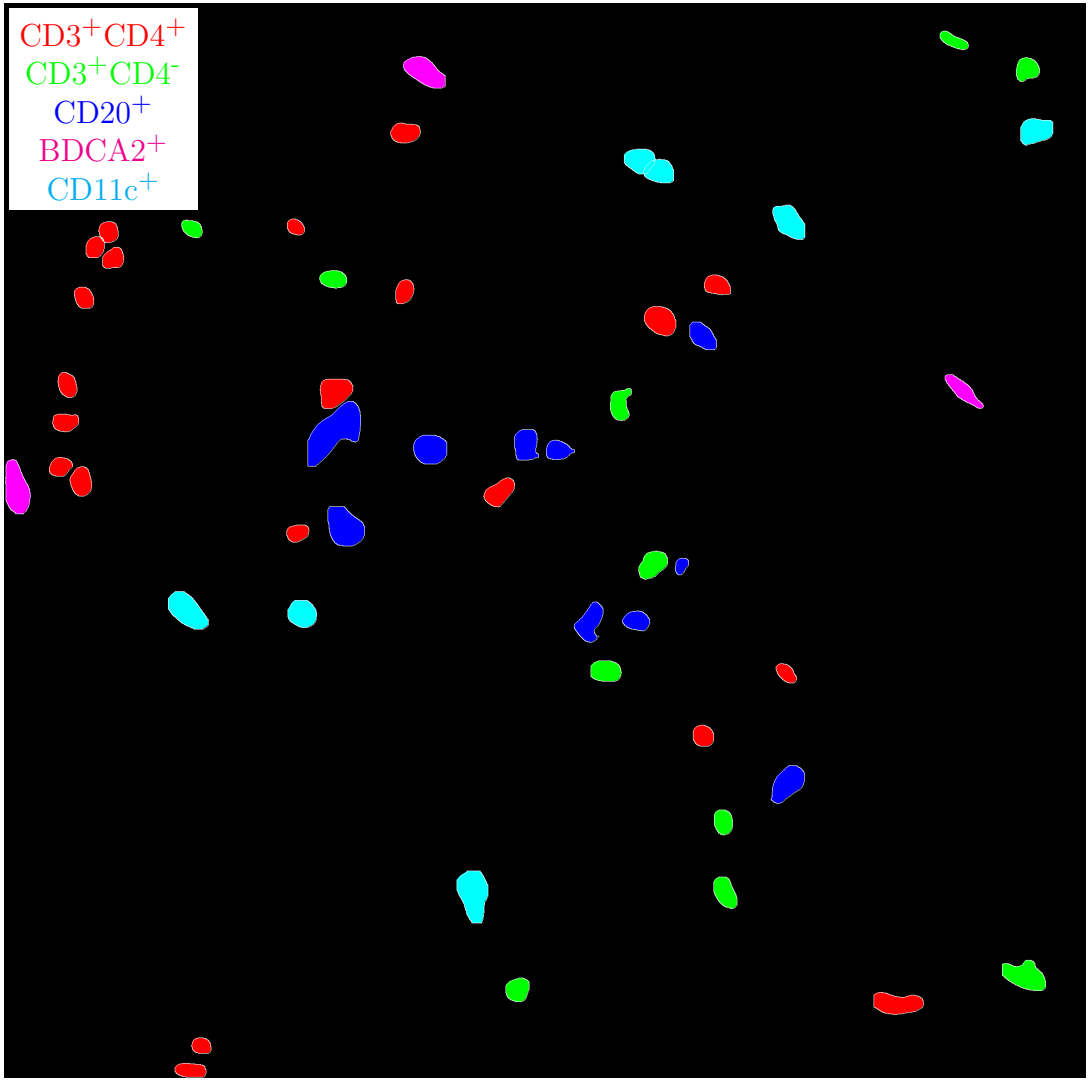

Figure H.83: **Label for H.84**. True label matrix size is 1939 × 1946. Depicted label matrix size is 1939 × 1946.

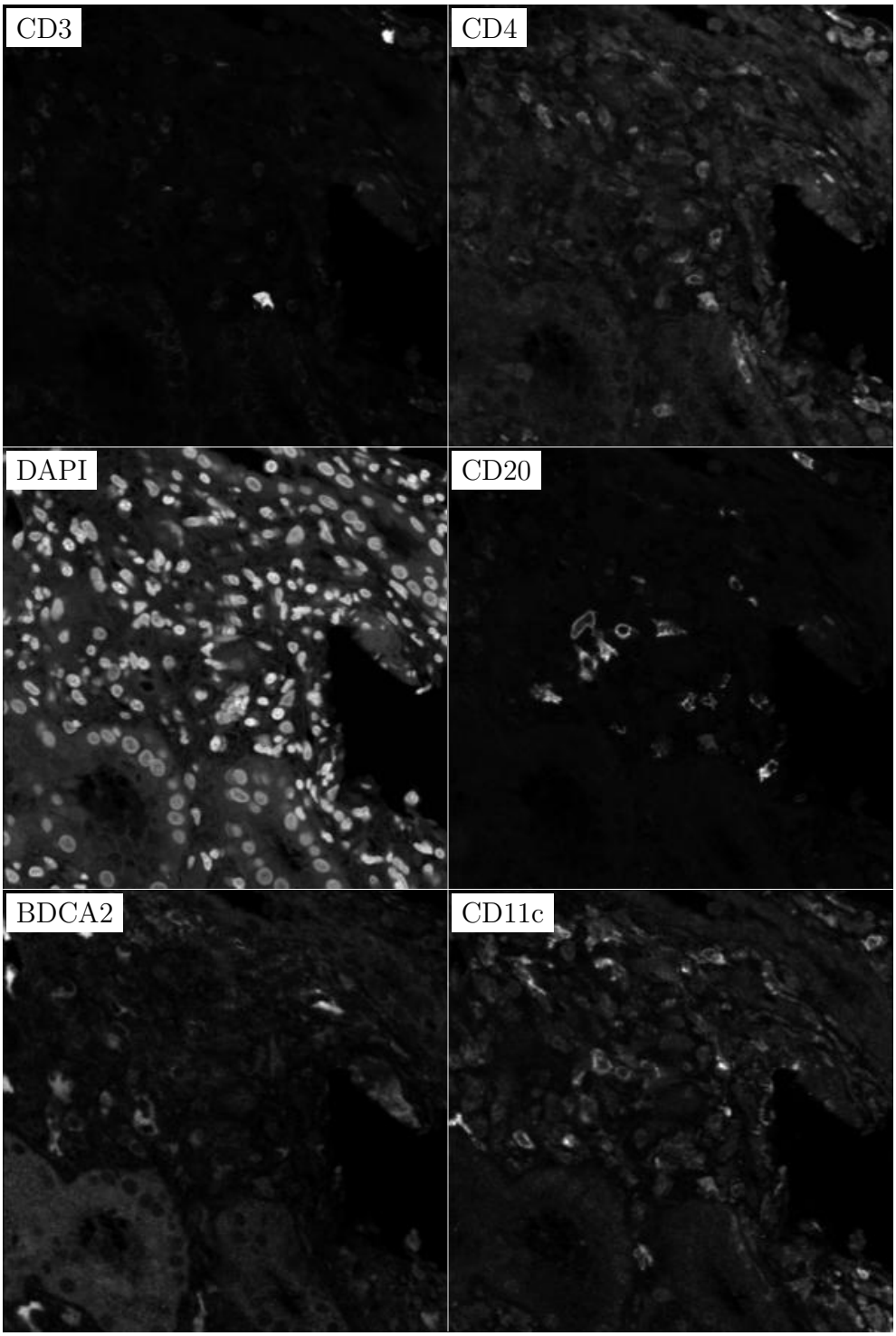

Figure H.84: **ROI for H.83**. True image channel matrix size is 1939 × 1946. Depicted image channel matrix size is 298 × 300.

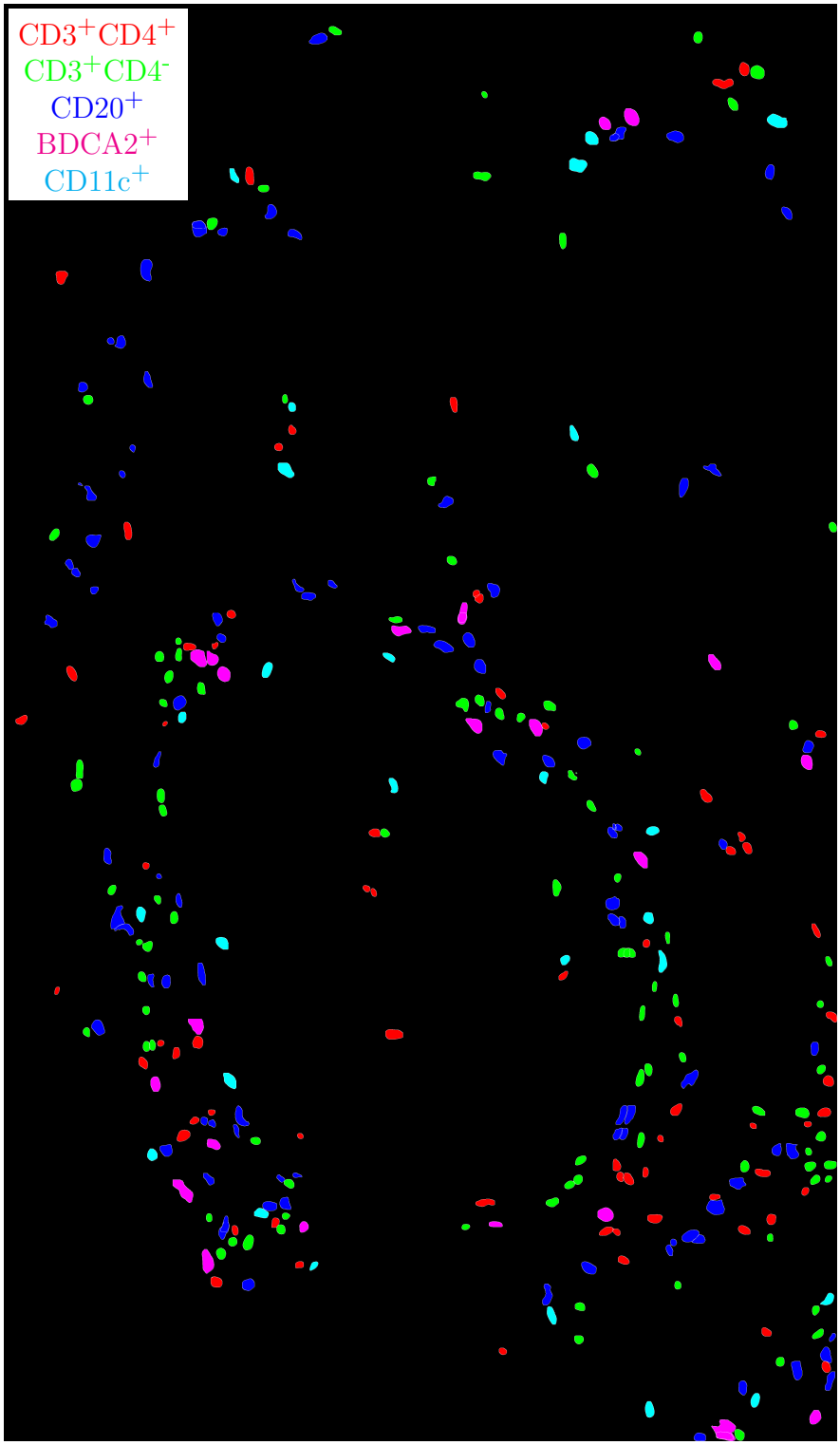

Figure H.85: **Label for H.86**. True label matrix size is 6543×3795. Depicted label matrix size is 6543×3795.

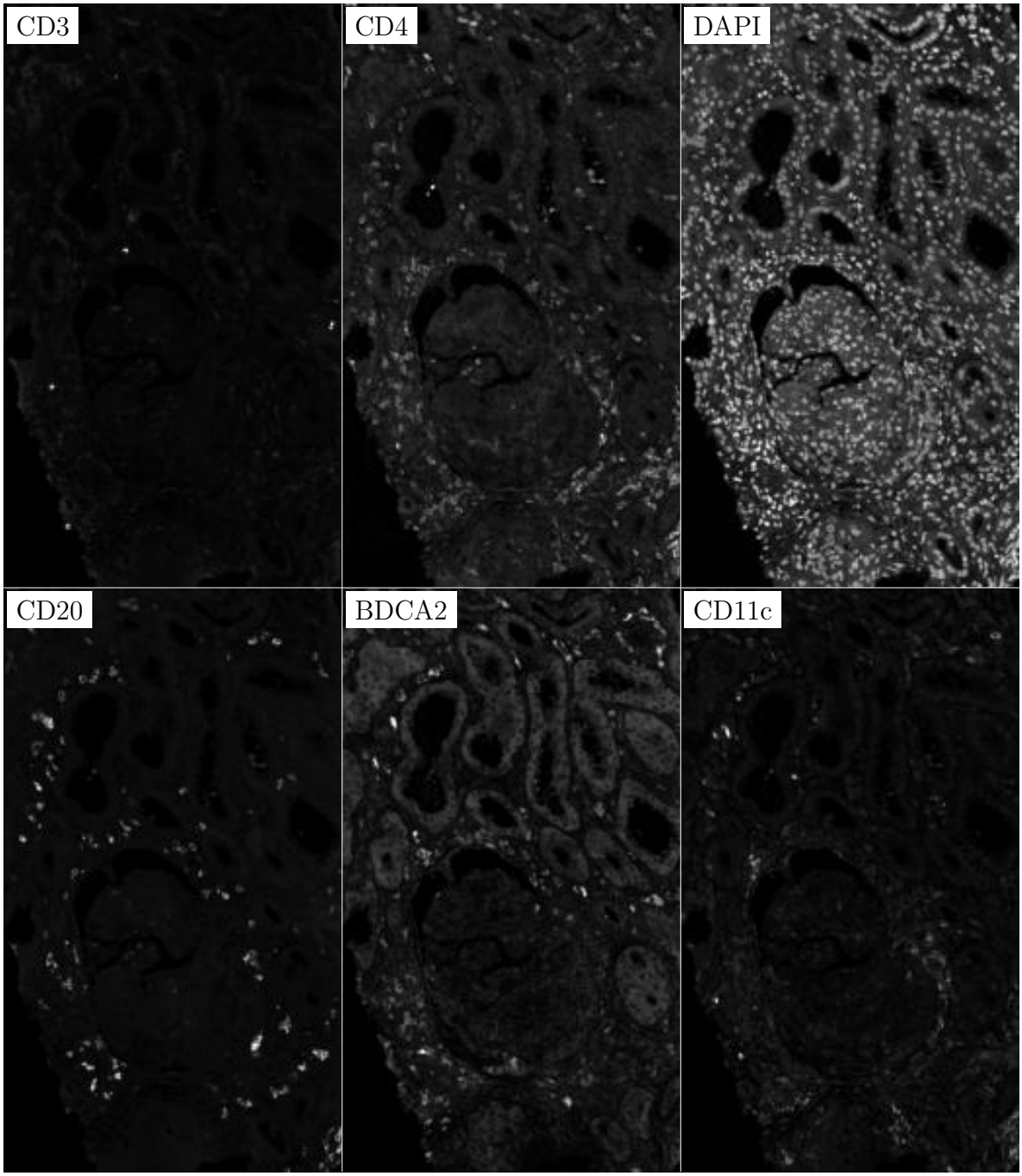

Figure H.86: **ROI for H.85**. True image channel matrix size is 6543×3795. Depicted image channel matrix size is 300×174.

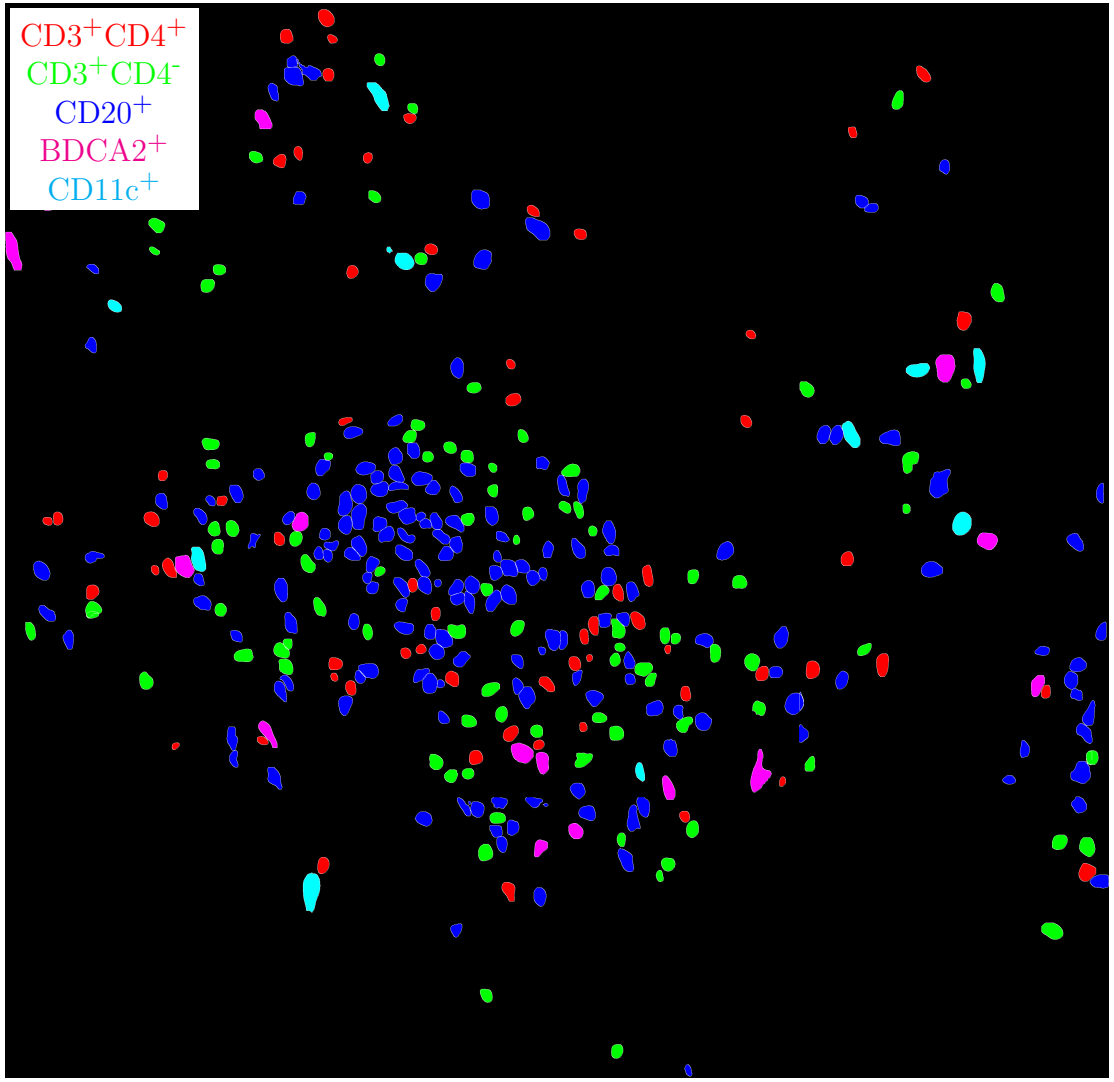

Figure H.87: **Label for H.88**. True label matrix size is 3752×3854. Depicted label matrix size is 3752×3854.

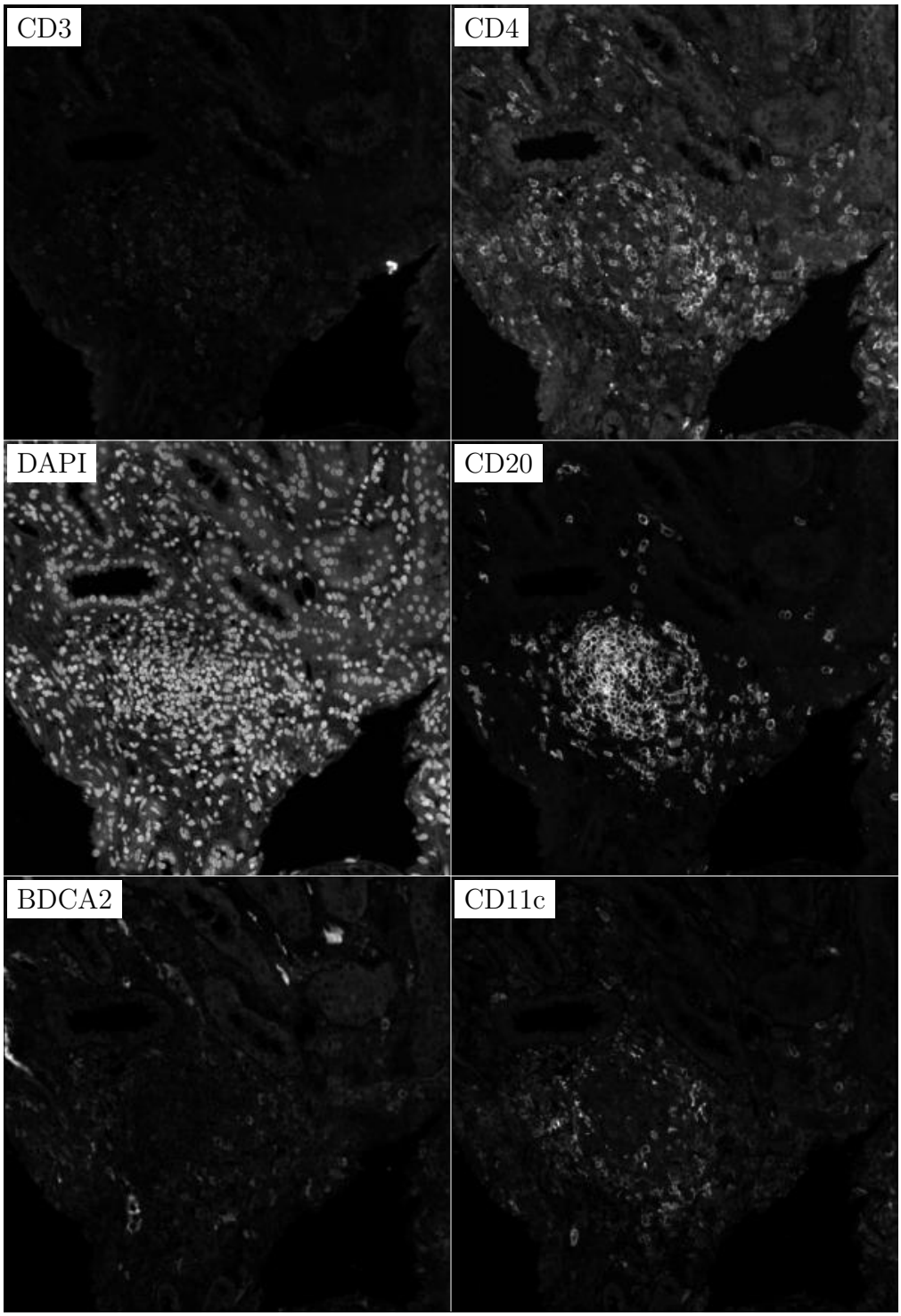

Figure H.88: **ROI for H.87**. True image channel matrix size is 3752 × 3854. Depicted image channel matrix size is 292 × 300.

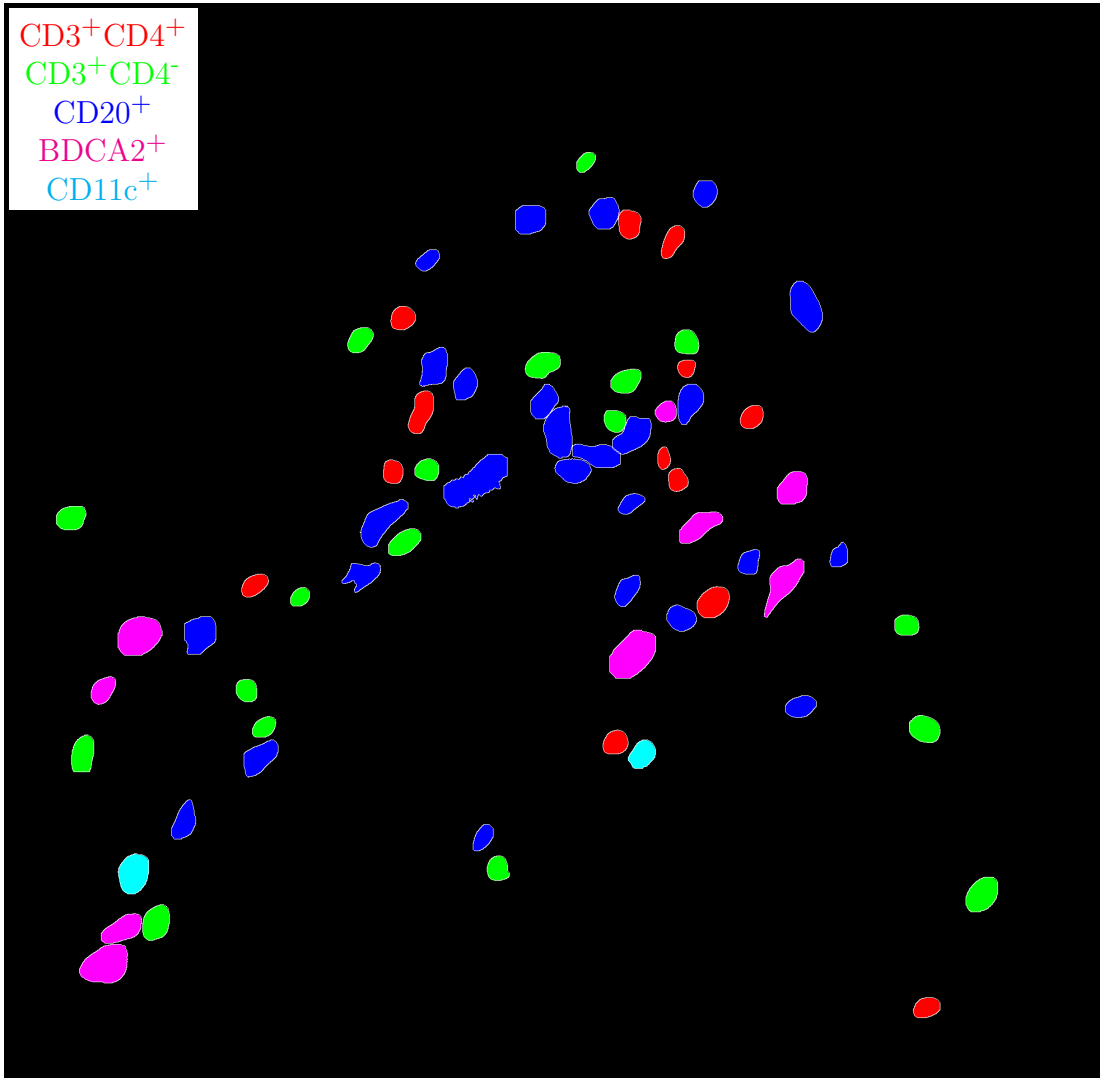

Figure H.89: **Label for H.90**. True label matrix size is 1919 × 1957. Depicted label matrix size is 1919 × 1957.

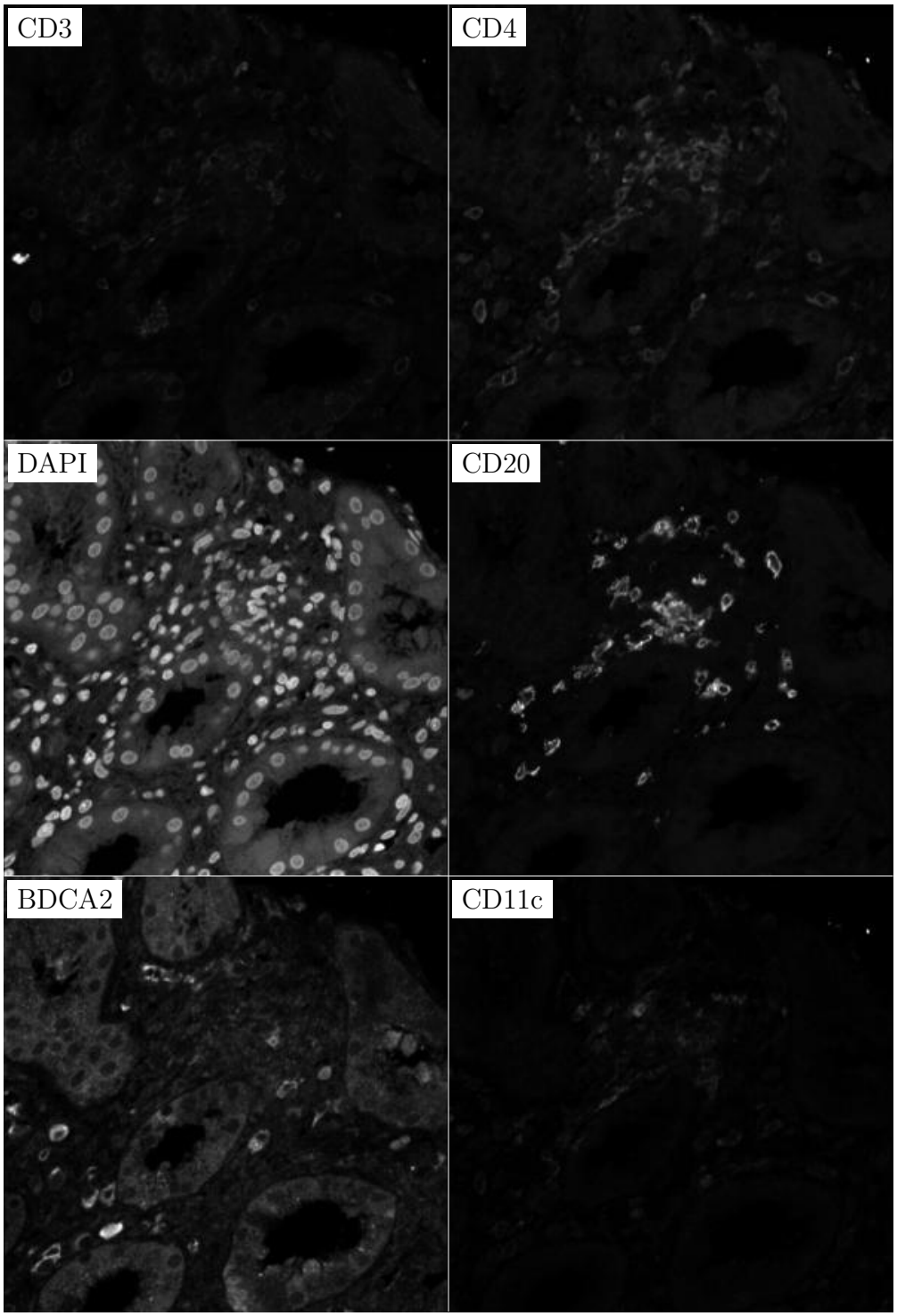

Figure H.90: **ROI for H.89**. True image channel matrix size is 1919 × 1957. Depicted image channel matrix size is 294 × 300.

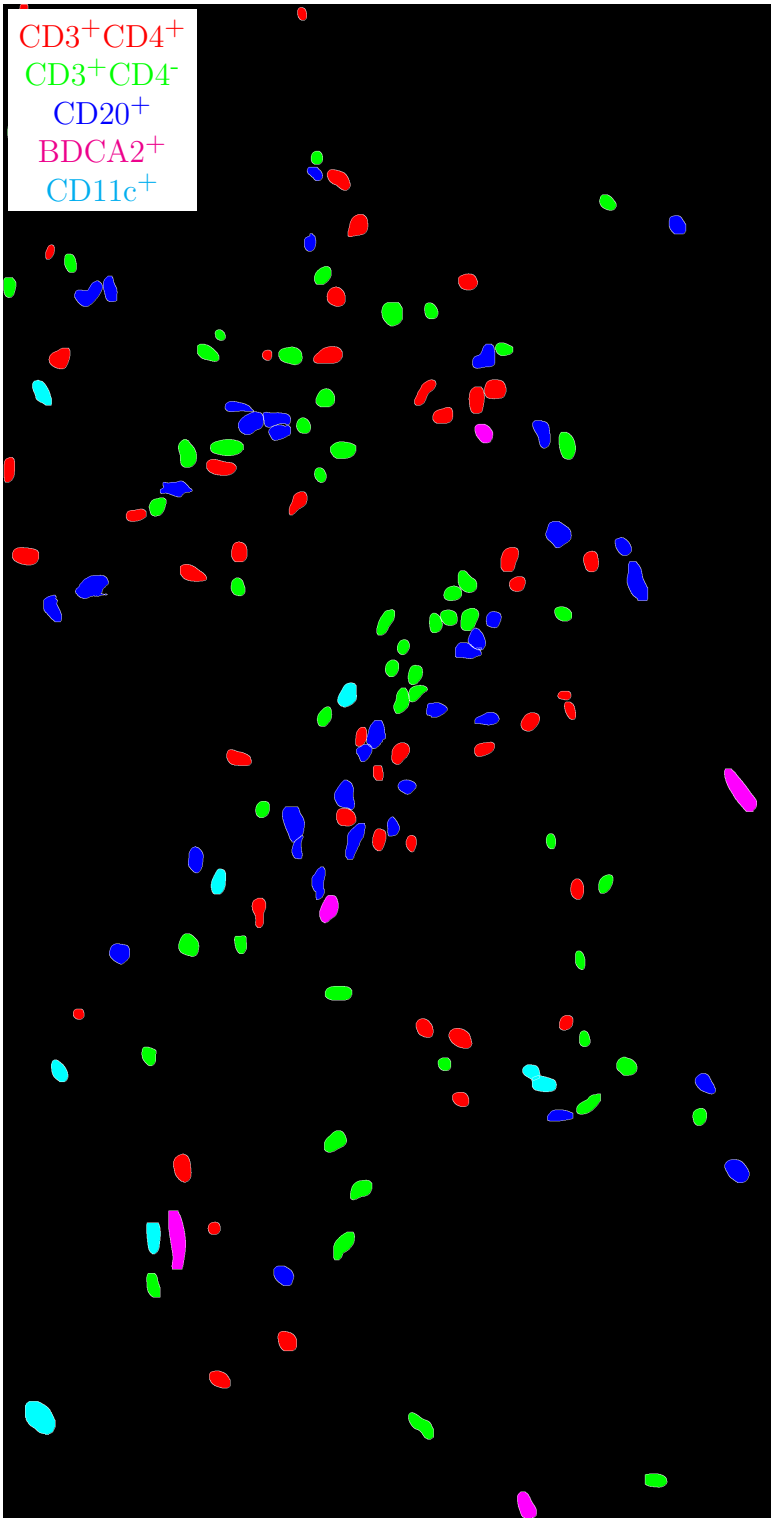

Figure H.91: **Label for H.92**. True label matrix size is 3782×1918. Depicted label matrix size is 3782×1918.

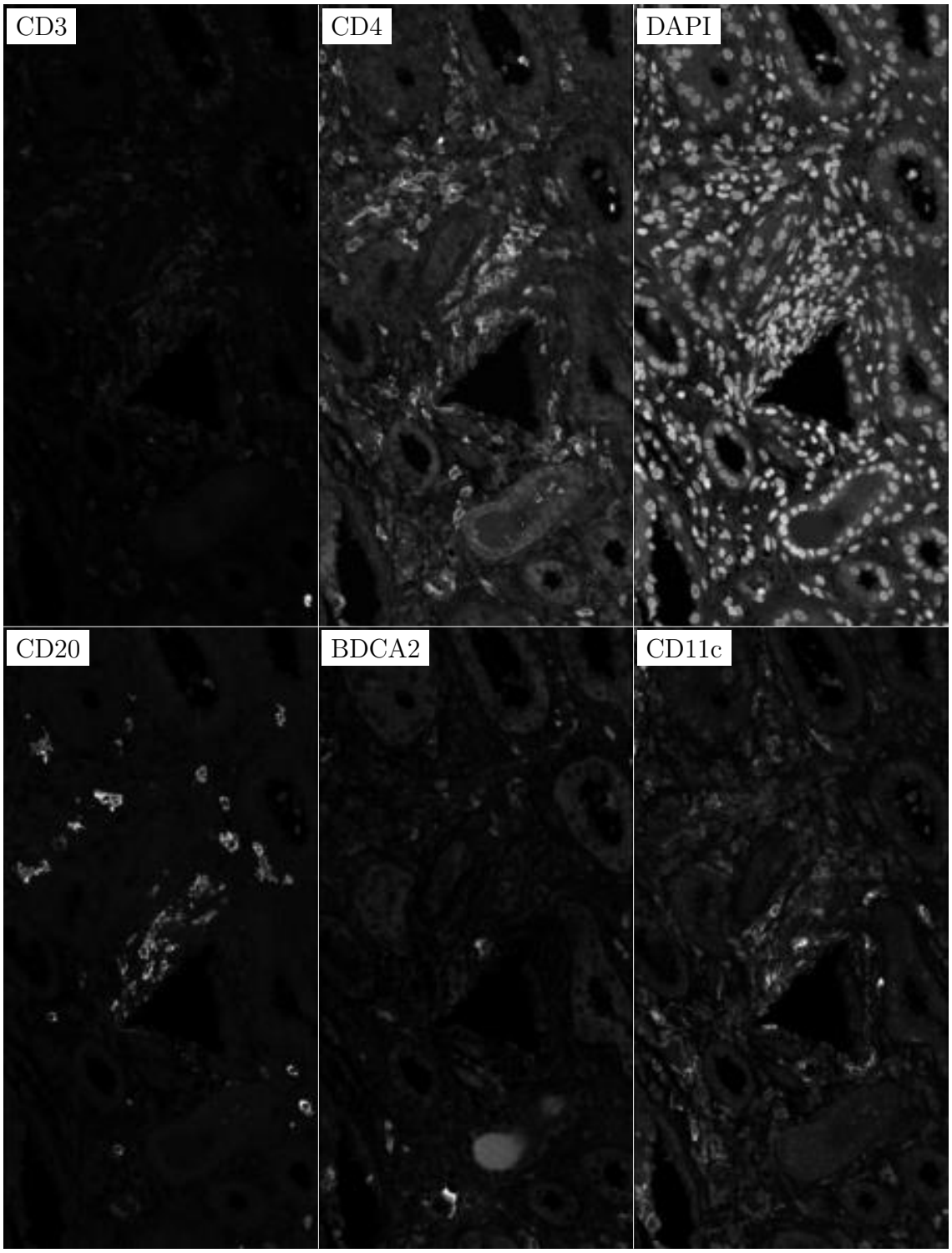

Figure H.92: **ROI for H.91**. True image channel matrix size is 3782 × 1918. Depicted image channel matrix size is 300 × 152.

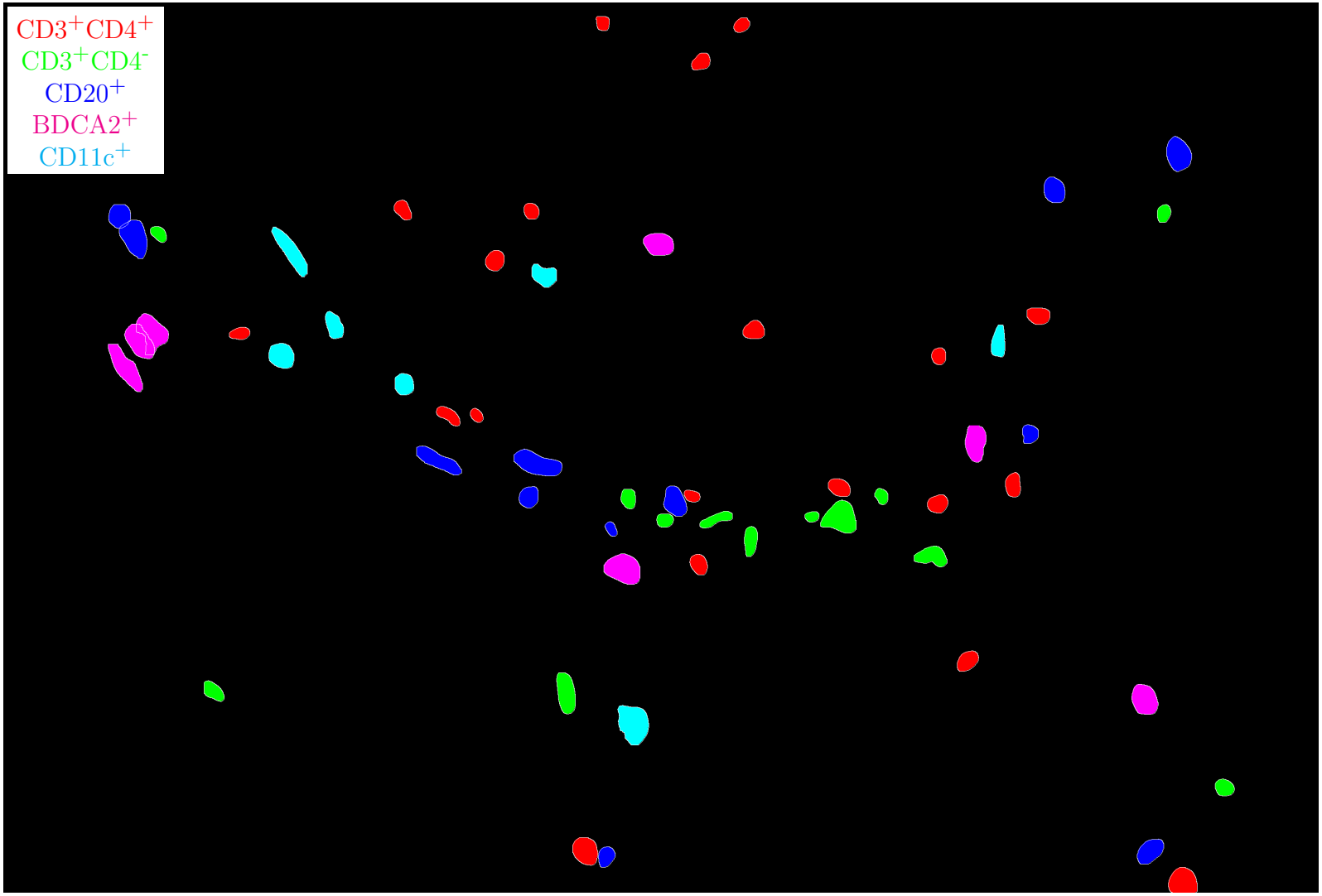

Figure H.93: **Label for H.94**. True label matrix size is 1945 × 2866. Depicted label matrix size is 1945 × 2866.

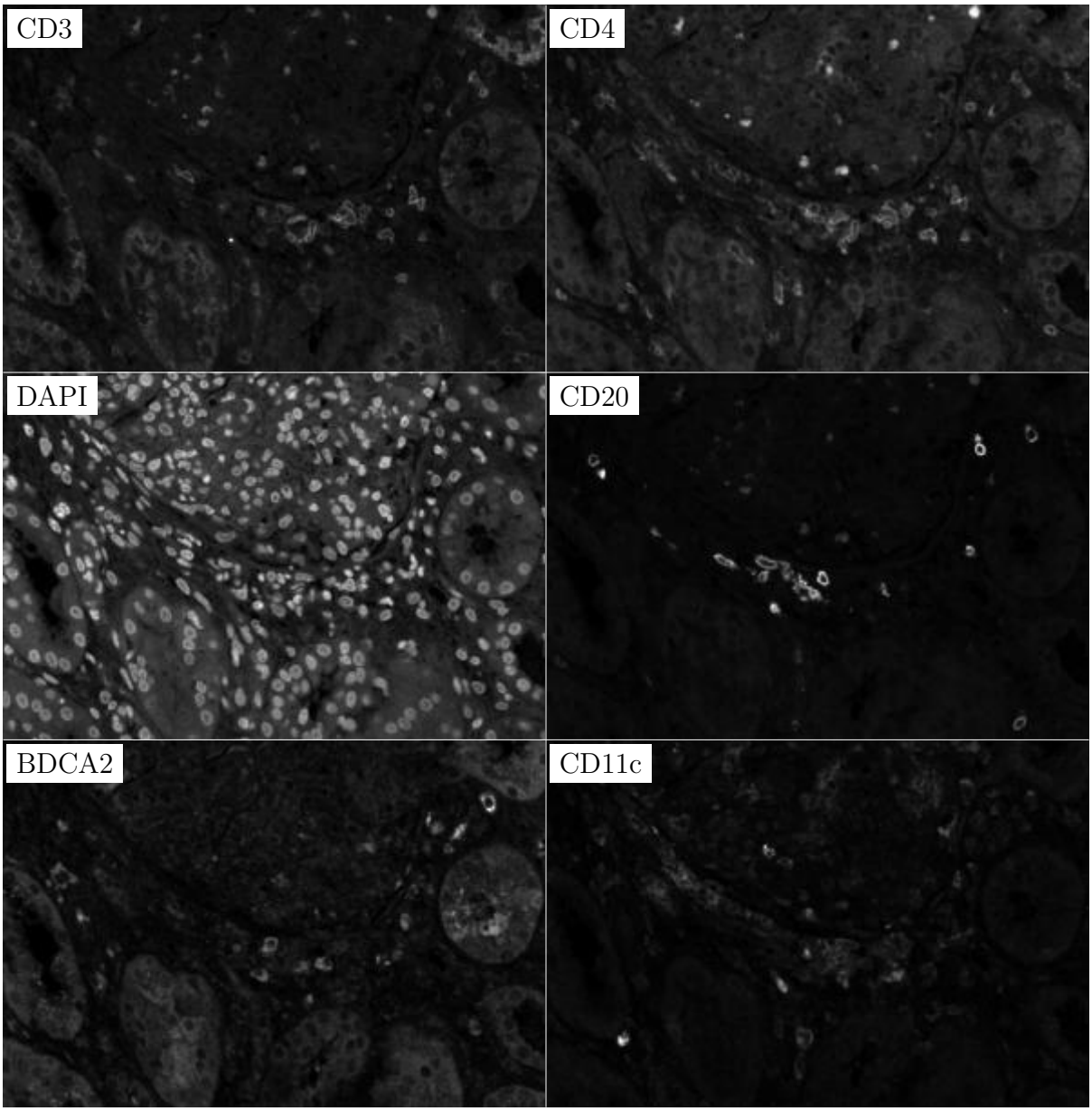

Figure H.94: **ROI for H.93**. True image channel matrix size is 1945 × 2866. Depicted image channel matrix size is 203 × 300.

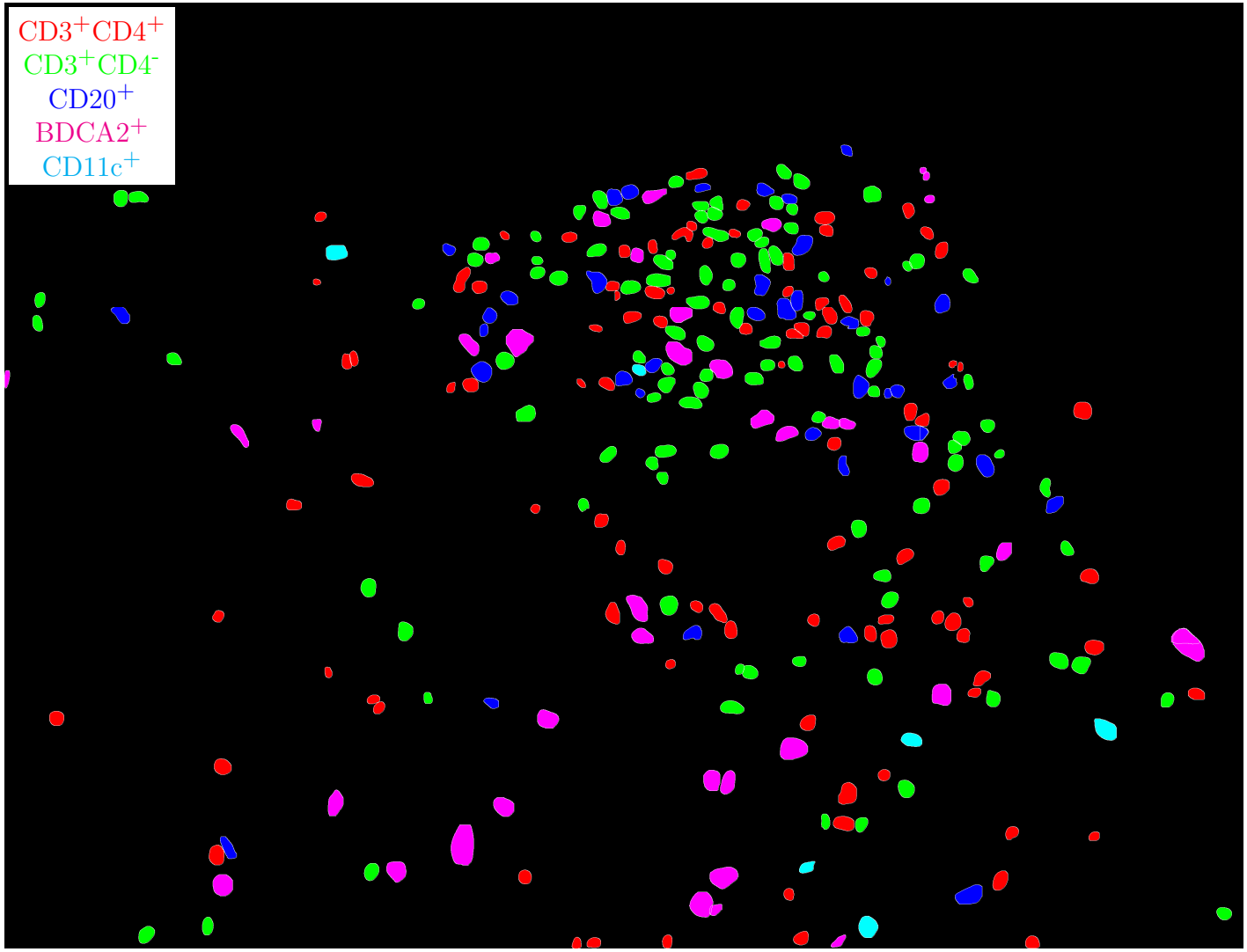

Figure H.95: **Label for H.96**. True label matrix size is 2871 × 3753. Depicted label matrix size is 2871 × 3753.

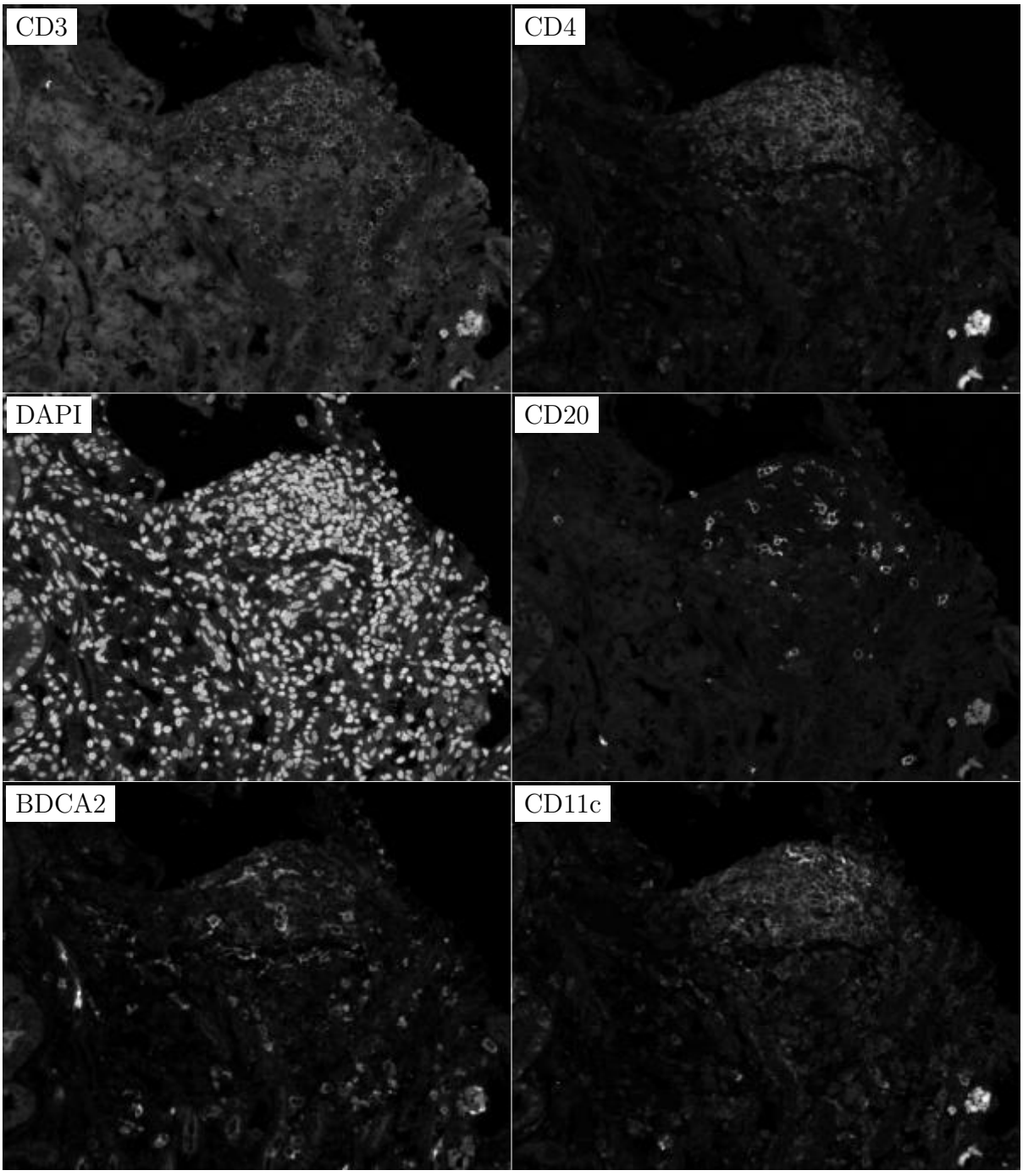

Figure H.96: **ROI for H.95**. True image channel matrix size is 2871×3753. Depicted image channel matrix size is 229×300.

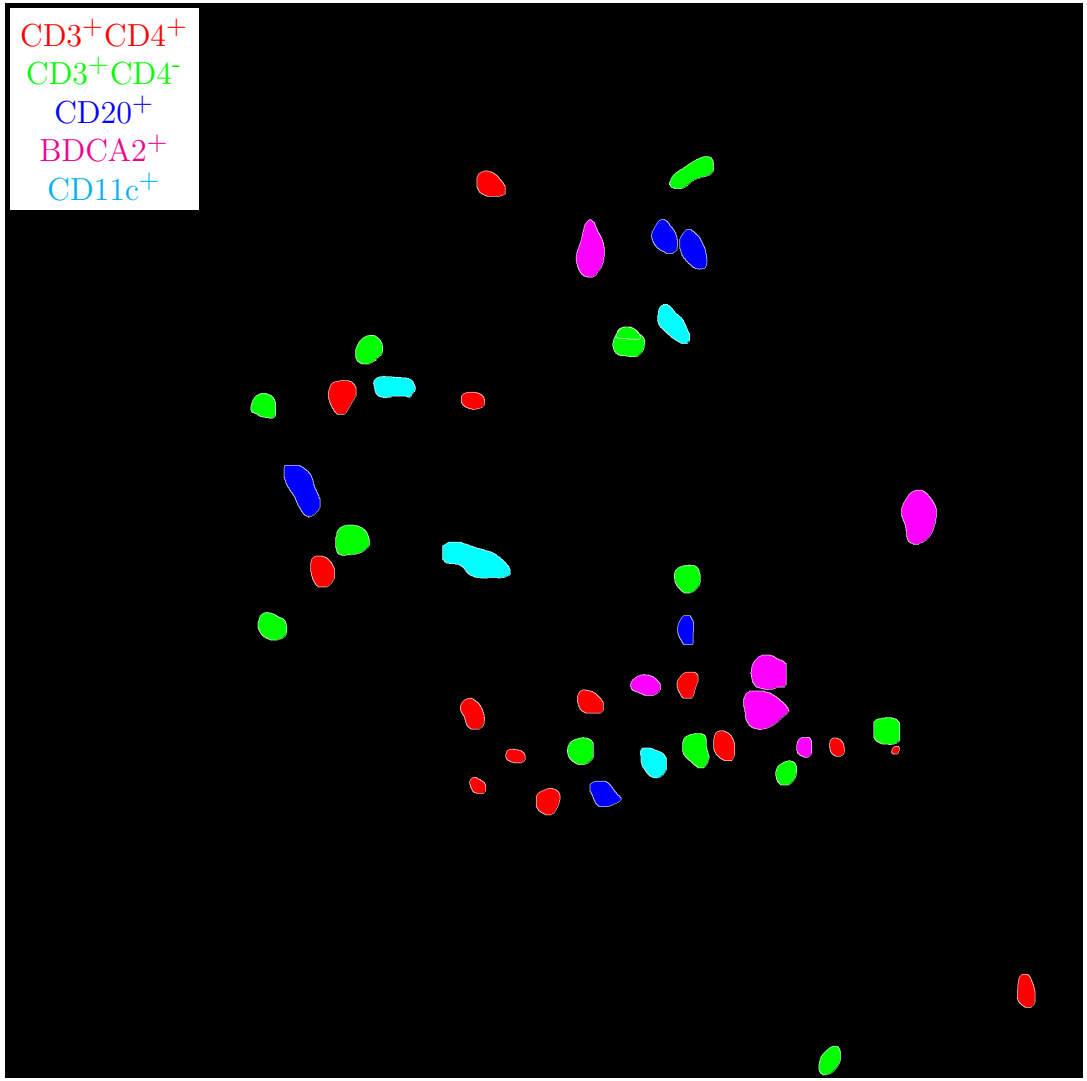

Figure H.97: **Label for H.98**. True label matrix size is 1943 × 1945. Depicted label matrix size is 1943 × 1945.

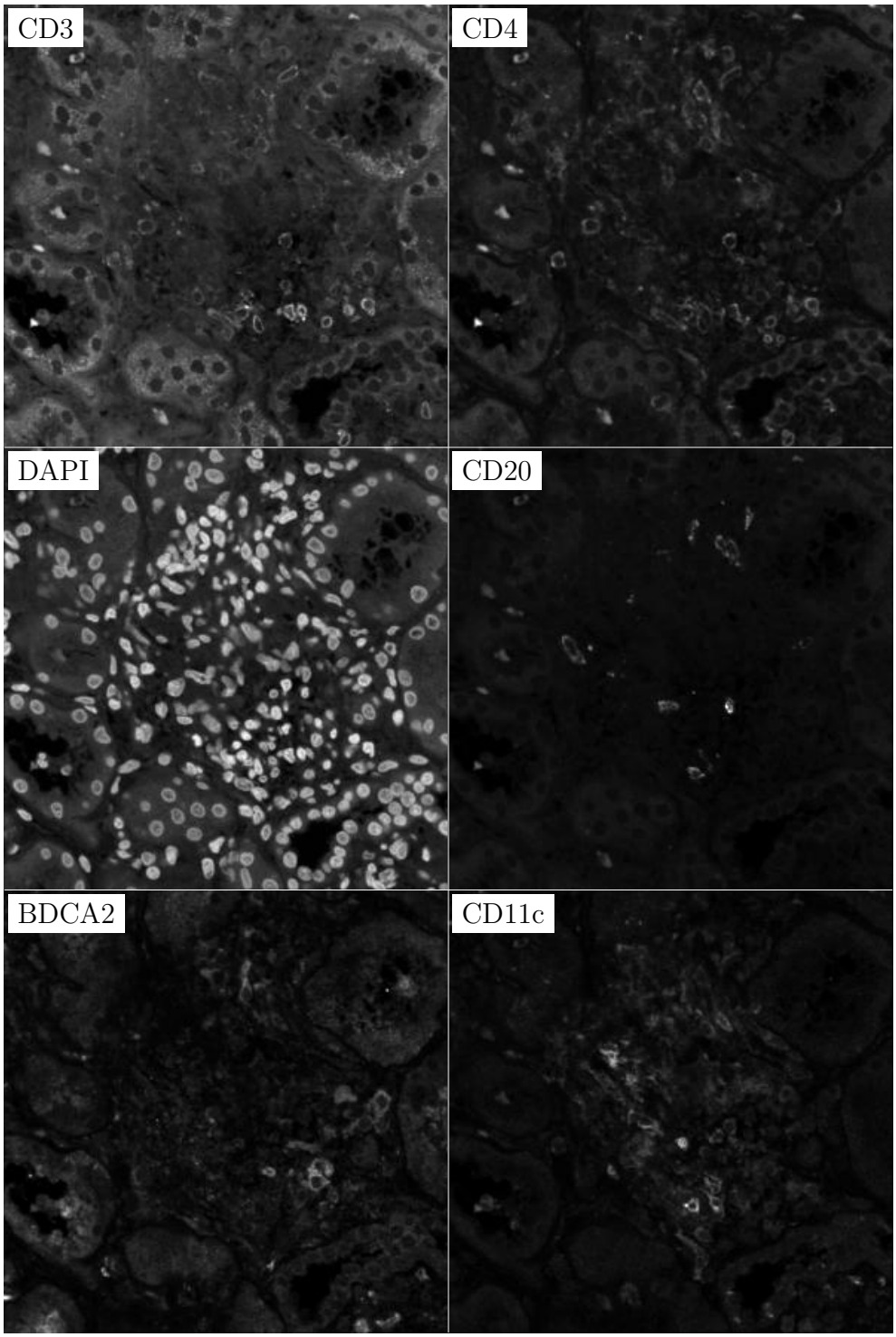

Figure H.98: **ROI for H.97**. True image channel matrix size is 1943 × 1945. Depicted image channel matrix size is 299 × 300.

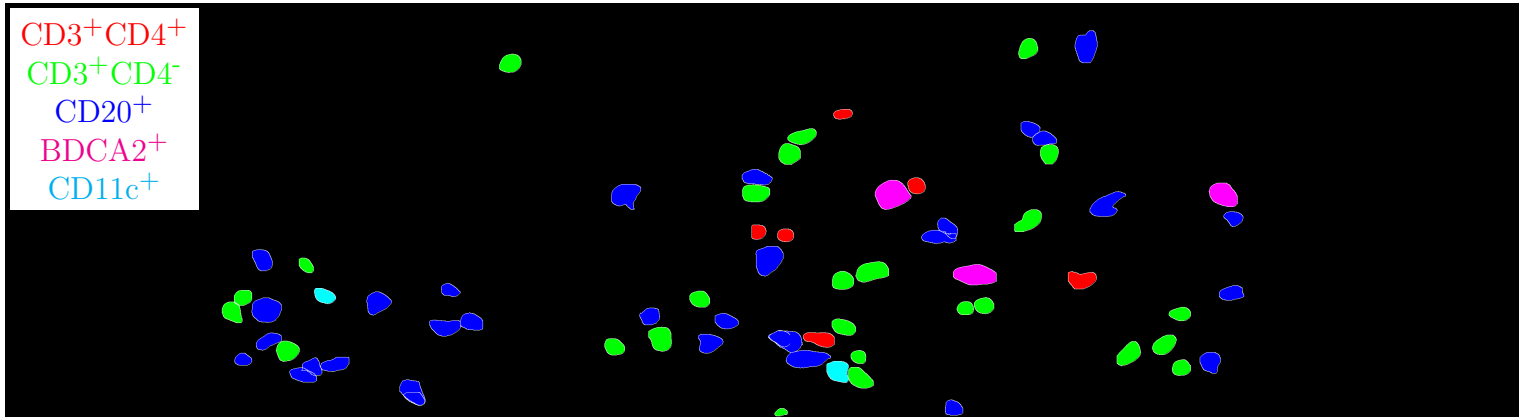

Figure H.99: **Label for H.100**. True label matrix size is 1027×3756. Depicted label matrix size is 1027×3756.

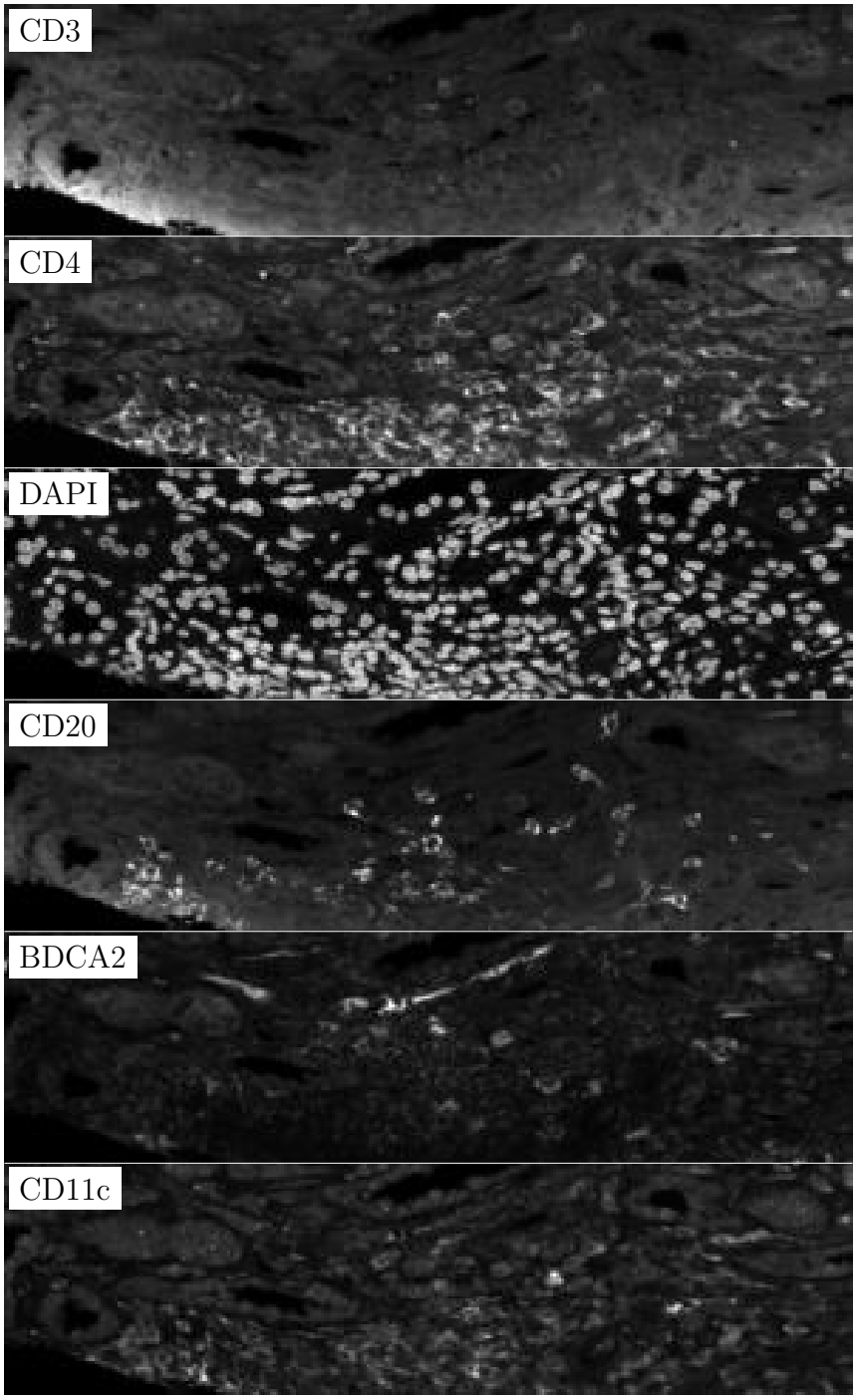

Figure H.100: **ROI for H.99**. True image channel matrix size is 1027×3756. Depicted image channel matrix size is 82×300.

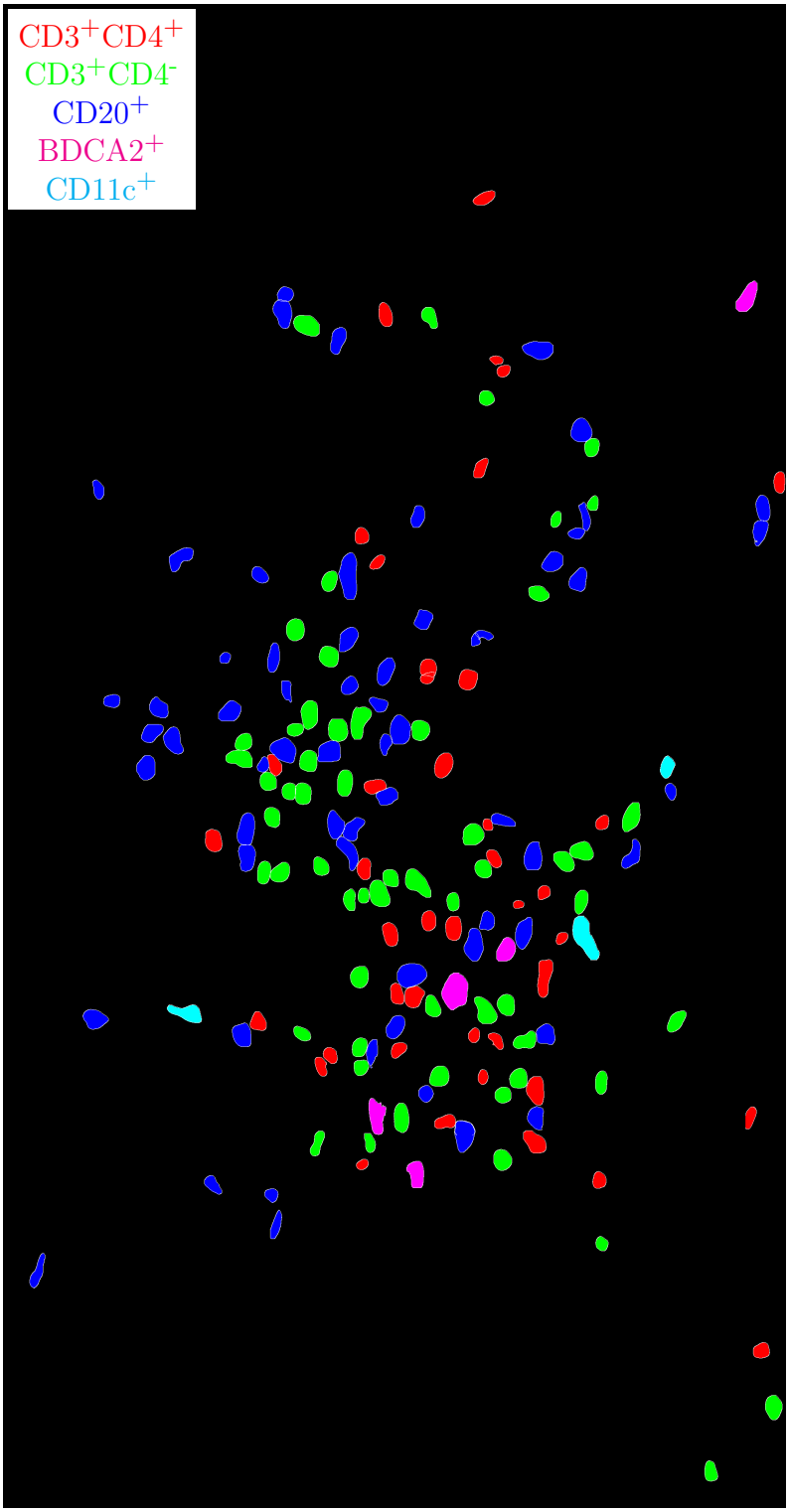

Figure H.101: **Label for H.102**. True label matrix size is 3728 × 1949. Depicted label matrix size is 3728 × 1949.

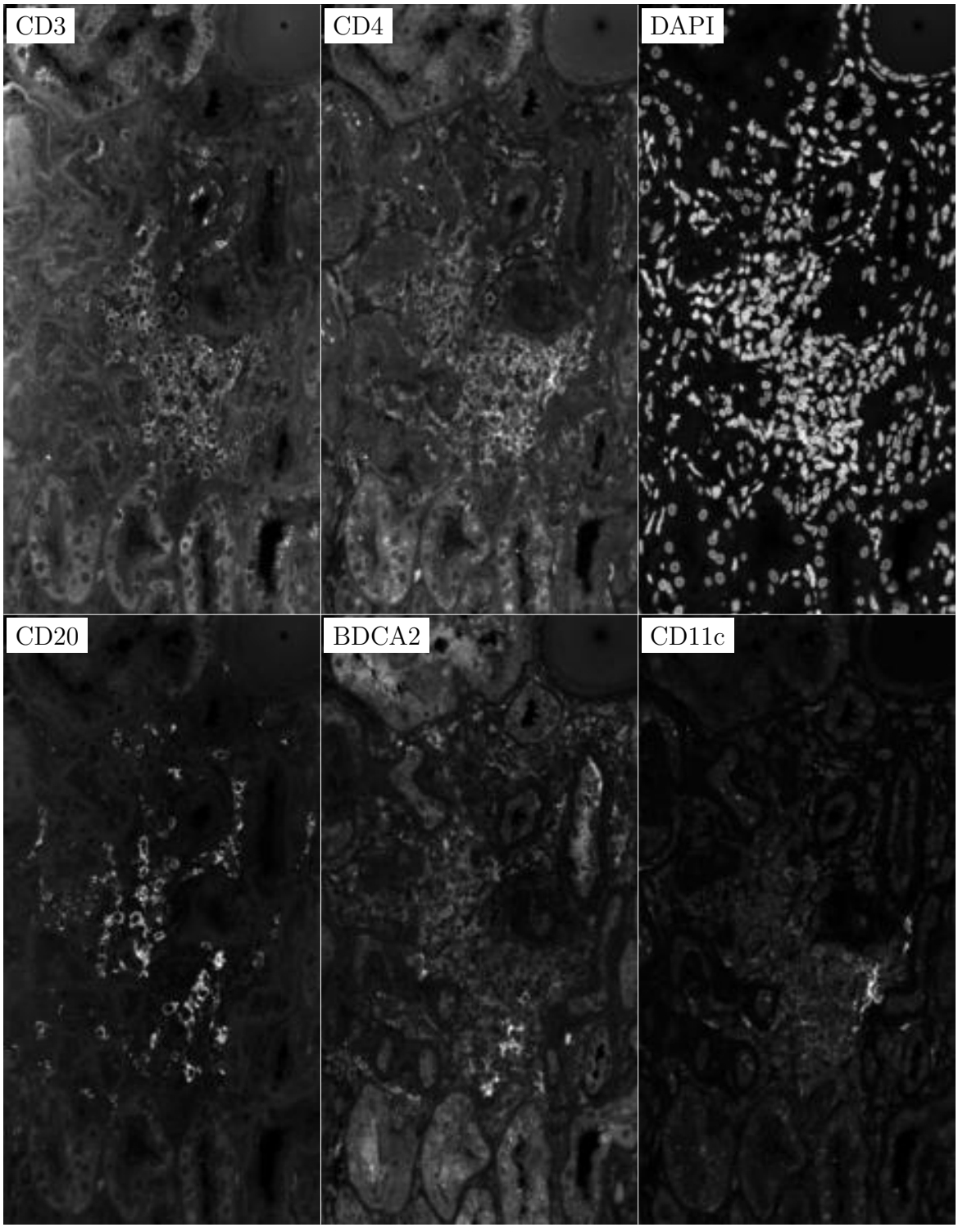

Figure H.102: **ROI for H.101**. True image channel matrix size is 3728×1949. Depicted image channel matrix size is 300×156.

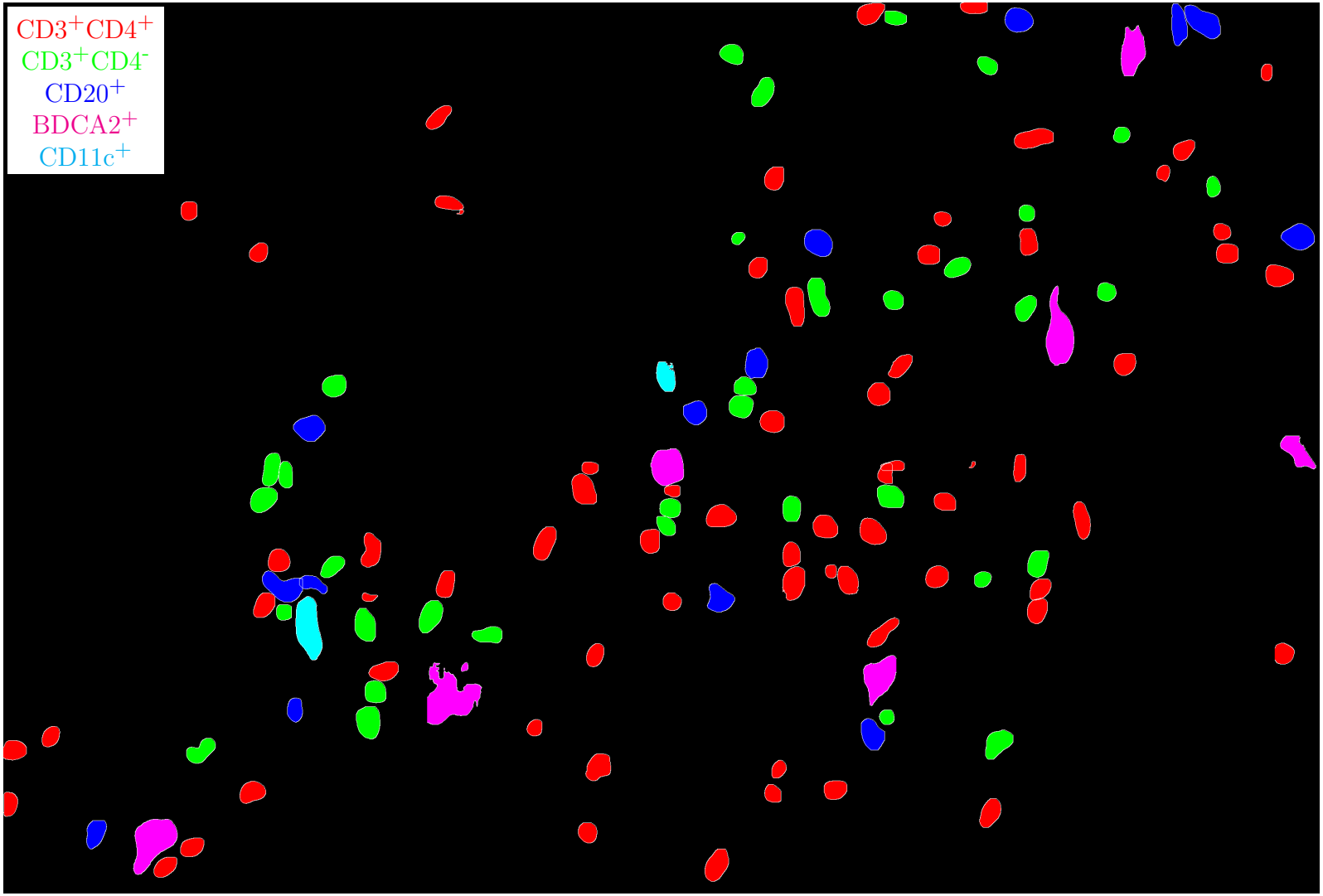

Figure H.103: **Label for H.104**. True label matrix size is 1948 × 2870. Depicted label matrix size is 1948 × 2870.

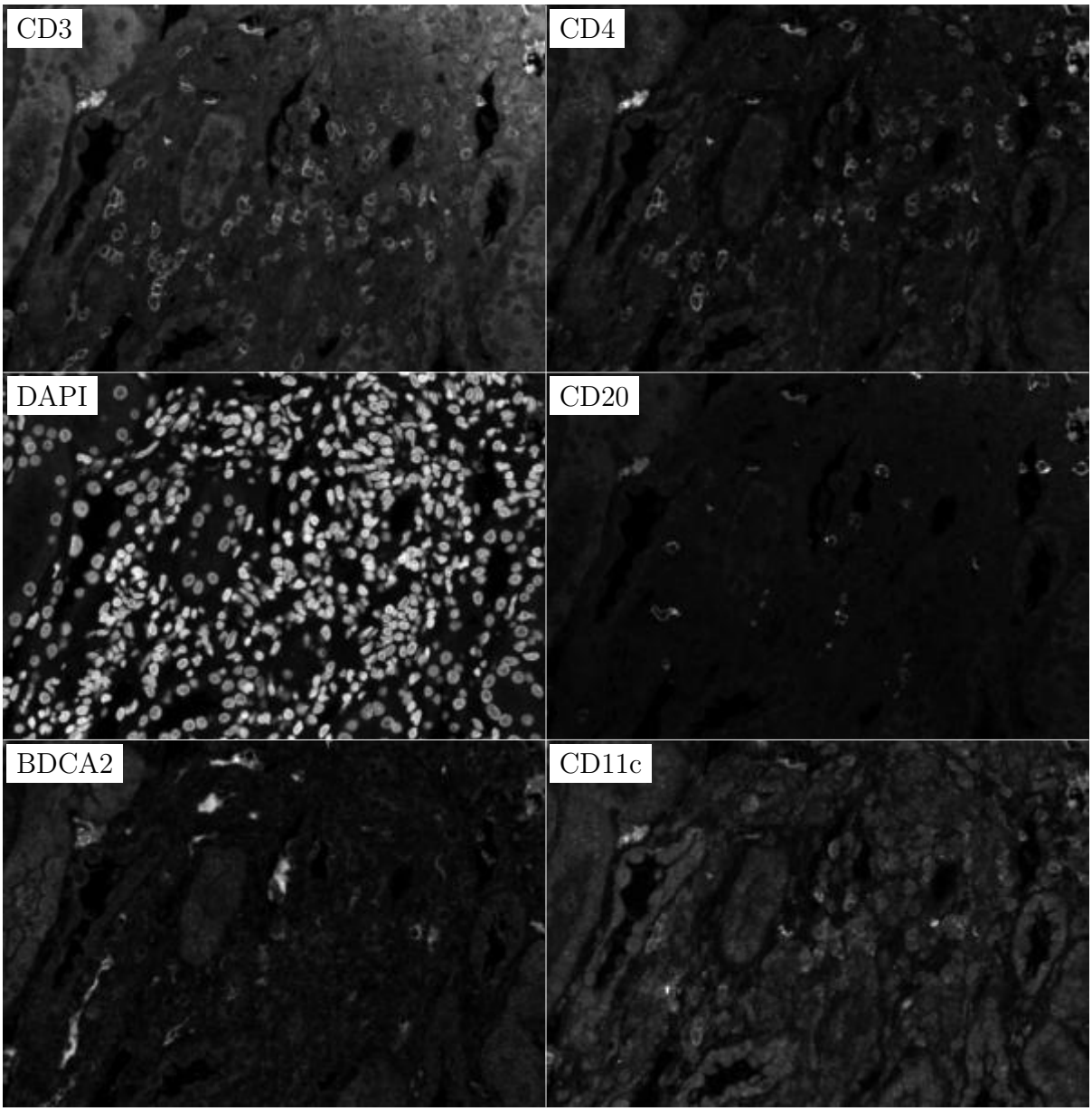

Figure H.104: **ROI for H.103**. True image channel matrix size is 1948×2870. Depicted image channel matrix size is 203×300.

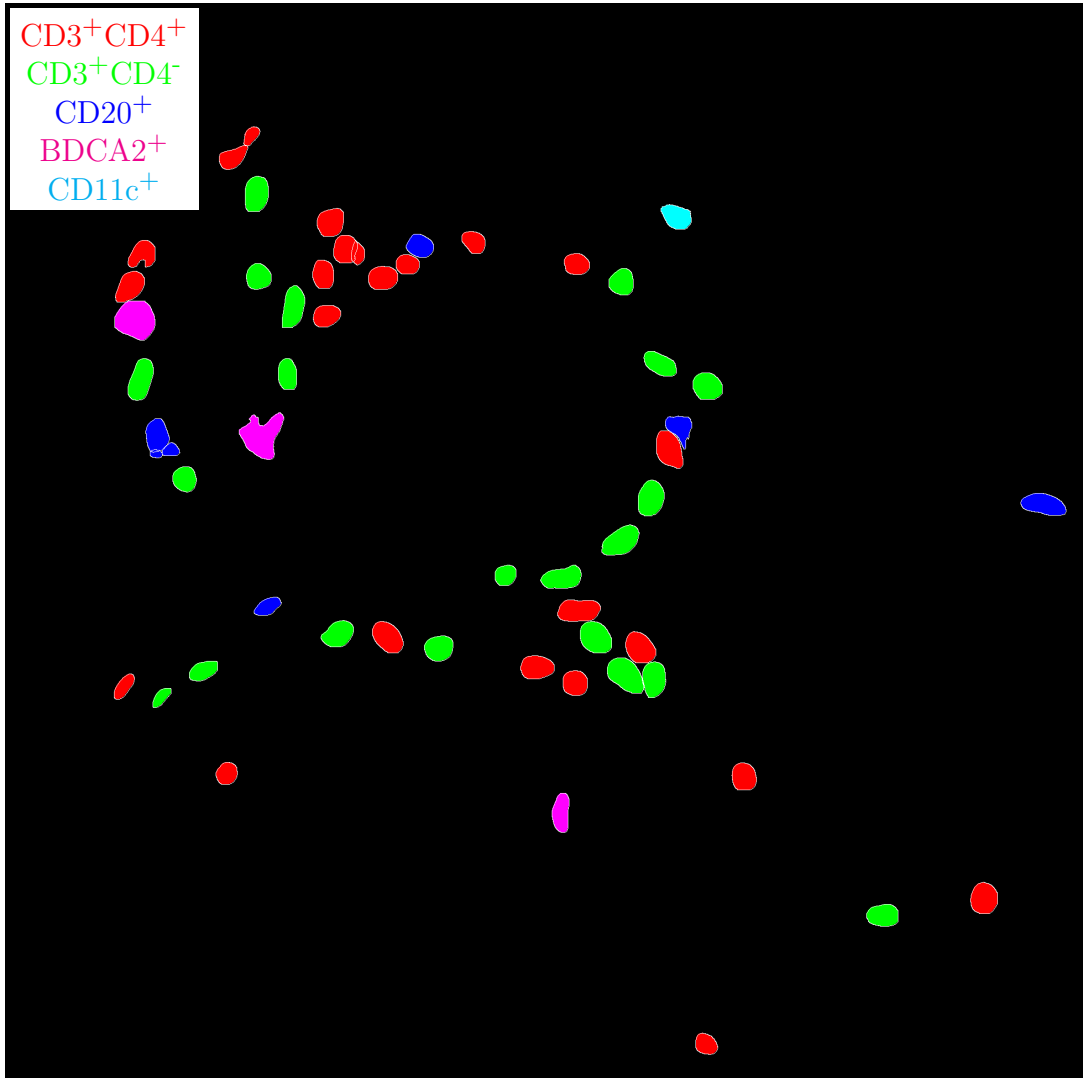

Figure H.105: **Label for H.106**. True label matrix size is 1946 × 1950. Depicted label matrix size is 1946 × 1950.

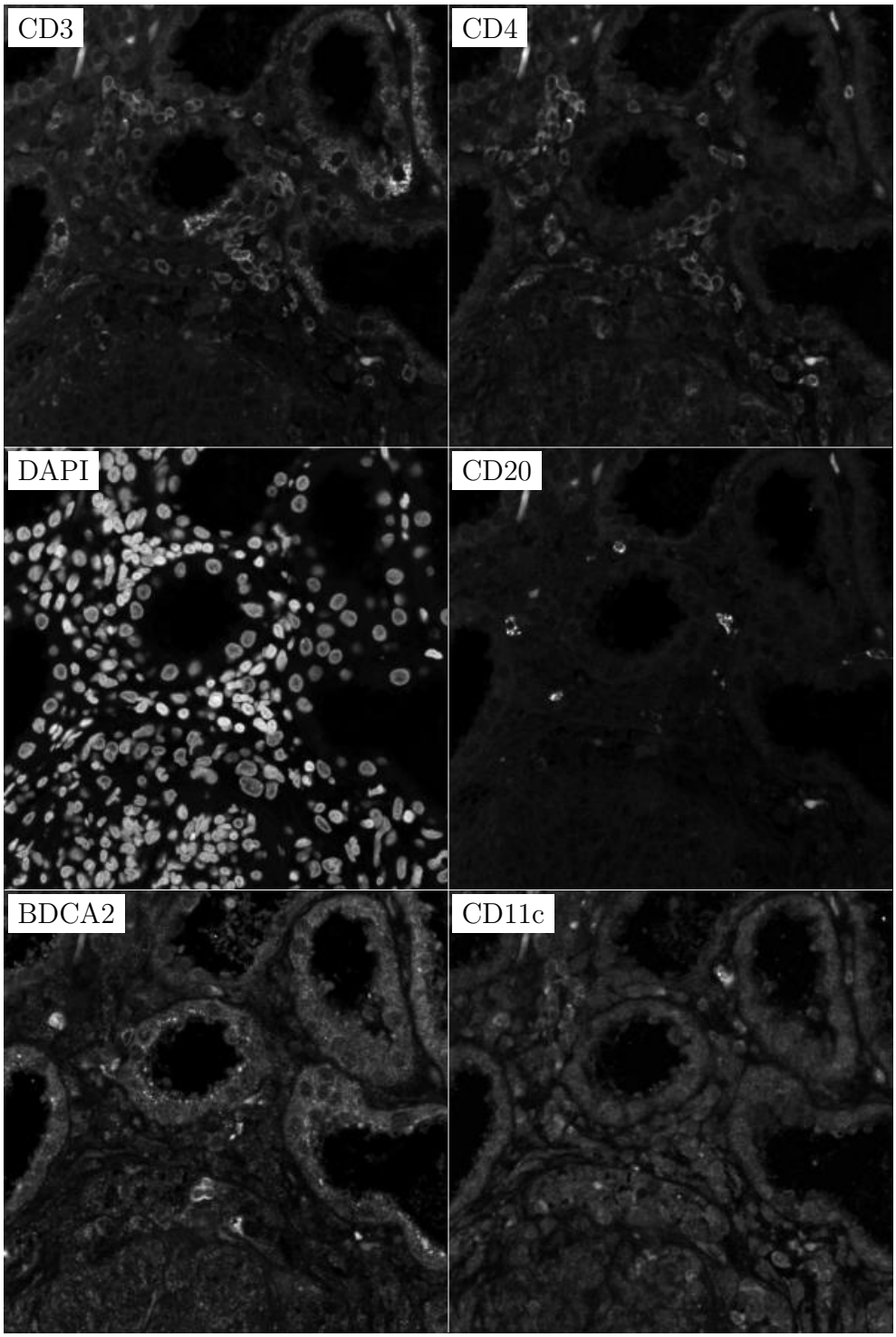

Figure H.106: **ROI for H.105**. True image channel matrix size is 1946 × 1950. Depicted image channel matrix size is 299 × 300.

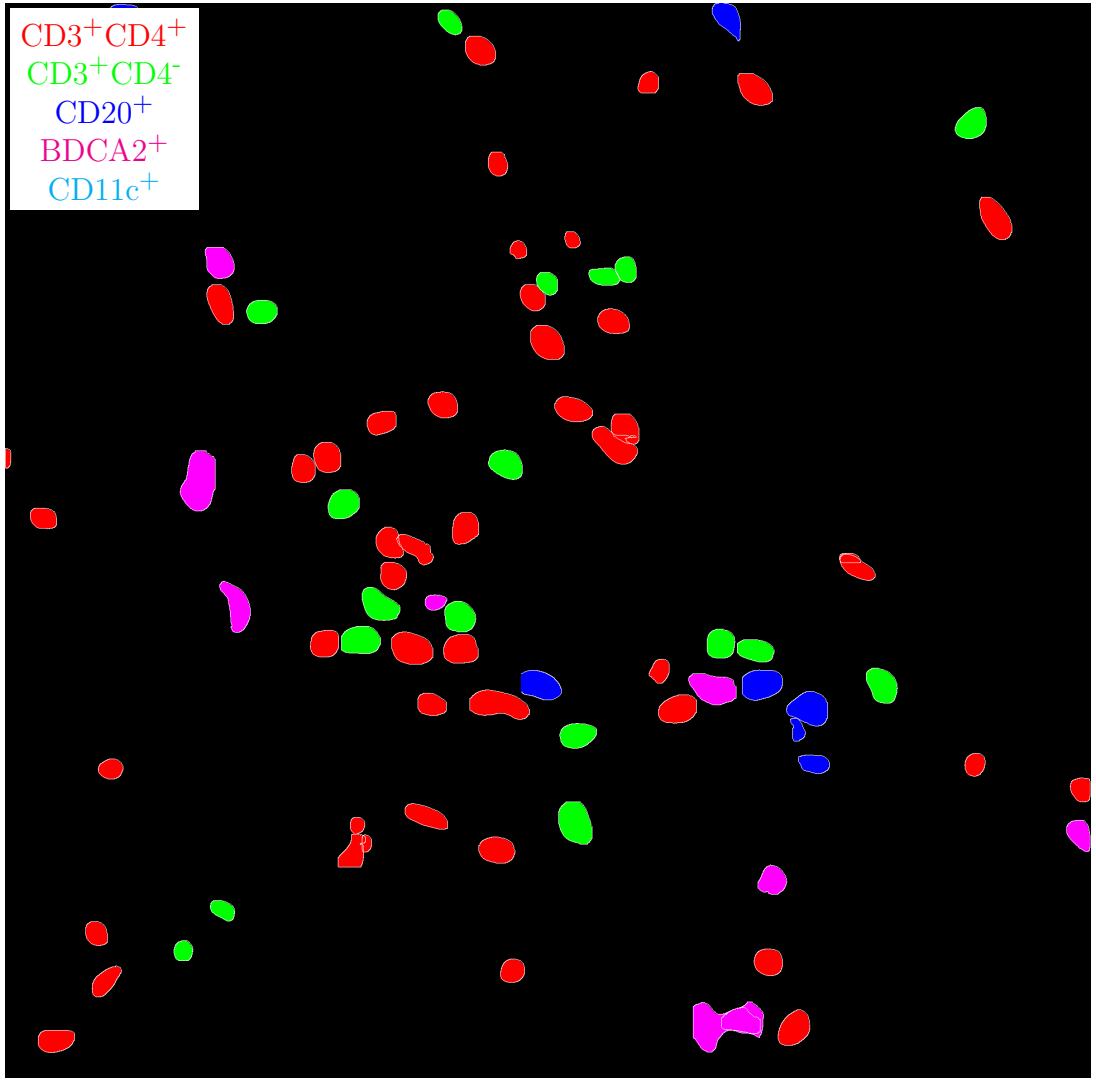

Figure H.107: **Label for H.108**. True label matrix size is 1929 × 1945. Depicted label matrix size is 1929 × 1945.

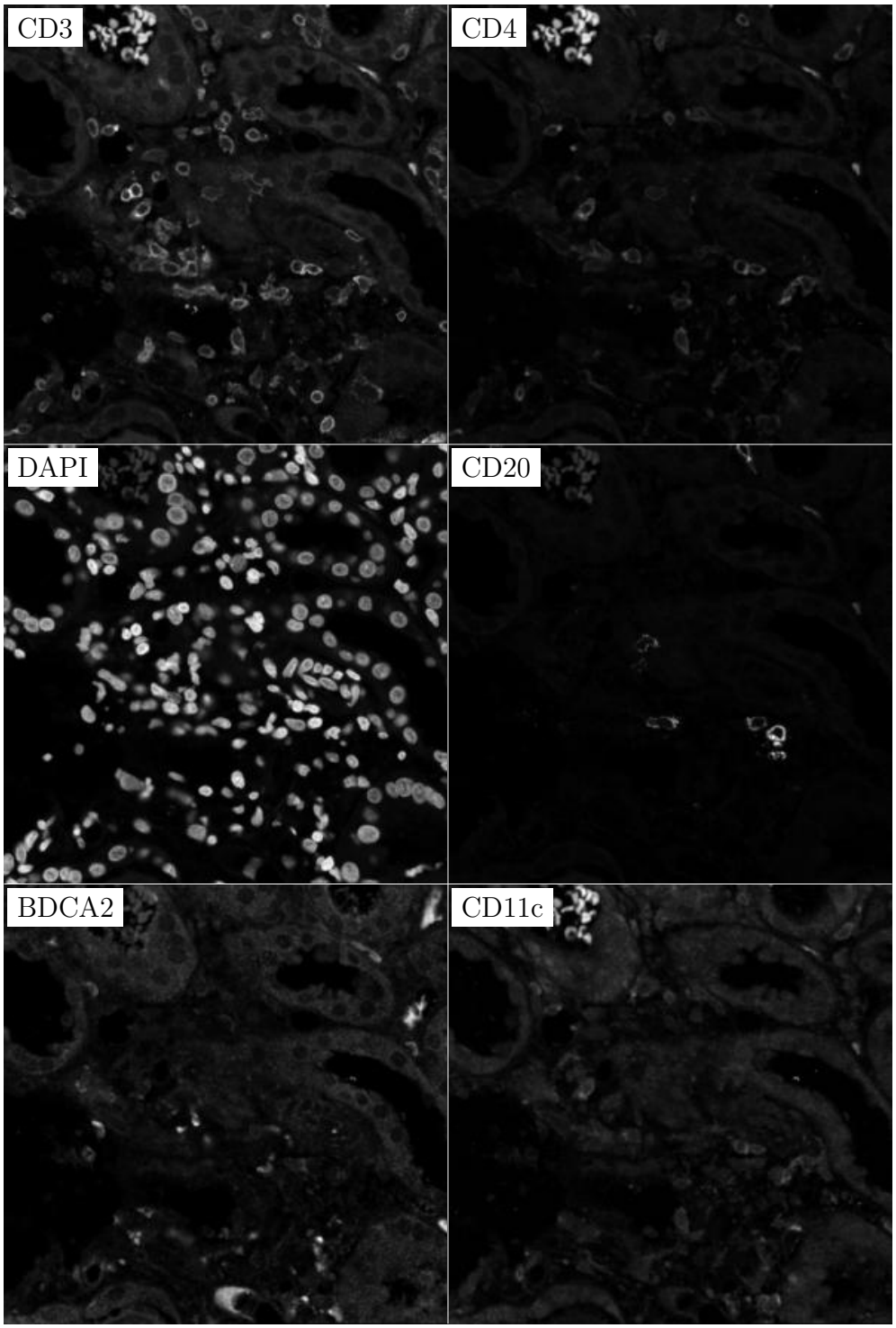

Figure H.108: **ROI for H.107**. True image channel matrix size is 1929 × 1945. Depicted image channel matrix size is 297 × 300.

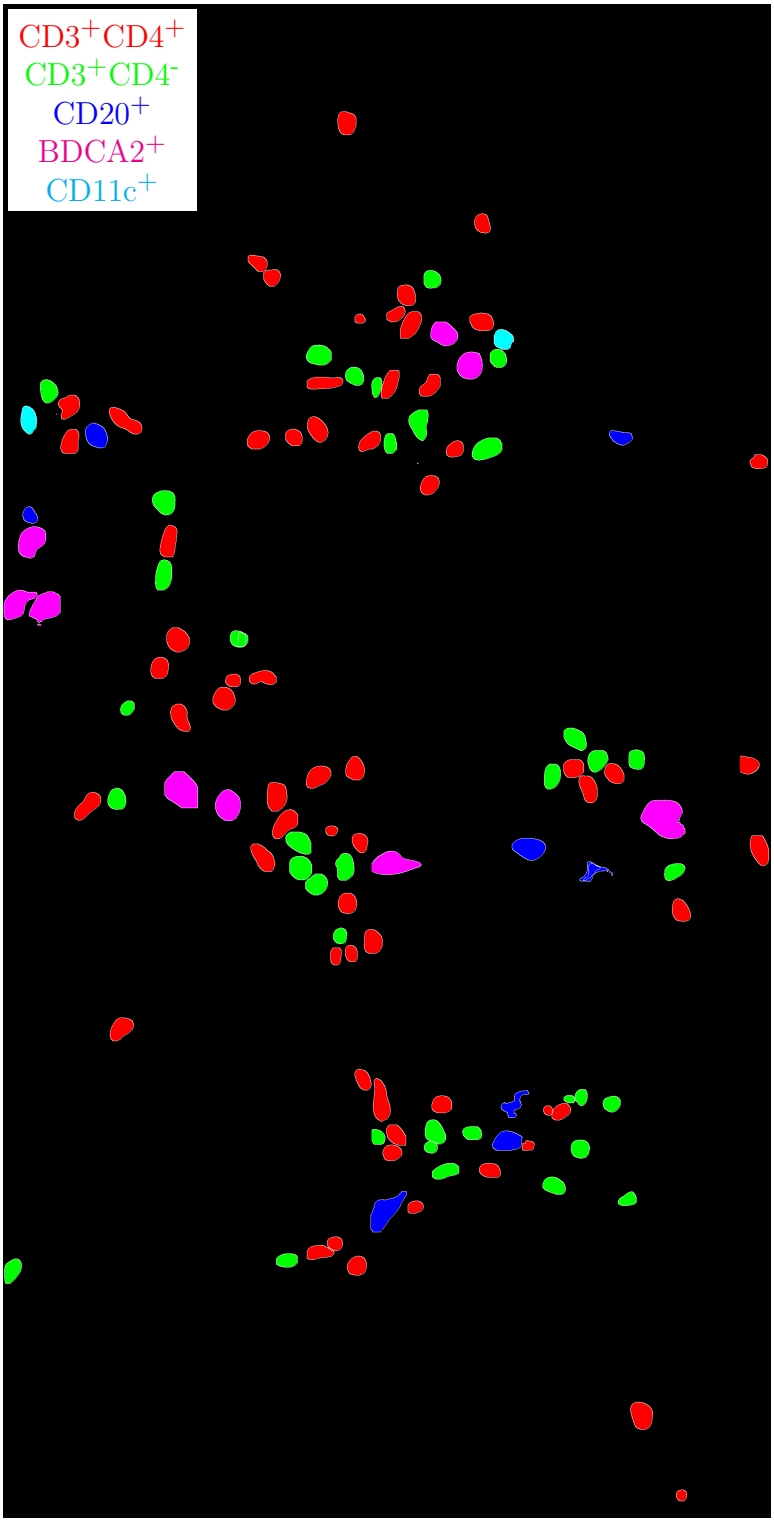

Figure H.109: **Label for H.110**. True label matrix size is 3788 × 1926. Depicted label matrix size is 3788 × 1926.

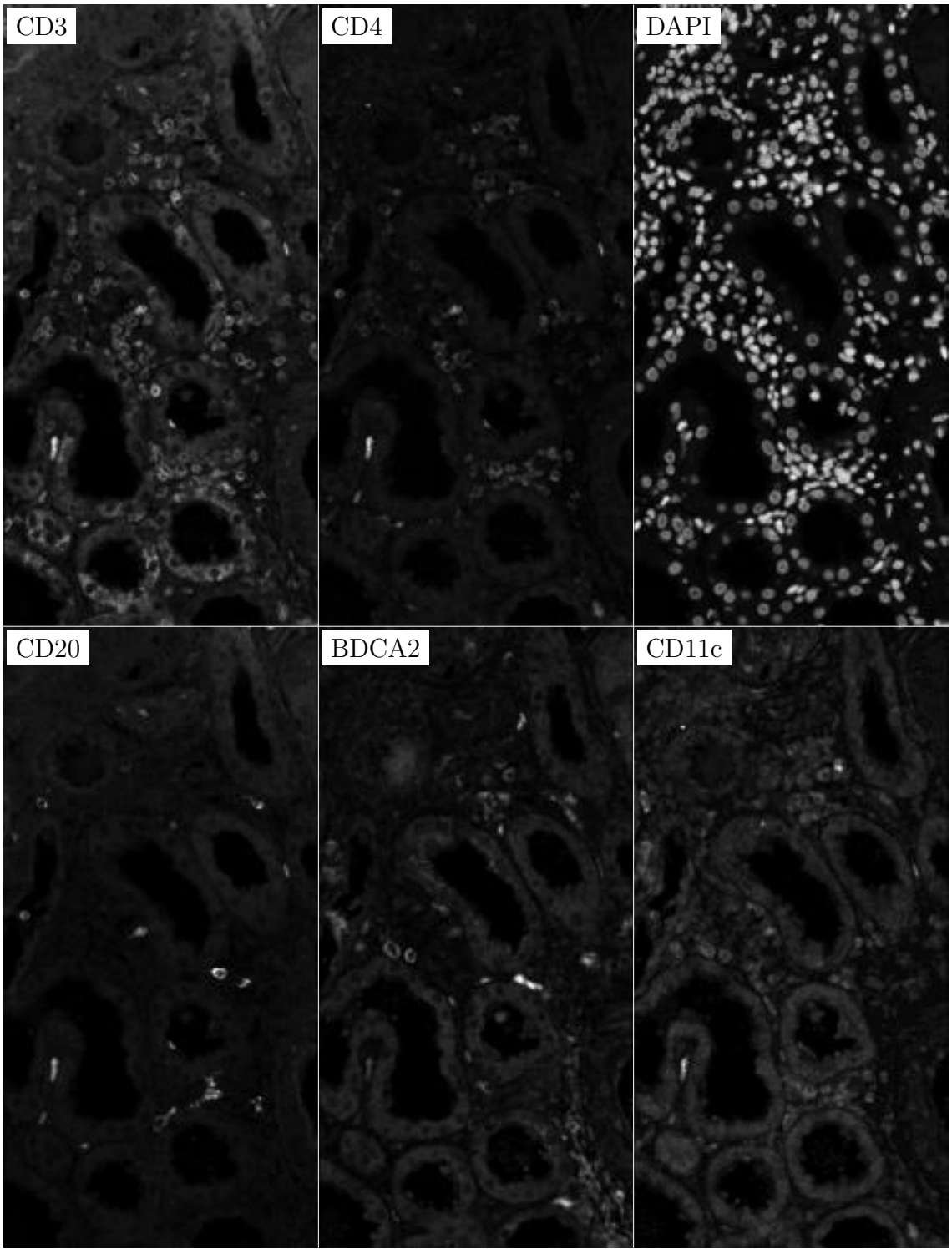

Figure H.110: **ROI for H.109**. True image channel matrix size is 3788×1926. Depicted image channel matrix size is 300×152.

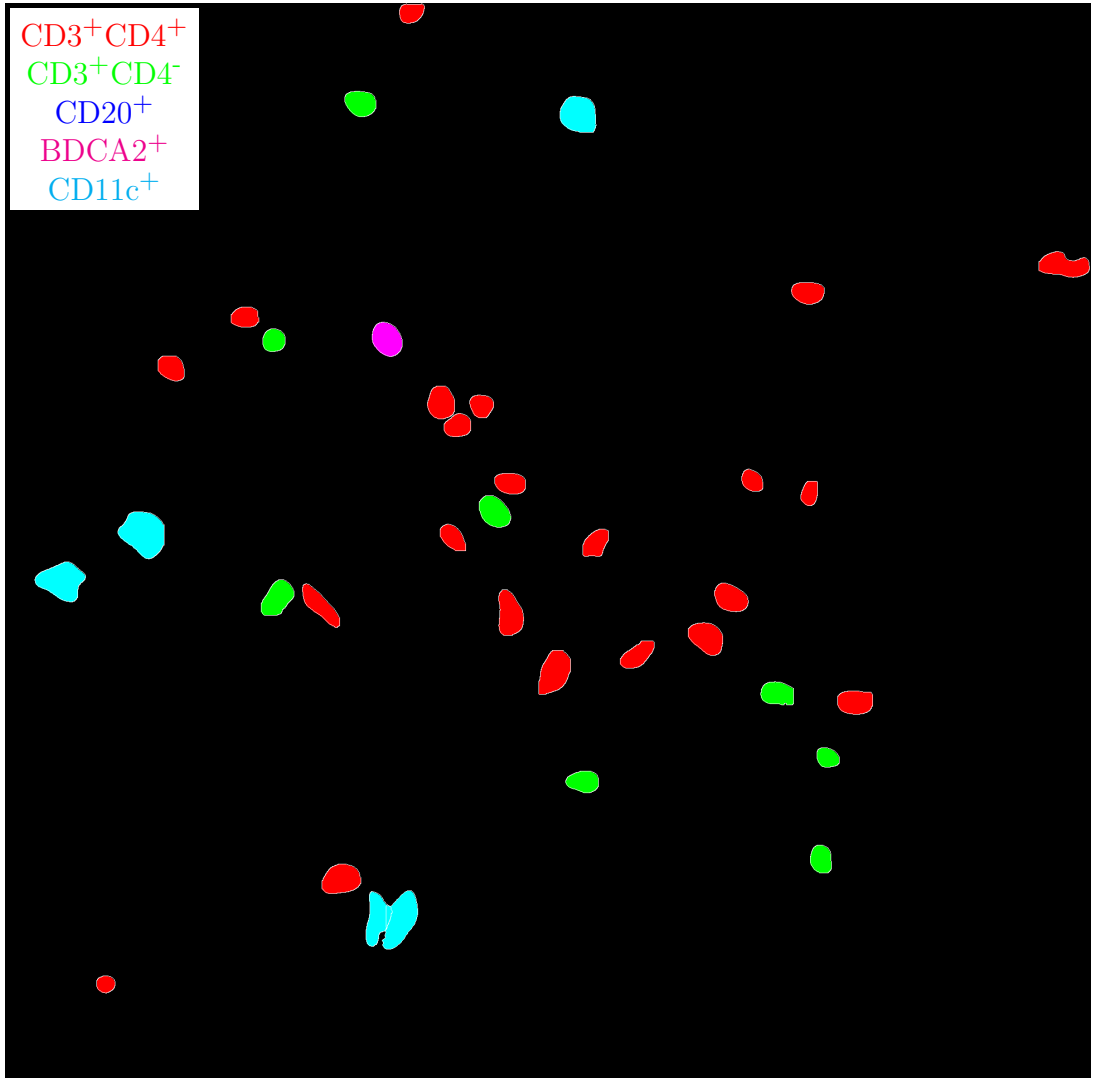

Figure H.111: **Label for H.112**. True label matrix size is 1932 × 1948. Depicted label matrix size is 1932 × 1948.

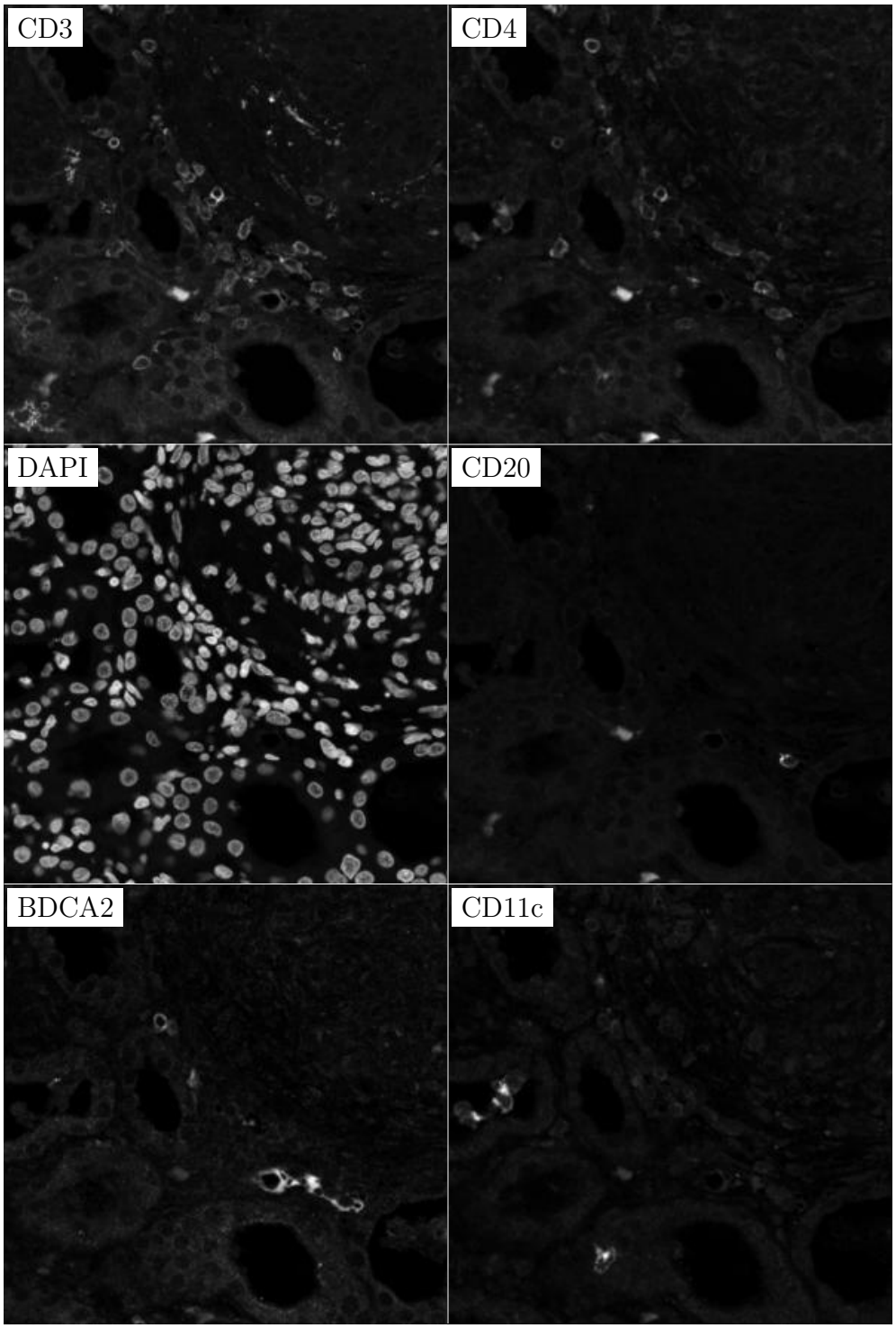

Figure H.112: **ROI for H.111**. True image channel matrix size is 1932×1948. Depicted image channel matrix size is 297×300.

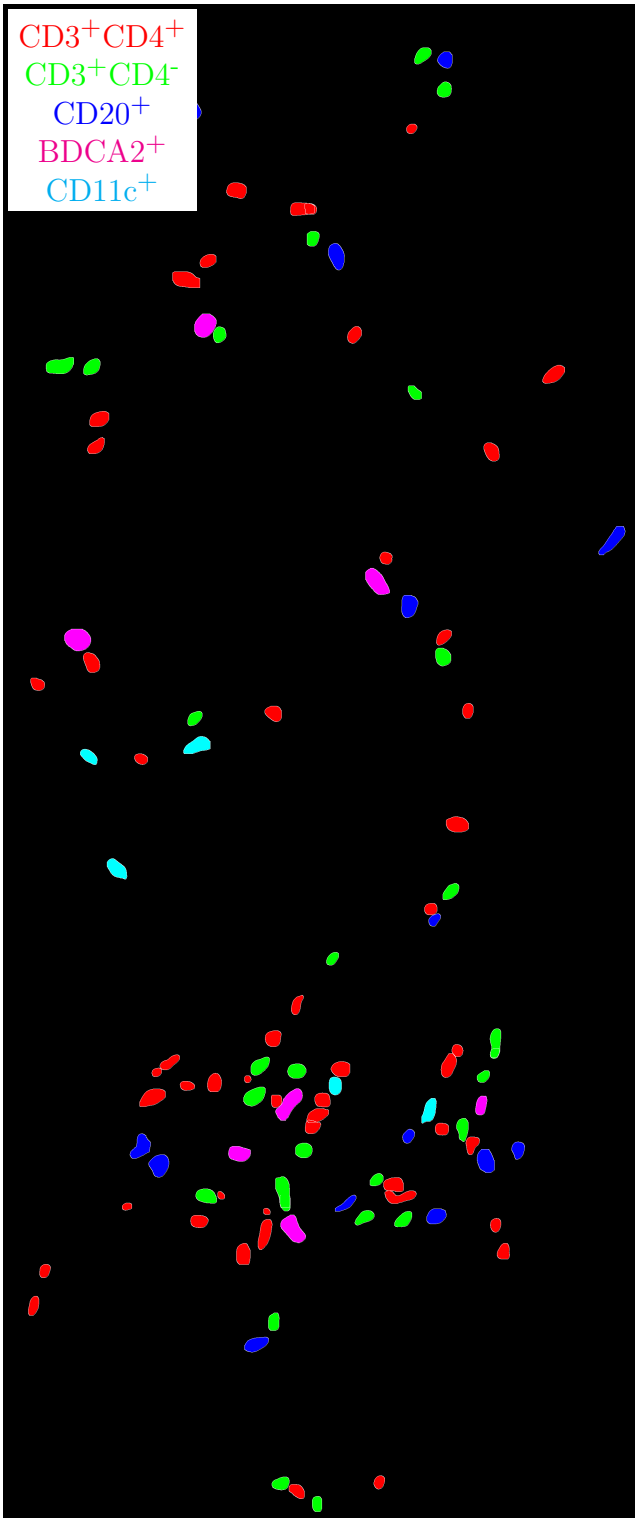

Figure H.113: **Label for H.114**. True label matrix size is 4698 × 1959. Depicted label matrix size is 4698 × 1959.

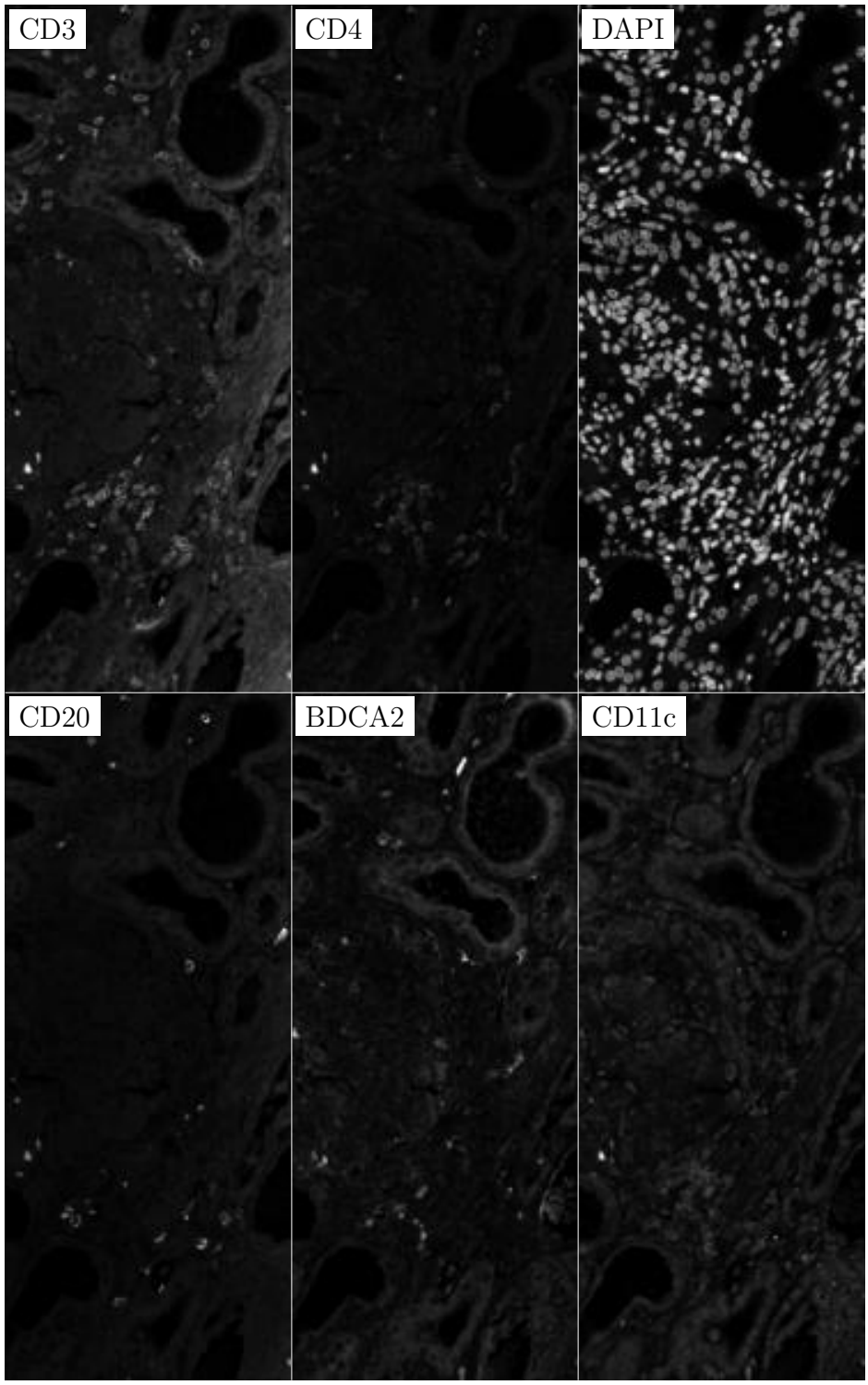

Figure H.114: **ROI for H.113**. True image channel matrix size is 4698×1959. Depicted image channel matrix size is 300×125.

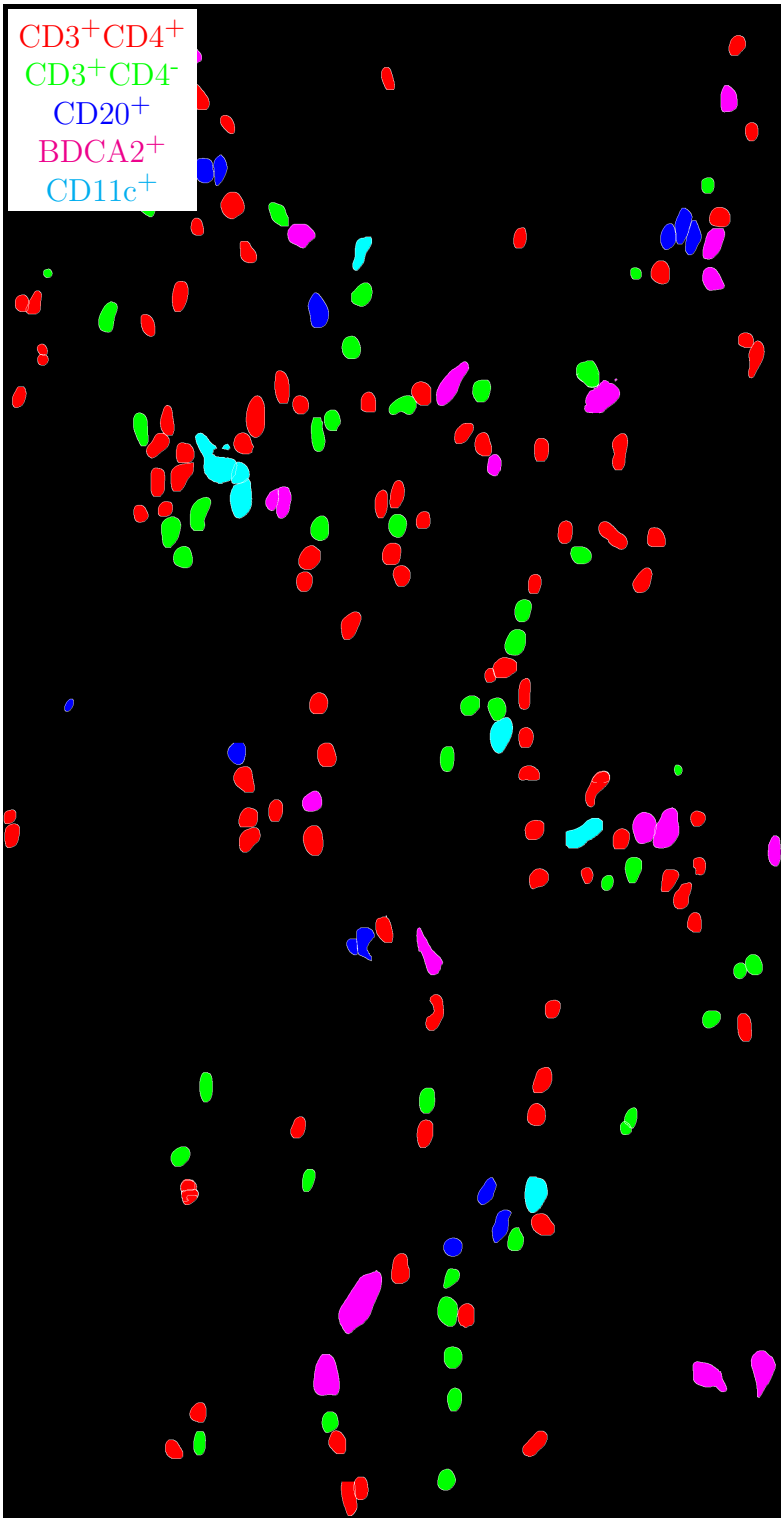

Figure H.115: **Label for H.116**. True label matrix size is 3787 × 1948. Depicted label matrix size is 3787 × 1948.

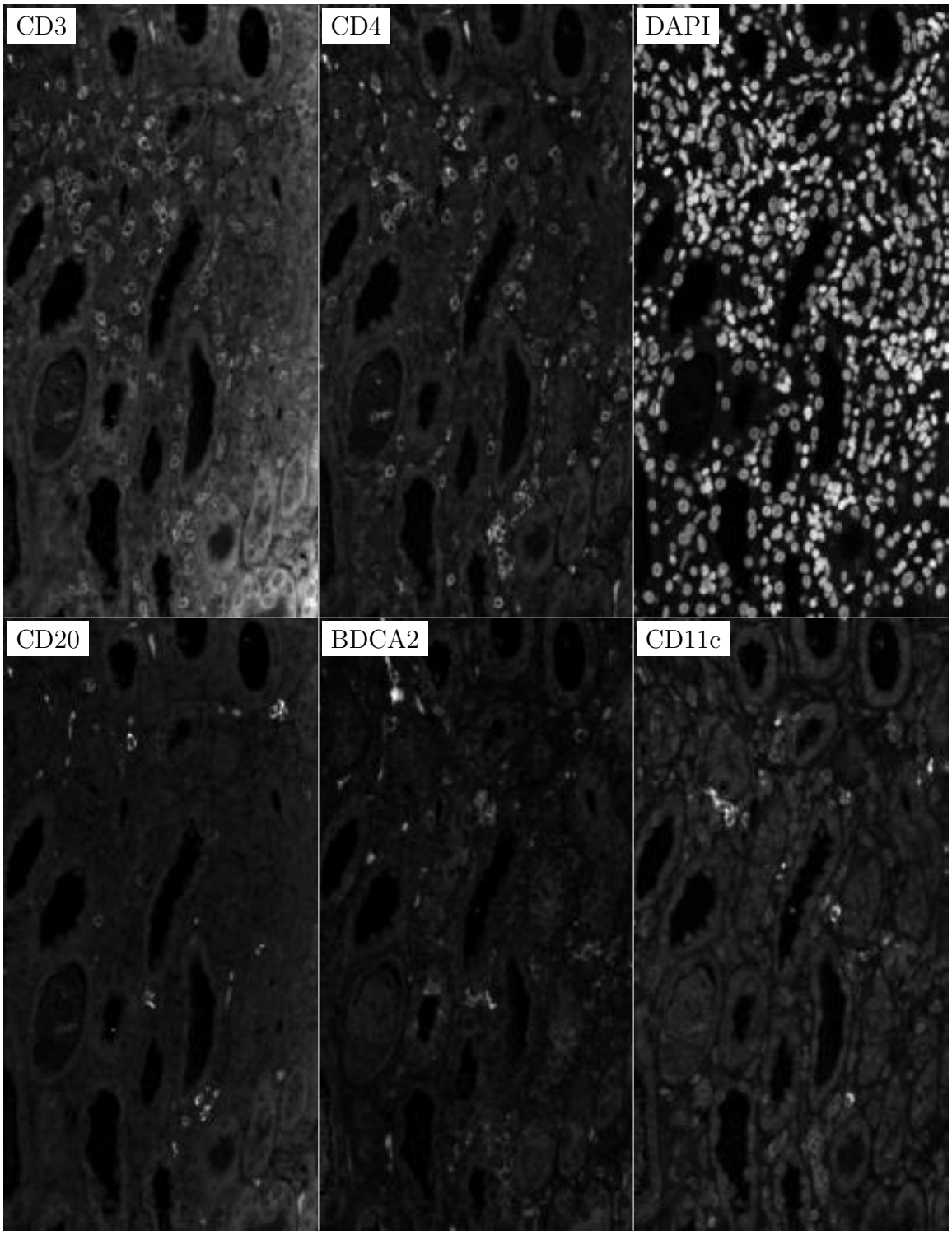

Figure H.116: **ROI for H.115**. True image channel matrix size is 3787 × 1948. Depicted image channel matrix size is 300 × 154.

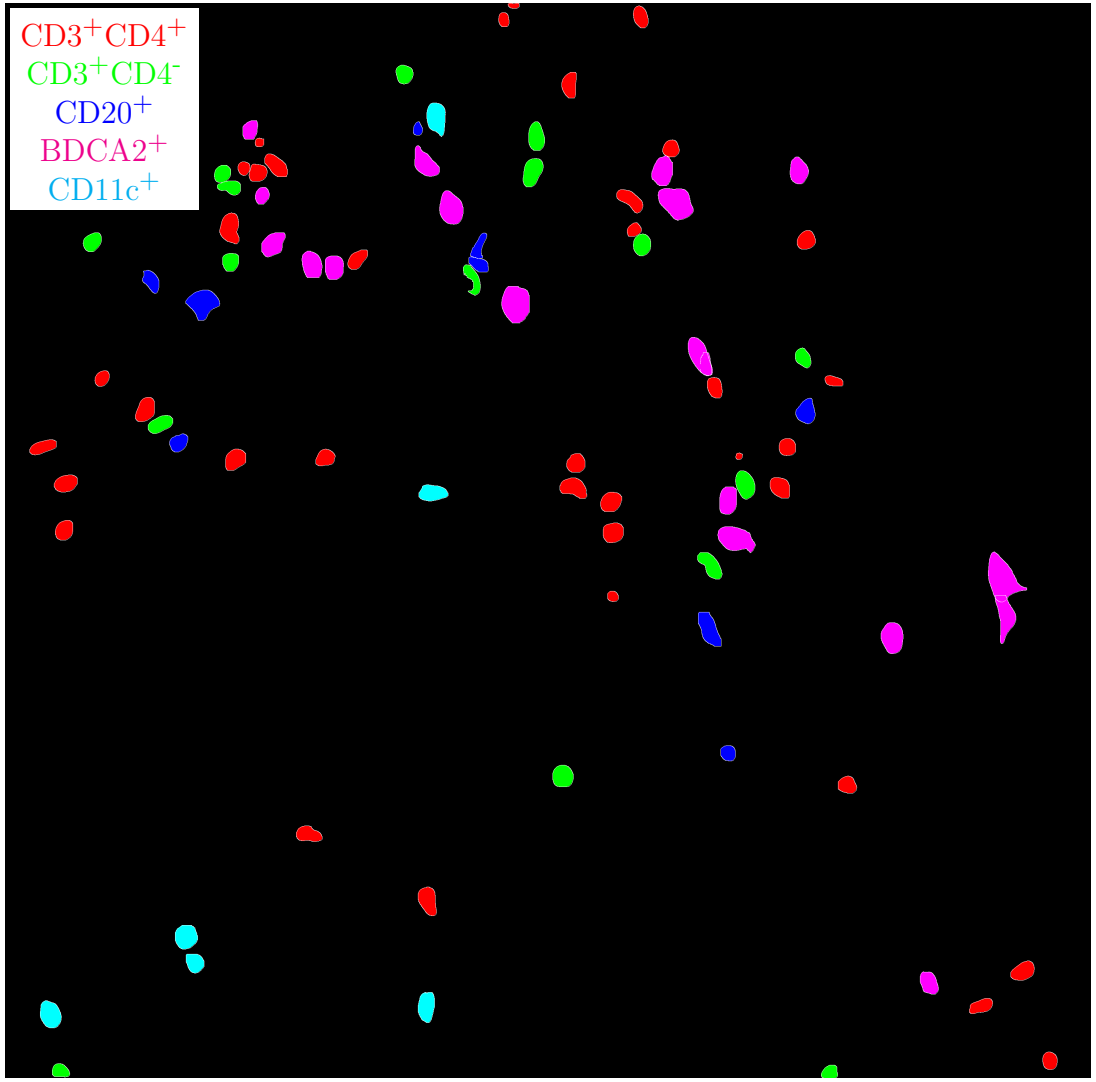

Figure H.117: **Label for H.118**. True label matrix size is 2867 × 2888. Depicted label matrix size is 2867 × 2888.

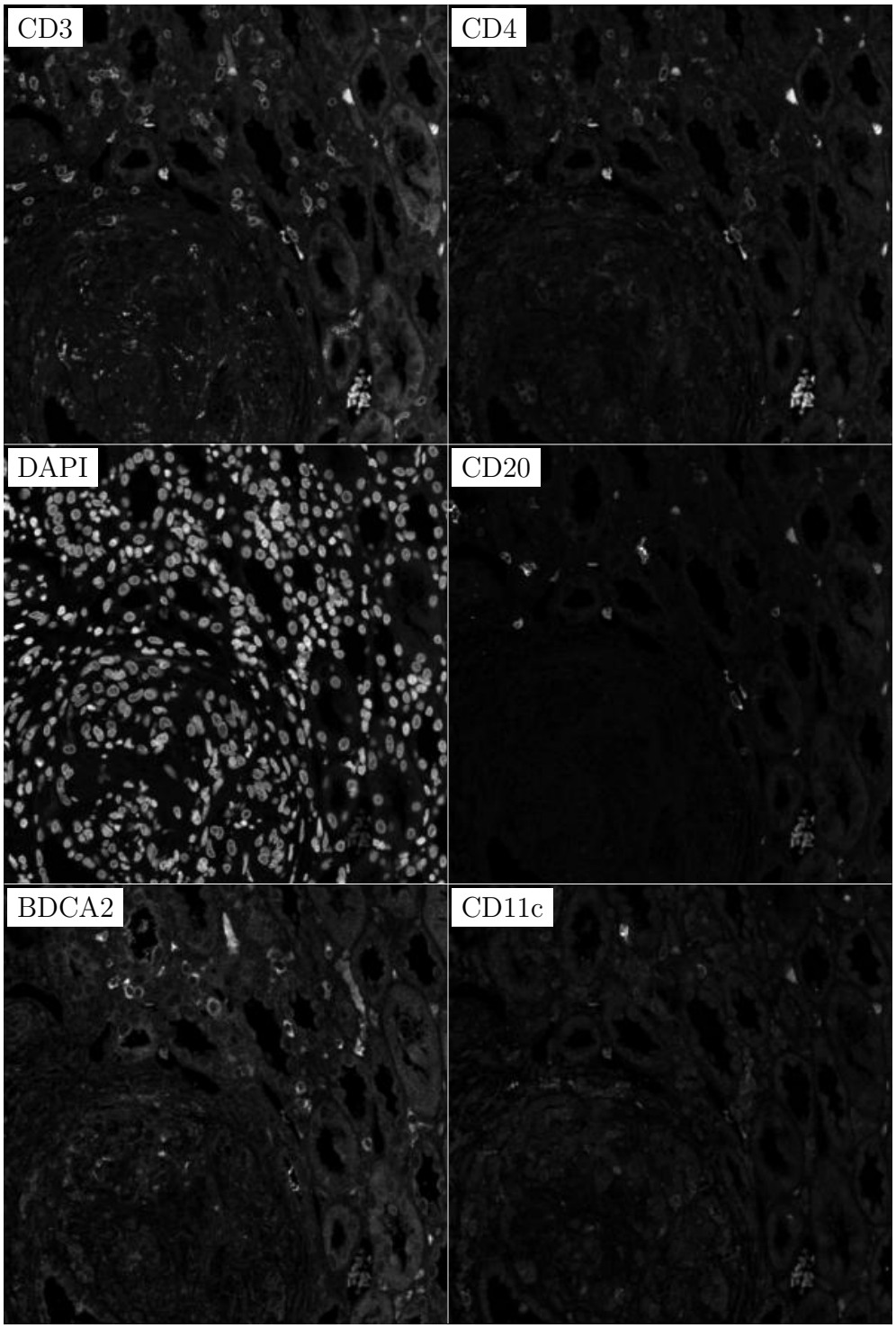

Figure H.118: **ROI for H.117**. True image channel matrix size is 2867 × 2888. Depicted image channel matrix size is 297 × 300.