Cell segmentation with traditional and deep learning methods on EVICAN - a partially annotated dataset of grayscale images of 30 different cell lines from multiple microscopes

EDWARD VENDROW* and ZIXIAN MA*, Stanford University, USA

ACM Reference Format:
Edward Vendrow and Zixian Ma. 2022. Cell segmentation with traditional and deep learning methods on EVICAN - a partially annotated dataset of grayscale images of 30 different cell lines from multiple microscopes. 1, 1 (April 2022), 4 pages.

1 INTRODUCTION

Improvements in microscopy have allowed researchers to acquire increasingly large amounts of cell image data, requiring human or automated systems for processing and screening. Using microscopy results to identify not just cells but cell types would be useful for scientific purposes and save dramatically on cost. An automated cell segmentation system would allow for live-cell imaging while reducing hands-on data labelling tasks.

One issue with cell datasets is cell heteroginy; many existing cell segmentation datasets feature just one type of cell. To this end, the EVICAN dataset [10] provides segmentation annotations for 30 cell types across 4600 images. We use this dataset to train a deep convolutional model with a U-Net architecture. Our model achieves impressive cell segmentation performance across many types of cells. We then demonstrate the performance of the model by comparing it against traditional segmentation methods.

2 BACKGROUND

Cell segmentation is the task of splitting a microscopic image into individual instances of cells [6]. As cellular morphology is an important indicator of a cell's physiological state, well-segmented images which capture biologically relevant morphological information can be extremely valuable in image-based cellular research [6]. With cell segmentation, scientists are able to analyze a wide range of biological features such as cell count, type, division, shape, etc. They can also quickly evaluate changes in cell features over time and in response to different experimental conditions. Therefore, cell segmentation is regarded as not only a fundamental step in many biomedical studies but also a driving force in drug discovery, diagnostics, and other important fields of biology, pharmacology, and personalized medicine [6].

Many methods have been developed for cell segmentation. All existing methods can be roughly categorized into traditional (i.e. non-deep-learning) and deep learning methods. Traditional methods include threshold-based approaches such as Otsu thresholding [7], as well as approaches based on feature extraction, level-set and graph-cut [11].

In recent years, as computational power of machines increased, deep learning methods have outperformed traditional methods in many prediction tasks. In the domain of image prediction, deep neural networks use successive convolutional layers to extract meaningful image information. Stacking these convolutional layers, which transform the input using convolution with a learnable and small-size kernel, leads to a Convolutional Neural Network (CNN). CNNs has been widely used and shown huge successes in image processing, including classification [5], detection [8], and most relevant to our task, segmentation [9] [2]. For example, the approach Mask R-CNN developed by[2] first proposes regions of interest (ROIs), then predicts masks over each ROI. Another prominent CNN network - U-Net - outputs the segmentation masks directly, and specifically targets biomedical image segmentation tasks [9]. Similar approaching utilizing deep neural networks have lead to important advances across the biocomputational domain [1] [4].

3 METHODS

We explore both traditional and deep-learning methods on the EVICAN cell segmentation dataset.

3.1 Traditional

For the traditional method, we mainly use Otsu thresholding [7]. Before obtaining the threshold, we preprocess the image with a high-pass filter and normalization. We've found Gaussian high-pass filter generates better results.

The aim of the Otsu's thresholding method is to find the optimal threshold value where the sum of foreground and background spreads is at its minimum [7]. It achieves this goal by iterating through all the possible threshold values and calculating a measure of spread for the pixel levels either fall in foreground or background [7].

3.2 Deep Learning

3.2.1 Data preparation. There are five steps in our data preprocessing pipeline.

- Step 1: Downsampling data. We downsample the images with blurred background in both the train and val folders and their corresponding masks to 224×224, and save them as our training (3714 images) and validation data (115 images) respectively.
- Step 2: Constructing a customized dataset: We use PyTorch's Dataset() class as a base class to construct a customized dataset called *EvicanDataset* where we define additional data preprocessing in the *__getitem__()* function (Step 4 and 5).

*Both authors contributed equally to this research.

Authors' address: Edward Vendrow, evendrow@stanford.edu; Zixian Ma, zixianma@stanford.edu, Stanford University, 620 Mayfield Avenue, Stanford, CA, USA, 94305.

XXXX-XXXX/2022/4-ART $15.00
https://doi.org/

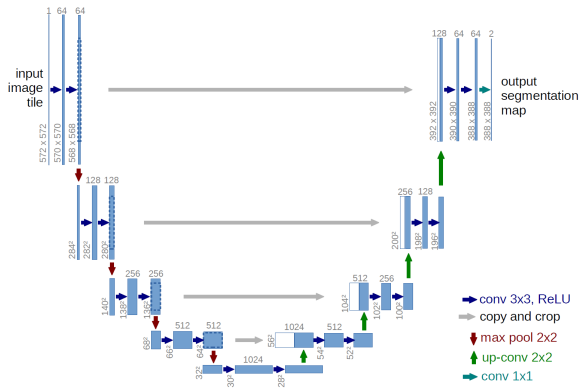

Fig. 1. U-Net model architecture using a contracting path to capture local context and a symmetric expanding path for precise localization [9].

- Step 3: Normalizing data. For additional data preprocessing, we first normalize both the raw cell images and the segmentation masks, since their pixel values vary and don't have a fixed range. In particular, we normalize the pixel values of the raw cell images to the [0, 1] range. We also turn the masks into binary masks by normalizing their pixel values to be either 0 or 1.
- Step 4: Additional preprocessing of images into trainable tensors: In order to fit our images into a PyTorch model for training, there are two additional steps: converting images into tensors, and permuting their dimensions. First, we convert the image into tensors. Second, we transpose/permute the dimensions of the data with *torch.Tensor.permute(2, 0, 1)* so that the color channel of the image is indicated by the first dimension instead of the last one. We perform this step because this is the conventional tensor representation of an image.
- Step 5: Data augmentation: Finally, we perform on-the-fly data augmentation with 4 different random transformations during training. Since we only have 3714 raw images, which is far from enough for training a deep learning model for this challenging segmentation task with 30 different cell lines, we employ data augmentation and expand our training dataset to 4 times larger with 4 transformations. Additionally, we make our data augmentation on-the-fly to optimize for space usage.

3.2.2 Network architecture. We choose the deep neural network architecture U-net, as it has shown state-of-the-art performance on segmentation tasks [9].The U-net is an encoder-decoder network which can take an image as its input, downsample and then upsample it to output an equal-sized image [9].

In Figure 1, we can see the U-net's architecture can be represented by a "U", and it consists of two paths. The left part (also called a contracting path) of the "U" is downsamping the image with a few convolutional and max pooling layers to get a more compact feature representation of the image. This path can be represented as: **conv_layer1 -> conv_layer2 -> max_pooling -> dropout(optional)** . More specifically, according to the U-net paper [9], this path consists of "the repeated application of two 3x3 convolution, each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for downsampling." The right part (or, the expansive path) then unsamples the image via the following layers: **conv_2d_transpose -> concatenate -> conv_layer1 -> conv_layer2**.

In details, every step in this path consists of an upsampling of the feature map followed by a 2x2 convolution, a concatenation with the cropped feature map from the contracting path, and two 3x3 convolutions. Each step is followed by the nonlinear unit ReLU. Finally, a 1x1 convolution serves to map each 64-component feature vector to the target number of classes at the last layer.

The reason why this U-net architecture works well on image segmentation problems, where we need to convert feature map into a vector and reconstruct an image from this vector, is that U-net utilizes the learned feature mapping of an image and use it (instead of a vector) to convert back to an image. This technique of using the feature map would preserve the structural integrity of the image and help reduce distortion.

3.2.3 Training procedure. We describe the major components of our training procedure below:

- data loader: we use PyTorch's *DataLoader* class for loading data with batches and multiple threads. We activate random shuffling for the training data.
- on-the-fly data augmentation: we perform 4 image transformations with the transform classes from torchvision.transforms, including horizontal flip, horizontal and vertical translation by 10%, rotating by [-10, 10] degrees, and scaling up and down by 10%, on the fly during each training epoch.
- loss function: we use *nn.BCEWithLogitsLoss()* for calculating our loss from output logits, since our groundtruth masks are binary.
- optimizer: we use the *Adam()* optimizer with an initial learning rate of 1e-4, and we decrease the learning rate to 5e-6 at the end.
- checkpoints saving: we save the state dictionary of the model with the best validation metrics, including mean Average Precision (mAP) and Jaccard Index (discussed below).

3.2.4 Evaluation. To draw fair comparisons between our method and the EVICAN MaskRCNN baseline, we use two metrics mAP and Jaccard Index to evaluate our deep learning method, as these are the metrics reported in their paper [10].

- mAP: The average precision metric is commonly used to evaluate a model's performance on object detection. The EVICAN paper counted predicted instances co-localized with corresponding ground truth instances as true positives when they have an intersection over union (IoU) scores above a certain threshold [10]. They calculated AP at IoU thresholds above 0.5 (AP0.5) and 0.75 (AP0.75) and report the average over all test images [10]. However, since the output of our model contains only segmentation masks but not bounding boxes, we can't obtain the IoU of each distance. Therefore, we calculate AP differently by counting each pixel as a true positive if it

Difficulty	Easy	Medium	Difficult	Average
Otsu thresholding	0.2055	0.2055	0.2090	0.2067
U-net	**0.8224**	**0.8224**	**0.8185**	**0.8211**

Table 1. The evaluation mAP scores of traditional (Otsu thresholding) and deep learning (U-net) method across various difficulties of evaluation data.

Difficulty	Easy	Medium	Difficult	Average
Otsu thresholding	0.1498	0.1498	0.1523	0.1506
U-net	**0.4232**	**0.4232**	**0.4126**	**0.4197**

Table 2. The evaluation Jaccard Index values of traditional (Otsu thresholding) and deep learning (U-net) method across various difficulties of evaluation data.

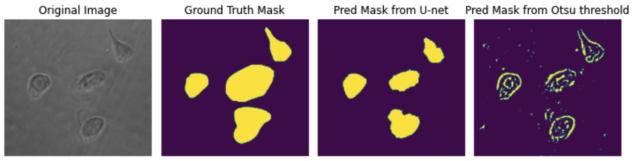

Fig. 2. Sample predictions from U-net and Otsu thresholding on a cell image. The predicted mask from U-net closely matches the ground truth mask, whereas the one from Otsu thresholding is much less accurate.

matches the corresponding pixel value in the ground truth mask. Specifically, we use the AP implementation in *torchmetrics* and obtain the mAP by take the average across all evaluation images. Note that while it might not be meaningful to directly compare our mAP score to the one reported in the EVICAN paper (i.e. 61.6%), we can draw fair comparison between the mAP scores of the traditional and deep learning method.

- Jaccard Index: The Jaccard Index, or Jaccard similarity coefficient, is a statistic used in understanding the similarities between sample sets [3]. The measurement is formally defined as the size of the intersection divided by the size of the union of the sample sets [3], and the mathematical representation of the index is written as:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

4 RESULT AND ANALYSIS

In Table 1, we see the performance of our deep learning model and Otsu thresholding as evalauted by the mAP score on the dataset evaluation partition across varying difficulties of segmentation. The U-net achieves a significantly higher mAP in all categories. Table 2 similarly shows performance of our deep learning model and Otsu thresholding as evalauted by the Jaccard Index on the same evaluation partitions. Again, the U-net achieves significantly higher Jaccard Index in all vategories. We also notice that the mAP score and Jaccard Index which decreases as segmentation difficulty increases, as expected.

Figure 2 shows an example of the segmentation performed by our deep learning model and Otsu thresholding. We create this mask from the deep learning model output by applying a Sigmoid function to the output, then using a 0.5 threshold to separate positive from negative predictions. We observe that qualitatively, the segmentation mask predicted by the U-Net is significantly better than the segmentation mask predicted by Otsu thresholding. The predicted mask from the U-net closely approximates the ground truth mask, capturing an accurate mask for each cell with little to no noise. Conversely, the predicted mask from Otsu thresholding mostly captures the outlines of the cells as well as a small amount of their contents. The Otsu thresholding mask also contains significant noise, with sparse noise spread throughout parts of the image.

Our results confirm the high performance of deep learning methods in cell segmentation tasks as compared to traditional image processing methods. Using the EVICAN dataset, we show that even a relatively small dataset is sufficient to train an accuarate model. As an predictor of cell morphology, count, type, division, and shape, cell segmentation is a fundamentally important step in many biomedical studies. We hope that our work will advance research and discussions on multi-type cell segmentation and thus contribute towards biomedical innnovation.

5 CONTRIBUTIONS

- Edward Vendrow: I processed EVICAN image and segmentation to prepare for training, and trained a U-net model. I also wrote evaluation scripts for the EVICAN evaluation set. For writing, I wrote the Introduction and Results and Analysis sections, and contributed to the background section.
- Zixian Ma: For implementation, I compared different cell image datasets, did some intial processing of the EVICAN data, trained the U-net model with and without data augmentations, and evaluated the traditional Otsu thresholding method. For writing, I was mainly in charge of the background and methods sections.

REFERENCES

[1] Varun Gulshan, Lily Peng, Marc Coram, Martin C Stumpe, Derek Wu, Arunachalam Narayanaswamy, Subhashini Venugopalan, Kasumi Widner, Tom Madams, Jorge Cuadros, et al. 2016. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *Jama* 316, 22 (2016), 2402–2410.

[2] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*. 2961–2969.

[3] Paul Jaccard. 1912. THE DISTRIBUTION OF THE FLORA IN THE ALPINE ZONE.1. *New Phytologist* 11, 2 (1912), 37–50. https://doi.org/10.1111/j.1469-8137.1912.tb05611.x arXiv:https://nph.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1469-8137.1912.tb05611.x

[4] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. 2021. Highly accurate protein structure prediction with AlphaFold. *Nature* 596, 7873 (2021), 583–589.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012), 1097–1105.

[6] Filip Lux and Petr Matula. 2020. Cell Segmentation by Combining Marker-Controlled Watershed and Deep Learning. arXiv:2004.01607 [eess.IV]

[7] N. Otsu. 1979. A threshold selection method from gray level histograms. *IEEE Trans. Systems, Man and Cybernetics* 9 (March 1979), 62–66. minimize inter class variance.

[8] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference*

on computer vision and pattern recognition. 779–788.

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv:1505.04597 [cs.CV]

[10] Mischa Schwendy, Ronald E Unger, and Sapun H Parekh. 2020. EVICAN—a balanced dataset for algorithm development in cell and nucleus segmentation. *Bioinformatics* 36, 12 (04 2020), 3863–3870. https://doi.org/10.1093/bioinformatics/btaa225 arXiv:https://academic.oup.com/bioinformatics/article-pdf/36/12/3863/33437424/btaa225.pdf

[11] Tomas Vicar, Jan Balvan, Josef Jaros, Florian Jug, Radim Kolar, Michal Masarik, and Jaromir Gumulec. 2019. Cell segmentation methods for label-free contrast microscopy: review and comprehensive comparison. *BMC bioinformatics* 20, 1 (2019), 1–25.