

—Supplemental Document —  
**The Impact Label Noise and Choice of Threshold has on  
Cross-Entropy and Soft-Dice in Image Segmentation**

Marcus Nordström<sup>1,2</sup>   Atsuto Maki<sup>1</sup>   Henrik Hult<sup>1</sup>  
<sup>1</sup>KTH Royal Institute of Technology   <sup>2</sup>RaySearch Laboratories  
`{marcno,atsuto,hult}@kth.se`

**Contents**

|   |          |
|---|----------|
| <b>1. Proofs</b>                            | <b>2</b> |
| 1.1. Proof of Proposition 1 . . . . .       | 2        |
| 1.2. Proof of Proposition 2 . . . . .       | 2        |
| 1.3. Proof of Proposition 3 . . . . .       | 3        |
| <b>2. Experiments</b>                       | <b>5</b> |
| 2.1. Computing the marginals . . . . .      | 5        |
| 2.2. Sampling noisy segmentations . . . . . | 5        |
| 2.3. Dice optimal segmentations . . . . .   | 5        |
| <b>3. Results</b>                           | <b>7</b> |

## 1. Proofs

### 1.1. Proof of Proposition 1

First, since the components of  $X$  are independent Gaussian fields with covariance kernel

$$k_{a,b}(\omega, \omega') = a^2 \exp\left(-\frac{\|\omega - \omega'\|_2^2}{2b^2}\right), \quad \omega, \omega' \in \mathbb{R}^n, \quad (1)$$

it follows that

$$\text{Var}((X_1(\omega), \dots, X_n(\omega))^T) = (k_{a,b}(\omega, \omega), \dots, k_{a,b}(\omega, \omega))^T = (a^2, \dots, a^2)^T, \quad (2)$$

and consequently that the marginal probability densities  $X(\omega), \omega \in \Omega$  are given by

$$X(\omega) \sim p_{a^2}(x), x \in \mathbb{R}^n. \quad (3)$$

Now, let

$$\hat{l}(\omega) = \begin{cases} l(\omega) & \text{if } \omega \in \Omega, \\ 0 & \text{otherwise,} \end{cases} \quad (4)$$

which means that

$$L(\omega) \stackrel{d}{=} \hat{l}(\omega + X(\omega)), \quad (5)$$

where  $\stackrel{d}{=}$  denotes equal in distribution. This together with the law of the unconscious statistician and the symmetry  $p_{a^2}(x) = p_{a^2}(-x), x \in \mathbb{R}^n$  then for any  $\omega \in \Omega$  yields

$$\mathbb{E}[L(\omega)] = \mathbb{E}[\hat{l}(\omega + X(\omega))] \quad (6)$$

$$= \int_{\mathbb{R}^n} \hat{l}(\omega + x) p_{a^2}(x) \lambda(dx) \quad (7)$$

$$= \int_{\mathbb{R}^n} \hat{l}(\omega') p_{a^2}(\omega' - \omega) \lambda(d\omega') \quad (8)$$

$$= \int_{\mathbb{R}^n} \hat{l}(\omega') p_{a^2}(\omega - \omega') \lambda(d\omega') \quad (9)$$

$$= \int_{\Omega} l(\omega') p_{a^2}(\omega - \omega') \lambda(d\omega'). \quad (10)$$

This completes the proof. □

### 1.2. Proof of Proposition 2

By Proposition 1, changing the order of integration and recalling that  $\int_{\mathbb{R}^n} p_{a^2}(\omega - \omega') \lambda(d\omega) = 1$  for any  $\omega' \in \mathbb{R}^n$ , it follows that

$$\mathbb{E} [\|L\|_1] = \mathbb{E} \left[ \int_{\Omega} |L(\omega)| \lambda(d\omega) \right] \quad (11)$$

$$= E \left[ \int_{\Omega} L(\omega) \lambda(d\omega) \right] \quad (12)$$

$$= \int_{\Omega} \mathbb{E}[L(\omega)] \lambda(d\omega) \quad (13)$$

$$= \int_{\Omega} \int_{\Omega} l(\omega') p_{a^2}(\omega - \omega') \lambda(d\omega') \lambda(d\omega) \quad (14)$$

$$= \int_{\Omega} l(\omega') \left[ \int_{\mathbb{R}^n} p_{a^2}(\omega - \omega') \lambda(d\omega) \right] \lambda(d\omega') - \int_{\Omega} l(\omega') \left[ \int_{\mathbb{R}^n \setminus \Omega} p_{a^2}(\omega - \omega') \lambda(d\omega) \right] \lambda(d\omega') \quad (15)$$

$$= \int_{\Omega} l(\omega') \lambda(d\omega') - \xi \quad (16)$$

$$= \int_{\Omega} |l(\omega')| \lambda(d\omega') - \xi \quad (17)$$

$$= \|l\|_1 - \xi. \quad (18)$$

This completes the proof.  $\square$

### 1.3. Proof of Proposition 3

Independently scattered measures are introduced more generally for  $\alpha$ -stable distributions in [3, Section 3.3]. The Gaussian case used in this work is simply the special case when  $\alpha = 2$ . Let  $W$  be an independently scattered Gaussian measure on  $\mathbb{R}^n$ , with control measure  $\lambda$ . That is, with  $\mathcal{B}_0$  being the Lebesgue measurable sets with finite measure, for any finite collection,  $A_1, \dots, A_k$  of disjoint sets in  $\mathcal{B}_0$ , the random variables  $W(A_1), \dots, W(A_k)$  are independent, and  $W(A_i)$  has centered Gaussian distribution with variance  $\lambda(A_i)$ . For  $f \in L^2(\mathbb{R}^n)$  the stochastic integral  $I(f) = \int_{\mathbb{R}^n} f(\omega) W(d\omega)$  is well defined with centered Gaussian distribution with variance  $\|f\|_{L^2}^2$ . In fact,  $\{I(f), f \in L^2(\mathbb{R}^n)\}$  is a Gaussian process indexed by  $L^2(\mathbb{R}^n)$ . In particular, for a given  $f \in L^2(\mathbb{R}^n)$ , the process  $Y = \{Y(\omega), \omega \in \mathbb{R}^n\}$  given by

$$Y(\omega) = \int_{\mathbb{R}^n} f(\omega - \omega') W(d\omega'), \quad (19)$$

is a centered Gaussian process with covariance kernel given by

$$k(\omega, \omega') = \int_{\mathbb{R}^n} f(\omega - u) f(\omega' - u) \lambda(du). \quad (20)$$

For the squared exponential kernel

$$k(\omega, \omega') = a^2 \exp \left( -\frac{\|\omega - \omega'\|^2}{2b^2} \right) \quad (21)$$

we can identify  $f$  as

$$f(\omega) = \frac{a}{(\pi b^2/2)^{n/4}} \exp \left( -\frac{\|\omega\|^2}{b^2} \right). \quad (22)$$

Indeed, for any  $\omega, \omega' \in \mathbb{R}^n$  it follows, after a completion of the square, that

$$k(\omega, \omega') = \int_{\mathbb{R}^n} f(\omega - u) f(\omega' - u) \lambda(du) \quad (23)$$

$$= \frac{a^2}{(\pi b^2/2)^{n/2}} \int_{\mathbb{R}^n} \exp\left(-\frac{\|\omega - u\|_2^2}{b^2}\right) \exp\left(-\frac{\|\omega' - u\|_2^2}{b^2}\right) \lambda(du) \quad (24)$$

$$= \frac{a^2}{(\pi b^2/2)^{n/2}} \exp\left(-\frac{\|\omega - \omega'\|_2^2}{2b^2}\right) \int_{\mathbb{R}^d} \exp\left(-\frac{\|u - \frac{\omega + \omega'}{2}\|_2^2}{2(b/2)^2}\right) \lambda(du) \quad (25)$$

$$= \frac{a^2 (2\pi(b/2)^2)^{d/2}}{(\pi b^2/2)^{d/2}} \exp\left(-\frac{\|\omega - \omega'\|_2^2}{2b^2}\right) \quad (26)$$

$$= a^2 \exp\left(-\frac{\|\omega - \omega'\|_2^2}{2b^2}\right) \quad (27)$$

Now consider the isotropic normal density in dimension  $d$  with variance  $b^2/2$

$$p_{b^2/2}(\omega) = \frac{1}{(\pi b^2)^{d/2}} \exp\left\{-\frac{\|\omega\|_2^2}{b^2}\right\} \quad (28)$$

and note that it can be used to rewrite  $f$  as follows

$$f(\omega) = a(2\pi b^2)^{n/4} p_{b^2/2}(\omega). \quad (29)$$

Consequently, it follows that

$$Y(\omega) = a(2\pi b^2)^{d/4} \int_{\mathbb{R}^n} p_{b^2/2}(\omega - \omega') W(dw'). \quad (30)$$

Now, let  $W_1, \dots, W_n$  be independent copies of  $W$ , then it follows that

$$X(\omega) \stackrel{d}{=} \begin{pmatrix} a(2\pi b^2)^{n/4} \int_{\mathbb{R}^n} p_{b^2/2}(\omega - \omega') W_1(dw') \\ \vdots \\ a(2\pi b^2)^{n/4} \int_{\mathbb{R}^n} p_{b^2/2}(\omega - \omega') W_n(dw') \end{pmatrix}, \quad (31)$$

where  $\stackrel{d}{=}$  denotes equal in distribution. This completes the proof.  $\square$

## 2. Experiments

The experimental code is composed of two parts. The first part is for extracting the data. The second part is for training the model. Most of the code is straight forward and similar to what would be found in any standard implementation of a UNet trained with either cross-entropy or soft-Dice. There are however three methods based on the theory described in this paper that needs clarification.

### 2.1. Computing the marginals

```
import numpy as np
import scipy.ndimage

def get_noisy_marginals(l,a):
    scaled_a = np.array(a)*np.array(l.shape)
    noisy_marginals = scipy.ndimage.gaussian_filter(m,scaled_a,mode='constant')

    return noisy_marginals
```

Listing 1. Python code for computing the marginals associated with a noisy segmentation that is formed by the noise free segmentation  $l$  and the noise strength parameter  $a$ . The code is a direct implementation of the Proposition 1.

The first method is for computing the marginals associated with a particular noisy segmentation and is based on Proposition 1. The code for this method is listed in Listing 1. The input is composed of:  $l$  a discretized version of the noise-free reference segmentation represented as a multidimensional numpy array and  $a$  the parameter to the noise model encoding the strength of the noise represented as a floating point number. The output is a discretized version of the marginals represented as a multidimensional numpy array.

### 2.2. Sampling noisy segmentations

```
import numpy as np
import scipy.ndimage

def get_noisy_sample(l,a):
    b = 0.15*np.sqrt(2)
    scaled_a = np.array(a)*np.array(l.shape)
    scaled_b = np.array(b)*np.array(l.shape)
    weight = scaled_a*(2*np.pi*scaled_b**2)**(len(np.shape(m))/4)

    perb = np.array([weight*scipy.ndimage.gaussian_filter(
        np.random.normal(size=l.shape),scaled_b[i]/np.sqrt(2),mode='constant')
        for i in range(len(l.shape))])
    grid_mesh = np.meshgrid(*[range(l.shape[i]) for i in range(len(l.shape))],indexing='ij')
    noisy_sample = np.round(scipy.ndimage.map_coordinates(l,grid_mesh+perb, mode='nearest'))

    return noisy_sample
```

Listing 2. Python code for computing a random sample associated with a noisy segmentation that is formed by the noise free segmentation  $l$  and the noise strength parameter  $a$ . The code is a direct implementation of Proposition 3.

The second method is for sampling noisy segmentations and is based on Proposition 3. The code for this method is listed in Listing 2. The input is composed of:  $l$  a discretized version of the noise-free reference segmentation represented as a multidimensional numpy array and  $a$  the parameter to the noise model encoding the strength of the noise represented as a floating point number. The output is a discretized version of a random sample associated with the noisy segmentation and is represented as a multidimensional numpy array. The method can be broken down into three steps. Firstly the constant used for rescaling is computed. Secondly, the vector of Gaussian fields is generated, which in the numerical setting is approximated by drawing i.i.d. Gaussian variables for each entry in  $l$  and then processing the resulting array with a Gaussian filter. Thirdly, the noise-free segmentation  $l$  is deformed with the resulting random deformation array.

### 2.3. Dice optimal segmentations

```
import numpy as np
```

```

def get_opt_dice_seg(m):
    psi = np.flip(np.sort(m.flatten()))
    d = 2*np.cumsum(psi)/(np.sum(m)+np.arange(1, len(psi)+1))
    t = np.max(d)/2
    s = 1.0*(m>=t)
    return s

```

Listing 3. Python code for generating a Dice optimal segmentation from a marginal function  $m$ .

The third method is for computing the optimal segmentation with respect to Dice. The code for this method is listed in Listing 3 and is taken from [2]. It is an efficient variation of a method proposed in binary classification [1]. The input is composed of:  $m$  a discretized version of the the marginal function represented as a multidimensional numpy array. The output is a discretized version of the optimal segmentation with respect to Dice represented as a multidimensional numpy array. The idea is to sort the voxels in  $m$  from largest to smallest, and then compute the Dice score associated with the segmentations that are formed by taking the first set of voxels associated with this sorted list. This is done for every possible number of voxels. The maximal score is used to form a threshold which is used to formulate the final segmentation.

### 3. Results

| Organ     | $a$    | $CE^{(0)}$ | $SD^{(0)}$ | $CE^{(*)}$ |
|-----------|--------|------------|------------|------------|
| Kidney    | 0.0000 | 0.9611     | 0.9634     | 0.9615     |
|           | 0.0100 | 0.8762     | 0.8794     | 0.8774     |
|           | 0.0200 | 0.7883     | 0.7914     | 0.7909     |
|           | 0.0300 | 0.6947     | 0.7080     | 0.7055     |
| Aorta     | 0.0000 | 0.9525     | 0.9515     | 0.9524     |
|           | 0.0100 | 0.8639     | 0.8654     | 0.8653     |
|           | 0.0200 | 0.7557     | 0.7569     | 0.7600     |
|           | 0.0300 | 0.6215     | 0.6513     | 0.6560     |
| Esophagus | 0.0000 | 0.8552     | 0.8603     | 0.8602     |
|           | 0.0100 | 0.6671     | 0.6722     | 0.6814     |
|           | 0.0200 | 0.4168     | 0.4527     | 0.4829     |
|           | 0.0300 | 0.1441     | 0.3105     | 0.3489     |

Table 1. Table over the results from the experiments for each organ and noise level  $a$ . The entries are average Dice scores obtained from the average over the five folds.

## References

- [1] Zachary C Lipton, Charles Elkan, and Balakrishnan Naryanaswamy. Optimal Thresholding of Classifiers to Maximize F1 Measure. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 225–239. Springer, 2014. [6](#)
- [2] Marcus Nordstrom, Henrik Hult, Fredrik Löfman, and Jonas Söderberg. On image segmentation with noisy labels: Characterization and volume properties of the optimal solutions to accuracy and dice. *Advances in Neural Information Processing Systems*, 35:34321–34333, 2022. [6](#)
- [3] Gennady Samoradnitsky and Murad S. Taqqu. *Stable non-Gaussian random processes: stochastic models with infinite variance*. Chapman and Hall, 1994. [3](#)