SHAPE-PRESERVING LOSS IN DEEP LEARNING FOR CELL SEGMENTATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

By
Furkan Hüseyin
July 2020

SHAPE-PRESERVING LOSS IN DEEP LEARNING FOR CELL SEGMENTATION
By Furkan Hüseyin
July 2020

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Çiğdem Gündüz Demir(Advisor)

Shervin Rahimzadeh Arashloo

Alptekin Temizel

Approved for the Graduate School of Engineering and Science:

Ezhan Karaşan
Director of the Graduate School

ABSTRACT

SHAPE-PRESERVING LOSS IN DEEP LEARNING FOR CELL SEGMENTATION

Furkan Hüseyin
M.S. in Computer Engineering
Advisor: Çiğdem Gündüz Demir
July 2020

Fully convolutional networks (FCNs) have become the state-of-the-art models for cell instance segmentation in microscopy images. These networks are trained by minimizing a loss function, which typically defines the loss of each pixel separately and aggregates these pixel losses by averaging or summing. Since this pixel-wise definition of a loss function does not consider the spatial relations between the pixels' predictions, it does not sufficiently impose the network to learn a particular shape(s). On the other hand, this ability of the network might be important for better segmenting cells, which commonly show similar morphological characteristics due to their natures. In response to this issue, this thesis introduces a new dynamic shape-preserving loss function to train an FCN for cell instance segmentation. This loss function is a weighted cross-entropy whose pixel weights are defined as prior-shape-aware. To this end, it calculates the weights based on the similarity between the shape of the segmented objects that the pixels belong to and the shape-priors estimated on the ground truth cells. This thesis uses Fourier descriptors to quantify the shape of a cell and proposes to define a similarity metric on the distribution of these Fourier descriptors. Working on four different medical image datasets, the experimental results demonstrate that this proposed loss function outperforms its counterpart for the segmentation of instances in these datasets.

Keywords: Deep learning, convolutional neural networks, cell instance segmentation, medical image analysis, shape-preserving loss, Fourier descriptors.

ÖZET

HÜCRE BÖLÜTLENMESI IÇIN DERIN ÖĞRENMEDE ŞEKIL-KORUYAN KAYIP

Furkan Hüseyin
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Danışmanı: Çiğdem Gündüz Demir
Temmuz 2020

Tam evrişimsel sinir ağları (TEA'lar), mikroskop görüntülerinde hücre bölütlemesi için en gelişmiş modeller haline gelmiştir. Bu ağlar, tipik olarak her pikselin kaybını ayrı ayrı tanımlayan ve bu piksel kayıplarının ortalamasını veya toplamını alan bir kayıp fonksiyonunu küçülterek eğitilir. Kayıp fonksiyonunun bu piksel bazlı tanımı, piksellerin tahminleri arasındaki uzamsal ilişkileri dikkate almadığından, ağı belirli bir şekli (şekilleri) öğrenmeye yetecek kadar eğitemez. Öte yandan, ağın bu yeteneği, doğaları gereği yaygın olarak benzer morfolojik özellikler gösteren hücreleri daha iyi bölütlemek için önemli olabilir. Bu soruna yanıt aramak adına, bu tez, hücre bölütlemesinde bir TEA'yi eğitmek için yeni bir dinamik şekil koruyan kayıp fonksiyonu ortaya koyar. Bu kayıp fonksiyonu, piksel ağırlıklarını öncül şekil bilgisine duyarlı olarak tanımlayan ağırlıklı bir çapraz düzensizliktir. Bu amaçla, ağırlıkları piksellerin ait olduğu bölütlenmiş nesnelerin şekli ile gerçek bölütleme (ground truth) hücrelerinde tahmin edilen şekil öncülleri arasındaki benzerlik temelinde hesaplar. Bu tez, bir hücrenin şeklini ölçmek için Fourier tanımlayıcılarını kullanır ve bu Fourier tanımlayıcılarının dağılımı üzerine bir benzerlik ölçütü tanımlamayı önerir. Dört farklı medikal görüntü veri seti üzerinde alınan deneysel sonuçlar, önerilen bu kayıp fonksiyonunun, bu veri setlerindeki örneklerin bölütlemesinde karşıtından daha iyi performans gösterdiğini ortaya koymuştur.

Anahtar sözcükler: Derin öğrenme, konvolüsyonal sinir ağları, hücre bölütlemesi, medikal görüntü analizi, şekil-koruyan kayıp, Fourier tanımlayıcıları .

Acknowledgement

I would like to express my gratitude to my advisor Assoc. Prof. Dr. Çiğdem Gündüz Demir for her continuous support, patience, and guidance. I have gained so many experience in my M.S. studies and it was all thanks to her. I would like to thank to my thesis committee, Asst. Prof. Shervin Rahimzadeh Arashloo and Prof. Alptekin Temizel, for reviewing this thesis.

I would like to thank to my parents Selver Hüseyin and Aleatin Hüseyin. I would like to thank to my sister Merve Hüseyin. Their everlasting support, understanding and love gave me the strength that I needed.

I would like to thank to my friends Melis Yılmaz, Anıl Özarslan, Gülşah Kaşdoğan and Eren Bellisoy for bringing joy to my life.

My special thanks goes to my girlfriend Berna Pekince. I would not find my motivation without her love and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Automated cell instance segmentation is a critical step in the analysis of cell morphology, diagnosis, and cell tracking. Manual segmentation of cell instances is a cumbersome task for experts. Moreover, manual segmentation is prone to error due to many factors such as lack of expertise or fatigue. Hence, automated cell instance segmentation is critical for a more robust workflow for medical image analysis. However, automated cell instance segmentation is a difficult task due to touching cells, variance in color and intensity, microscopy scanning defects, and faded boundaries.

Instance segmentation in traditional medical image analysis uses morphological, structural, and textural features in designing models and delineating boundaries of the instances [2–7]. With the advancements in deep learning, deep neural networks have become widely used models in instance segmentation. Early methods in deep neural networks for instance segmentation process images by cutting them into small patches and classifying each patch using a convolutional neural network (CNN). These early methods find instance boundaries by post-processing the predictions for image patches. There are studies that segment instances using shape-based methods on the classified pixels [8]. There are also other studies that detect instances by thresholding the posterior probabilities of the instance class [9] or finding regional maxima on these posteriors [10]. Recent studies in

instance segmentation focus on fully convolutional networks (FCNs), which process all pixels in parallel to generate a semantic segmentation map for a given image [11–14]. With the promising results of FCNs, their popularity and usage are increasing in the literature.

In this thesis, we focus on designing a loss function for FCNs, in which we embed shape information. Since morphological correctness of the segmented instances is important for better analysis of cell images, we focus on improving the morphological correctness of the state-of-the-art cell segmentation methods. Then, we extend our study beyond cell segmentation and run our method on different medical image segmentation tasks.

1.1 Motivation

Pixel-wise classification using encoder-decoder based architectures achieve state-of-the-art results in medical image segmentation [11–13]. Loss functions that are used in these architectures are defined in a pixel-wise manner and the choice of the loss function is an important factor in order to use FCNs at their full potential. There are many proposed loss functions to train segmentation networks [15, 16]. Loss weighting is a widely used method for improving the performance of the network training. Increasing the weight of a certain group of pixels in a loss function makes networks give more attention to those pixels during training. If foreground pixels are under-represented in the dataset, the weight of the foreground pixels can be increased to solve the class imbalance problem. In instance segmentation, background pixels locating between two near cells can be given more attention for a better separation of instances. However, these attention methods are static and give the same weight even though the instances are already separated.

Segmenting each instance in a given cell image, a human annotator has a general idea about the shape of a cell. For example, the annotator may look at the ground truth data of a cell type and may see that cells have a circular or tubular shape. Then s/he uses this information during segmentation. In Figure 1.1, one

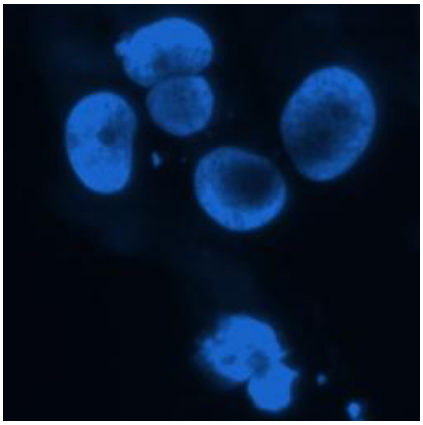


Figure 1.1: An example image of cells taken by a fluorescence microscope.

can see black spots inside cells. If these black spots are cropped and we are asked to classify whether they belong to the foreground or the background, it would be hard to decide because their intensity values are very close to those of background pixels. The human annotator labels these regions as foreground by using the information that these regions are part of a cell and cell instances are of circular. Hence, the segmentation process naturally involves using prior shape information. The aforementioned attention methods ignore such prior shape information during training.

Our motivation is to make segmentation tasks similar to manual segmentation, which involves understanding a general pattern about the shapes, by examining different image samples, and using this pattern during segmentation afterwards. In this thesis, we embed prior shape information in FCN training in order to achieve a pipeline similar to manual segmentation. Our proposed method learns the prior information about the data, just like a human annotator obtaining a general idea about shapes before segmenting. Then this prior-shape is embedded into the pixel-wise loss in order to use the FCN architectures at their full potential. Another motivation is to make a dynamic attention mechanism that weights an instance only if necessary, unlike the aforementioned static weighting methods.

1.2 Contributions

In this thesis, we propose a novel loss function, which we call *Fourier loss*, in order to use FCNs at their full potential by introducing a dynamic attention mechanism that uses prior shape information. Our method is defined on the output probability map of FCNs. Hence it can be used in different architectures. Our loss adjusts the weights of the regions by measuring the importance of these regions in terms of shape.

In the literature, there are models that use predefined attention mechanisms [11, 12, 17]. The same static attention map is used during training without considering the current performance of the model. Our method changes its attention map dynamically by considering the current performance of the model. To the best of our knowledge, the only work that is similar to ours has proposed to use star-shape-prior [1]. This work increases the weights that do not satisfy the star-shape prior. Unlike, the use of star-shape-prior, our method uses dataset-specific shape-priors by learning the shape prior from the training set so that our method can be used in the datasets that do not satisfy the star-shape-prior.

In the literature, most shape-preserving methods have been defined only for foreground/background segmentation. We have defined our method to work with multiclass segmentation problems as well. Prior shape information may also change from one class to another in a dataset. Our method is able to learn the shape prior of each class.

Different classes can have similar shape priors. Assume that there are two foreground classes whose shape priors are similar. If an instance belongs to the first class but the network classifies that instance as the second class, the shape prior cannot give attention to that instance because two classes have similar shape priors. Our method is defined to handle such cases. If an instance is classified incorrectly, our method still gives high weights for that instance even though its shape is correct.

1.3 Outline

The outline of this thesis is as follows. In Chapter 2, we give the background of deep learning and the related work on shape-preserving losses in deep learning. In Chapter 3, we explain the methodology and the formulations of our method. In Chapter 4, we introduce datasets and evaluation metrics that are used in our experiments. In Chapter 5, we give the quantitative and the visual results of our experiments. Then we discuss the results and explain the effects of our loss on FCN training. In Chapter 6, we conclude the thesis and discuss possible future research directions.

Chapter 2

Background

Medical image analysis literature can be classified into three categories: classification, detection, and segmentation. Classification is the process of assigning a label to each image [18–21]. Detection is the process of locating the objects in a given image without exact delineation of the object boundaries [9, 22–24]. Segmentation is the process of separating certain regions in the image by classifying each pixel as background or another foreground label (e.g., miscellaneous, inflammatory, epithelial or spindle-shaped) [12, 17, 22, 25–37]. Deep learning techniques are widely used in medical image segmentation in recent years. Loss functions of deep learning architectures have a significant effect on the learning process. Since morphological correctness of a segmented instance is an important factor in medical image segmentation, there are many proposed shape-preserving losses in deep learning. In this chapter, we give the related work on deep learning and the use of a shape-preserving loss in deep learning.

2.1 Deep Learning

The field of deep learning studies a family of neural network architectures and tools to train these neural networks (e.g., optimizers, losses, and initializers).

Deep learning is a subfield of machine learning. Conventional machine learning algorithms typically involves two steps: extracting features and learning the correlation between these features. Deep learning uses stacked layers with nonlinear transformations in order to learn the feature extraction. With its feature learning ability, deep learning achieved state-of-the-art results in many fields. As larger datasets have become available and GPU development has decreased the training and inference time, deep learning has gained popularity in recent years.

Deep neural network architectures have become larger, deeper, and more complex in recent years. However, the idea behind these architectures comes from a simple idea of a perceptron. A perceptron is an early version of a deep neural network. It is a linear mapping of a vector followed by a nonlinear transformation. Stacking these perceptrons creates deep neural networks. Each layer in a deep neural network generates a feature vector by taking the input generated by the previous layer.

Let $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{b} \in \mathbb{R}^m$ be vectors. Let $\mathbf{W}$ be an $m \times n$ matrix. Let σ be a nonlinear differentiable mapping. A simple perceptron $f(\mathbf{x})$ is defined as follows.

$$f(\mathbf{x}) = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{2.1}$$

A deep neural network $f(\mathbf{x})$ with L layers is defined as a series of stacked perceptrons which is given as follows.

$$f(\mathbf{x}) = \sigma(\mathbf{W}_L\sigma(\ldots \mathbf{W}_2\sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2 \ldots) + \mathbf{b}_L) \tag{2.2}$$

We can change the behavior of a neural network by changing the weights of the neural network. We will refer to all weights in a neural network as $\mathbf{W}$. Given a set of vector pairs $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$, we want our neural network to perform the following behavior. $f(\mathbf{x}_i) \simeq \mathbf{y}_i$, $\forall i \in \{1, 2, \ldots N\}$. Then finding the optimal values of weights $\mathbf{W}$ is defined as follows.

$$\mathbf{W}^* = \underset{\mathbf{W}}{\operatorname{argmin}} \sum_{i=1}^{N} \|f(\mathbf{x}_i) - \mathbf{y}_i\| \tag{2.3}$$

Since these stacked layers are differentiable, we can find the first-order derivative of the loss function with respect to weights using the backpropagation algorithm [38]. Then the loss can be minimized using these first-order derivatives.

The procedure explained above is called supervised learning. Neural networks are also used in an unsupervised way [39–42]. One unsupervised usage of neural networks is autoencoder. In an autoencoder architecture, an encoder maps a vector to a latent space and a decoder maps the vector in the latent space to the original space [see Figure 2.1]. So the input and output vectors have the same number of dimensions. Autoencoders are trained to reconstruct the original vector. Then the encoder can be used as a dimensionality reduction technique [43]. Another interesting training method uses generative adversarial networks (GANs) proposed by Goodfellow *et al.* [44]. GANs are trained using two networks. One of them tries to generate data and the other one tries to predict whether the given data is real or fake. This unique design has achieved promising results and gathered attention in deep learning community [39, 45].

The layer definition given in Equation 2.1 is called a fully connected layer. Fully connected layers assume that there is a correlation between each dimension. This assumption does not hold for image data. The correlation of two pixels in an image decreases as their distance increases. In order to achieve human performance in computer vision tasks, CNNs are developed by imitating the visual cortex by using convolutions to learn the spatial correlations in the image [46, 47]. CNNs use pooling layers to increase the receptive field and then the final outputs are converted to a vector to be used in fully connected layers to make the final prediction [see Figure 2.2]. These architectures achieved very good results in image recognition, localization, and detection [48–54]. Long *et al.* propose FCNs that take an input image and output a probability map for the entire image [55]. Later, FCN architectures have been improved and become an autoencoder-like form with skip connections [see Figure 2.3]. Skip connections propagate the fine

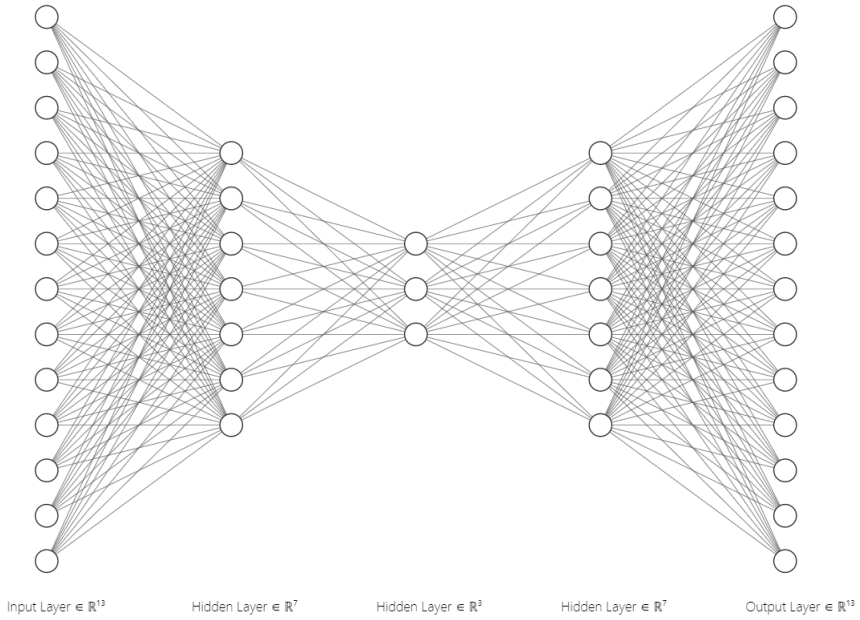


Figure 2.1: Example of an autoencoder arhitecture. Adapted from http://alexlenail.me/NN-SVG/index.html.

details which are lost due to the pooling operation. FCNs achieve the state-of-the-art results in image segmentation [11–13, 17, 56, 57]. Different tools have been designed and added to neural networks such as batch normalization, dropout, and spatial dropout [58–60]. With these additions, we have seen that there is more potential in deep learning awaits to be discovered.

2.2 Related Work on Shape-Preserving Loss in Deep Learning

The loss function is one of the most important parts of a neural network. A poor choice of the loss function can affect object-level metrics drastically. Since FCNs are defined as a classification problem per pixel, loss functions are defined in a pixel-wise manner. Although FCNs are designed to learn spatial correlations, they may fail to learn these spatial correlations due to this pixel-wise nature of losses. Previous studies show that incorporating shape information into segmentation

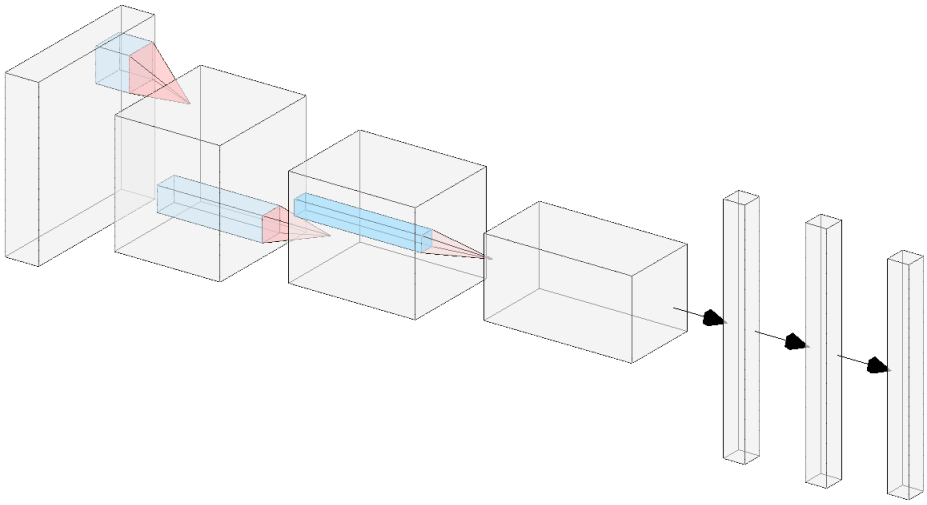


Figure 2.2: Example of a CNN architecture. Adapted from http://alexlenail.me/NN-SVG/AlexNet.html.

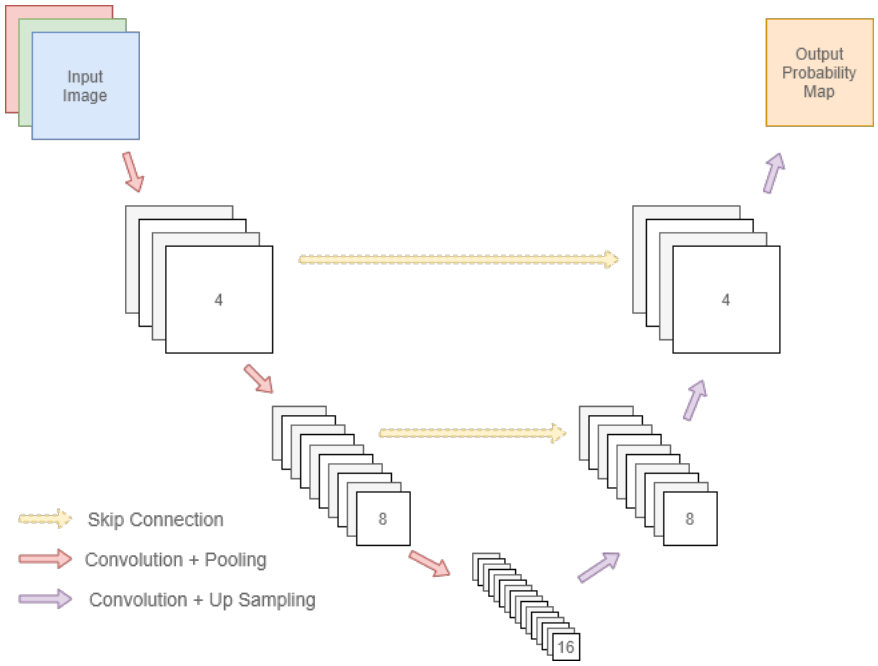


Figure 2.3: Example of an FCN architecture.

improves results in former segmentation methods [2, 61–64]. There exist loss functions in the literature that incorporate shape to solve the given segmentation problem.

We can classify shape-preserving losses using two dimensions: instance vs. semantic and prior vs. non-prior [see Figure 2.4]. Instance-based shape-preserving losses use connected components of the segmentation to extract shape information, on the other hand, semantic-based shape-preserving losses use the probability map or the thresholded segmentation map to extract shape information from the image instead of using these connected components. Non-prior-based shape-preserving losses use the ground truth data during training while prior-based shape-preserving losses do not use the ground truth data during training but make an external assumption on these objects' shapes. These two dimensions create four categories of shape-preserving loss in deep learning: prior-instance, prior-semantic, non-prior-instance, and non-prior-semantic. Loss functions designed using one of these four methods are used in FCN training.

Non-prior-instance methods use the ground truth during training and they consider connected components rather than the entire segmentation map [69–71]. One example of a non-prior-instance method could be pairing each segmented instance with the respective ground truth object in order to measure the shape similarity and use this similarity in a weight map. This would increase object-level metrics but with a high cost due to a drastic increase in training time. This approach is applicable when there are one or few objects in the image as it is the case in [70]. Since the pairing process takes too much time, non-prior-instance methods attempt to extract other information from the connected components so that the comparison with the ground truth can be faster. Hu *et al.* propose topology-preserving loss [69]. Topology-preserving loss uses different threshold values to generate segmentations from a given probability map. This process is called filtration. At each filtration, the number of connected components and holes, which are 0-dimensional and 1-dimensional homology structures [83], changes. These homology structures change exactly at critical points. The probabilities of these critical points are called persistent dots. They use the Wasserstein distance between the persistent dots of the ground truth and the segmentation map as the

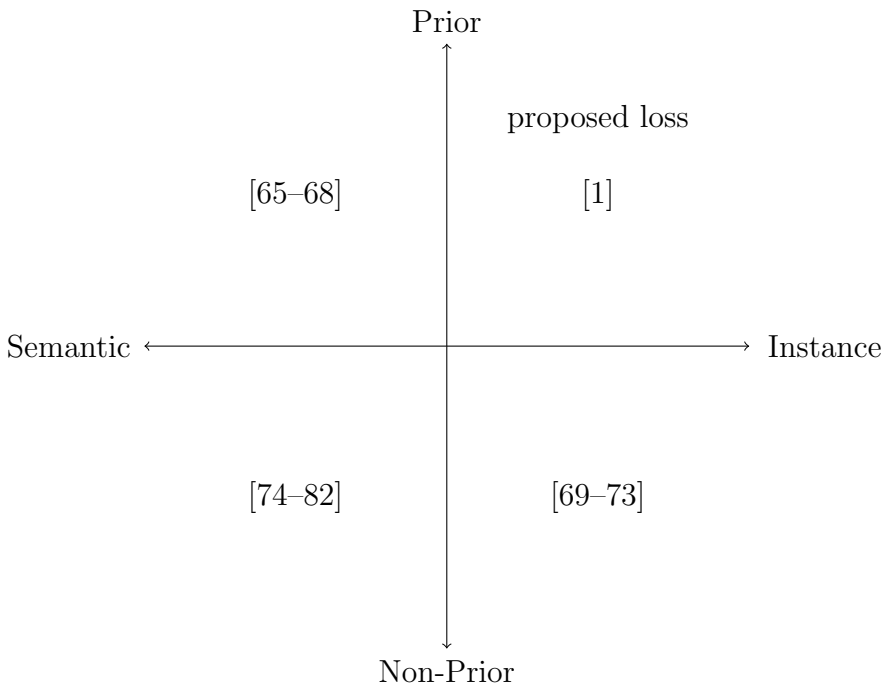


Figure 2.4: Taxonomy of shape-preserving losses in deep learning. Note that the proposed *Fourier loss* and the loss of [1] are different in terms of how they define this prior. This previous study [1] makes an external assumption on the objects' shape; it assumes objects are of the star-shape. Different than this approach, our proposed method learns the shape priors directly on the training data and does not neccessitate to make such an external assumption.

loss function, which is differentiable with respect to the weights of the neural network and the derivative flows to the critical points. Karimi *et al.* use distance transform to approximate one-sided Hausdorff distance as a differentiable loss [72]. These processes are examples of how a non-prior-instance segmentation could be achieved without an exhaustive pairing process.

Non-prior-semantic methods extract shape information from the entire image and also use the ground truth data during training [74–82]. This extracted information is expected to be the same for both segmentation and ground truth. Yan *et al.* propose a non-prior-semantic-based shape-preserving loss using the skeletal similarity metric [74, 84]. They find the boundaries of the segmentation and match these boundary segments with the ground truth boundary segments by searching the respective boundary segment in a limited range. Then they fit two cubic polynomials to these two segments and use the polynomial similarity to generate a weight map. Chen *et al.* derive a differentiable loss from an energy function defined on the ground truth and segmentation [75].

Prior-semantic methods extract shape information from the entire segmentation and do not use the ground truth. Tofighi *et al.* use predefined filters as a shape-prior [66]. Although they also embed ground truth in their loss function, their shape term is generated by annotators. Later they redesign their architecture to learn these shape priors as well. Zotti *et al.* use the probability of a voxel being foreground as the prior shape information in MR segmentation [68].

Prior-instance methods find connected components of the segmentation and use a prior shape without using the ground truth during training [1, 73]. Our proposed loss is also a prior-instance method. To the best of our knowledge, the only work that is similar to ours is the one that uses the star shape-prior [1]. The star shape prior is defined as follows. For any given two pixels in an instance, all points between these two pixels should also be in that instance. In [1], a pixel is weighted whether it satisfies this star shape definition or not. Our method does not use a predefined shape prior. Instead, our method learns the prior shape from the ground truth data before training and uses that prior shape information during training. For example, some gland objects may not satisfy the star shape

prior. As another example, the peripheral zone of pancreas images is crescent-shaped, which does not satisfy the star shape prior as well. On the other hand, our proposed method is able to learn these priors directly from training data, without making any assumptions on the prior beforehand. We have tested our method on such datasets to show that our method can work with a larger range of priors by learning them.

Chapter 3

Methodology

Our method defines a new loss function, which we call *Fourier loss*, and proposes to use this loss function for training a fully convolutional network (FCN). This is a weighted cross-entropy loss function that increases the weight of the instances whose segmentations are morphologically wrong. Fourier descriptors (FDs), which are used in shape characterization in the literature, are vectors generated from the boundaries of the objects. We assume that these vectors follow a multivariate distribution on a space, which we call shape space, and thus, we estimate this distribution with a mixture of Gaussians. This thesis proposes to define a dissimilarity metric using Mahalanobis distance on this distribution. By finding this dissimilarity metric for any given instance and using them in calculating pixel weights, it defines a weighted categorical cross-entropy loss to train FCNs for segmentation. The proposed method is illustrated in Figure 3.1 and its details are given in the following sections.

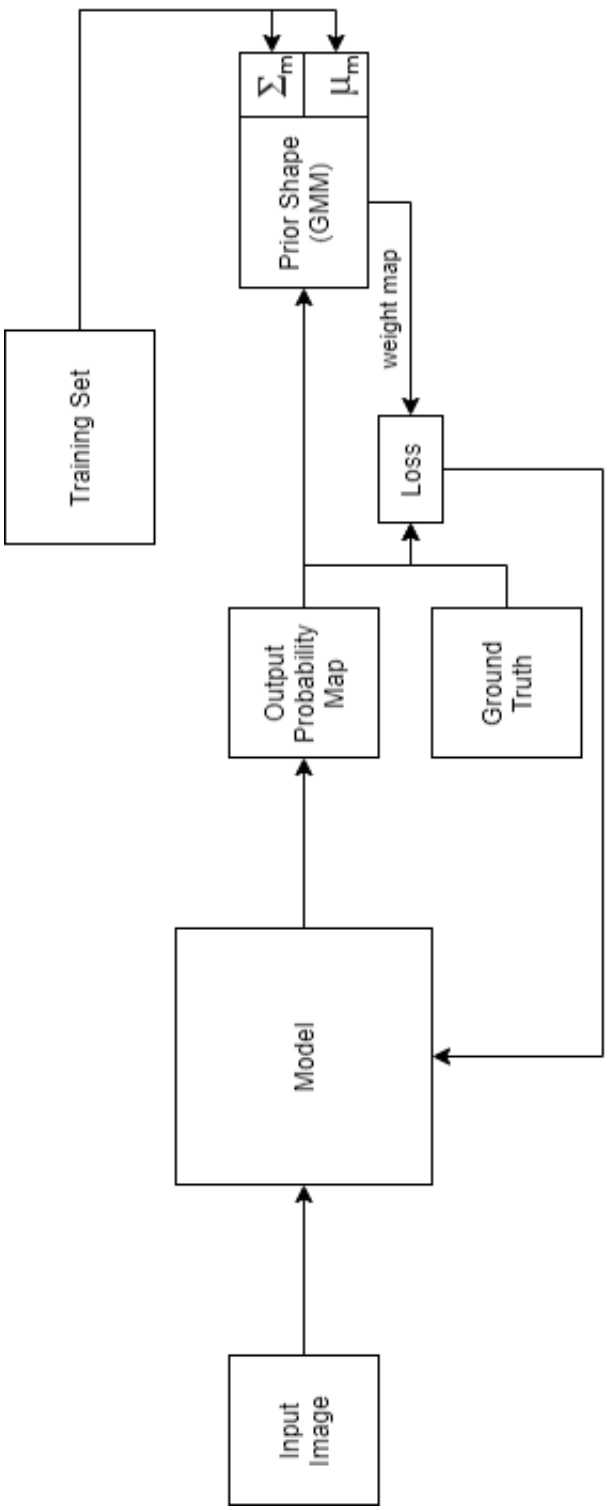


Figure 3.1: Overview of the proposed method, which uses our *Fourier loss* function.

3.1 Weighted Cross-Entropy Loss

Semantic segmentation is defined as a pixel-wise classification task. Fully convolutional networks predict a label for each pixel. Let $\hat{p}_c(j)$ be the probability of a pixel j for the c-th class and $p_c(j)$ be the ground truth of that pixel for the c-th class. In other words, $p_c(j) = 1$ if j-th pixel belongs to the class c and $p_c(j) = 0$ otherwise. Then the weighted categorical cross-entropy (or weighted cross-entropy for short) for pixel j is defined as follows.

$$L_{WCE}(j) = -w(j) \sum_{c=1}^{C} p_c(j) \log \left(\hat{p}_c(j)\right) \tag{3.1}$$

The weight, $w(j)$, of the pixel j can be changed to realize different attention mechanisms. For example, Ronnenberger *et al.* propose to use the distance from a pixel to the boundary of the two closest objects to define the weight for this pixel [11]. This weight is given in Equation 3.2 where w_c is the class imbalance weight for the class c that the pixel j belongs to, $d_1(j)$ and $d_2(j)$ are the distances to the first and second nearest cells from the pixel j, respectively, and σ and w_0 are two tunable hyperparameters.

$$w(j) = w_c + w_0 \exp\left(-\frac{\left(d_1(j) + d_2(j)\right)^2}{2\sigma^2}\right) \tag{3.2}$$

In this thesis, we mean such weighting when we refer the weighted cross-entropy. This weighting achieves the best results in instance segmentation so far. We will compare our loss function with this loss proposed in [11]. In our method, we redefine $w(j)$ in order to change its attention mechanism and make cross-entropy loss prior-shape aware. To this end, we first define a dissimilarity metric to calculate $w(j)$. This dissimilarity metric is defined on Fourier descriptors and the proposed *Fourier loss* is formulated on this metric. The details of this calculation and formulation are given in the following sections.

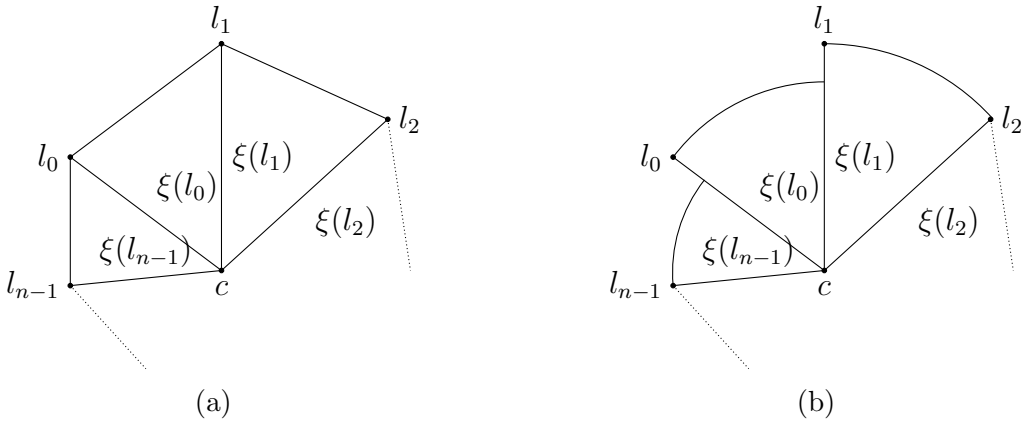


Figure 3.2: Examples of generating continuous curves from discrete points of the boundaries of an instance using two types of interpolation. Two curves look different in the figure because the neighbor points are drawn too separated from each other for demonstration purposes. (a) Line interpolation. (b) Circular arc interpolation.

3.2 Fourier Descriptors

The boundary of a segmented instance is a simple closed curve. Since the contour of an object in a digital image is a set of finitely many discrete points, we assume an interpolation between these discrete points to define a continuous curve [see Figure 3.2]. Since neighbor pixels in a boundary are very close to each other, the choice of an interpolation will not have a significant impact on the calculated value but it may ease especially the integral calculations. Thus, we use them interchangeably in our definitions.

Let $\gamma(o)$ be a simple closed continous curve, which is the interpolation of n pixels $\{p_0, p_1, ..., p_{n-1}\}$, of the contour of an object o. Let the length of $\gamma(o)$ be L. We define FDs on the domain of length $l_x \in [0, L]$ where l_x denotes the arc length of a section of the curve $\gamma(o)$ from its starting point p_0 to the point p_x of the same curve. So we have $l_0 = 0$ and $l_n = L$ because p_0 and p_n represent the same point. Let c be the center or mean point of all n points.

We define $\xi(l_x)$ as the distance of p_x to the point c where the point p_x lies on the curve at length l_x. We can expand $\xi(l_x)$ using Fourier series as shown below.

$$\xi(l_x) = a_0 + \sum_{k=1}^{\infty} \left[a_k \cos\left(\frac{2\pi k l_x}{L}\right) + b_k \sin\left(\frac{2\pi k l_x}{L}\right) \right] \tag{3.3}$$

where

$$a_k = \frac{2}{L} \int_0^L \xi(l_x) \cos\left(\frac{2\pi k l_x}{L}\right) dl_x \tag{3.4}$$

$$b_k = \frac{2}{L} \int_0^L \xi(l_x) \sin\left(\frac{2\pi k l_x}{L}\right) dl_x \tag{3.5}$$

We will only calculate a_k here. The calculation of b_k follows similar steps and only signs and trigonometric functions change. We can divide the integral in Equation 3.4 into n intervals of $[l_{i-1}, l_i)$. So the equation becomes as follows.

$$a_k = \frac{2}{L} \sum_{i=1}^{n} \int_{l_{i-1}}^{l_i} \xi(l_x) \cos\left(\frac{2\pi k l_x}{L}\right) dl_x$$

Assume that $\forall l_x \in [l_i, l_{i+1}), \xi(l_x) = \xi(l_i)$. This assumption will interpolate boundaries as in Figure 3.2b. This assumption allows taking $\xi(l_x)$ outside the integral. Note that this assumption is used only for $\xi(l_x)$ and we still assume the line interpolation when calculating the length of a curve.

$$
\begin{aligned}
a_k =& \frac{2}{L}\sum_{i=1}^{n}\xi(l_{i-1})\int_{l_{i-1}}^{l_i}\cos\left(\frac{2\pi k l_x}{L}\right)dl_x \\
a_k =& \frac{1}{\pi k}\sum_{i=1}^{n}\xi(l_{i-1})\left[\sin\left(\frac{2\pi k l_i}{L}\right)-\sin\left(\frac{2\pi k l_{i-1}}{L}\right)\right] \\
a_k =& \frac{1}{\pi k}\left[\xi(l_0)\sin\left(\frac{2\pi k l_1}{L}\right)-\xi(l_0)\sin\left(\frac{2\pi k l_0}{L}\right)\right. \\
&+\xi(l_1)\sin\left(\frac{2\pi k l_2}{L}\right)-\xi(l_1)\sin\left(\frac{2\pi k l_1}{L}\right) \\
&\vdots \\
&+\xi(l_{n-2})\sin\left(\frac{2\pi k l_{n-1}}{L}\right)-\xi(l_{n-2})\sin\left(\frac{2\pi k l_{n-2}}{L}\right) \\
&\left.+\xi(l_{n-1})\sin\left(\frac{2\pi k l_n}{L}\right)-\xi(l_n)\sin\left(\frac{2\pi k l_{n-1}}{L}\right)\right]
\end{aligned}
$$

Since $\gamma(o)$ is a closed curve, the last point p_n is indeed the starting point p_0, and thus, $\xi(l_0)=\xi(l_n)$ and $\sin(\frac{2\pi k l_0}{L})=\sin(\frac{2\pi k l_n}{L})$. By defining $\Delta\xi_i=\xi(l_{i-1})-\xi(l_i)$

$$
a_k = \frac{1}{\pi k}\sum_{i=1}^{n}\Delta\xi_i\ \sin\left(\frac{2\pi k l_i}{L}\right) \tag{3.6}
$$

Following similar steps, the coefficient b_k is expressed as

$$
b_k = -\frac{1}{\pi k}\sum_{i=1}^{n}\Delta\xi_i\ \cos\left(\frac{2\pi k l_i}{L}\right) \tag{3.7}
$$

Let A_k and α_k be k-th harmonic amplitude and k-th harmonic phase, respectively. So $A_k=\sqrt{{a_k}^2+{b_k}^2}$ and $\alpha_k=\arctan(b_k/a_k)$. This work uses the first K harmonic amplitudes of a truncated expansion of $\xi(l_x)$. Let $FD_\xi(\gamma(o))=[A_1,A_2,\ldots,A_K]$. We call $FD_\xi(\gamma(o))$ the Fourier descriptor of the contour $\gamma(o)$. Note that when $K\to\infty$ the curve can be reconstructed using these harmonic amplitudes together with their corresponding harmonic phases. However, we do not use the harmonic phases to define a descriptor as they provide less shape related information [85].

We will call this method of calculating FDs the center method. Zahn *et al.* use the cumulative angular function $\phi(l_x)$ instead of the distance function $\xi(l_x)$ [85].

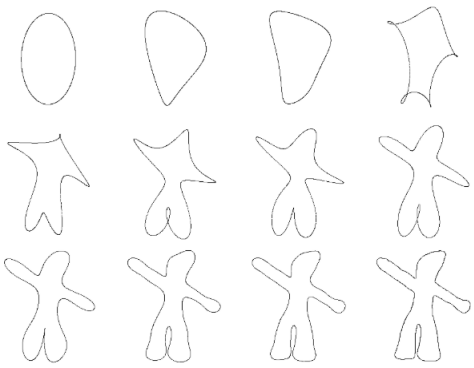


Figure 3.3: From left to right and top to bottom, K increases and the contour approaches to the original shape. Adapted from http://fourier.eng.hmc.edu/e161/lectures/fd/node1.html

Their formula is almost similar to ours. The only difference is that their formula replaces the change of distance $\Delta\xi_i$ to the change of angle $\Delta\phi_i$. We will call the method in [85] the angle method. We will refer to to Fourier descriptor calculated by the angle method using FD_ϕ. We will use both methods in calculating the *Fourier loss*.

We can reconstruct an approximation of the original curve using harmonic amplitudes and phase angles. The reconstructed curve will approach to the original curve as we increase K [see Figure 3.3]. Since the original contour can be reconstructed using harmonic amplitudes and phase angles, they contain shape information on each dimension. In Figure 3.3, we can only reconstruct the general outline of the object using a small number of K. So larger dimensions contain finer details about the contour and smaller dimensions contain more general information about the contour. So the choice of K is important to capture the prior shape information.

The process of calculating FDs can be seen in Figure 3.4. With FDs, we can convert any given instance to a vector in a K-dimensional space, which we call the shape space. The important highlight about FDs is that the FDs of similar shapes are close to each other in the shape space. For that reason, FDs are used in shape retrieval and shape discrimination [85–87].

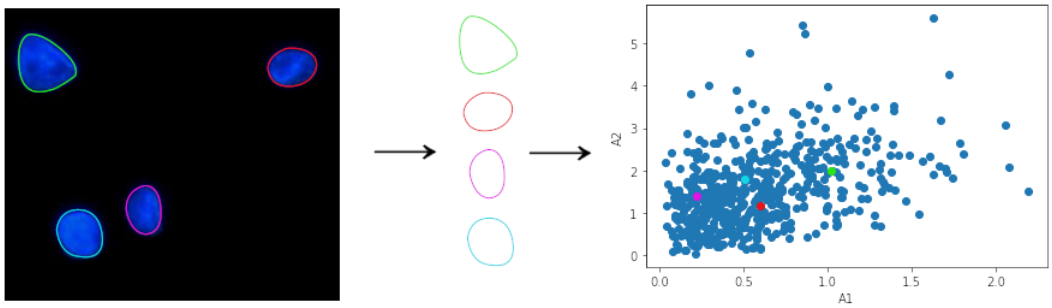


Figure 3.4: Representation of cell boundaries in the shape space. First two dimensions (A_1 and A_2) of the shape space are shown in this figure.

In Figure 3.4, only the first two dimensions of the shape space are shown for simplicity. After calculating the FDs of boundaries, each given instance falls into a point in the shape space. The FDs of the training set generate a multivariate distribution on this shape space. In the next section, we will estimate this multivariate distribution and use this estimation to define a dissimilarity metric for our loss.

3.3 Dissimilarity Metric

We calculate the FD of a given boundary using the procedure explained above. The FDs of the training set instances generate a multivariate distribution. We can estimate this distribution by fitting a mixture of Gaussians [88]. Gaussian mixture models (GMMs) are used for clustering and distribution estimation and they are trained using the expectation-maximization algorithm. Although there are many other methods for distribution estimations, GMMs are suitable for our method because we can use the mean vectors and the covariance matrices in order to define a dissimilarity metric.

Assume that there are C classes in a given dataset. For each class c, we can use a different K value. We fit a GMM with M_c Gaussians to the distribution of FDs of the objects in class c. Let $\gamma(o)$ be the contour of a given object o which is classified as class c. Let μ_c^m and Σ_c^m be the mean and the covariance of m-th

Gaussian of class c, respectively. We define the Mahalanobis distance of the curve $\gamma(o)$ to the m-th Gaussian of the class c as follows.

$$\tilde{d}_c^m(\gamma(o)) = \sqrt{(FD_\xi(\gamma(o)) - \mu_c^m)^T (\Sigma_c^m)^{-1} (FD_\xi(\gamma(o)) - \mu_c^m)} \tag{3.8}$$

Note that Equation 3.8 uses the center method for calculating the FDs. Since each class can use a different method for calculating the FDs, FD_ξ in Equation 3.8 can be changed to FD_ϕ.

The Mahalanobis distance normalizes distances for all distributions in a K-dimensional space. However, the distances are not scaled for all values of K. If we increase K, the average Mahalanobis distance of the points that are drawn from a K-dimensional Gaussian will also increase. This is because the distance in the introduced dimension will be directly added to the Mahalanobis distance. Since $\tilde{d}_c^m$ uses a different K value for each c, distances across classes are not scaled. We define $m_c(\gamma(o))$ as

$$m_c(\gamma(o)) = \underset{m}{\operatorname{argmin}}\, \tilde{d}_c^m(\gamma(o)) \tag{3.9}$$

Using all objects in the training set, we define $\tilde{\mu}_c^{m_c(\gamma(o))}$ and $\tilde{\sigma}_c^{m_c(\gamma(o))}$ as follows.

$$\tilde{\mu}_c^{m_c(\gamma(o))} = \frac{1}{n_o} \sum_{\substack{\gamma(o) \text{ s.t.} \\ o \in \text{class } c}} \tilde{d}_c^{m_c(\gamma(o))}(\gamma(o)) \tag{3.10}$$

$$\tilde{\sigma}_c^{m_c(\gamma(o))} = \sqrt{\frac{1}{n_o - 1} \sum_{\substack{\gamma(o) \text{ s.t.} \\ o \in \text{class } c}} \left(\tilde{d}_c^{m_c(\gamma(o))}(\gamma(o)) - \tilde{\mu}_c^{m_c(\gamma(o))} \right)^2} \tag{3.11}$$

Then we define the normalized Mahalanobis distance $d_c^m(\gamma(o))$ for the class c and the m-th Gaussian as follows.

$$d_c^m(\gamma(o)) = \max\left(0,\ \frac{\tilde{d}_c^m(\gamma(o)) - \tilde{\mu}_c^m}{\tilde{\sigma}_c^m}\right) \tag{3.12}$$

The Mahalanobis distance is defined on the range $[0, \infty]$. Since we are subtracting the mean, we introduce negative numbers when normalizing. A negative distance means that it is smaller than the average distance. In our case, this means that the curve satisfies our prior shape. So we map negative values to zero in order to map the normalized distances to the range $[0, \infty]$.

In order to provide better readibility, we remove the $m_c(\gamma(o))$ term from $d_c^{m_c(\gamma(o))}(\gamma(o))$, and thus, $d_c(\gamma(o)) \equiv d_c^{m_c(\gamma(o))}(\gamma(o))$.

So far, we assumed that there exists only one curve for an object. Although this assumption holds for the objects in the training set, it may not hold for the objects that are predicted by the FCN. If a predicted object has holes in it, there will be more than one contour for that object. Let $\gamma_i(o)$ be the i-th contour of the object o. We define the dissimilarity metric of the object o as follows.

$$d_c^*(o) = \max_i d_c(\gamma_i(o)) \tag{3.13}$$

For the sake of simplicity, we remove c term from $d_c^*(o)$. The term c is the class that object o belongs to, and thus, $d_c^*(o) \equiv d^*(o)$.

Since our dissimilarity metric $d^*(o)$ is defined on the range $[0, \infty]$, it can get arbitrarily large. Since weights of the pixels are multiplied with the gradients, large weights can regularize training too much and the network underfits without learning. For that reason, we set a maximum value d_{max} and we do not allow distances to be larger than d_{max}. The hyperparameter d_{max} determines the degree of regularization in our loss. Larger d_{max} means more regularization. In our experiments, d_{max} is chosen as 50. We have also tried 100 to see the regularization effect. We define the bounded dissimilarity metric of the object o as follows.

$$d(o) = \min\left(d_{max}, d^*(o)\right)$$

3.4 Fourier Loss

We have defined the dissimilarity metric for the object o as $d(o)$. In this section, we will define the *Fourier loss* using this dissimilarity metric. Let Y and $\hat{Y}$ be the ground truth and the segmented image, respectively. Let $c(j)$ and $\hat{c}(j)$ be the ground truth and predicted class for the j-th pixel in Y and $\hat{Y}$, respectively. We assume that zero is the background class. Let o_j be the object that the j-th pixel belongs to, provided that the j-th pixel is not classified as background. We define d_{min} as the minimum weight that a pixel can get so that the network does not forget the learned features. In our experiments, d_{min} is chosen as one.

Our dissimilarity metric can generate a weight for each segmented instance. Therefore, it cannot directly define a weight for false positive pixels. In order to give weights also to false positive pixels, we define a maximum distance of an image, $d_{\hat{Y}}$. The maximum value of an image cannot be defined if there exists no segmented instance in the predicted segmentation. For those reasons, we use an upper bound d_{max} if there exists no segmented object. At the end, for an entire segmentation $\hat{Y}$, we define $d_{\hat{Y}}$ as

$$d_{\hat{Y}} = \begin{cases} d_{max} & \text{if } \forall j \in \hat{Y}, \hat{c}(j) = 0 \\ \max\limits_{j \in \hat{Y}} d(o_j) & \text{otherwise} \end{cases}$$

We define the Fourier weight $w(j)$ for the pixel j as follows.

$$w(j) = \begin{cases} d_{min} & \text{if } c(j) = 0 \text{ and } \hat{c}(j) = 0 \\ d_{\hat{Y}} & \text{if } c(j) > 0 \text{ and } \hat{c}(j) = 0 \\ d_{\hat{Y}} & \text{if } c(j) > 0 \text{ and } \hat{c}(j) > 0 \text{ and } c(j) \neq \hat{c}(j) \\ \max\left(d_{min}, d(o_j)\right) & \text{if } c(j) = 0 \text{ and } \hat{c}(j) > 0 \\ \max\left(d_{min}, d(o_j)\right) & \text{if } c(j) > 0 \text{ and } \hat{c}(j) > 0 \text{ and } c(j) = \hat{c}(j) \end{cases}$$

In this equation, the first and second conditions correspond to true negatives and false negatives, respectively. The third condition corresponds to correctly segmented objects but the classes do not match. This case happens when there exist more than one cell type in a given image. The fourth condition corresponds to false positives. The last condition corresponds to true positives. This is the case where classes also match.

Then we define the *Fourier loss* as follows.

$$L_{Fourier}(j) = -w(j) \sum_{c=1}^{C} p_c(j) \log\left(\hat{p}_c(j)\right) \tag{3.14}$$

Equation 3.14 is used in FCN training. Although it uses a pixel-wise loss, the term $w(j)$ is now prior-shape aware. This defined loss has a dynamic attention mechanism. It pays more attention to the pixels that affect the morphological correctness of the segmentation.

Chapter 4

Experiments

We have tested our loss function on four different datasets: Nucleus Segmentation dataset, CoNSeP dataset, Decathlon Task 5 dataset, and Gland Segmentation dataset. We show that our proposed loss function can learn the prior information from various datasets and it can improve the results for these common tasks in medical image segmentation. We have used six different object-level metrics to measure the quality of the segmentation: precision, recall, F-score, Hausdorff distance, Dice score, intersection over union (IoU), and F-classification-detection-score. In this chapter, we give the details of the datasets, metrics, models, and implementation. Then we present quantitative and visual results in the next chapter.

4.1 Datasets

4.1.1 Nucleus Segmentation Dataset

The Nucleus Segmentation dataset consists of 37 fluorescence microscopy images. There are a total of 2661 nuclei in this dataset. Images are taken from the Huh7 and HepG2 liver cancer cell lines. Five Huh7 and five HepG2 images are randomly

selected to create the training set. The remaining 27 images are left as the test set. The Huh7 test set contains 11 images with 891 nuclei. The HepG2 test set contains 16 images with 985 nuclei. Training data further split into two and eight images to generate validation and training sets, respectively. Further information about this dataset can be found in [7]. This dataset is publicly available at http://www.cs.bilkent.edu.tr/ gunduz/downloads/NucleusSegData/.

4.1.2 Gland Segmentation Dataset

The Gland Segmentation dataset consists of 200 colon biopsy images. There are a total of 2102 glands in this dataset. There are 80 images with 891 glands in the training set. There are 20 images with 223 glands in the validation set. There are 100 images with 988 glands in the test set. Tissues are stained using hematoxylin-and-eosin (H&E). Images contain normal and colon adenocarcinomatous (cancerous) glands. The image resolution is 480×640. Further information about this dataset can be found in [89].

4.1.3 CoNSeP Dataset

The Colorectal Nuclear Segmentation and Phenotypes (CoNSeP) dataset consists of 41 H&E stained images. The image resolution is 1000×1000. There are a total of 24319 nuclei in this dataset. Each nucleus is annotated as one of the following classes: miscellaneous, inflammatory, healthy epithelial, dysplastic/malignant epithelial, fibroblast, muscle, or endothelial. Healthy epithelial and dysplastic/malignant epithelial nuclei are combined into one class which is called the epithelial class. Fibroblast, muscle, and endothelial nuclei are combined into one class which is called the spindle-shaped class. The validation set is generated randomly by taking 20 percent of the training set. Further information about this dataset can be found in [17]. The dataset is publicly available at https://warwick.ac.uk/fac/sci/dcs/research/tia/data/hovernet/.

4.1.4 Decathlon Task 5 Dataset

Medical Segmentation Decathlon Challenge consists of ten different tasks which are the segmentation of brain tumor, heart, liver, hippocampus, prostate, lung, pancreas, hepatic vessel, spleen, and colon. All of them are semantic segmentation tasks. We have chosen the fifth task, which is pancreas segmentation, for our experiments. In this dataset, there are a total of 48 images. Thirtytwo of them belong to the training set and 16 of them belong to the test set. The validation set is generated randomly by taking 20 percent of the training set. There are two classes which are transitional zone and peripheral zone. Images are CT scans. Each section of the same CT image is considered as one image in our experiments. This dataset is publicly available at https://decathlon.grand-challenge.org/.

4.2 Implementation Details

All code that is used in this thesis is written in Python 3 and C. The calculations of Hausdorff distance, Dice index and intersection over union are written in C and called from a Python 3 code using Python C Extensions. All neural networks are implemented and trained using the Keras framework [90]. Networks are trained on a Linux server with three GPUs (two GeForce GTX 1080 Ti and one GeForce GTX 2080 Ti).

Implementation details for each network and dataset are summarized in Table 4.1. All networks are randomly initialized using Glorot Uniform initialization [91]. We have used early stopping on validation loss. The batch size is 8 for the CoNSeP Segmentation dataset and 2 for others. We have used Adam optimizer [92] for the CoNSeP Segmentation dataset and AdaDelta optimizer [93] for others. In the Micro-Net architecture [13], dropout [59] with 0.5 probability is used for the last hidden layers only. In the U-Net and DCAN architectures, dropout with 0.2 probability is used for all hidden layers. Input images are normalized between 0 and 1 for the CoNSeP Segmentation dataset. Input images are standardized (subtracting the mean and dividing by the standard deviation for all

channels) for other datasets. We have used normalization on the Nucleus Segmentation dataset for one experiment to show the robustness of our loss. Since Micronet is defined with a specific input resolution (252×252), we crop the images into 252×252 patches on datasets that use Micro-Net. We cut the images by skipping certain pixels on horizontal and vertical axes. We call the number of skipped pixels stride. We use 128×128 stride to generate training and the validation patches and 64×64 stride to generate test patches on the CoNSeP Segmentation dataset. In instance segmentation tasks, annotations of objects touch each other. We remove the dilated boundaries from objects in order to separate instances in the segmetation map so that the network can learn to separate instances. Since the Decathlon Task 5 Segmentation dataset is a semantic segmentation task and there is maximum one object per class in each image, we do not remove the boundary from the objects. Since miscellaneous cells are too small in the CoNSeP dataset, we have removed the boundary from the objects without applying any dilation. Since Micro-Net models overfit easily due to its larger number of learnable parameters, we have used horizontal flip, vertical flip, rotation, Gaussian blur, and median blur while training an FCN for the CoNSeP dataset.

4.3 Evaluation

Segmentation results are quantitatively evaluate using the object-level F-score, Hausdorff distance, object-level Dice score, intersection over union (IoU), and F-classification-detection-score [17]. In this section, we will give the definitions and intuitive explanations of these metrics.

4.3.1 Object-Level F-Score

The precision metric measures that out of all segmented instances how many of them are really true segmentations. The recall metric measures that out of all ground truth instances how many of them are detected. We calculate the

precision and recall metrics as follows. We match all segmented instances with the respective maximally overlapping ground truth objects. Then we classify each object pair or object into one of three categories: true positive (TP), false positive (FP), and false negative (FN). We define them as follows.

- **True Positive (TP):** We call a pair of a ground truth object and a segmented instance a true positive if the segmented instance intersects with the ground truth object more than 50 percent.
- **False Positive (FP):** We call a segmented instance a false positive if this segmented instance does not intersect with a ground truth object more than 50 percent.
- **False Negative (FN):** We call a ground truth object a false negative if there is no segmented instance that matches with this ground truth object.

Then we define precision and recall as follows.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

In many segmentations models, segmentation maps are postprocessed such that instances whose areas are less than an area threshold are eliminated in order to increase the performance of these models. We use a similar small area elimination in our work. As we increase the area threshold, we will get higher precision values because we are eliminating false positives. In the meantime, we will get lower recall values after a certain threshold value because we are also eliminating some of the true positives and making them false negatives. So we have to choose an area threshold which increases precision as much as possible without decreasing recall. For that reason, we need another metric that takes both precision and recall into account. F-Score is the metric that considers both

precision and recall in the same metric. F-Score is the harmonic mean of precision and recall and it is calculated as follows.

$$F\text{-}Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

4.3.2 Object-Level Dice Index

The object-level Dice index measures the overlap of the segmented instances and the ground truth objects. Let X be a set of pixels in a segmented instance and Y be a set of pixels in a ground truth object. Then we define the Dice index as follows.

$$Dice(X, Y) = \frac{2|X \cap Y|}{|X| + |Y|}$$

Given the segmented instances, we match all instances with the respective maximally overlapping ground truth object as in object-level F-score definition.

- Let S be the set of all segmented instances.
- Let G be the set of all ground truth objects.
- Let $s_i \in S$ be the i-th segmented instance in S.
- Let $g_i \in G$ be the ground truth object which maximally overlaps s_i given that g_i and s_i belong to the same image.
- Let $\tilde{g_i} \in G$ be the i-th ground truth object in G.
- Let $\tilde{s_i} \in S$ be the segmented instance that maximally overlaps $\tilde{g_i}$ given that $\tilde{g_i}$ and $\tilde{s_i}$ belong to the same image.

Then we define the object-level Dice index as follows.

$$Dice_{object}(G,S) = \frac{1}{2}\left[\sum_{i=1}^{|S|}\omega_i Dice(g_i,s_i) + \sum_{i=1}^{|G|}\tilde{\omega}_i Dice(\tilde{g}_i,\tilde{s}_i)\right]$$

$$\omega_i = |s_i| / \sum_{j=1}^{|S|}|s_j|,\ \tilde{\omega}_i = |\tilde{g}_i| / \sum_{j=1}^{|G|}|\tilde{g}_j|$$

Note that this equation takes a weighted summation of the Dice indices calculated for the pairs of segmented and ground truth instances. Here the weights (ω_i and $\tilde{\omega}_i$) are calculated based on the areas of the instances.

4.3.3 Hausdorff Distance

The object-level Hausdorff distance measures the similarity between the segmented instances and the ground truth objects. Let X be a set of pixels in a segmented instance and Y be a set of pixels in a ground truth object. We define the Hausdorff distance between X and Y as follows.

$$H(X,Y) = \max\{\sup_{x\in X}\inf_{y\in Y}\|x-y\|, \sup_{y\in Y}\inf_{x\in X}\|x-y\|\}$$

Our definitions for G, S, s_i, $\tilde{g}_i$, ω_i, and $\tilde{\omega}_i$ hold for the definition of object-level Hausdorff distance. We redefine g_i and s_i as follows.

- Let $g_i \in G$ be the ground truth object which maximally overlaps s_i given that g_i and s_i belong to the same image. If there exists no ground truth object which overlaps with g_i, we define g_i as the ground truth object that has the minimum Hausdorff distance from s_i given that g_i and s_i belong to the same image.

- Let $\tilde{s}_i \in S$ be the segmented instance that maximally overlaps $\tilde{g}_i$ given that $\tilde{g}_i$ and $\tilde{s}_i$ belong to the same image. If there exists no segmented instance

which overlaps with $\tilde{s_i}$, we define $\tilde{s_i}$ as the segmented instance that has the minimum Hausdorff distance from $\tilde{g_i}$ given that $\tilde{g_i}$ and $\tilde{s_i}$ belong to the same image.

Then we define the object-level Hausdorff distance as follows.

$$H_{object}(G, S) = \frac{1}{2}\left[\sum_{i=1}^{|S|} \omega_i H(g_i, s_i) + \sum_{i=1}^{|G|} \tilde{\omega_i} H(\tilde{g_i}, \tilde{s_i})\right]$$

4.3.4 Intersection over Union (IoU)

Intersection over union (IoU) is a commonly used metric in object detection. It is also used in instance segmentation to evaluate the morphological correctness. Let X be a set of pixels in a segmented instance and Y be a set of pixels in a ground truth object. Then we define the IoU as follows.

$$IoU(X, Y) = \frac{|X \cap Y|}{|X \cup Y|}$$

For each ground truth and segmented instance pair, we calculate the IoU. Let t be the threshold value that we choose. Then we define TP_t, FP_t, and FN_t as follows.

- **True Positive (TP_t):** A ground truth object whose intersection over union with a segmented instance is larger than t.
- **False Positive (FP_t):** A segmented instance whose intersection over union with all ground truth objects is less than or equal to t.
- **False Negative (FN_t):** A ground truth object whose intersection over union with all segmented instance is less than or equal to t.

Then we define IoU for a threshold t as follows.

$$IoU_t = \frac{TP_t}{TP_t + FP_t + FN_t}$$

In order to perform a better evaluation, it is very common to use multiple thresholds. Let T be a set of thresholds. Then the IoU for a segmentation is defined as

$$IoU = \frac{1}{|T|} \sum_{t \in T} IoU_t$$

In our experiments, we have chosen $T = \{0.50, 0.55, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, 0.90\}$.

4.3.5 F-Classification-Detection-Score

This metric is proposed in [17]. For multiclass segmentation problems, we calculate metrics for each class independently in order to evaluate classification quality. Previously defined metrics ignore the detection quality. This metric considers class-independent detection as evaluating the classification for each class. For each ground truth and segmented instance, we calculate the centroid and create a set of centroids with the class information attached to them. We pair ground truth and segmented instances using linear sum assignment [94] with the distance between centroids. Ignoring class values, we will have TP, FP, and FN values from this pairing. When we include class information, we have correctly classified instances of class c (TP_c), correctly classified instances of other classes (TN_c), incorrectly classified instances of class c (FP_c), and incorrectly classified instances of other types (FN_c). Then we will calculate F_c as follows.

$$F_c = \frac{2(TP_c + TN_c)}{2(TP_c + TN_c + FP_c + FN_c) + FP + FN}$$

Then we calculate a general detection quality F_{all} as follows.

$$F_{all} = \frac{2TP}{2TP + FP + FN}$$

4.4 Parameter Selection

The proposed *Fourier loss* has two sets of hyperparameters. The first set of hyperparameters is K, M, method type, and covariance type. These parameters are optimized in order to find the best prior shape estimation. The second set of hyperparameters are the structuring element sizes in erosion (E) and dilation (D), and the area threshold (A) which are used in postprocessing. We explain these hyperparameters and their optimizations in the following subsections.

4.4.1 Parameters of Fourier Loss Calculation

There are four hyperparameters involved in calculating the proposed *Fourier loss*: K, M, covariance type, and method type. For simplicity, we have dropped the class subscripts because each class uses different K, M, method type, and covariance type values and we have made a hyperparameter search for each class independently. K is the dimensionality of the shape space. M is the number of Gaussians in the GMM. The method type corresponds to either using the center or the angle method for calculating FDs. The covariance type is the parameter in GMM estimation.

The value of M, method type, and covariance type directly affect the distribution. In order to find a better estimation of the prior shape distribution, we need to make a hyperparameter search for these parameters. The K value determines how much detail we consider in our segmentation. Since we want to capture only the necessary shape information, but not all the details, we need an optimal value of K.

We fit the FDs of the training set to a GMM. Since the instances of the validation data are also morphologically correct, we expect a small average loss if we define the loss as our dissimilarity metric. We try N from 1 to 100; K from 1 to 20; covariance type as full, diagonal, and spherical; and the method type as center and angle. We select a combination that minimizes the average dissimilarity score of the FDs of the validation data. Since our hyperparameter space is too large, grid search and random search are poor choices for their selection. The tree-structured parzen estimator (TPE) models the joint probability of hyperparameters and loss [95]. Then it makes more educated guesses for hyperparameters to find the minimum point in the hyperparameter space. We use the TPE algorithm to find the values of the hyperparameters used in our loss function calculation. This hyperparameter search is done only once for a dataset. Then the same values are used for training multiple networks. The chosen hyperparameter values are given in Table 4.2.

4.4.2 Post-Processing

After obtaining the class labels for each pixel by feeding the images to the trained network, we apply post-processing to all instances of all classes to obtain the final segmentation. The post-processing procedure is given in Algorithm 1. There are three hyperparameters in post-processing procedure: the structuring element size in erosion(E) and dilation(D), and the area threshold (A). Note that the average instance area of each class is different than the others. The class with a larger average area can tolerate larger E and A and might need larger D in case of under-segmentation. For that reason, we use different hyperparameters for different classes. We perform a grid search in the hyperparameter space and select the optimal values for the hyperparameters which give the maximum IoU for the combination of the training set and the validation set. The chosen hyperparameter values for postprocessing are given in Table 4.3.

We define the hyperparameters as follows.

Algorithm 1 A pixel in $cPred$ is 1 if it is segmented as class c and 0 otherwise.

```
1: procedure POSTPROCESSING(cPred, E, D, A)
2:     Erode cPred with a structuring element of size (E × E)
3:     components ← connectedComponents(cPred)
4:     for each component in components do
5:         Dilate component with a structuring element of size (E × E)
6:         Dilate component with a structuring element of size (D × D)
7:         if area(component) < A then
8:             Remove component from components
       return components
```

- **The structuring element size E for erosion:** The given segmented image is eroded with a structuring element of size $(E \times E)$ in order to separate the instances that are barely touching each other. After eroding the image, connected components are found and the remaining procedures are applied to each connected component. Since the image is eroded, we dilate each component with the same structuring element in order to make them the same size before the erosion operation. While performing the grid search, we have used the following values $E = \{0, 3, 5, 7, 9\}$.

- **The structuring element size D for dilation:** Each connected component is dilated with a structuring element of size $(D \times D)$. Since we remove the boundary from each connected component during training, we need to dilate each component in order to compare it with the test set. Dilating each connected component also helps with the under-segmentation problem to some extent. While performing the grid search, we have used the following values $D = \{3, 5, 7, 9, 11\}$.

- **Area threshold (A):** In order to eliminate false positives in the segmentation, we choose an area threshold value and remove all components whose areas are less than the chosen area threshold. While performing the grid search, we have used the following values $A = \{0, 50, 100, 250, 500\}$.

Table 4.1: Implementation details for all experiments.

Dataset	Model	Optimizer	Dropout	Batch Size	Image Resolution	Model Input Resolution	Stride	Boundary Dilate
Nucleus Segmentation	U-Net	AdaDelta	0.2	2	768×1024	768×1024	-	5
	DCAN	AdaDelta	0.2	2	768×1024	768×1024	-	5
Gland Segmentation	U-Net	AdaDelta	0.2	2	480×640	480×640	-	5
Decathlon Task 5 Segmentation	U-Net	AdaDelta	0.2	2	320×320	320×320	-	-
CoNSeP Segmentation	Micro-Net	Adam	0.5	8	1000×1000	252×252	128×128	1

Table 4.2: Chosen hyperparameter values for all datasets.

Dataset	Class id	K	M	Covariance type	Method
Nucleus Segmentation Dataset	1	93	2	Diagonal	Center
Gland Segmentation Dataset	1	68	1	Diagonal	Center
CoNSeP Segmentation Dataset	1	26	9	Diagonal	Angle
	2	92	4	Diagonal	Angle
	3	64	8	Diagonal	Angle
	4	2	1	Spherical	Center
Decathlon Task 5 Dataset	1	20	2	Diagonal	Angle
	2	1	4	Spherical	Angle

Table 4.3: Hyperparameter values used in post-processing for the proposed method as well as the comparison algorithms.

Dataset	Model	Preprocess	Class id	Method	E	D	A
Nucleus Segmentation	U-Net	Standardization	1	*Weighted Crossentropy*	7	5	250
				Fourier Loss ($d_{max} = 50$)	9	7	250
				Fourier Loss ($d_{max} = 100$)	7	7	250
	DCAN	Standardization	1	*Weighted Crossentropy*	7	9	500
				Fourier Loss	3	11	250
CoNSeP	Micro-Net	Normalization	1	*Weighted Crossentropy*	0	11	100
				Fourier Loss	0	11	100
			2	*Weighted Crossentropy*	0	5	100
				Fourier Loss	5	3	100
			3	*Weighted Crossentropy*	0	9	250
				Fourier Loss	7	5	250
			4	*Weighted Crossentropy*	0	9	100
				Fourier Loss	0	5	100
Gland Segmentation	U-Net	Normalization	1	*Weighted Crossentropy*	9	7	500
				Fourier Loss	9	7	500
Decathlon Task 5 Segmentation	U-Net	Standardization	1	*Weighted Crossentropy*	9	3	500
				Fourier Loss	0	3	500
			2	*Weighted Crossentropy*	0	3	500
				Fourier Loss	7	3	500

Chapter 5

Results

In this chapter, we give the quantitative and visual results of our method, which uses the proposed *Fourier loss*, and the network that is trained using the standard weighted cross-entropy loss. Ronnenberger *et al.* propose a weighting which uses distance to the nearest cell boundary in order to pay more attention to the boundary pixels [11]. This weighting gives the best results for instance segmentation datasets. So we have chosen this weighted loss for instance segmentation tasks as the baseline method and we refer to this loss when we say the weighted cross-entropy loss in instance segmentation tasks. Since the Decathlon Task 5 Segmentation dataset is a semantic segmentation task, we mean class imbalance weighting when we say the weighted cross-entropy loss in the Decathlon Task 5 Segmentation dataset.

We have used three state-of-the-art networks (U-Net, DCAN, Micro-Net) [11–13] as baseline models. All models are autoencoder-like FCNs which produce a prediction map for segmentation. Since each model has its own strength according to the dataset, we have chosen a different model for different datasets as the baseline models.

5.1 Baseline Models

5.1.1 U-Net

The U-Net model is an autoencoder-like architecture that uses skip connections to carry the spatial details to the decoder path because such details are lost in the pooling process [11]. U-Net is originally defined on the extracted patches and changes the input resolution at the output probability map with convolutional layers. In order to use the full resolution of the images, we have used padding to make the resolution of the output probability map equal to the resolution of the input image.

5.1.2 DCAN

DCAN is an autoencoder-like architecture proposed in [12]. The encoder and decoder paths are the same with those of the U-Net model. The only difference is that there are two identical decoder paths for multitask learning. One decoder path learns the boundary map as the other decoder path learns the segmentation map. During inference, two maps are fused together to create the final segmentation. The fusion procedure is done by removing the boundary prediction from the segmentation map in order to separate the instances. The DCAN model has skip connections to both decoder paths from the encoder path. Our loss is defined on mask prediction maps only. Since the second decoder of DCAN is a boundary map, we only apply our loss to the mask prediction branch of DCAN.

5.1.3 Micro-Net

Micro-Net is an autoencoder-like architecture proposed in [13]. This model resembles the architecture of U-Net but it takes four input images. It takes the original image in the first input layer. At each hidden layer of the encoder path,

it takes a reshaped image that matches the resolution of the layer. Images are reshaped using cubic interpolation. It also has auxiliary outputs at each layer of the decoder path. This network is defined with a specific input and output resolution of 252×252 pixels. Micro-Net has a larger number of learnable parameters compared to the U-Net and DCAN models and it can overfit easily. Since the foreground/background instance segmentation task is easier compared to multi-class segmentation problem, we do not use Micro-Net in foreground/background segmentation tasks. With the proper augmentation, Micro-Net achieves state-of-the-art results in multi-class segmentation tasks such as CoNSeP.

5.2 Nucleus Segmentation Dataset

In Nucleus Segmentation dataset, we have used two models (U-Net and DCAN) to compare our proposed loss function with the weighted cross-entropy loss. In Table 5.1, quantitative results of the baseline and our proposed loss can be seen. The U-Net architecture with 64 number of features and 0.2 dropout probability is used for both methods. The d_{max} value in our function determines how much we penalize shape. So it is a degree of shape regularization. Increasing the maximum value of d_{max} for *Fourier loss* from 50 to 100, we have observed better results due to more shape regularization. Our loss achieves better Hausdorff distance, Dice, IoU, and F-score results compared to the weighted cross-entropy loss.

Visual results obtained on the Huh7 and HepG2 test sets are given in Figure 5.1 and Figure 5.2, respectively. The use of the weighted cross-entropy (WCE) under-segments the cells that are close to each other. On the other hand, when the Fourier loss is used, instances that are close to each other are separated without under-segmentation.

In Table 5.2, we compare our method with the baseline when the DCAN architecture is used. From all given quantitative and visual results, it can be seen that our method outperforms the use of the standard weighted cross-entropy loss.

Table 5.1: Test results obtained on the Nucleus Segmentation dataset. Both methods are trained using the same parameters on the U-Net network. (a) Object-level precision, recall and F-score for the Huh7 and HepG2 test sets. (b) Object-level Dice index, Hausdorff distance, and IoU for the Huh7 and HepG2 test sets.

Data Type	Method	Precision	Recall	F-Score
Huh7	*Weighted Crossentropy*	98.45	92.93	95.61
	Fourier Loss ($d_{max} = 50$)	**98.61**	**95.29**	**96.92**
	Fourier Loss ($d_{max} = 100$)	98.04	**95.29**	96.64
HepG2	*Weighted Crossentropy*	92.16	85.89	88.91
	Fourier Loss ($d_{max} = 50$)	95.36	85.48	90.15
	Fourier Loss ($d_{max} = 100$)	**95.96**	**86.70**	**91.09**

(a)

Data Type	Method	Hausdorff	Dice	IoU
Huh7	*Weighted Crossentropy*	6.62	90.48	71.68
	Fourier Loss ($d_{max} = 50$)	4.76	92.26	77.01
	Fourier Loss ($d_{max} = 100$)	**4.57**	**92.76**	**78.52**
HepG2	*Weighted Crossentropy*	10.86	85.30	51.44
	Fourier Loss ($d_{max} = 50$)	10.99	85.69	54.28
	Fourier Loss ($d_{max} = 100$)	**9.50**	**87.35**	**58.49**

(b)

Table 5.2: Test results obtained on the Nucleus Segmentation dataset using the DCAN architecture. (a) Object-level precision, recall and F-score for the Huh7 and HepG2 test sets. (b) Object-level Dice index, Hausdorff distance, and IoU for the Huh7, and HepG2 test sets.

Data Type	Method	Precision	Recall	F-Score
Huh7	*Weighted Crossentropy*	**98.10**	87.09	92.27
	Fourier Loss	97.66	**93.49**	**95.53**
HepG2	*Weighted Crossentropy*	**93.47**	77.06	84.47
	Fourier Loss	90.87	**89.95**	**90.41**

(a)

Data Type	Method	Hausdorff	Dice	IoU
Huh7	*Weighted Crossentropy*	8.37	87.93	63.90
	Fourier Loss	**5.79**	**91.03**	**72.74**
HepG2	*Weighted Crossentropy*	12.43	82.11	44.96
	Fourier Loss	**9.10**	**86.87**	**56.03**

(b)

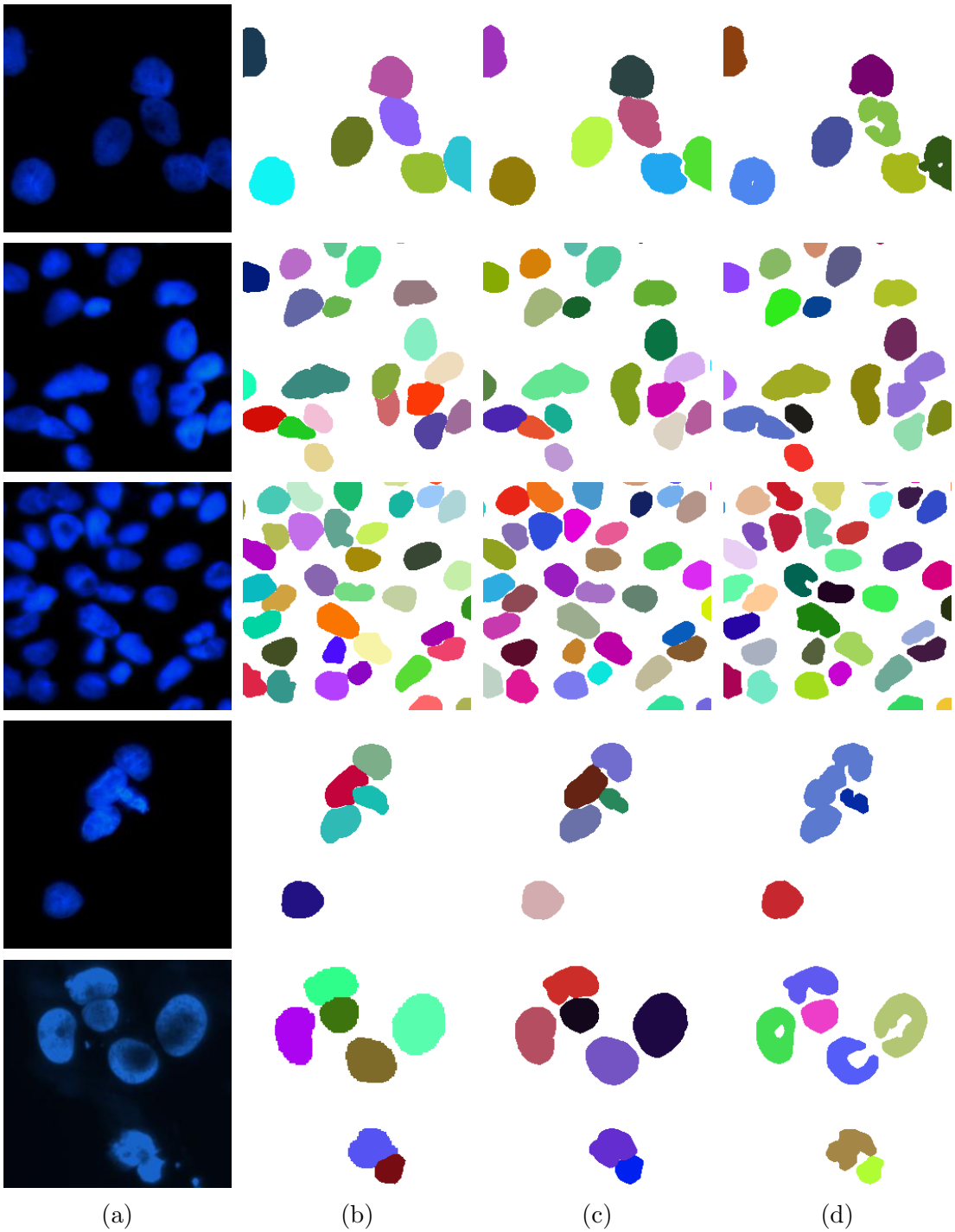
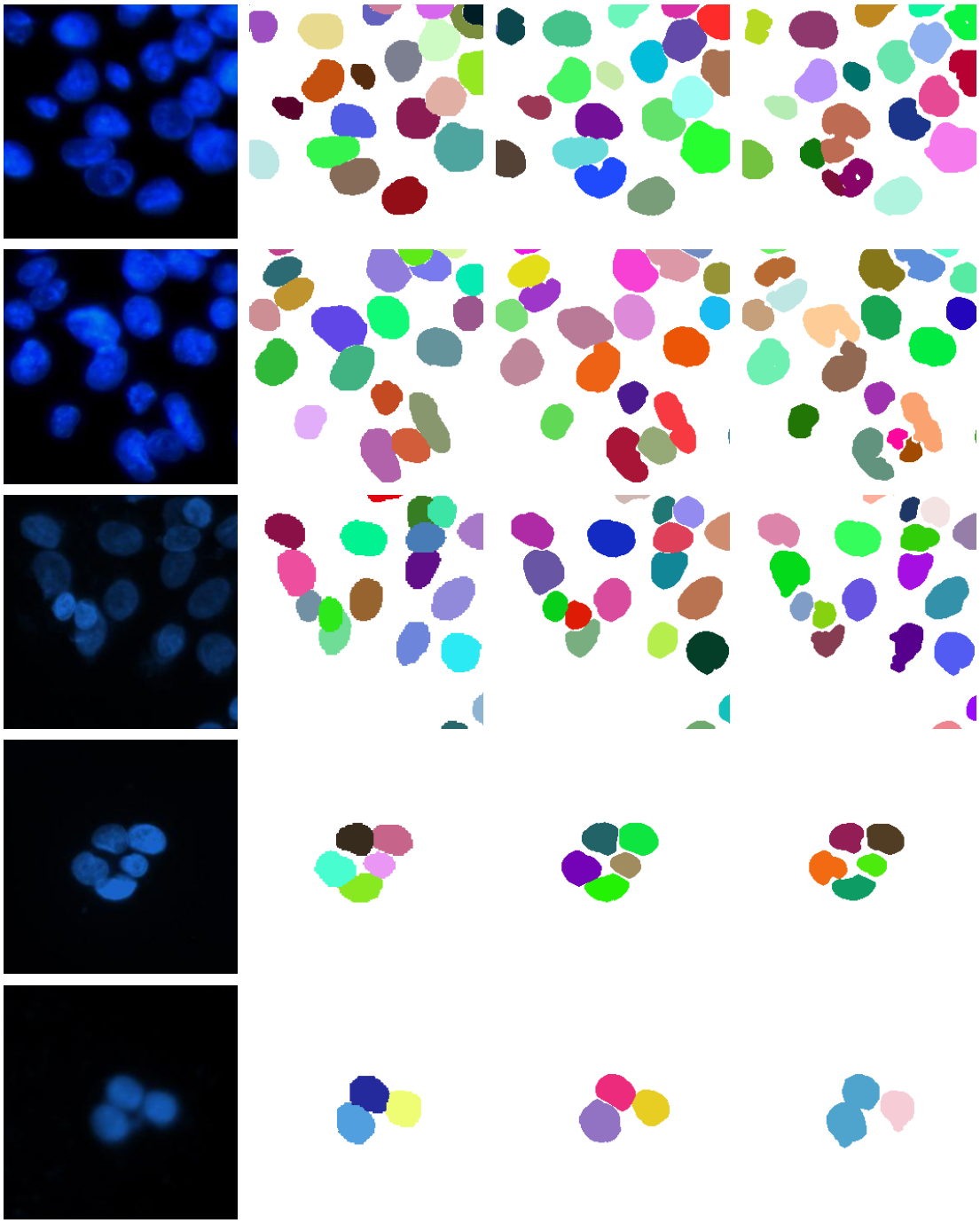


Figure 5.1: Visual results obtained on the Huh7 test set of Nucleus Segmentation dataset. (a) Input images. (b) Ground truths. (c) Results when the *Fourier loss* is used. (d) Results when the *WCE* loss is used.



(a) (b) (c) (d)

Figure 5.2: Visual results obtained on the HepG2 test set of Nucleus Segmentation dataset. (a) Input images. (b) Ground truths. (c) Results when the *Fourier loss* is used. (d) Results when the *WCE* loss is used.

In our experiments, we choose the class with the largest probability. In foreground-background segmentation tasks, this choice corresponds to choosing the probability threshold as 0.5. However, neural networks are not calibrated for that threshold due to many factors, e.g., regularization and batch normalization [96]. We have observed the probability calibration for both methods. For each threshold in $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$, we have postprocessed the probabilities. Here, the parameters E, D, and A are separately selected by a hyperparameter search. Then we have calculated test results for each threshold. The test results as a function of the selected parameter are given in Figure 5.3.

5.3 Gland Segmentation Dataset

Gland instances can have more complex shapes than cell instances. In order to test the prior shape learning abilities of our method, we have experimented on the Gland Segmentation dataset. The quantitative results of our method and the baseline, which uses the standard weighted cross-entropy, are given in Table 5.3. The U-Net architecture with 64 number of features and 0.2 dropout probability is used for both methods. Our method outperforms the baseline in F-score, Dice index, and IoU. The use of the weighted cross-entropy achieves a better Hausdorff distance with only 0.1 margin. The effectiveness of our method can be seen visually in Figure 5.4.

5.4 CoNSeP Dataset

For the CoNSeP Segmentation dataset, we have used the Micro-Net model to compare the use of our proposed loss function and the weighted cross-entropy loss. We have already seen that our loss function outperforms the use of the standard weighted cross-entropy loss in foreground/background instance segmentation tasks. Our loss function is designed to be compatible with multi-class

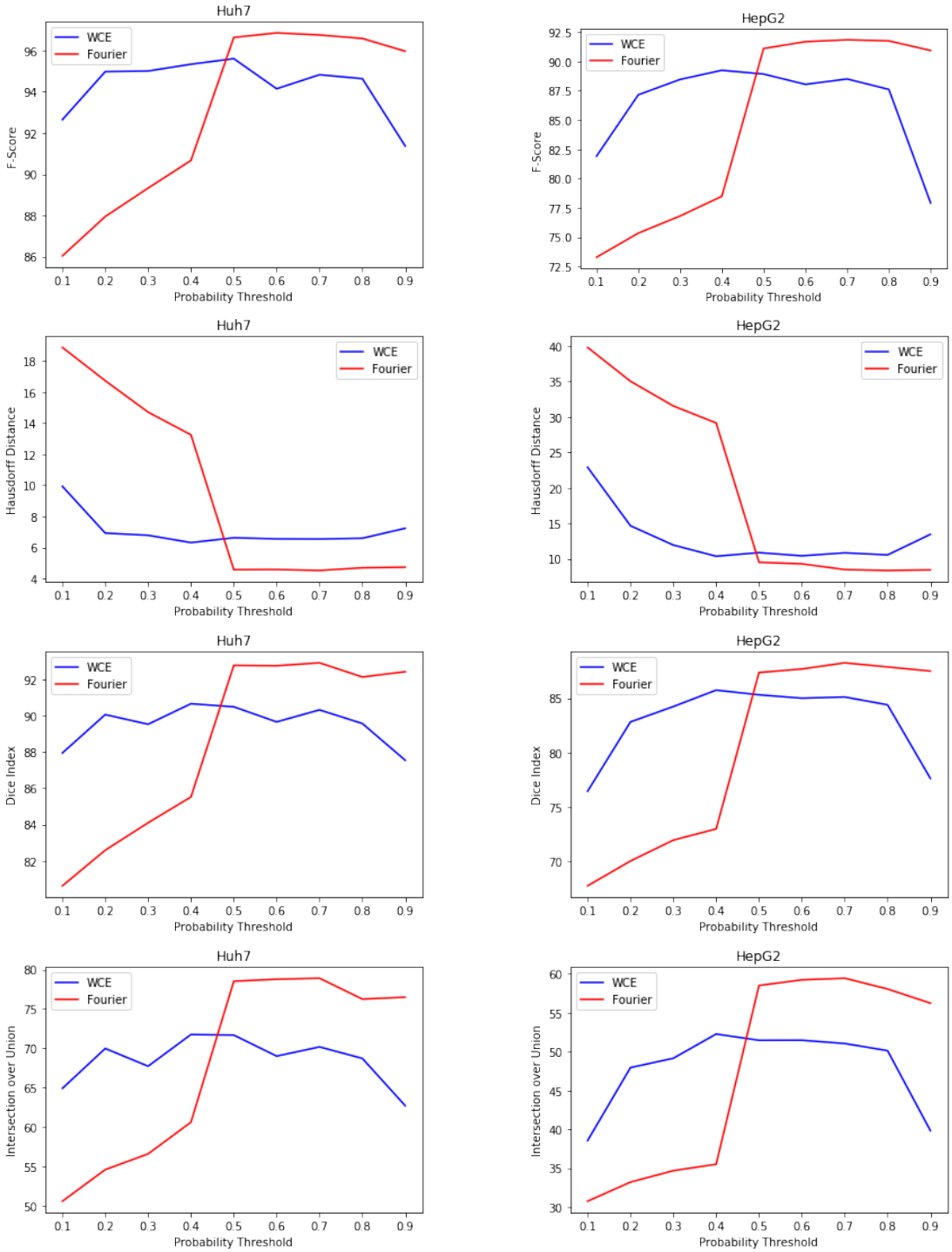


Figure 5.3: Test set results using different probability thresholds. These results are taken on the (a) Huh7 and (b) HepG2 test sets.

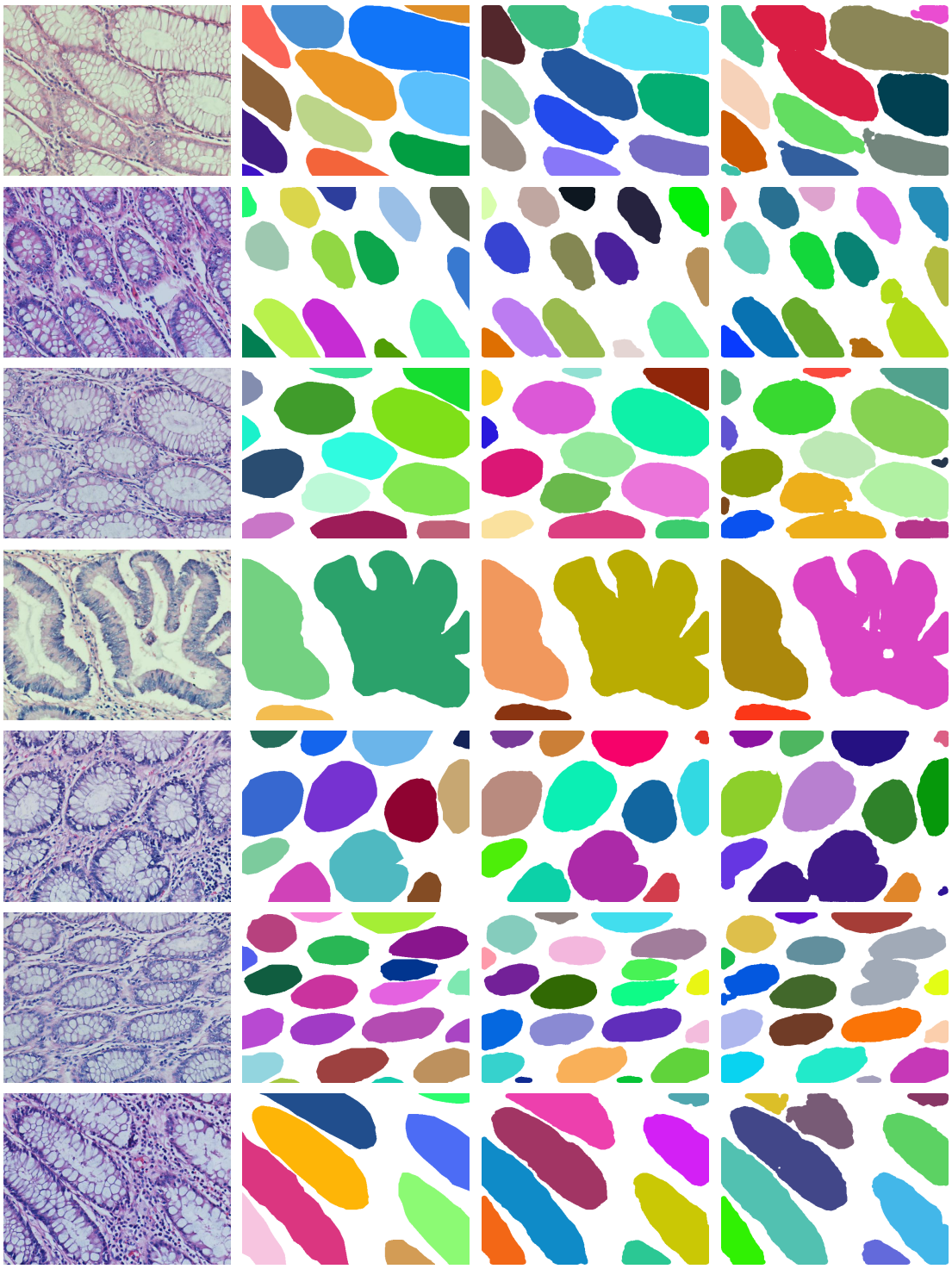


Figure 5.4: Visual results obtained on the test set of Gland Segmentation dataset. (a) Input images. (b) Ground truths. (c) Results when the *Fourier loss* is used. (d) Results when the *WCE* loss is used.

Table 5.3: Test results obtained on the Gland Segmentation dataset. Both methods are trained using the same parameters of the U-Net network. (a) Object-level precision, recall, and F-score for the test set. (b) Object-level Dice index, Hausdorff distance, and IoU for the test set.

Method	Precision	Recall	F-Score
Weighted Crossentropy	89.64	**89.46**	89.55
Fourier Loss	**93.39**	88.75	**91.01**

(a)

Method	Hausdorff	Dice	IoU
Weighted Crossentropy	**49.43**	90.91	66.27
Fourier Loss	49.53	**91.18**	**68.86**

(b)

segmentation tasks. Our loss learns a different prior for each class and normalizes the distances to have scaled weights for all classes. We have used the CoNSeP dataset to test our loss function in a multi-class segmentation task. The quantitative results are given in Tables 5.4 and 5.5. The use of the weighted cross-entropy loss achieves a better IoU in miscellaneous class with 0.03 margin and a better Hausdorff distance in inflammatory class with 0.99 margin. In all remaining metrics and classes, the use of our loss outperforms. Visual results are given in Figure 5.5. As seen in this figure, our method better predicts spindle-shaped structures in the segmentation for the classes with spindle-shaped objects since the use of the standard weighted cross-entropy cannot generate spindle-shaped predictions.

5.5 Decathlon Task 5 Dataset

Although we have designed our loss function for cell instance segmentation, we have extended our work to CT image segmentation, which is another common task in medical image segmentation. In the Decathlon Task 5 Segmentation dataset, we have used U-Net with 64 number of features and 0.2 dropout probability. Since this dataset is a semantic segmentation and there exists only one instance per class in each image, the weighted categorical cross-entropy does not have boundary attention. The quantitative results are given in Table 5.6. Our loss

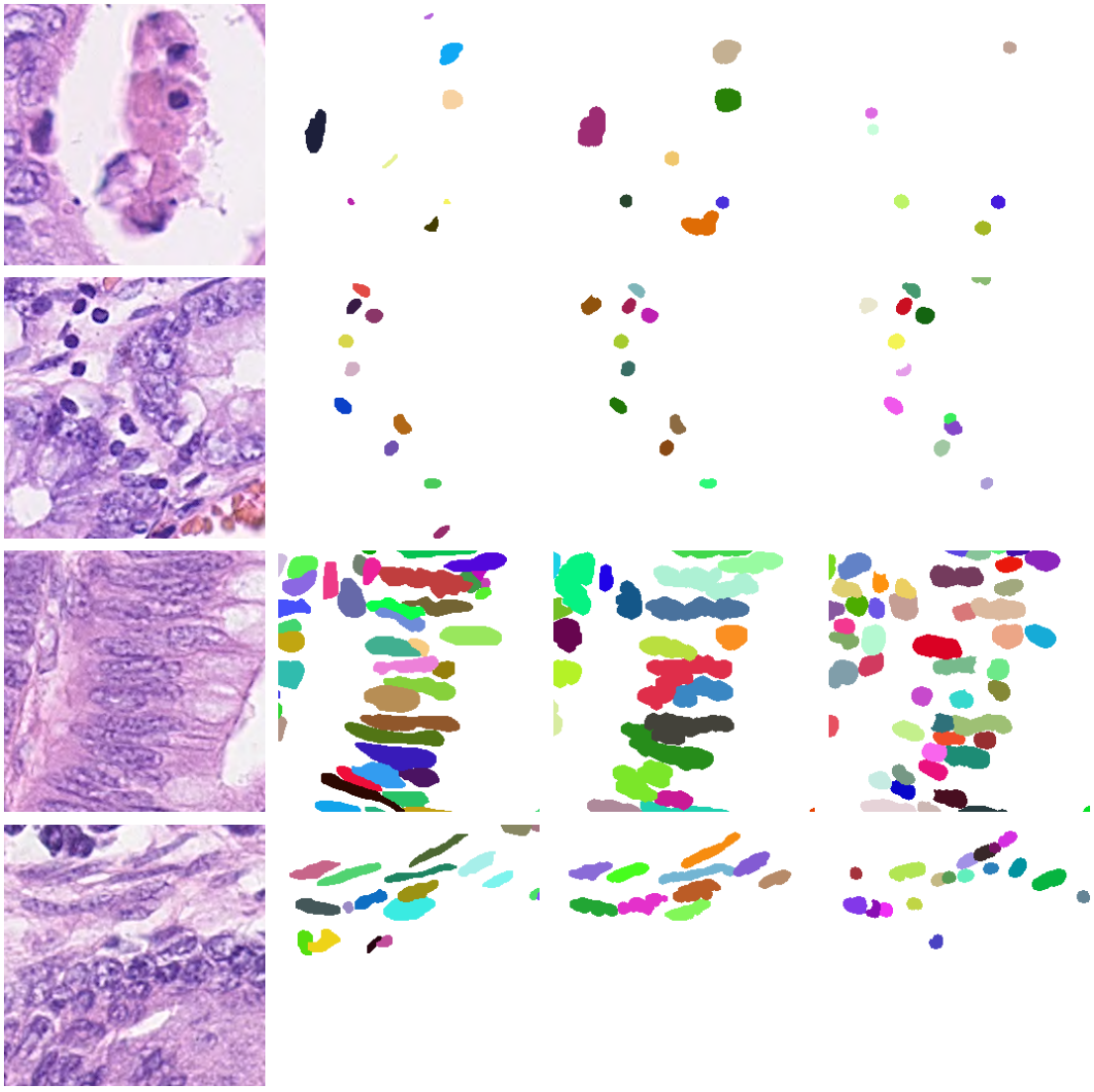
Table 5.4: Test results obtained on the CoNSeP Segmentation dataset. Both methods are trained using the same parameters on the Micro-Net model. (a) Object-level precision, recall, and F-score for the test set. (b) Object-level dice, Hausdorff distance, and IoU for the test set.

Class	Method	Precision	Recall	F-Score
1 (miscallaneous)	*Weighted Cross Entropy*	46.13	28.70	35.38
	Fourier Loss	**57.91**	**38.50**	**46.25**
2 (inflammatory)	*Weighted Cross Entropy*	76.74	**69.90**	73.16
	Fourier Loss	**82.46**	67.46	**74.21**
3 (epithelial)	*Weighted Cross Entropy*	60.44	**50.09**	54.78
	Fourier Loss	**84.05**	48.35	**61.39**
4 (spindle-shaped)	*Weighted Cross Entropy*	40.97	39.33	40.13
	Fourier Loss	**70.47**	**47.32**	**56.62**
All Classes	*Weighted Cross Entropy*	63.11	55.61	59.12
	Fourier Loss	**86.07**	**57.08**	**68.64**

(a)

Class	Method	Hausdorff	Dice	IoU
1 (miscallaneous)	*Weighted Cross Entropy*	61.35	26.99	**0.97**
	Fourier Loss	**52.08**	**34.82**	0.94
2 (inflammatory)	*Weighted Cross Entropy*	**15.48**	69.97	33.03
	Fourier Loss	16.47	**70.78**	**38.34**
3 (epithelial)	*Weighted Cross Entropy*	19.15	58.81	11.76
	Fourier Loss	**18.62**	**64.59**	**17.85**
4 (spindle-shaped)	*Weighted Cross Entropy*	26.93	44.39	4.06
	Fourier Loss	**20.51**	**57.80**	**13.78**
All Classes	*Weighted Cross Entropy*	14.67	60.65	12.78
	Fourier Loss	**14.49**	**66.87**	**20.89**

(b)



(a) (b) (c) (d)

Figure 5.5: Visual results obtained on the test set of the CoNSeP Segmentation dataset. One sample is given for each class. From top to bottom, classes are miscallaneous, inflammatory, epithelial, and spindle-shaped. The shape preserving effect can be observed in tubular objects for epithelial and spindle-shaped classes. (a) Input images. (b) Ground truths. (c) Results when the *Fourier loss* is used. (d) Results when the *WCE* loss is used.

function outperforms the weighted cross-entropy loss in F-score, Dice index, and IoU. The weighted cross-entropy loss outperforms our loss in Hausdorff distance of peripheral zone by only 0.15 margin and Hausdorff distance of transitional zone by 2.42 margin. Visual results are given in Figure 5.6.

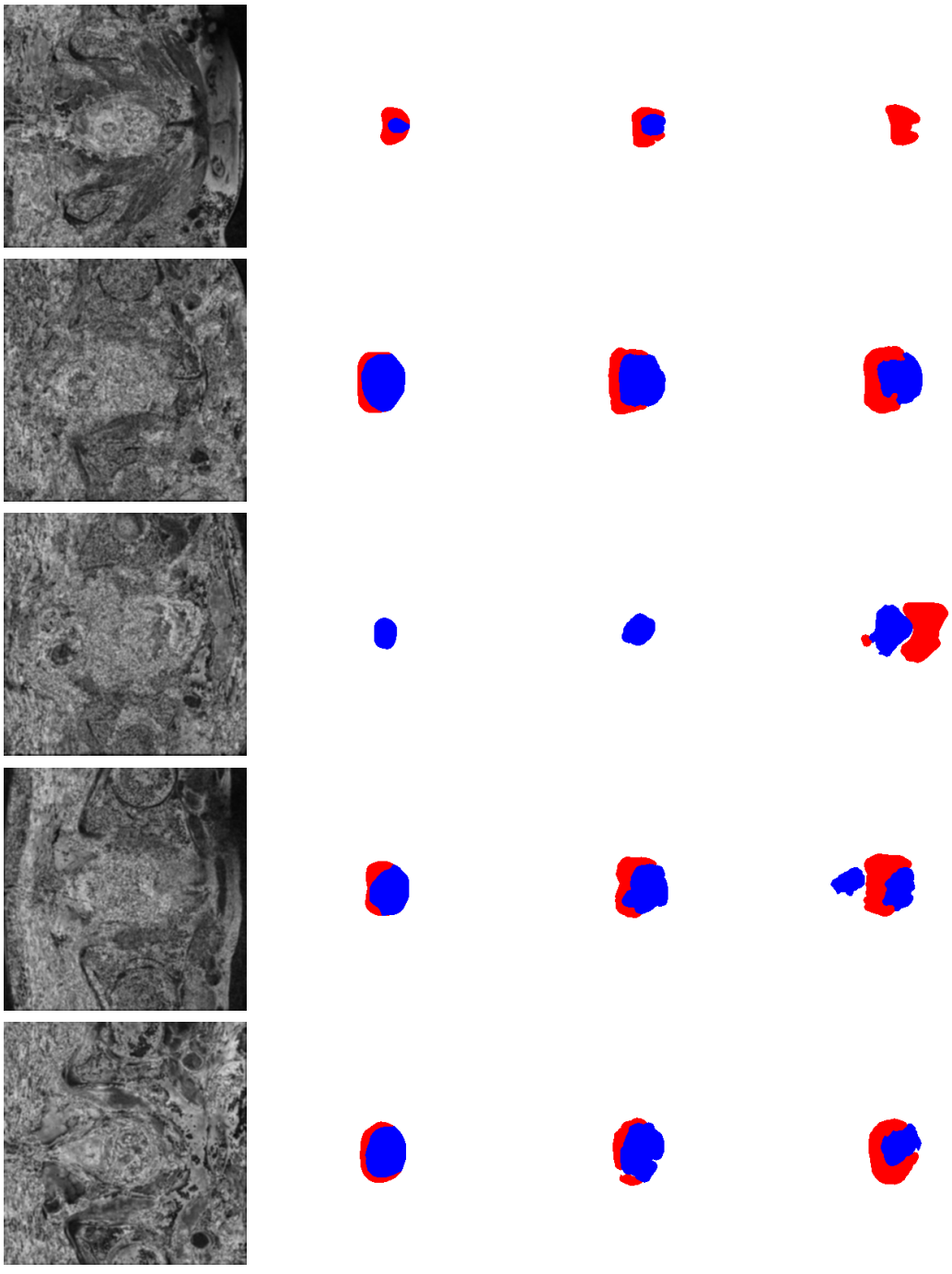
5.6 Discussion

In this thesis, we have proposed a shape-preserving loss weighting which pays more attention to deformed instances. There are four attributes of our method:

- It is an instance-based method that considers each instance in the segmentation map in order to improve object-level scores.
- It uses prior-shape information that increases the generalization abilities of the network. Using prior shape also makes our loss calculations faster than matching ground truth objects with segmented instances.
- It learns the prior-shape information so that it can be used for different datasets.
- Some shape-preserving losses are only defined for foreground/background segmentation. Our method is also defined for multi-class segmentation problems.

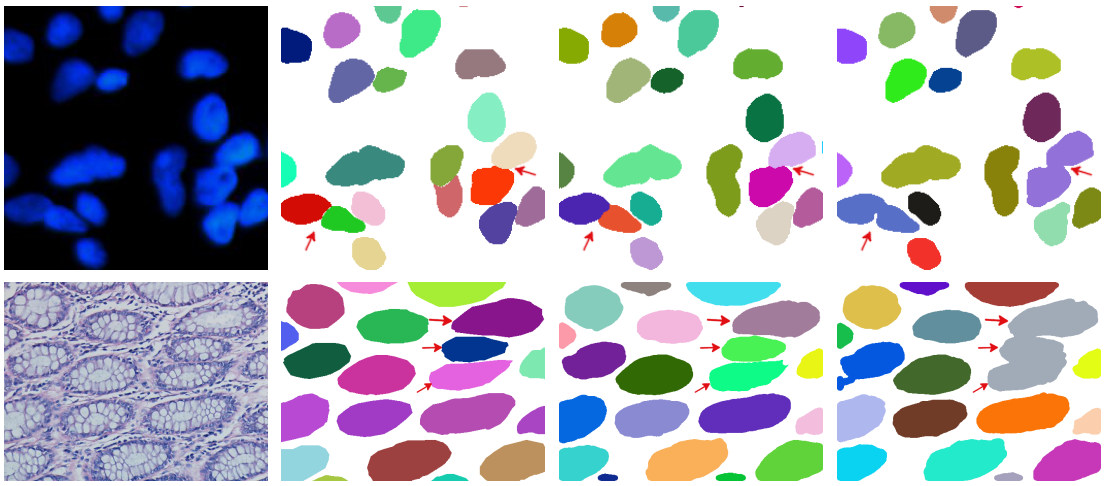
With the help of these attributes, our loss outperforms the standard weighted cross-entropy loss in different datasets and different networks. From the visual results, we have observed four different effects of our loss function: better separation, filling, shape fixing, and cavity fixing. In this section, we will explain and visually show these four effects.

Although the weighted cross-entropy that uses distances to nearest cell as



(a) (b) (c) (d)

Figure 5.6: Visual results obtained on the test set of the Decathlon Task 5 Segmentation dataset. Blue regions are transitional zone and red regions are peripheral zone. Since input image was too noisy, we have smoothed it with Gaussian filter for demonstration purposes only. (a) Input images. (b) Ground truths. (c) Results when the *Fourier loss* is used. (d) Results when the *WCE* loss is used.

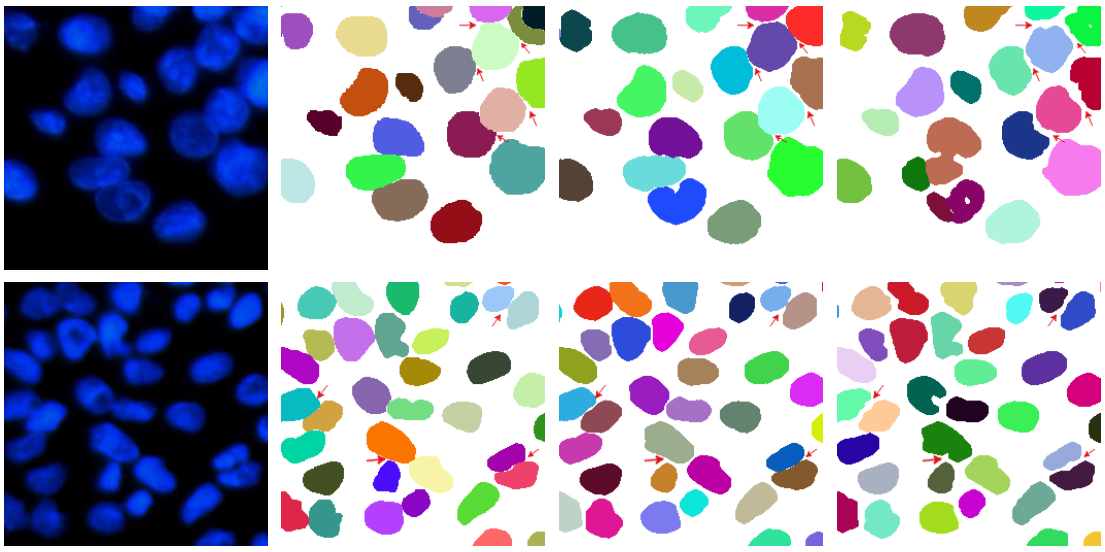


(a) (b) (c) (d)

Figure 5.7: Separation effect of using the *Fourier loss*. (a) Input images. (b) Ground truths. (c) Results when the *Fourier loss* is used. (d) Results when the *WCE* is used.

weights can separate overlapped instances better than other losses, our loss outperforms this weighted cross-entropy in separating instances. The weighted cross-entropy pays the same attention to the pixels between separated instances and the pixels between unseparated instances. Our loss pays attention to instances only if they are not separated. Note that our loss function does not consider whether two instances are separated or not by design. It is only designed to pay more attention to the deformed instances. Since non-separated instances will have a high dissimilarity score, they will naturally get more attention in our loss. This dynamic attention behavior achieves better separation than the weighted cross-entropy loss which is specifically designed to separate instances. In Figure 5.7, red arrows show the regions where our loss function is able to separate the instances while cross-entropy loss could not.

When the weighted cross-entropy loss is used, instances are separated too much because of the static attention mechanism. This causes cavities on the instances that are close to each other. Red arrows in Figure 5.8 point to such cavities. For comparison, the same arrows are also placed in the images of ground truth and those generated by our method. Our loss separates instances without making such cavities, and as a result, it increases the object-level Hausdorff distance,

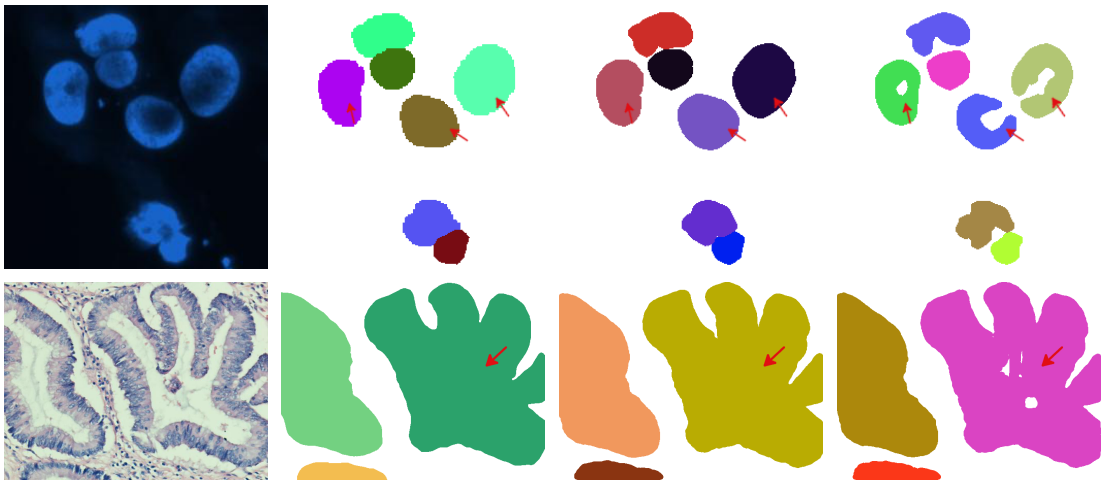


(a) (b) (c) (d)

Figure 5.8: Cavity fixing effect of using the *Fourier loss*. (a) Input images. (b) Ground truths. (c) Results when the *Fourier loss* is used. (d) Results when the *WCE* is used.

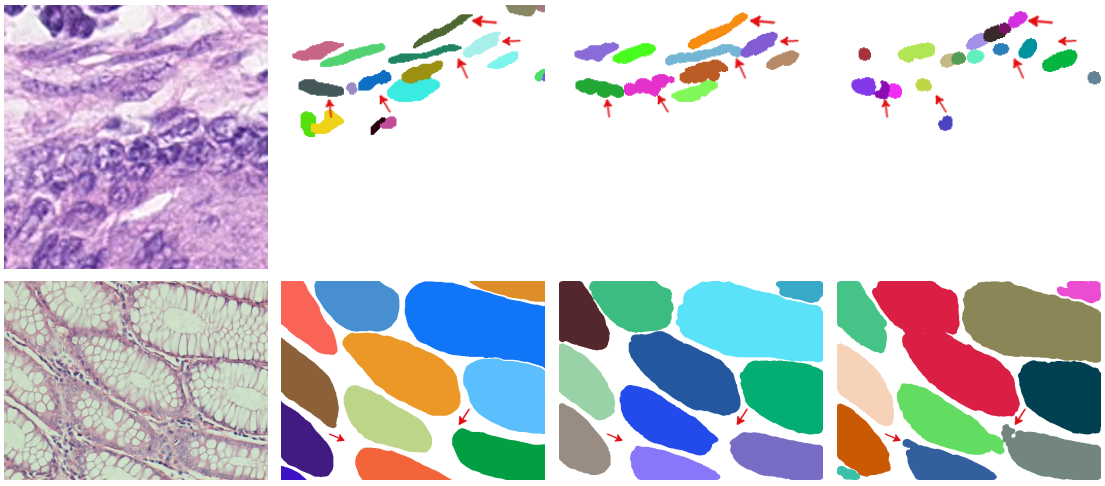
object-level Dice index, and IoU.

If there are multiple boundaries for an instance, the dissimilarity score is calculated for all boundaries and the maximum dissimilarity score is chosen. Even though an outer boundary of an instance has a low dissimilarity score, it will have a high dissimilarity score if it has a hole, which creates another boundary for the same instance. Hence, our method also fixes holes inside objects. In the first row of Figure 5.9, it can be seen that such holes are caused by the intensity values which are similar to the background values. If we are given only a region of the black area inside a cell, we cannot conclude that it belongs to a cell and we might annotate that region as background. If we are given the entire cell, we can recognize that the black area is inside a cell so the region must be foreground. As humans, we use spatial and shape information while making such deductions during annotation. Convolutional neural networks are capable of learning this spatial information but they cannot learn enough when they are trained with pixel-wise losses as can be seen in Figure 5.9. When we use our loss in exactly the same network, the network can better learn how to use its spatial recognition abilities.



(a) (b) (c) (d)

Figure 5.9: Filling effect of using the *Fourier loss.*(a) Input images. (b) Ground truths. (c) Results when the *Fourier loss* is used. (d) Results when the *WCE* is used.



(a) (b) (c) (d)

Figure 5.10: Shape fixing effect of using the *Fourier loss.* (a) Input images. (b) Ground truths. (c) Results when the *Fourier loss* is used. (d) Results when the *WCE* is used.

In the first row of Figure 5.10, we see that ground truth cells have a spindle shape. The use of the weighted cross-entropy loss is able to locate the exact places but it divides long shapes into small circular shapes. On the other hand, our method is able to predict contiguous spindle-shaped cells. Hence, our method performs better in object-level metrics. In the second row of Figure 5.10, it can be seen that the use of the weighted cross-entropy leads to deformed instances. Our method is able to fix such deformations in the segmentation map. From the results of our experiments, we have observed that the performance of convolutional neural networks can be improved with shape attention. Both quantitative and visual results show that our loss outperforms weighted cross-entropy loss in different models and datasets.

Table 5.5: Test results obtained on the CoNSeP Segmentation dataset. Both methods are trained using the same parameters of the Micro-Net model. F-classification-detection-scores are given.

Class	Method	F_c
1 (miscallaneous)	*Weighted Crossentropy*	0.157
	Fourier Loss	**0.301**
2 (inflammatory)	*Weighted Crossentropy*	0.618
	Fourier Loss	**0.648**
3 (epithelial)	*Weighted Crossentropy*	0.538
	Fourier Loss	**0.589**
4 (spindle-shaped)	*Weighted Crossentropy*	0.451
	Fourier Loss	**0.526**
All Classes	*Weighted Crossentropy*	0.618
	Fourier Loss	**0.703**

Table 5.6: Test results obtained on Decathlon Task 5 dataset. Both methods are trained using the same parameters of the U-Net network. (a) Object-level precision, recall and f-score for the test set. (b) Object-level Dice index, Hausdorff distance, and IoU for the test set.

Data type	Method	Precision	Recall	F-Score
Peripheral Zone	*Weighted Crossentropy*	42.70	**41.15**	41.91
	Fourier Loss	**66.67**	35.42	**46.26**
Transitional Zone	*Weighted Crossentropy*	73.02	70.23	71.60
	Fourier Loss	**77.10**	**77.10**	**77.10**

(a)

Data type	Method	Hausdorff	Dice	IoU
Peripheral Zone	*Weighted Crossentropy*	**21.43**	40.81	1.34
	Fourier Loss	21.58	**53.13**	**3.41**
Transitional Zone	*Weighted Crossentropy*	**17.82**	74.55	19.77
	Fourier Loss	20.24	**77.93**	**25.76**

(b)

Chapter 6

Conclusion

In this thesis, we propose a novel shape-preserving loss function to be used in deep learning for medical image segmentation. We generate vectors from segmented instances. These vectors are called Fourier descriptors, which are used in shape characterization. We use a Gaussian mixture model to estimate the shape distribution of the instances. Then we use Mahalanobis distance on this distribution as a dissimilarity metric for instances. We pay more attention to certain instances using this dissimilarity metric during training. We have experimented on four different datasets and compared our method to widely used weighted cross-entropy loss function in instance segmentation, which gives static attention for instance separation. We have also tested our method in different models.

The visual results show that our method improves the segmentation in four ways. The segmented instances are more close to the ground truth in terms of shape using our method. Our method fills holes inside objects. Our method does not create cavities between close instances like the use of the weighted cross-entropy loss does. Our method can separate instances better than the use of the weighted cross-entropy loss.

Although our loss improves the instance segmentation quality, it can be improved by defining a differentiable loss function using a similar idea. One could

use other shape similarity metrics to change the weighting defined in our method. Other shape discriminators like the curvature scale space [97] can also be used to see whether they outperform Fourier descriptors or not. In the literature, topology and shape are studied separately. A loss function that uses topology and shape at the same time can be developed by extending this thesis.

Bibliography

[1] Z. Mirikharaji and G. Hamarneh, "Star shape prior in fully convolutional networks for skin lesion segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 737–745, Springer, 2018.

[2] C. Gunduz-Demir, M. Kandemir, A. B. Tosun, and C. Sokmensuer, "Automatic segmentation of colon glands using object-graphs," *Medical Image Analysis*, vol. 14, no. 1, pp. 1–12, 2010.

[3] C. Jung and C. Kim, "Segmenting clustered nuclei using h-minima transform-based marker extraction and contour parameterization," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 10, pp. 2600–2604, 2010.

[4] Y. Gao, L. Wang, Y. Shao, and D. Shen, "Learning distance transform for boundary detection and deformable segmentation in ct prostate images," in *International Workshop on Machine Learning in Medical Imaging*, pp. 93–100, Springer, 2014.

[5] K. Sirinukunwattana, D. R. Snead, and N. M. Rajpoot, "A stochastic polygons model for glandular structures in colon histology images," *IEEE Transactions on Medical Imaging*, vol. 34, no. 11, pp. 2366–2378, 2015.

[6] F. Xing, H. Su, J. Neltner, and L. Yang, "Automatic ki-67 counting using robust cell detection and online dictionary learning," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 3, pp. 859–870, 2013.

[7] C. F. Koyuncu, R. Cetin-Atalay, and C. Gunduz-Demir, "Object-oriented segmentation of cell nuclei in fluorescence microscopy images," *Cytometry Part A*, vol. 93, no. 10, pp. 1019–1028, 2018.

[8] Y. Song, E.-L. Tan, X. Jiang, J.-Z. Cheng, D. Ni, S. Chen, B. Lei, and T. Wang, "Accurate cervical cell segmentation from overlapping clumps in pap smear images," *IEEE Transactions on Medical Imaging*, vol. 36, no. 1, pp. 288–300, 2016.

[9] J. Xu, L. Xiang, Q. Liu, H. Gilmore, J. Wu, J. Tang, and A. Madabhushi, "Stacked sparse autoencoder (ssae) for nuclei detection on breast cancer histopathology images," *IEEE Transactions on Medical Imaging*, vol. 35, no. 1, pp. 119–130, 2015.

[10] Y. Xie, X. Kong, F. Xing, F. Liu, H. Su, and L. Yang, "Deep voting: A robust approach toward nucleus localization in microscopy images," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 374–382, Springer, 2015.

[11] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Springer, 2015.

[12] H. Chen, X. Qi, L. Yu, and P.-A. Heng, "Dcan: deep contour-aware networks for accurate gland segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2487–2496, 2016.

[13] S. E. A. Raza, L. Cheung, M. Shaban, S. Graham, D. Epstein, S. Pelengaris, M. Khan, and N. M. Rajpoot, "Micro-net: A unified model for segmentation of various objects in microscopy images," *Medical Image Analysis*, vol. 52, pp. 160–173, 2019.

[14] C. F. Koyuncu, G. N. Gunesli, R. C. Atalay, and C. Gunduz-Demir, "Deepdistance: A multi-task deep regression model for cell detection in inverted microscopy images," *Medical Image Analysis*, p. 101720, 2020.

[15] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2980–2988, 2017.

[16] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. J. Cardoso, "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 240–248, Springer, 2017.

[17] S. Graham, Q. D. Vu, S. E. A. Raza, A. Azam, Y. W. Tsang, J. T. Kwak, and N. Rajpoot, "Hover-net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images," *Medical Image Analysis*, vol. 58, p. 101563, 2019.

[18] A. Janowczyk and A. Madabhushi, "Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases," *Journal of Pathology Informatics*, vol. 7, 2016.

[19] H. Sharma, N. Zerbe, I. Klempert, O. Hellwich, and P. Hufnagl, "Deep convolutional neural networks for automatic classification of gastric carcinoma using whole slide images in digital histopathology," *Computerized Medical Imaging and Graphics*, vol. 61, pp. 2–13, 2017.

[20] Y. Zheng, Z. Jiang, F. Xie, H. Zhang, Y. Ma, H. Shi, and Y. Zhao, "Feature extraction from histopathological images based on nucleus-guided convolutional neural network for breast lesion classification," *Pattern Recognition*, vol. 71, pp. 14–25, 2017.

[21] Z. Al-Milaji, I. Ersoy, A. Hafiane, K. Palaniappan, and F. Bunyak, "Integrating segmentation with deep learning for enhanced classification of epithelial and stromal tissues in h&e images," *Pattern Recognition Letters*, vol. 119, pp. 214–221, 2019.

[22] Y. Al-Kofahi, W. Lassoued, W. Lee, and B. Roysam, "Improved automatic detection and segmentation of cell nuclei in histopathology images," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 4, pp. 841–852, 2009.

[23] H. Xu, C. Lu, R. Berendt, N. Jha, and M. Mandal, "Automatic nuclei detection based on generalized laplacian of gaussian filters," *IEEE Journal of Biomedical and Health Informatics*, vol. 21, no. 3, pp. 826–837, 2016.

[24] M. Saha, C. Chakraborty, and D. Racoceanu, "Efficient deep learning model for mitosis detection using breast histopathology images," *Computerized Medical Imaging and Graphics*, vol. 64, pp. 29–40, 2018.

[25] X. Pan, L. Li, H. Yang, Z. Liu, J. Yang, L. Zhao, and Y. Fan, "Accurate segmentation of nuclei in pathological images via sparse reconstruction and deep convolutional networks," *Neurocomputing*, vol. 229, pp. 88–99, 2017.

[26] W. Zhang and H. Li, "Automated segmentation of overlapped nuclei using concave point detection and segment grouping," *Pattern Recognition*, vol. 71, pp. 349–360, 2017.

[27] M. A. A. Dewan, M. O. Ahmad, and M. Swamy, "A method for automatic segmentation of nuclei in phase-contrast images based on intensity, convexity and texture," *Ieee Transactions On Biomedical Circuits And Systems*, vol. 8, no. 5, pp. 716–728, 2014.

[28] M. E. Plissiti and C. Nikou, "Overlapping cell nuclei segmentation using a spatially adaptive active physical model," *IEEE Transactions on Image Processing*, vol. 21, no. 11, pp. 4568–4580, 2012.

[29] C. Jung, C. Kim, S. W. Chae, and S. Oh, "Unsupervised segmentation of overlapped nuclei using bayesian classification," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 12, pp. 2825–2832, 2010.

[30] S. Arslan, T. Ersahin, R. Cetin-Atalay, and C. Gunduz-Demir, "Attributed relational graphs for cell nucleus segmentation in fluorescence microscopy images," *IEEE Transactions on Medical Imaging*, vol. 32, no. 6, pp. 1121–1131, 2013.

[31] F. Xing, Y. Xie, and L. Yang, "An automatic learning-based framework for robust nucleus segmentation," *IEEE Transactions on Medical Imaging*, vol. 35, no. 2, pp. 550–566, 2015.

[32] M. Saha and C. Chakraborty, "Her2net: A deep framework for semantic segmentation and classification of cell membranes and nuclei in breast cancer evaluation," *IEEE Transactions on Image Processing*, vol. 27, no. 5, pp. 2189–2200, 2018.

[33] Y. Song, L. Zhang, S. Chen, D. Ni, B. Lei, and T. Wang, "Accurate segmentation of cervical cytoplasm and nuclei based on multiscale convolutional network and graph partitioning," *IEEE Transactions on Biomedical Engineering*, vol. 62, no. 10, pp. 2421–2433, 2015.

[34] Y. Xu, Y. Li, Y. Wang, M. Liu, Y. Fan, M. Lai, I. Eric, and C. Chang, "Gland instance segmentation using deep multichannel neural networks," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 12, pp. 2901–2912, 2017.

[35] S. Manivannan, W. Li, J. Zhang, E. Trucco, and S. J. McKenna, "Structure prediction for gland segmentation with hand-crafted and deep convolutional features," *IEEE Transactions on Medical Imaging*, vol. 37, no. 1, pp. 210–221, 2017.

[36] G. Li, S. E. A. Raza, and N. M. Rajpoot, "Multi-resolution cell orientation congruence descriptors for epithelium segmentation in endometrial histology images," *Medical Image Analysis*, vol. 37, pp. 91–100, 2017.

[37] J. Xu, X. Luo, G. Wang, H. Gilmore, and A. Madabhushi, "A deep convolutional neural network for segmenting and classifying epithelial and stromal regions in histopathological images," *Neurocomputing*, vol. 191, pp. 214–223, 2016.

[38] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[39] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

[40] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[41] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[42] Q. V. Le, "Building high-level features using large scale unsupervised learning," in *2013 IEEE international Conference on Acoustics, Speech and Signal Processing*, pp. 8595–8598, IEEE, 2013.

[43] Y. Wang, H. Yao, and S. Zhao, "Auto-encoder based dimensionality reduction," *Neurocomputing*, vol. 184, pp. 232–242, 2016.

[44] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

[45] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," *arXiv preprint arXiv:1502.04623*, 2015.

[46] L. Itti and C. Koch, "Computational modelling of visual attention," *Nature Reviews Neuroscience*, vol. 2, no. 3, pp. 194–203, 2001.

[47] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of Physiology*, vol. 160, no. 1, p. 106, 1962.

[48] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

[49] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI Conference on Artificial Intelligence*, 2017.

[50] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European Conference on Computer Vision*, pp. 630–645, Springer, 2016.

[51] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[52] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

[53] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[54] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[55] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

[56] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[57] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2961–2969, 2017.

[58] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[59] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[60] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 648–656, 2015.

[61] T. Werner, "A linear programming approach to max-sum problem: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 7, pp. 1165–1179, 2007.

[62] T. F. Chan, S. Esedoglu, and M. Nikolova, "Algorithms for finding global minimizers of image segmentation and denoising models," *SIAM Journal on Applied Mathematics*, vol. 66, no. 5, pp. 1632–1648, 2006.

[63] M. Klodt, F. Steinbrücker, and D. Cremers, "Moment constraints in convex optimization for segmentation and tracking," in *Advanced Topics in Computer Vision*, pp. 215–242, Springer, 2013.

[64] L. Kostrykin, C. Schnörr, and K. Rohr, "Globally optimal segmentation of cell nuclei in fluorescence microscopy images using shape and intensity information," *Medical Image Analysis*, vol. 58, p. 101536, 2019.

[65] C. Ventura, J. Pont-Tuset, S. Caelles, K.-K. Maninis, and L. Van Gool, "Iterative deep learning for network topology extraction," *arXiv preprint arXiv:1712.01217*, 2017.

[66] M. Tofighi, T. Guo, J. K. Vanamala, and V. Monga, "Prior information guided regularized deep learning for cell nucleus detection," *IEEE Transactions on Medical Imaging*, vol. 38, no. 9, pp. 2047–2058, 2019.

[67] C. Zotti, Z. Luo, A. Lalande, and P.-M. Jodoin, "Convolutional neural network with shape prior applied to cardiac mri segmentation," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 3, pp. 1119–1128, 2018.

[68] C. Zotti, Z. Luo, O. Humbert, A. Lalande, and P.-M. Jodoin, "Gridnet with automatic shape prior registration for automatic mri cardiac segmentation,"

in *International Workshop on Statistical Atlases and Computational Models of the Heart*, pp. 73–81, Springer, 2017.

[69] X. Hu, F. Li, D. Samaras, and C. Chen, "Topology-preserving deep image segmentation," in *Advances in Neural Information Processing Systems*, pp. 5658–5669, 2019.

[70] S. M. R. Al Arif, K. Knapp, and G. Slabaugh, "Shape-aware deep convolutional neural network for vertebrae segmentation," in *International Workshop and Challenge on Computational Methods and Clinical Applications in Musculoskeletal Imaging*, pp. 12–24, Springer, 2017.

[71] B. Murugesan, K. Sarveswaran, S. M. Shankaranarayana, K. Ram, J. Joseph, and M. Sivaprakasam, "Psi-net: Shape and boundary aware joint multi-task deep network for medical image segmentation," in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 7223–7226, IEEE, 2019.

[72] D. Karimi and S. E. Salcudean, "Reducing the hausdorff distance in medical image segmentation with convolutional neural networks," *IEEE Transactions on Medical Imaging*, vol. 39, no. 2, pp. 499–513, 2019.

[73] J. R. Clough, I. Oksuz, N. Byrne, V. A. Zimmer, J. A. Schnabel, and A. P. King, "A topological loss function for deep-learning based image segmentation using persistent homology," *arXiv preprint arXiv:1910.01877*, 2019.

[74] Z. Yan, X. Yang, and K.-T. T. Cheng, "A deep model with shape-preserving loss for gland instance segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 138–146, Springer, 2018.

[75] X. Chen, B. M. Williams, S. R. Vallabhaneni, G. Czanner, R. Williams, and Y. Zheng, "Learning active contour models for medical image segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11632–11640, 2019.

[76] A. Mosinska, P. Marquez-Neila, M. Koziński, and P. Fua, "Beyond the pixel-wise loss for topology-aware delineation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3136–3145, 2018.

[77] G. G. Pihlgren, F. Sandin, and M. Liwicki, "Improving image autoencoder embeddings with perceptual loss," *arXiv preprint arXiv:2001.03444*, 2020.

[78] H. He, D. Yang, S. Wang, S. Wang, and Y. Li, "Road extraction by using atrous spatial pyramid pooling integrated encoder-decoder network and structural similarity loss," *Remote Sensing*, vol. 11, no. 9, p. 1015, 2019.

[79] R. Li, M. Li, and J. Li, "Connection sensitive attention u-net for accurate retinal vessel segmentation," *arXiv preprint arXiv:1903.05558*, 2019.

[80] H. Ravishankar, R. Venkataramani, S. Thiruvenkadam, P. Sudhakar, and V. Vaidya, "Learning and incorporating shape models for semantic segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 203–211, Springer, 2017.

[81] G. Qu, W. Zhang, Z. Wang, X. Dai, J. Shi, J. He, F. Li, X. Zhang, and Y. Qiao, "Stripnet: Towards topology consistent strip structure segmentation," in *Proceedings of the 26th ACM International Conference on Multimedia*, pp. 283–291, 2018.

[82] O. Oktay, E. Ferrante, K. Kamnitsas, M. Heinrich, W. Bai, J. Caballero, S. A. Cook, A. De Marvao, T. Dawes, D. P. O'Regan, *et al.*, "Anatomically constrained neural networks (acnns): application to cardiac image enhancement and segmentation," *IEEE Transactions on Medical Imaging*, vol. 37, no. 2, pp. 384–395, 2017.

[83] D. Freedman and C. Chen, "Algebraic topology for computer vision," *Computer Vision*, pp. 239–268, 2009.

[84] Z. Yan, X. Yang, and K.-T. Cheng, "A skeletal similarity metric for quality evaluation of retinal vessel segmentation," *IEEE Transactions on Medical Imaging*, vol. 37, no. 4, pp. 1045–1057, 2017.

[85] C. T. Zahn and R. Z. Roskies, "Fourier descriptors for plane closed curves," *IEEE Transactions on Computers*, vol. 100, no. 3, pp. 269–281, 1972.

[86] D. Zhang and G. Lu, "A comparative study of curvature scale space and fourier descriptors for shape-based image retrieval," *Journal of Visual Communication and Image Representation*, vol. 14, no. 1, pp. 39–57, 2003.

[87] F. Larsson and M. Felsberg, "Using fourier descriptors and spatial models for traffic sign recognition," in *Scandinavian Conference on Image Analysis*, pp. 238–249, Springer, 2011.

[88] D. A. Reynolds, "Gaussian mixture models.," *Encyclopedia of Biometrics*, vol. 741, 2009.

[89] G. N. Gunesli, C. Sokmensuer, and C. Gunduz-Demir, "Attentionboost: Learning what to attend by boosting fully convolutional networks," *arXiv preprint arXiv:1908.02095*, 2019.

[90] F. Chollet *et al.*, "Keras." https://keras.io, 2015.

[91] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," pp. 249–256, 2010.

[92] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[93] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[94] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[95] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Advances in Neural Information Processing Systems*, pp. 2546–2554, 2011.

[96] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," *arXiv preprint arXiv:1706.04599*, 2017.

[97] F. Mokhtarian, S. Abbasi, and J. Kittler, "Robust and efficient shape indexing through curvature scale space," in *British Machine Vision Conference*, Citeseer, 1996.