RankSEG: A Consistent Ranking-based Framework for Segmentation

Ben Dai  
Department of Statistics  
The Chinese University of Hong Kong  
Hong Kong SAR  

Chunlin Li  
School of Statistics  
University of Minnesota  
MN 55455 USA  

Editor: Christoph Lampert

Abstract

Segmentation has emerged as a fundamental field of computer vision and natural language processing, which assigns a label to every pixel/feature to extract regions of interest from an image/text. To evaluate the performance of segmentation, the Dice and IoU metrics are used to measure the degree of overlap between the ground truth and the predicted segmentation. In this paper, we establish a theoretical foundation of segmentation with respect to the Dice/IoU metrics, including the Bayes rule and Dice-/IoU-calibration, analogous to classification-calibration or Fisher consistency in classification. We prove that the existing thresholding-based framework with most operating losses are not consistent with respect to the Dice/IoU metrics, and thus may lead to a suboptimal solution. To address this pitfall, we propose a novel consistent ranking-based framework, namely \textit{RankDice}/\textit{RankIoU}, inspired by plug-in rules of the Bayes segmentation rule. Three numerical algorithms with GPU parallel execution are developed to implement the proposed framework in large-scale and high-dimensional segmentation. We study statistical properties of the proposed framework. We show it is Dice-/IoU-calibrated, and its excess risk bounds and the rate of convergence are also provided. The numerical effectiveness of \textit{RankDice}/\textit{mRankDice} is demonstrated in various simulated examples and \textit{Fine-annotated CityScapes}, \textit{Pascal VOC} and \textit{Kvasir-SEG} datasets with state-of-the-art deep learning architectures. Python module and source code are available on GitHub at https://github.com/statmlben/rankseg.

Keywords: Segmentation, Bayes rule, ranking, Dice-calibrated, excess risk bounds, Poisson-binomial distribution, normal approximation, GPU computing

1. Introduction

Segmentation is one of the key tasks in the field of computer vision and natural language processing, which groups together similar pixels/features of an input that belong to the same class (Ronneberger et al., 2015; Badrinarayanan et al., 2017). It has become an essential part of image and text understanding with applications in autonomous vehicles (Assidiq et al., 2008), medical image diagnostics (Wang et al., 2018), face/fingerprint recognition (Xin et al., 2018), and video action localization (Shou et al., 2017).

The primary aim of segmentation is to label each foreground feature/pixel of an input with a corresponding class. Specifically, for a feature vector or an image $X \in \mathbb{R}^d$, a segmentation function $\delta$: 
\( \mathbb{R}^d \rightarrow \{0,1\}^d \) yields a predicted segmentation \( \delta(X) = (\delta_1(X), \cdots, \delta_d(X))^\top \), where \( \delta_j(X) \) represents the predicted segmentation for the \( j \)-th feature \( X_j \), and \( I(\delta(X)) = \{ j : \delta_j(X) = 1 \} \) for \( j = 1, \cdots, d \) is the index set of the segmented features of \( X \) provided by \( \delta \). Correspondingly, \( Y \in \{0,1\}^d \) is a feature-wise label of a ground truth segmentation, where \( Y_j = 1 \) indicates that the \( j \)-th feature/pixel \( X_j \) is segmented, and \( I(Y) = \{ j : Y_j = 1 \} \) is the index set of the ground-truth features.

To access the performance for a segmentation function \( \delta \), the Dice and IoU metrics are introduced and widely used in the literature (Milletari et al. 2016), both of which measure the overlap between the ground truth and the predicted segmentation:

\[
\text{Dice}_\gamma(\delta) = \mathbb{E} \left( \frac{2|I(Y) \cap I(\delta(X))| + \gamma}{|I(Y)| + |I(\delta(X))| + \gamma} \right) = \mathbb{E} \left( \frac{2Y^\top \delta(X) + \gamma}{\|Y\|_1 + \|\delta(X)\|_1 + \gamma} \right),
\]

\[
\text{IoU}_\gamma(\delta) = \mathbb{E} \left( \frac{|I(Y) \cap I(\delta(X))| + \gamma}{|I(Y) \cup I(\delta(X))| + \gamma} \right) = \mathbb{E} \left( \frac{Y^\top \delta(X) + \gamma}{\|Y\|_1 + \|\delta(X)\|_1 - \|\delta(X)\|_1 + \gamma} \right),
\]

where \( | \cdot | \) is the cardinality of a set, and \( \gamma \geq 0 \) is a smoothing parameter. When \( \gamma = 0 \), \( \text{Dice}_\gamma(\delta) = \mathbb{E} (2\text{TP} / (2\text{TP} + \text{FP} + \text{FN})) \). \( \text{IoU}_\gamma(\delta) = \mathbb{E} (\text{TP} / (\text{TP} + \text{FP} + \text{FN})) \) where \( \text{TP} \) is the true positive, \( \text{FP} \) is the false positive, and \( \text{FN} \) is the false negative. Both metrics are similar and will be treated in a unified fashion; however, as to be seen in the sequel, searching for the optimal segmentation function with respect to the IoU metric may require extra computation than its Dice counterpart. Thus, for clarity of presentation, we first focus on the Dice metric and postpone the discussion on the relationships between the Dice and IoU metrics in Section 4.2.

The recent success of fully convolutional networks has enabled significant progress in segmentation. In literature, the mainstream of recent works devoted to designing and developing neural network architectures under different segmentation scenarios, including FCN (Long et al., 2015), U-Net (Ronneberger et al. 2015), DeepLab (Chen et al., 2018), and PSPNet (Zhao et al., 2017). Despite their remarkable performance, most existing models primarily focus on predicting segmentation using a classification framework, without considering the inherent disparities between classification and segmentation (as discussed in Section 1.1). We find this framework leading to an inconsistent solution and suboptimal performance with respect to the Dice/IoU metrics, and we address this pitfall by developing a novel consistent ranking-based framework, namely RankSEG (RankDice to the Dice metric and RankIoU to the IoU metric), to improve the segmentation performance.

1.1 Existing methods

Most existing segmentation methods are developed under a threshold-based framework with two types of loss functions.

As indicated in Figure 1, the existing threshold-based segmentation framework, inspired by binary classification, provides a predicted segmentation via a two-stage procedure: (i) estimating a decision function or a probability function based on a loss; (ii) predicting feature-wise labels by thresholding the estimated decision function or probabilities. Specifically, given a training dataset \( \{(x_i, y_i)\}_{i=1}^n \), the prediction provided by the threshold-based framework for a new instance \( x \) can be formulated as:

\[
\tilde{q} = \arg\min_{q \in \mathcal{Q}} \frac{1}{n} \sum_{i=1}^n l(y_i, q(x_i)) + \lambda \|q\|^2, \quad \delta(x) = 1(\tilde{q}(x) \geq 0.5),
\]

where \( \mathcal{Q} \) is the set of possible decision functions, \( l \) is the loss function, and \( \lambda \) is a weighting parameter.

The recent success of fully convolutional networks has enabled significant progress in segmentation. In literature, the mainstream of recent works devoted to designing and developing neural network architectures under different segmentation scenarios, including FCN (Long et al., 2015), U-Net (Ronneberger et al. 2015), DeepLab (Chen et al., 2018), and PSPNet (Zhao et al., 2017). Despite their remarkable performance, most existing models primarily focus on predicting segmentation using a classification framework, without considering the inherent disparities between classification and segmentation (as discussed in Section 1.1). We find this framework leading to an inconsistent solution and suboptimal performance with respect to the Dice/IoU metrics, and we address this pitfall by developing a novel consistent ranking-based framework, namely RankSEG (RankDice to the Dice metric and RankIoU to the IoU metric), to improve the segmentation performance.

1.1 Existing methods

Most existing segmentation methods are developed under a threshold-based framework with two types of loss functions.

As indicated in Figure 1, the existing threshold-based segmentation framework, inspired by binary classification, provides a predicted segmentation via a two-stage procedure: (i) estimating a decision function or a probability function based on a loss; (ii) predicting feature-wise labels by thresholding the estimated decision function or probabilities. Specifically, given a training dataset \( \{(x_i, y_i)\}_{i=1}^n \), the prediction provided by the threshold-based framework for a new instance \( x \) can be formulated as:

\[
\tilde{q} = \arg\min_{q \in \mathcal{Q}} \frac{1}{n} \sum_{i=1}^n l(y_i, q(x_i)) + \lambda \|q\|^2, \quad \delta(x) = 1(\tilde{q}(x) \geq 0.5),
\]
Figure 1: The existing and the proposed (RankDice) frameworks for segmentation. The upper panel is the existing threshold-based segmentation framework, and the lower panel is the proposed RankDice framework.

where \( I(\cdot, \cdot) \) is an operating loss, \( q : \mathbb{R}^d \rightarrow [0, 1]^d \) is a candidate probability function with \( q_j \) being the candidate probability of the \( j \)-th pixel, \( Q \) is a class of candidate probability functions, \( \|q\| \) is a regularization term, \( \lambda > 0 \) is a tuning parameter to balance the overfitting and underfitting, and \( 1(\cdot) \) is an indicator function. For ease of presentation, \( q_j(x) \) is specified as a probability function and a predicted segmentation is produced by thresholding at 0.5, yet it can be equally extended to a general decision function. For example, we may formulate \( q_j(x) \) as a decision function with range in \( \mathbb{R} \), and the prediction is produced by thresholding at 0, analogous to SVMs in classification (Cortes and Vapnik, 1995). Next, under the framework (2), two different types of operating loss functions are considered, namely the classification-based losses and the Dice-approximating losses. Classification-based losses completely characterize segmentation as a classification problem, with examples such as the cross-entropy (CE; Cox (1958)) and the focal loss (Focal; Lin et al. (2017)):

\[
\text{(CE)} \quad l_{\text{CE}}(y, q(x)) = -\sum_{j=1}^{d} \left( y_j \log (q_j(x)) + (1-y_j) \log (1-q_j(x)) \right),
\]

\[
\text{(Focal)} \quad l_{\text{focal}}(y, q(x)) = -\sum_{j=1}^{d} \left( y_j (1-q_j(x))^\vartheta \log (q_j(x)) + (1-y_j) q_j^\vartheta(x) \log (1-q_j(x)) \right),
\]

where \( \vartheta \geq 0 \) is a hyperparameter for the focal loss (Lin et al., 2017). Other margin-based losses such as the hinge loss, in principle, can be included as classification-based losses with a decision function ranged in \( \mathbb{R} \) thresholding at 0, although they are less frequently used in a multiclass problem (Tewari and Bartlett, 2007). Therefore, we focus on the probability-based classification loss in the sequel. Dice-approximating losses aim to approximate the Dice/IoU metric and conduct a direct optimization. Typical examples are the soft-Dice (Sudre et al. 2017) and the Lovasz-softmax loss (Berman et al. 2018):

\[
\text{(Soft-Dice)} \quad l_{\text{SoftD}}(y, q(x)) = 1 - \frac{2y^T q(x)}{\|y\|_1 + \|q(x)\|_1},
\]

\[
\text{(Lovasz-softmax)} \quad l_{\text{Softmax}}(y, q(x)) = \nabla_{\text{lovasz}}(y \circ (1-q(x)) + (1-y) \circ q(x)),
\]

where \( \nabla(\cdot) \) is the Lovasz extension of the mis-IoU error (Berman et al., 2018), and \( \circ \) is the element-wise product. Specifically, the soft-Dice loss replaces the binary segmentation indicator \( \delta(x) \in \)
\{0,1\}^d in the Dice metric by a candidate probability function \(q(x) \in [0,1]^d\) to make the computation feasible. The Lovasz-softmax directly takes a convex extension of IoU based on a softmax transformation. Moreover, other losses including the Tversky loss (Salehi et al., 2017), the Lovasz-hinge loss (Berman et al., 2018), and the log-Cosh Dice loss (Jadon, 2021) can be also categorized as Dice-approximating losses.

The threshold-based framework (2) with a classification-based loss or a Dice-approximating loss is a commonly used approach for segmentation. Although encouraging performance is delivered, we show that the minimizer from (2) (based on the cross-entropy and the focal loss) is \textbf{inconsistent} (or suboptimal) to the Dice metric, see Lemma 9. For Lovasz convex losses, it is still unclear if they are able to yield an optimal segmentation (convex closure is usually not enough to ensure the consistency (Bartlett et al., 2006)). Moreover, in practice, only a small number of pixel predictions are taken into account in one stochastic gradient step. Therefore, the Lovasz-softmax loss cannot directly optimize the segmentation metric (see Section 3.1 in Berman et al. (2018)).

Another line of work (Bao and Sugiyama, 2020; Nordström et al., 2020; Lipton et al., 2014) has been centered on a linear-fractional approximation of the Dice and IoU metrics which are not decomposable per instance. In particular, Bao and Sugiyama (2020) and Nordström et al. (2020) proposed to approximate the utility by tractable surrogate functions with a sample-splitting procedure and showed that their methods are consistent in optimizing the target utility. Yet, splitting the sample may undermine the efficiency. Lipton et al. (2014) indicated that the optimal threshold maximizing the \(F_1\) metric is equal to half of the optimal \(F_1\) value. However, their definitions of Dice and IoU metrics may overlook small instances, which is undesirable in many applications; see Appendix A for technical discussion.

To summarize, the current frameworks with existing losses may either yield a suboptimal solution or suffer from an inappropriate target metric, demanding efforts to further improve the performance, robustness and sustainability of the existing segmentation framework.

1.2 Our contribution

In this paper, we propose a novel Dice-calibrated ranking-based segmentation framework, namely \textit{RankDice}, to address the inconsistency of the existing framework. \textit{RankDice} is primarily inspired by the Bayes rule of Dice-segmentation. We summarize our major contribution as follows:

1. To our best knowledge, the proposed ranking-based segmentation framework \textit{RankDice}, is the first consistent segmentation framework with respect to the Dice metric (Dice-calibrated).

2. Three numerical algorithms with GPU parallel execution are developed to implement the proposed framework in large-scale and high-dimensional segmentation.

3. We establish a theoretical foundation of segmentation with respect to the Dice metric, such as the Bayes rule and Dice-calibration. Moreover, we present Dice-calibrated consistency (Lemma 10) and a convergence rate of the excess risk (Theorem 11) for the proposed \textit{RankDice} framework, and indicate inconsistent results for the existing methods (Lemma 9).

4. Our experiments in three simulated examples and three real datasets (CityScapes dataset, Pascal VOC 2021 dataset, and Kvasir-SEG dataset) suggest that the improvement of \textit{RankDice} over the existing framework is significant for various loss functions and network architectures (see Tables 9-6).
It is worth noting that the results are equally applicable to the proposed RankIoU framework in terms of the IoU metric.

2. RankDice

It is reasonable to assume that all information on a feature-wise label is solely based on input features, that is, \( Y_i \perp Y_j | X \) for any \( i \neq j \). In Appendix B.3 we provide a probabilistic perspective to suggest the necessity of this assumption in segmentation tasks. Without loss of generality, we further assume that \( p_j(X) := P(Y_j = 1|X) \) are distinct for \( j = 1, \cdots, d \) with probability one.

2.1 Bayes segmentation rule

To begin with, we call segmentation with respect to the Dice metric as “Dice-segmentation”. Then, we discuss Dice-segmentation at the population level, and present its Bayes (optimal) segmentation rule in Theorem 1 akin to the Bayes classifier for classification.

**Theorem 1 (The Bayes rule for Dice-segmentation)** A segmentation rule \( \delta^* \) is a global maximizer of \( \text{Dice}_\gamma(\delta) \) if and only if it satisfies that

\[
\delta^*_j(x) = \begin{cases} 
1 & \text{if } p_j(x) \text{ ranks top } \tau^*(x), \\
0 & \text{otherwise}.
\end{cases}
\]

The optimal volume (the optimum total number of segmented features) \( \tau^*(x) \) is given as

\[
\tau^*(x) = \arg\max_{\tau \in \{0, 1, \cdots, d\}} \left( \sum_{j \in J_j(x)} \frac{2}{\tau + l + \gamma + 1} p_j(x) P(\Gamma_j(x) = l) + \sum_{l=0}^{d} \frac{\gamma}{\tau + l + \gamma} P(\Gamma(x) = l) \right),
\]

where \( J_j(x) = \{ j : \sum_{j'=1}^{d} 1( p_{j'}(x) \geq p_j(x) ) \leq \tau \} \) is the index set of the \( \tau \)-largest conditional probabilities with \( J_0(x) = \emptyset, \Gamma(x) = \sum_{j=1}^{d} B_j(x), \text{and } \Gamma_{-j}(x) = \sum_{j \neq j} B_{j'}(x) \) are Poisson-binomial random variables, and \( B_j(x) \) is a Bernoulli random variable with the success probability \( p_j(x) \). See the definition of the Poisson-binomial distribution in Appendix B.2.

Two remarkable observations emerge from Theorem 1. First, the Bayes segmentation operator can be obtained via a two-stage procedure: (i) ranking the conditional probability \( p_j(x) \), and (ii) searching for the optimal volume of the segmented features \( \tau(x) \). Second, both the Bayes segmentation rule \( \delta^*(x) \) and the optimal volume function \( \tau^*(x) \) are achievable when the conditional probability \( \mathbf{p}(x) = (p_1(x), \cdots, p_d(x))^T \) is well-estimated. Therefore, our proposed framework RankDice is directly inspired by a general plug-in rule of the Bayes segmentation rule.

Moreover, Lemma 9 (and examples provided in its proof) indicates that the segmentation rule produced by existing frameworks, such as the threshold-based framework, can significantly differ from the optimum in Theorem 1. In fact, Lemma 9 further proves that the cross-entropy loss, the focal loss, and even a general classification-calibrated loss (Zhang, 2004; Bartlett et al., 2006), are not Dice-calibrated. See the definition of Dice-calibrated (Definition 8), and the negative results for existing frameworks (Lemma 9) in Section 4.1. Besides, the Bayes rule for IoU-segmentation is presented in Lemma 13.
Remark 2 (Suboptimal of a fixed-thresholding framework) The Bayes rule of Dice-segmentation can be also regarded as adaptive thresholding of conditional probabilities. Specifically, for each input \( \mathbf{x} \), the optimal segmentation rule can be rewritten as:

\[
\delta_j^*(\mathbf{x}) = 1(p_j(\mathbf{x}) \geq p_{j^*}(\mathbf{x})) , \quad \text{where } p_{j^*}(\mathbf{x}) \text{ is the top-} \tau^* \text{ largest probability over } p(\mathbf{x}).
\]

Alternatively, Theorem 1 indicates that the Bayes rule for Dice-segmentation is unlikely to be obtained by a fixed thresholding framework, since the optimal threshold \( p_{j^*}(\mathbf{x}) \) varies greatly across different inputs. Therefore, (tuning a threshold on) a fixed thresholding-based framework leads to a suboptimal solution.

Remark 2 also explains the heterogeneity of optimal thresholds in various datasets indicated in Bice et al. (2021), and the suboptimality of a fixed-thresholding framework is also empirically supported by Table 10 and Figure 7 for simulated examples, and Table 7 and Figure 4 for real datasets. Furthermore, the fact that fixed-thresholding is suboptimal for Dice-segmentation should be compared with the existing results in classification. For binary classification, the optimal threshold \( p_{j^*}(\mathbf{x}) \) varies greatly across different inputs. Therefore, (tuning a threshold on) a fixed thresholding-based framework leads to a suboptimal solution.

2.2 Proposed framework

Suppose a training dataset \((\mathbf{x}_i, y_i)_{i=1}^n\) is given, where \( \mathbf{x}_i \in \mathbb{R}^d \) and \( y_i \in \{0, 1\}^d \) are the input features and the true label for the \( i \)-th instance. Inspired by Theorem 1, we develop a ranking-based framework \textit{RankDice} for Dice-segmentation (Steps 1-3).

**Step 1 (Conditional probability estimation):** Estimate the conditional probability based on logistic regression (the cross-entropy loss):

\[
\hat{q}(\mathbf{x}) = \arg\min_{q \in \mathcal{Q}} - \sum_{i=1}^n \sum_{j=1}^d \left( y_{ij} \log(q_j(\mathbf{x}_i)) + (1 - y_{ij}) \log(1 - q_j(\mathbf{x}_i)) \right) + \lambda \|q\|^2, \tag{5}
\]

where \( \mathcal{Q} \) is a class of candidate probability functions, \( \|q\| \) is a regularization for a candidate function, and \( \lambda > 0 \) is a hyperparameter to balance the loss and regularization. For example, \( q \in \mathcal{Q} \) is usually a deep convolutional neural network for image segmentation, and \( \|q\| \) can be a matrix norm of weight matrices in the network.

**Step 2 (Ranking):** Given a new instance \( \mathbf{x} \), rank its estimated conditional probabilities decreasingly, and denote the corresponding indices as \( j_1, \cdots, j_d \), that is, \( \tilde{q}_{j_1}(\mathbf{x}) \geq \tilde{q}_{j_2}(\mathbf{x}) \geq \cdots \geq \tilde{q}_{j_d}(\mathbf{x}) \).

**Step 3 (Volume estimation):** From (4), we estimate the volume \( \tilde{\tau}(\mathbf{x}) \) by replacing the true conditional probability \( p(\mathbf{x}) \) by the estimated one \( \hat{q}(\mathbf{x}) \):

\[
\tilde{\tau}(\mathbf{x}) = \arg\max_{\tau \in \{0, \cdots, d\}} \sum_{i=0}^{\tau} \sum_{l=0}^{d-1} \frac{2}{\tau + l + 1} \tilde{q}_{j_i}(\mathbf{x}) \mathbb{P}(\hat{\Gamma}_{-j_i}(\mathbf{x}) = l) + \sum_{i=0}^{d} \frac{\gamma}{\tau + l + \gamma} \mathbb{P}(\hat{\Gamma}(\mathbf{x}) = l), \tag{6}
\]

where \( \hat{\Gamma}(\mathbf{x}) = \sum_{j=1}^d \hat{B}_j(\mathbf{x}) \) and \( \hat{\Gamma}_{-j_i}(\mathbf{x}) = \sum_{j \neq j_i} \hat{B}_j(\mathbf{x}) \) are Poisson-binomial random variables, and \( \hat{B}_j(\mathbf{x}) \) are independent Bernoulli random variables with success probabilities \( \hat{q}_j(\mathbf{x}) \); for \( j = 1, \cdots, d \).
Finally, the predicted segmentation $\hat{\delta}(x) = (\hat{\delta}_1(x), \cdots, \hat{\delta}_d(x))^T$ is given by selecting the indices of the top-$\hat{\tau}(x)$ conditional probabilities:

$$\hat{\delta}_j(x) = 1, \text{ if } j \in \{j_1, \cdots, j_{\hat{\tau}(x)}\}; \quad \hat{\delta}_j(x) = 0, \text{ otherwise}. \quad (7)$$

The proposed RankDice framework (Steps 1-3) provides a feasible solution to the Bayes segmentation rule in terms of the Dice metric. Note that Step 1 is a standard conditional probability estimation, and Step 2 simply ranks the estimated conditional probabilities. Next, we focus on developing a scalable computing scheme for Step 3.

### 2.3 Scalable computing schemes

This section develops scalable computing schemes for volume estimation in (6). Note that (6) can be rewritten as:

$$\hat{\tau}(x) = \arg\max_{\tau \in \{0, \cdots, d\}} \Pi_{\tau}(x); \quad \Pi_{\tau}(x) = \omega_{\tau}(x) + \nu_{\tau}(x),$$

$$\omega_{\tau}(x) = \frac{d-1}{\tau + l + \gamma + 1}, \quad \omega_{\tau,l}(x) = \sum_{s=1}^{\tau} \hat{q}_{js}(x) \mathbb{P}^{\hat{\Gamma}_{-js}(x)} = l, \quad \nu_{\tau}(x) = \sum_{l=0}^{d} \gamma \hat{\nu}(\hat{\Gamma}(x) = l) \frac{\tau + l + \gamma}{\tau + l + \gamma} \quad (8)$$

The computational complexity of solving (8) is intimately related to the dimension of input features. Therefore, we develop numerical algorithms for low- and high-dimensional segmentation separately.

#### 2.3.1 Exact algorithms for low-dimensional segmentation

**Exact algorithm based on FFT.** When the dimension is low ($d \leq 500$), we consider an exact algorithm to evaluate $\omega_{\tau}$ and $\nu_{\tau}$. According to the definition of $\omega_{\tau,l}$, it can be computed by the following recursive formula ($\tau = 1, \cdots, d$):

$$\omega_{\tau}(x) = \omega_{\tau-1}(x) + \hat{q}_{js}(x) (\mathbb{P}^{\hat{\Gamma}_{-js}(x) = 0}, \cdots, \mathbb{P}^{\hat{\Gamma}_{-js}(x) = d - 1})^T, \quad \omega_0(x) = 0 \quad (9)$$

where $\omega_{\tau}(x) = (\omega_{\tau,0}(x), \cdots, \omega_{\tau,d-1}(x))^T$. On this ground, it suffices to evaluate $\mathbb{P}^{\hat{\Gamma}(x) = l}$ and $\mathbb{P}^{\hat{\Gamma}_{-js}(x) = l}$, which are the probability mass functions of Poisson-binomial random variables $\hat{\Gamma}(x)$ and $\hat{\Gamma}_{-js}(x)$, respectively. As indicated in Hong (2013), they can be efficiently evaluated by a fast Fourier transformation (FFT). Based on the numerical results in Hong (2013), the computing time for FFT evaluation with $d \leq 500$ is generally negligible (less than ten milliseconds). Moreover, it is worth noting that our algorithm in (9) needs not store the entire auxiliary matrix $(\mathbf{w}_1(x), \cdots, \mathbf{w}_d(x))$, since the $\tau$-th row $\mathbf{w}_{\tau}$ can be computed from the previous row $\mathbf{w}_{\tau-1}$. Hence, only $O(d)$ storage is required in (9). The detailed algorithm is summarized in Algorithm 1 with approx=False.

#### 2.3.2 Approximation algorithms for high-dimensional segmentation

For high-dimensional segmentation, especially for image segmentation, it is challenging to solve (6) by a grid searching over $\tau \in \{0, \cdots, d\}$. To address this issue, we use shrinkage and approximation techniques to reduce the computational complexity of (6). First, Lemma 3 is developed to shrink the searching range of $\tau$. 

---

7
Lemma 3 (Shrinkage) If \( \sum_{j=1}^{\tau} q_{jx}(x) \geq (\tau + \gamma + d) \hat{q}_{jx+1}(x) \), then \( \overline{\pi}_x(x) \geq \overline{\pi}_{\tau'}(x) \) for all \( \tau' > \tau \).

Lemma 3 can be viewed as an early stopping of the grid searching, which draws from the property of Poisson binomial distribution (c.f. Lemma 17). Accordingly, we can shrink the grid search in (6) from \( \{0, \cdots, d\} \) to \( \{0, \cdots, d_0(x)\} \), which significantly reduces the computational complexity. Specifically,

\[
\tau(x) = \arg\max_{\tau \in \{0, \cdots, d_0(x)\}} \pi_x(x); \quad d_0(x) = \min\{\tau = 1, \cdots, d : \sum_{s=1}^{\tau} q_{jx}(x)/\hat{q}_{jx+1}(x) \geq \tau + \gamma + d\}. \tag{10}
\]

In many applications, \( d_0(x) \) is upper bounded by a small integer, that is, \( d_0(x) < d_U \ll d \), called the well-separated segmentation. For example, there exists a small number of features/ pixels whose probabilities (close to 1) are much larger than the others.

**Truncated refined normal approximation (T-RNA).** Note that the cumulative distribution function (CDF), and thus the probability mass function, of a Poisson-binomial random variable can be efficiently evaluated by a refined normal approximation when the dimension is large (Hong, 2013; Neammanee, 2005). For instance,

\[
\mathbb{P}(\overline{\Gamma}(x) \leq l) \leftarrow \mathbb{P}(\overline{\Gamma}(x) \leq l) := \Psi(\hat{\sigma}^{-1}(l + 1/2 - \hat{\mu}(x))), \\
\mathbb{P}(\overline{\Gamma}_{jx}(x) \leq l) \leftarrow \mathbb{P}(\overline{\Gamma}_{jx}(x) \leq l) := \Psi_j(\hat{\sigma}_j^{-1}(l + 1/2 - \hat{\mu}_j(x))), \tag{11}
\]

where \( \Psi(u) = \Phi(u) + \hat{\eta}(x)(1 - u^2)\phi(u)/6 \) and \( \Psi_j(u) = \Phi(u) + \hat{\eta}_j(x)(1 - u^2)\phi(u)/6 \) are two skew-corrected refined normal CDFs, \( \Phi(\cdot) \) is the CDF of the standard normal distribution, and \((\hat{\mu}(x), \hat{\mu}_j(x)), (\hat{\sigma}^2(x), \hat{\sigma}^2_j(x)), (\hat{\eta}(x), \hat{\eta}_j(x))\) are mean, variance and skewness of the Poisson-binomial random variables \( \overline{\Gamma}(x) \) and \( \overline{\Gamma}_{jx}(x) \), respectively. See the definitions of variance and skewness of the Poisson-binomial distribution in Appendix B.2.

On this ground, it is unnecessary to compute all \( \mathbb{P}(\overline{\Gamma}_{jx}(x) = l) \) and \( \mathbb{P}(\overline{\Gamma}(x) = l) \) for \( l = 1, \cdots, d \), since they are negligibly close to zero when \( l \) is too small or too large. In other words, many \( \mathbb{P}(\overline{\Gamma}_{jx}(x) = l) \) and \( \mathbb{P}(\overline{\Gamma}(x) = l) \) can be ignored when evaluating \( \overline{\sigma}_x \) and \( \overline{\nu}_x \). Therefore, according to the refined normal approximation in (11), \( \overline{\sigma}_x \) and \( \overline{\nu}_x \) can be approximated by only taking a partial sum over a subset of \( l = 0, \cdots, d \):

\[
\overline{\sigma}_x(x) = \sum_{l \in \mathcal{L}(x)} \frac{2\overline{\sigma}_{lx}(x)}{\tau + l + \gamma + 1}, \quad \overline{\nu}_x(x) = \sum_{l \in \mathcal{L}(x)} \frac{\gamma \mathbb{P}(\overline{\Gamma}(x) = l)}{\tau + l + \gamma}, \quad \overline{\sigma}_{jx}(x) = \sum_{s=1}^{\tau} q_{jx}(x) \mathbb{P}(\overline{\Gamma}_{jx}(x) = l),
\]

\[
\mathcal{L}(x) = \left\{ [\hat{\sigma}(x)\Psi^{-1}(\epsilon) + \hat{\mu}(x) - \frac{3}{2}], \cdots, [\hat{\sigma}(x)\Psi^{-1}(1-\epsilon) + \hat{\mu}(x) - \frac{1}{2}] \right\} \cap \{0, \cdots, d\}, \tag{12}
\]

where \( \mathbb{P}(\overline{\Gamma}(x) = l) := \mathbb{P}(\overline{\Gamma}(x) \leq l) - \mathbb{P}(\overline{\Gamma}(x) \leq l - 1), \mathbb{P}(\overline{\Gamma}_{jx}(x) = l) := \mathbb{P}(\overline{\Gamma}_{jx}(x) \leq l) - \mathbb{P}(\overline{\Gamma}_{jx}(x) \leq l - 1) \), \( \Psi^{-1}(\cdot) \) is the quantile function of the refined normal distribution, and \( \epsilon \) is a custom tolerance error. The detailed algorithm is summarized in Algorithm 1 with approx = ‘T-RNA’. Moreover, the following lemma quantifies the approximation error of the proposed approximate algorithm.

**Lemma 4** For \( (\overline{\sigma}_x, \overline{\nu}_x) \) and \( (\overline{\sigma}_{jx}, \overline{\nu}_{jx}) \) defined in (8) and (12) respectively, if \( \hat{\sigma}^2(x) \geq 25 \), then for any \( \tau \in \{0, \cdots, d\} \), we have

\[
|\overline{\sigma}_x - \overline{\sigma}_x| \leq \frac{4}{\tau + \gamma + 1}(\epsilon + \frac{C_0}{\hat{\sigma}^2(x)}) + \frac{C_0 \min(\hat{\mu}_x, \tau)}{\hat{\sigma}^2(x) - 1/4} \left( \log (1 + d) + 1 \right),
\]

\[
|\overline{\nu}_x - \overline{\nu}_x| \leq \frac{2\gamma}{\tau + \gamma}(\epsilon + \frac{C_0}{\hat{\sigma}^2(x)}) + \frac{C_0 \gamma}{\hat{\sigma}^2(x)} \left( \log (1 + d) + 1 \right),
\]
where $C_0 = 0.1618$ if $\tilde{\sigma}^2(x) \geq 100$, and $C_0 = 0.3056$ if $\tilde{\sigma}^2(x) \geq 25$. Moreover, when $d \to \infty$, we have

$$|\tilde{\pi}_\tau - \bar{\pi}_\tau| \leq \left( \frac{4\tau}{\tau + \gamma + 1} + \frac{2\gamma}{\tau + \gamma} \right) \epsilon + O(\frac{\min(\mu(x), \tau) \log(d)}{\tilde{\sigma}^2(x)}).$$

Here we define $0/0 := 0$ for $\gamma/(\tau + \gamma)$ when $\tau = \gamma = 0$.

Note that $\tilde{\sigma}^2(x) = \sum_{j=1}^{d} \tilde{q}_j(x)(1 - \tilde{q}_j(x))$ is the variance of $\tilde{\Gamma}(x)$, which often tends to infinity as $d \to \infty$. In image segmentation, the dimension $d$ varies from 1024 (32x32) to 262144 (512x512). Therefore, the error bound is practical for high-dimensional segmentation. Moreover, $\epsilon$ is the tolerance error to balance the approximation error and computation complexity. For instance, when $\epsilon$ becomes smaller, the approximation error decreases, the computation complexity increases with an enlarged $L(\epsilon)$. Typically, $\Psi^{-1}(1 - \epsilon) - \Psi^{-1}(\epsilon) \leq 7.438$ when $\epsilon = 1e-4$. Based on the approximation formula (12), the worst-case computational complexity is reduced to $O(d\tilde{\sigma}(x))$ for general segmentation and $O(\tilde{\sigma}(x))$ for well-separated segmentation ($d_0(x) \leq d_U$). The computational complexity is summarized in Table 1 (based on $\epsilon = 1e-4$).

Although T-RNA significantly reduces the computational complexity, yet this algorithm uses recursive updates, making it difficult in parallel computing on GPUs. For example, due to the memory issues, recursive function calls are restricted in the CUDA (Compute Unified Device Architecture) kernels (Nickolls et al., 2008). Next, we propose the blind approximation algorithm to exploit GPU-computing for the proposed RankDice.

**Blind approximation (BA).** In high-dimensional segmentation, the difference in distributions between $\tilde{\Gamma}(x)$ and $\Gamma_{\mathcal{J}}(x)$ is negligible. Inspired by this fact, we propose a novel approximation algorithm, namely the blind approximation, to simultaneously evaluate the score functions with a small error tolerance. Specifically, based on the refined normal approximation, we further simplify the evaluation formulas by replacing $\bar{P}(\tilde{\Gamma}_{\mathcal{J}}(x) = l)$ as $\bar{P}(\tilde{\Gamma}(x) = l)$:

$$\bar{w}_\tau^b(x) = 2 \sum_{s=1}^{\tau} \tilde{q}_j(x) \sum_{l \in \mathcal{L}(\epsilon)} \frac{\bar{P}(\tilde{\Gamma}(x) = l)}{\tau + l + \gamma + 1}, \quad \bar{v}_\tau^b(x) = \bar{v}_\tau(x) = \sum_{l \in \mathcal{L}(\epsilon)} \frac{\gamma \bar{P}(\tilde{\Gamma}(x) = l)}{\tau + l + \gamma},$$

where $\mathcal{L}(\epsilon) = \{l_U, \cdots, l_U\}$ is defined in (12). Then, for any $1 \leq d_0 \leq d$, $\bar{\omega}_{1:T}^b = (\bar{\omega}_1^b, \cdots, \bar{\omega}_T^b)^T$ and $\bar{v}_{1:T}^b = (\bar{v}_1^b, \cdots, \bar{v}_T^b)^T$ can be simultaneously computed over $\tau = 0, \cdots, d_0$ based on cross-correlation (flipped convolution (Tahmasebi et al., 2012)):

$$\bar{\omega}_{1:d_0}^b = 2\tilde{s}_{1:d_0}(x) \circ \left( \bar{P}(\tilde{\Gamma}(x) = l_L), \cdots, \bar{P}(\tilde{\Gamma}(x) = l_U) \right)^T \ast \left( \frac{1}{l_I + \gamma + 1}, \cdots, \frac{1}{l_U + \gamma + d_0 + 1} \right),$$

$$\bar{v}_{1:d_0}^b = \gamma \left( \bar{P}(\tilde{\Gamma}(x) = l_L), \cdots, \bar{P}(\tilde{\Gamma}(x) = l_U) \right)^T \ast \left( \frac{1}{l_I + \gamma}, \cdots, \frac{1}{l_U + \gamma + d_0} \right),$$

where $\tilde{s}(x) = (\tilde{s}_1(x), \cdots, \tilde{s}_d(x))^T$ and $\tilde{s}_K(x) = \sum_{i=1}^{K} \tilde{q}_i(x)$ is the cumulative sum of sorted estimated probabilities, $\circ$ is element-wise product of two vectors, and $\ast$ is the cross-correlation operator (flipped convolution) of two vectors (Tahmasebi et al. 2012). Notably, the cross-correlation can be efficiently implemented by a fast Fourier transform (Bracewell and Bracewell, 1986) with time complexity $O((d_0 + \tilde{\sigma}(x)) \log(d_0 + \tilde{\sigma}(x)))$. Besides, the overall time complexity with CUDA implementation on GPUs can be further reduced by parallel computing. The detailed algorithm is summarized in Algorithm 1 with approximate BA. Next, Lemma 5 shows the approximation error of the proposed blind approximation algorithm.
Algorithm 1: Computing schemes for the proposed RankDice framework.

Input: Training set: \((x_i, y_i)_{i=1}^{n}\); A new testing instance: \(x\); Approx method: \text{approx}

Output: The predicted segmentation for the testing instance \(\mathbf{\delta}(x)\)

1. **Conditional prob est.** Estimate conditional prob \(\hat{q}\) via (5) based on training set \((x_i, y_i)_{i=1}^{n}\);

2. **Ranking.** Rank estimated conditional probabilities for \(x\), and denote the indices in sorted order as \(j_1, \cdots, j_d\), that is, \(\hat{q}_{j_1}(x) \geq \hat{q}_{j_2}(x) \geq \cdots \geq \hat{q}_{j_d}(x)\);

3. **Volume estimation.**

4. Compute and store the cumulative sum of sorted estimated probabilities \(\hat{s}(x)\);

5. Compute \(d_0(x)\) based on \(\hat{s}(x)\) via (10);

6. **if approx is None then**

7. \(L = \{0, \cdots, d\}\);

8. **else**

9. \(L = L(\epsilon) = \{l_L, \cdots, l_U\}\) based on (12);

10. **end**

11. Compute and store \(P(\hat{\Gamma}(x) = l)\) for \(l \in L\);

12. **if approx is BA then**

13. Compute and store \(\tilde{\omega}_1^{b}(x)\) and \(\tilde{\nu}_1^{b}(x)\) based on (13);

14. **else**

15. Initialize \(\omega_{old} = 0\);

16. **for** \(\tau = 1, \cdots, d_0(x)\) **do**

17. **Update** \(\omega_{new}\) as

\[
\omega_{new,l} \leftarrow \omega_{old,l} + \hat{q}_{j_\tau}(x) P(\hat{\Gamma}_{j_\tau}(x) = l), \text{ for } l \in L, \quad \omega_{old} \leftarrow \omega_{new},
\]

where \(\hat{\Gamma}_{j_\tau}(x)\) is Poisson binomial r.v. with the success probabilities \(\hat{q}_{j_\tau}(x)\);

18. **Compute and store** \(\pi_\tau = \delta_\tau + \nu_\tau = \sum_{l \in L} \frac{1}{\tau+l+1} \omega_{new,l} + \sum_{l \in L} \frac{\gamma}{\tau+l+2} P(\hat{\Gamma}(x) = l)\);

19. **end**

20. **end**

21. Estimate \(\hat{\tau}(x) = \arg\max_{\tau=0, \cdots, d_0(x)} \pi_\tau\) via (8);

22. **Prediction.** The predicted segmentation is provided as:

\[
\hat{\delta}_j(x) = 1, \text{ if } j \in \{j_1, \cdots, j_{\hat{\tau}(x)}\}; \quad \hat{\delta}_j(x) = 0, \text{ otherwise}.
\]

**return** The predicted segmentation \(\hat{\mathbf{\delta}}(x) = (\hat{\delta}_1(x), \cdots, \hat{\delta}_d(x))^T\).
Lemma 5 For \((\overline{\beta}_b, \overline{\nu}_b)\) defined in (8) and (13) respectively, if \(\hat{\sigma}^2(x) \geq 25\), then for any \(\tau \in \{0, \ldots, d\}\), and any \(\gamma \geq 0\), we have

\[
|\overline{\beta}_b - \overline{\nu}_b| \leq \frac{4\tau}{\tau + \gamma + 1} (\epsilon + \frac{C_0}{\hat{\sigma}^2(x)} + \frac{C_0 \min(\hat{\mu}(x), \tau)}{\hat{\sigma}^2(x) - 1/4}) (\log(1 + d) + 1)
+ \frac{1}{4\sqrt{2\pi}} \left(\frac{1/2(\epsilon)}{\hat{\sigma}^2(x) - 1/4} + \frac{4}{\sqrt{\hat{\sigma}^2(x) - 1/4}} + \frac{4\hat{m}_3(x)}{((\hat{\sigma}^2(x) - 1/4)^{3/2})} \right) (\log(1 + d) + 1),
\]

where \(\hat{m}_3(x) = \sum_{j=1}^d \overline{\nu}_j(x)(1 - \overline{\nu}_j(x))(1 - 2\overline{\nu}_j(x))\), \(C_0 = 0.1618\) if \(\hat{\sigma}^2(x) \geq 100\), and \(C_0 = 0.3056\) if \(\hat{\sigma}^2(x) \geq 25\). Specifically, when \(d \to \infty\), we have

\[
|\overline{\beta}_b - \overline{\nu}_b| \leq \left(\frac{4\tau}{\tau + \gamma + 1} + \frac{2\gamma}{\tau + \gamma}\right) \epsilon + O\left(\left(\frac{\min(\hat{\mu}(x), \tau)}{\hat{\sigma}^2(x)} + \frac{1}{\hat{\sigma}(x)}\right) \log(d)\right).
\]

In contrast to the truncated refined normal approximation, the blind approximation algorithm significantly reduces the time complexity and enables GPU parallel execution. On the other hand, the blind approximation leads to an additional \(\hat{\sigma}^{-1}(x)\) in Lemma 5 compared with that of T-RNA in Lemma 4, yet the error bound is still acceptable in practice.

Taken together, we summarize the foregoing computational schemes in Algorithm 1, and their inference (after obtaining conditional probabilities) computational complexity (worst-case) is indicated in Table 1. For the threshold-based framework, thresholding the estimated probabilities takes \(O(d)\) time complexity. For the proposed method, Step 2 takes \(O(d \log(d))\) based on the merge sort (Ajtai et al., 1983), and Step 3 takes \(O(d_0(x) \hat{\sigma}(x))\) based on T-RNA in (10) and (11), and takes \(O\left(\min(d(0(x)) \log (d_0(x)))\right)\) based on BA in (13).

<table>
<thead>
<tr>
<th>SEG framework</th>
<th>Time</th>
<th>Time (well-separated)</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threshold-based SEG</td>
<td>(O(d))</td>
<td>(O(d))</td>
<td>(O(d))</td>
</tr>
<tr>
<td>RankDice (our)</td>
<td>(O(d \log(d) + d\log(d_0(x))))</td>
<td>(O(d \log(d)))</td>
<td>(O(d))</td>
</tr>
<tr>
<td>RankDice-RNA (our)</td>
<td>(O(d \log(d) + \hat{\sigma}(x) d_0(x)))</td>
<td>(O(d \log(d)))</td>
<td>(O(d))</td>
</tr>
<tr>
<td>RankDice-BA (our)</td>
<td>(O(d \log(d) + d_0(x) \log (d_0(x))))</td>
<td>(O(d \log(d)))</td>
<td>(O(d))</td>
</tr>
</tbody>
</table>

Table 1: Inference (prediction) computational complexity for the segmentation frameworks. Here “Time” denotes the time complexity, “Memory” denotes the amount of storage needed including probability outcomes, “Time (well-separated)” denotes the time complexity on well-separated segmentation \(d_0(x) \leq d\), and \(d_0(x) \leq d\) is the reduced dimension defined in (10), \(\hat{\sigma}(x)\) is the standard deviation of \(\hat{\Gamma}(x)\) with at most an order of \(O(\sqrt{d})\). Note that the time complexity of RankDice-BA can be further reduced by GPU implementation. See more detailed discussion in Section 2.3.

3. mDice-segmentation and mRankDice

In this section, we discuss the extension of the proposed RankDice framework to segmentation with multiclass-multilabel outcomes evaluated by the mDice metric. Overall, multiclass-multilabel segmentation differs from Dice-segmentation in a number of important aspects. First, a new evaluation metric called mDice is introduced. Second, multiclass-multilabel outcomes provide two different ways of probabilistic modeling. Third, whether (or not) to allow for overlapping segmentation results among different classes leads to different problem setups. These aspects will be discussed in detail in the following subsections.
3.1 The mDice metric

For multiclass/multilabel segmentation, \((\mathbf{X}, \mathbf{Y}_1, \cdots, \mathbf{Y}_K)\) is available, where \(\mathbf{X} \in \mathbb{R}^d\) represents a feature vector, \(K\) is the total number of classes of interest, \(\mathbf{Y}_k \in \{0, 1\}^d\) is the true feature-wise segmentation label for the \(k\)-th class, where \(Y_{jk} = 1\) indicates that the \(j\)-th feature \(X_j\) is segmented under the \(k\)-th class, and \(I(\mathbf{Y}_k) = \{ j: Y_{jk} = 1; \text{ for } j = 1, \cdots, d \}\) is the class-specific index set for all segmented features.

On this ground, a class-specific segmentation operator \(\mathbf{\Delta}_k: \mathbb{R}^d \rightarrow \{0, 1\}^d (k = 1, \cdots, K)\) is introduced, where \(\mathbf{\Delta}_k(\mathbf{x}) \in \{0, 1\}\) is the predicted segmentation for the \(k\)-th class, and \(I(\mathbf{\Delta}_k(\mathbf{x})) = \{ j: \Delta_{jk}(\mathbf{x}) = 1; \text{ for } j = 1, \cdots, d \}\) is the class-specific index set of features for the predicted segmentation. Then, given a segmentation operator \(\mathbf{\Delta} = (\mathbf{\Delta}_1, \cdots, \mathbf{\Delta}_K)\), the multi-Dice (mDice) metric is defined as:

\[
\text{mDice}_\gamma(\mathbf{\Delta}) = \sum_{k=1}^K \alpha_k \text{Dice}_{\gamma,k}(\mathbf{\Delta}_k)
\]

\[
= \sum_{k=1}^K \alpha_k \mathbb{E}\left( \frac{2|I(\mathbf{Y}_k) \cap I(\mathbf{\Delta}_k(\mathbf{x}))|}{|I(\mathbf{Y}_k)| + |I(\mathbf{\Delta}_k(\mathbf{x}))| + \gamma} \right) = \sum_{k=1}^K \alpha_k \mathbb{E}\left( \frac{2\mathbf{Y}_k^\top \mathbf{\Delta}_k(\mathbf{x}) + \gamma}{\|\mathbf{Y}_k\|_1 + \|\mathbf{\Delta}_k(\mathbf{x})\|_1 + \gamma} \right),
\]

where \(\text{Dice}_{\gamma,k}(\cdot)\) is the Dice metric under the \(k\)-th class, \(\mathbf{\alpha} = (\alpha_1, \cdots, \alpha_K)^\top \geq \mathbf{0}_K\) is a weight vector for classes with \(\|\mathbf{\alpha}\|_1 = 1\). For example, \(\alpha_k = 1/K\) yields that each class has the same contribution to segmentation performance. More generally, \(\mathbf{\alpha} \geq \mathbf{0}_K\) can be a custom weight vector indicating the relative importance of segmentation classes. In practice, given a new instance \((\mathbf{x}, \mathbf{y})\), the weight is an average over classes excluding those that are not present and not predicted, that is,

\[
\alpha_k = 0, \text{ if } \|\mathbf{y}_k\|_1 = \|\mathbf{\Delta}_k(\mathbf{x})\|_1 = 0; \quad \alpha_k = \frac{1}{\sum_{k=1}^K \mathbf{1}(\|\mathbf{y}_k\|_1 + \|\mathbf{\Delta}_k(\mathbf{x})\|_1 > 0)}, \text{ otherwise.}
\]

Following our convention, we shall call multiclass/multilabel segmentation with respect to the mDice metric as “mDice-segmentation”. As indicated in Figure 2, unlike Dice-segmentation, mDice-segmentation provides more flexibility in probabilistic modeling (multiclass or multilabel) and the decision-making in prediction (taking argmax or thresholding at 0.5), resulting in different operating losses and the construction of predictive functions.

3.2 Multilabel/multiclass outcomes

In this section, we describe two probabilistic models (multilabel or multiclass) of \(\mathbf{Y}_j|\mathbf{X}\), where \(\mathbf{Y}_j = (Y_{j1}, \cdots, Y_{jK})^\top\) is the true label for the \(j\)-th feature.

For multilabel modeling, we assume each feature could be assigned to multiple classes, that is, \(\|\mathbf{Y}_j\|_1 \in \{0, \cdots, K\}\), for \(j = 1, \cdots, d\). As a result, the index sets of segmented features may overlap among classes. In this case, we formulate and estimate \(q_{jk}(\mathbf{x})\) under a multilabel probabilistic model. For example, for deep learning models, we use the sigmoid function as the output layer activation function of a neural network with the binary cross-entropy loss.

For multiclass modeling, each feature is assigned to one and only one class, that is, \(\|\mathbf{Y}_j\|_1 = 1\) for \(j = 1, \cdots, d\). An instance is the panoptic annotation in image segmentation. In this case, we formulate and estimate \(\mathbf{q}_j(\mathbf{x}) = (q_{j1}(\mathbf{x}), \cdots, q_{jK}(\mathbf{x}))^\top\) under a multiclass model with additional sum-to-one constraints \(\|\mathbf{q}_j(\mathbf{x})\|_1 = 1\) for \(j = 1, \cdots, d\) and \(\mathbf{x} \in \mathbb{R}^d\). Specifically, for deep learning models, we use the softmax function as the output layer activation function of a neural network, which
automatically enforces the sum-to-one constraints. Correspondingly, a multiclass loss is used as an operating loss, including the multiclass cross-entropy. Note that the Dice-approximating losses, such as the soft-Dice loss, can be adopted into the multilabel/multiclass modeling.

In literature, once the estimation is done, the predicted segmentation is produced by taking \( \text{argmax} \) or \( \text{thresholding} \) at 0.5 on the estimated probabilities. Indeed, \( \text{argmax} \) and \( \text{thresholding} \) are inspired by the decision-making in multiclass and multilabel classification, respectively. In segmentation, it is possible to attempt ad-hoc combinations, such as multiclass modeling + \( \text{thresholding} \), and multilabel modeling + \( \text{argmax} \). The major purpose of \( \text{argmax} \) is to provide non-overlapping prediction (e.g., the panoptic prediction in image segmentation). We discuss overlapping/non-overlapping segmentation in the next section.

Figure 2: The existing and the proposed frameworks under multiclass/multilabel modeling. The upper panel is threshold-/argmax-based segmentation, and the lower panel is the proposed mRankDice framework.

3.3 Overlapping or non-overlapping mDice-segmentation

Specifically, whether (or not) to allow for overlapping results among distinct classes leads to different decision-making procedures, namely overlapping/non-overlapping mDice-segmentation:

\[
\begin{align*}
\text{(Overlapping) } \text{argmax}_{\Delta} \text{mDice}_{\gamma}(\Delta), & \quad \text{(Non-overlapping) } \text{argmax}_{\Delta} \text{mDice}_{\gamma}(\Delta), \\
& \quad \sum_{k=1}^{K} \Delta_k(\cdot) = 1_d.
\end{align*}
\] (16)

In the overlapping setting, there is no restriction on a segmentation operator, thus the predicted segmentation for different classes may overlap. On the other hand, the overlapping is ruled out in non-overlapping formulation due the additional sum-to-one constraint. Lemma 6 presents the Bayes rule for overlapping segmentation, yielding that mDice-segmentation is reduced to class-specific Dice-segmentation subproblems if overlapping is allowed.

**Lemma 6 (The Bayes rule for overlapping mDice-segmentation)** An overlapping (allowing) segmentation operator \( \Delta^* \) is a global maximizer of \( \text{mDice}_{\gamma}(\Delta) \) if and only if \( \Delta^*_k \) is the Bayes rule (global maximizer) of \( \text{Dice}_{\gamma,k}(\Delta_k) \) on \((X,Y,k)\) for all \( k \in \{1 \leq k \leq K : \alpha_k > 0\} \).
Therefore, it suffices to consider Dice-segmentation of each class separately in overlapping mDice-segmentation, and the proposed RankDice is readily extended to mRankDice; see Section 3.4.

Next, we investigate the Bayes rule for non-overlapping mDice-segmentation. To proceed, we denote $\Delta^*$ as the Bayes rule (global maximizer) of non-overlapping mDice-segmentation in (16), and $\tau^*(\cdot)$ as the volume function of the Bayes segmentation rule: $\tau^*(x) = (\tau^1(x), \ldots, \tau^K(x))^T = (\|\Delta_1^*(x)\|_1, \ldots, \|\Delta_K^*(x)\|_1)^T$.

**Lemma 7** Suppose $\tau^*(\cdot)$ is pre-known, then solving the Bayes rule for non-overlapping mDice-segmentation in (16) is equivalent to an assignment problem.

As indicated in Lemma 7, even when the optimal volume function is pre-given, solving the Bayes rule for non-overlapping segmentation is nontrivial. For example, the most successful assignment algorithms, including the Hungarian method (Kuhn, 1955; Edmonds and Karp, 1972. Tomizawa, 1971) and its variants Jonker-Volgenant algorithm (Crouse, 2016), generally achieves an $O(d^3)$ running time complexity in our content, which is time-consuming for high-dimensional segmentation. In sharp contrast, for the overlapping case, when $\tau^*(x)$ is given, the Bayes rule is simply ranking all the conditional probabilities with $O(d\log(d))$. Moreover, when $\tau^*(x)$ is unknown, the optimization for non-overlapping segmentation becomes a nonlinear integer optimization which is NP-hard in general (Murty and Kabadi, 1985. D’Ambrosio et al. 2020). Therefore, a fast $O(d\log(d))$ greedy approximation algorithm is more feasible in practical implementation. We leave pursuing this topic as future work.

Next, we summarize the proposed mRankDice framework under three scenarios.

### 3.4 mRankDice

In this section, we present the proposed mRankDice framework for mDice-segmentation. Before we proceed, we highlight the different roles of multiclass/multilabel modeling and the overlapping/non-overlapping constraint. Multiclass/multilabel modeling determines the probabilistic models (say the softmax or the sigmoid activation in a neural network) and an operating loss in probability estimation (say the cross-entropy or the binary cross-entropy). Meanwhile, the overlapping/non-overlapping constraint affects the segmentation prediction after the probabilities are estimated.

Therefore, we consider following segmentation three cases: “multilabel + overlapping”, “multiclass + overlapping”, and “multilabel/multiclass + non-overlapping”.

**mRankDice (multilabel + overlapping segmentation)** is a straightforward extension of RankDice in Dice-segmentation (inspired by Lemma 6). Given a training dataset $(x_1, y_{1,1,d,1}, \ldots, y_{1,1,d,K})$, with the same manner, the conditional probability function is estimated under a multilabel logistic regression (the binary cross-entropy loss):

$$
\hat{Q}(x) = \arg\min_{Q \in \mathcal{Q}} \sum_{i=1}^{n} \sum_{j=1}^{d} \sum_{k=1}^{K} \left( y_{ijk} \log \left( q_{jk}(x_i) \right) + (1 - y_{ijk}) \log \left( 1 - q_{jk}(x_i) \right) \right) + \lambda \|Q\|^2, \quad (17)
$$

where $Q = (q_{jk}) : \mathbb{R}^d \rightarrow [0, 1]^{d \times K}$ is a matrix function, and $q_{jk}(x)$ is a candidate estimator of $p_{jk}(x)$. Then, given a new instance $x$, the class-specific segmentation $\hat{\Delta}_k(x)$ is generated based on Steps 2-3 in Section 2.2 with the estimated conditional probabilities $\hat{Q}_k(x)$ (the $k$-th column of $\hat{Q}(x)$). We summarize the foregoing computational scheme in Algorithm 2.
**mRankDice (multiclass + overlapping segmentation)** yields a different probability estimation procedure, where the conditional probability function is estimated under a multiclass logistic regression (the multiclass cross-entropy loss):

\[
\hat{Q}(x) = \arg\min_{Q \in \mathcal{Q}} \sum_{i=1}^{n} \sum_{j=1}^{d} \sum_{k=1}^{K} v_{ijk} \log (q_{jk}(x_i)) + \lambda \|Q\|_2^2, \text{ s.t. } \sum_{k=1}^{K} q_{jk}(\cdot) = 1; \text{ for } j = 1, \cdots, d, \tag{18}
\]

where \( Q = (q_{jk}) : \mathbb{R}^d \rightarrow [0, 1]^{d \times K} \) is a matrix function, and \( q_{j}(x) \) is a candidate estimator of \( p_{j}(x) \). Although the probability estimation (18) differs from (17), the downstream ranking and volume estimation remain the same according to Lemma 6. We also summarize the foregoing computational scheme in Algorithm 2.

**Algorithm 2: mRankDice** for overlapping mDice-segmentation.

- **Input**: Training set: \((x_i, y_i, 1:d, \cdots, y_i, 1:d, K)_{i=1}^{n} \); A new testing instance: \(x\)
- **Output**: The predicted segmentation for the testing instance \(\hat{\Delta}(x)\)

1. if multilabel outcome then
2. Multilabel Prob Est. Estimate the prob function \(\hat{Q}\) via (17);
3. end
4. if multiclass outcome then
5. Multiclass Prob Est. Estimate the prob function \(\hat{Q}\) via (18);
6. end
7. for \(k = 1, \cdots, K\) do
8. Class-specific RankDice. Obtain \(\hat{\Delta}_k(x)\) from Lines 2-22 in Algorithm 1 based on the estimated prob \(\hat{Q}_k(x)\).
9. end
10. return The predicted segmentation \(\hat{\Delta}(x) = (\hat{\Delta}_1(x), \cdots, \hat{\Delta}_K(x))\)

**mRankDice (multiclass/multilabel + non-overlapping segmentation)** is a quite difficult scenario for developing \(mRankDice\) from \(RankDice\) in Dice-segmentation. As indicated in Lemma 7, searching for an optimal non-overlapping mDice-segmentation operator is NP-hard in general. We leave pursuing this topic as future work.

### 4. Theory

In this section, we establish a theoretical foundation of segmentation, including the concept of Dice-calibration, the excess risk of the Dice metric, and the rate of convergence with respect to the excess risk of the proposed \(RankDice\) framework. For illustration, we focus on Dice-segmentation, and the results can be extended to mDice-segmentation, and \(RankIoU\) in terms of the IoU metric.

#### 4.1 Dice-calibrated segmentation

In Theorem 1 the Bayes rule of Dice-segmentation is obtained. To carry this agenda further, we propose concept of “Dice-calibrated”, following the same philosophy of Fisher consistency or classification-calibration in Lin (2004); Zhang (2004); Bartlett et al. (2006). Intuitively, Dice-calibration is the weakest possible condition on a consistent segmentation method with respect to
the Dice metric, this is, at population level, a method ultimately searches for the Bayes rule that achieves the optimal Dice metric. Figure 3 indicates the overview and logical relations among the theoretic results in this section.

**Figure 3:** Flowchart of the theory for RankDice in Section 4.1 indicating the logical relations among the developed theorems.

To proceed, let $\mathcal{P}$ be the class of all probability measures $P_{XY}$ on (Borel) measurable subsets of $\mathbb{R}^d \times \{0, 1\}^d$ such that $(X, Y) \sim P_{XY}$, $(X, Y) \in \mathbb{R}^d \times \{0, 1\}^d$, and $Y_i \perp Y_j | X$ for $i \neq j$. Denote $\mathcal{D}$ as the class of all (Borel) measurable segmentation operators $\delta : x \in \mathbb{R}^d \rightarrow \delta(x) \in \{0, 1\}^d$. The definition of Dice-calibrated segmentation is given as follows.

**Definition 8 (Dice-calibrated segmentation)** Given $\gamma \geq 0$, a (population) segmentation method $M_\gamma : \mathcal{P} \rightarrow \mathcal{D}$ is Dice-calibrated if, for any $P_{XY} \in \mathcal{P}$,

$$\text{Dice}_\gamma(M_\gamma(P_{XY})) = \text{Dice}_\gamma(\delta^*),$$

where $\delta^*$ is the Bayes rule for Dice-segmentation defined in Theorem 1.

Now, we show that most existing loss functions, under the framework (2), are not Dice-calibrated.

**Lemma 9** Given a loss function $l(\cdot, \cdot)$, define $M_\gamma(P_{XY})$ under the framework (2), that is,

$$M_\gamma(P_{XY})(x) := 1(\tilde{q}_l(x) \geq 0.5), \quad \tilde{q}_l = \arg\min_q \mathbb{E}(l(Y, q(X))).$$

Then, $M_\gamma(P_{XY})$ is not Dice-calibrated for $\gamma = 0$ when $l(\cdot, \cdot)$ is any classification-calibrated loss, including the cross-entropy loss and the focal loss.

Lemma 9 indicates that the existing framework (2) with most losses ultimately yields a suboptimal solution to Dice-segmentation, even if a “classification-calibrated” loss, such as the cross-entropy loss or the focal loss (Charoenphakdee et al., 2021), is used. Meanwhile, as indicated in Bertels et al. (2019), the optimization with the soft-Dice loss can introduce a volumetric bias for tasks with high inherent uncertainty. In sharp contrast, the proposed RankDice method is Dice-calibrated (Lemma 10) and its asymptotic convergence rate in terms of the Dice metric is provided in Theorem 11.

To proceed, we give the definition of RankDice at population level in Appendix B.1, which replaces the average in (5) by the population expectation. Moreover, the cross-entropy loss in (5) can be extended to an arbitrary strictly proper loss (Gneiting and Raftery. 2007). The most common strictly proper losses are the cross-entropy loss and the squared error loss.
Lemma 10 (Dice-calibrated) The proposed RankDice framework with a strictly proper loss is Dice-calibrated.

Next, we present an excess risk bound in terms of the Dice metric, that is, \( \text{Dice}(\delta^*) - \text{Dice}(\hat{\delta}) \).

Theorem 11 (Excess risk bounds) Given \( \gamma \geq 0 \), let \( \hat{\mathbf{q}}(\cdot) \) be an estimated probability of \( p(\cdot) \), and \( \hat{\delta}(\cdot) \) be the RankDice segmentation function defined in (7) based on \( \hat{\mathbf{q}}(\cdot) \), then

\[
\text{Dice}_\gamma(\delta^*) - \text{Dice}_\gamma(\hat{\delta}) \leq C_1 \mathbb{E}_X \| \hat{\mathbf{q}}(X) - p(X) \|_1,
\]

where \( C_1 > 0 \) is a universal constant depending only on \( \gamma \).

As indicated in Theorem 11, the excess risk of the Dice metric for the proposed RankDice framework is upper bounded by the total variation (TV) distance between the estimated probability \( \hat{\mathbf{q}} \) and the true probability \( p \). Note that the Kullback-Leibler divergence (the excess risk for the cross-entropy) dominates the TV distance. It follows that if the KL divergence between \( p_j \) and \( \hat{q}_j \) goes to 0, then \( \hat{\mathbf{q}} \) converges to \( p \) in the TV sense, and so does \( \text{Dice}_\gamma(\hat{\delta}) \) to \( \text{Dice}_\gamma(\delta^*) \).

Taken together, we present the rate of convergence for the empirical estimator obtained from the proposed RankDice framework (Steps 1-3) in Section 2.2.

Corollary 12 (Convergence rate) Let \( \hat{\mathbf{q}}(\cdot) \) and \( \hat{\delta}(\cdot) \) be obtained by the proposed RankDice framework (Steps 1-3) in Section 2.2 and

\[
\mathcal{E}_{CE}(\hat{\mathbf{q}}) := \mathbb{E}\left( l_{CE}(Y, \hat{\mathbf{q}}(X)) \right) - \mathbb{E}\left( l_{CE}(Y, p(X)) \right) = O_P(\epsilon_n),
\]

where \( l_{CE}(\cdot, \cdot) \) is defined in (3). Then,

\[
\text{Dice}_\gamma(\delta^*) - \text{Dice}_\gamma(\hat{\delta}) = O_P(\sqrt{\epsilon_n}^{1/2}).
\]

Note that \( \mathcal{E}_{CE} \) is the excess risk of the cross-entropy loss or the negative conditional log-likelihood in (5), and its asymptotics as well as a rate of convergence can be established based on statistical learning theory of empirical risk minimization (Pollard, 1984; Shen, 1997; Bartlett et al., 2005; Giné and Koltchinskii, 2006; Cucker and Zhou, 2007), which depends on the sample size and the complexity of the probability class. Then, the rate of convergence of the excess risk in terms of the Dice metric is obtained via (20). Note that both (19) and (20) are derived for a fixed dimension, and the upper bounds can be extended and improved when the dimension of segmentation grows with the sample size.

Finally, we briefly discuss the connections of the developed theory (i.e., Lemma 10, Theorem 11 and Corollary 12) with the existing results. For example, Popordanoska et al. (2021) derived an upper bound for the volume bias \( \| \mathbb{E}_X (\hat{\mathbf{q}}(X) - p(X)) \|_1 \) in terms of the TV distance. It is worth noting that the volume bias focuses on conditional probability estimation and a small volume bias may not necessarily yield a consistent segmentation rule in terms of the Dice metric. In contrast, our result on the excess risk \( \text{Dice}(\delta^*) - \text{Dice}(\hat{\delta}) \) characterizes the performance of segmentation rule \( \hat{\delta} \). Besides, Bao and Sugiyama (2020) proved the consistency of their method under a linear fractional approximation of Dice metric (see Appendix A1), which seems not directly comparable to ours.
4.2 Relation between Dice and IoU metrics

In this section, we consider the relation and difference between Dice and IoU metrics, and present the Bayes rule for IoU-segmentation.

**Lemma 13** A segmentation rule $\mathbf{\delta}^*$ is a global maximizer of $\text{IoU}_\gamma(\mathbf{\delta})$ if and only if it satisfies that

$$
\delta_j^*(x) = \begin{cases} 
1 & \text{if } p_j(x) \text{ ranks top } \tau^*(x), \\
0 & \text{otherwise.}
\end{cases}
$$

The optimal volume $\tau^*(x)$ is given as

$$
\tau^*(x) = \arg\max_{\tau \in \{0, 1, \ldots, d\}} \left( \sum_{j \in J_\tau(x)} p_j(x) + \gamma \right) \sum_{l=0}^{d-\tau} \frac{1}{\tau + l + \gamma} \mathbb{P}(\Gamma_{J_\tau(x)}(x) = l),
$$

where $J_\tau(x) = \{ j : \sum_{j=1}^d \mathbb{I}(p_j(x) \geq p_j(x)) \leq \tau \}$ is the index set of the $\tau$-largest conditional probabilities with $J_0(x) = \emptyset$, and $\Gamma_{J_\tau(x)}(x) = \sum_{j \notin J_\tau(x)} B_j(x)$ is a Poisson-binomial random variable, and $B_j(x)$ is a Bernoulli random variable with the success probability $p_j(x)$.

In view of Lemma 13, IoU-segmentation shares a substantial similarity with Dice-segmentation in terms of the Bayes rule. On this ground, a consistent RankIoU framework is also developed based on a plug-in rule by replacing $p(x)$ as $\hat{q}(x)$. Specifically, RankIoU comprises three steps, where Steps 1-2 are the same as in RankDice; see Section 2.2.

**Step 3’ (IoU volume estimation):** From (4), we estimate the volume $\hat{\tau}(x)$ by replacing the true conditional probability $p(x)$ with the estimated one $\hat{q}(x)$:

$$
\hat{\tau}(x) = \arg\max_{\tau \in \{0, 1, \ldots, d\}} \left( \sum_{j \in J_{\hat{\tau}}(x)} \hat{q}_j(x) + \gamma \right) \sum_{l=0}^{d-\tau} \frac{1}{\tau + l + \gamma} \mathbb{P}(\Gamma_{J_{\hat{\tau}}(x)}(x) = l),
$$

where $\hat{\Gamma}_{J_{\hat{\tau}}(x)}(x) = \sum_{j \notin J_{\hat{\tau}}(x)} \hat{B}_j(x)$ is a Poisson-binomial random variable, and $\hat{B}_j(x)$ are independent Bernoulli random variables with success probabilities $\hat{q}_j(x)$; for $j = 1, \ldots, d$.

Similar to RankDice, the predicted IoU-segmentation $\hat{\mathbf{\delta}}(x) = (\hat{\delta}_1(x), \cdots, \hat{\delta}_d(x))^\top$ is produced by taking the top-$\hat{\tau}(x)$ conditional probabilities:

$$
\hat{\delta}_j(x) = 1, \text{ if } j \in \{ j_1, \cdots, j_{\hat{\tau}(x)} \}; \quad \hat{\delta}_j(x) = 0, \text{ otherwise.}
$$

For multiclass/multilabel segmentation, the conditional probability estimation (17) and (18) are carried over into $m$RankIoU and the subsequent ranking and volume estimation remain the same as Step 2 and Step 3’ in binary segmentation.

Computationally, RankIoU involves the evaluation of $\mathbb{P}(\hat{\Gamma}_{J_{\hat{\tau}}(x)}(x) = l)$ in Step 3’. The FFT algorithm and the truncated refined normal approximation (T-RNA) are applicable after minor modifications; however, the blind approximation (BA) may not be appropriate due to the discrepancy of $\hat{\Gamma}(x)$ and $\hat{\Gamma}_{J_{\hat{\tau}}(x)}(x)$, especially when the size of $J_{\tau}(x)$ is large; see Section 2.3. Thus, the computation scheme of RankIoU might be relatively expensive in high-dimensional segmentation. Here, we present a parallel result of Lemma 3 to narrow down the searching range in Step 3’ of RankIoU.
Lemma 14  If

\[ \sum_{s=1}^{\tau} \hat{q}_s(x) + \gamma \geq \frac{\hat{q}_{j+1}(x)}{1 - \hat{q}_{j+1}(x)} \max \left( d + \gamma, \frac{(d - \tau)\hat{q}_{j+1}(x) + \tau + \gamma}{\tau + \gamma} \right), \]

then \( \sigma_{\tau}(x) \geq \sigma_{\tau'}(x) \) for all \( \tau' > \tau \), where

\[ \sigma_{\tau}(x) = \left( \sum_{j \in J_{\tau}(x)} \hat{q}_j(x) + \gamma \right) \sum_{l=0}^{d-\tau} \frac{1}{\tau + l + \gamma} \mathbb{P}(\hat{\Gamma} - J_{\tau}(x)(x) = l). \]

Theoretically, the concept of “IoU-calibrated” can be established (by replacing Dice as IoU in Definition 8) and the excess risk bounds can be derived in parallel to Dice-segmentation.

Theorem 15 Given \( \gamma \geq 0 \), let \( \hat{q}(\cdot) \) be an estimated probability of \( p(\cdot) \), and \( \hat{\delta}(\cdot) \) be the RankIoU segmentation function based on \( \hat{q}(\cdot) \), then

\[ \text{IoU}_{\gamma}(\delta^*) - \text{IoU}_{\gamma}(\hat{\delta}) \leq C_2 \mathbb{E}_X \| \hat{q}(X) - p(X) \|_1, \]

where \( C_2 > 0 \) is a universal constant depending only on \( \gamma \). Consequently, if \( \mathcal{E}_{CE}(\hat{q}) = O_P(\epsilon_n) \), then

\[ \text{IoU}_{\gamma}(\delta^*) - \text{IoU}_{\gamma}(\hat{\delta}) = O_P(\sqrt{d\epsilon_n^{1/2}}). \]

5. Numerical experiments

This section describes a set of simulations and real datasets that demonstrate the segmentation performance of the proposed RankDice and mRankDice frameworks compared with the existing argmax- and thresholding-based frameworks using various loss functions and network architectures. For illustration, the segmentation performances for all numerical experiments are evaluated by empirical Dice/IoU metrics with \( \gamma = 0 \), see Appendix A. For the mDice/mIoU metric, the class-specific weight is defined as in (15). All experiments are conducted using PyTorch and CUDA on an NVIDIA GeForce RTX 3080 GPU. All Python codes are available for download at our GitHub repository (https://github.com/statmlben/rankseg).

5.1 Simulation

In this section, we mainly compare the proposed RankDice framework with the thresholding-based framework (2) in various simulated examples. Note that for Dice-segmentation with binary outcomes, threshold- and argmax-based frameworks yield the same solution. Both frameworks require an estimation of conditional probability function in the first stage. Therefore, in order to convincingly demonstrate the difference between two frameworks, in our simulation, we assume the true conditional probabilities \( p_j(x) = \mathbb{P}(Y_j|x); j = 1, \ldots, d \) are perfectly estimated, and report the Dice metric of the downstream segmentation produced by two different frameworks.

Example 1. To mimic the spatial smoothness in practical segmentation problem, especially for image segmentation, the simulated dataset based on matrix response \( d = W \times H \) is generated as follows. First, the true probability matrix \( P = (p_{wh})_{W \times H} \) are generated by two patterns:

- Step decay: \( p_{wh} \sim U(0.5, 1) \), if \( w \leq \lfloor \rho W \rfloor \) and \( h \leq \lfloor \rho H \rfloor \); \( p_{wh} \sim N_{[0,1]}(\beta, 0.1) \), otherwise.
Exponential decay: $p_{wh} = \exp(-\beta(w+h))$, as visualized in the upper panel of Figure 6.

Linear decay: $p_{wh} = 1 - \beta(w+h)/(W+H)$, as visualized in the lower panel of Figure 6.

Here $\beta$ is a decay parameter, $U(0,1)$ is the uniform distribution, and $N_{[0,1]}(\beta,0.1)$ is the truncated normal distribution with mean $\beta$ and standard deviation 0.1. In our simulation, we consider $\beta = 0.1,0.3,0.5$ with $\rho = 0.1$ for step decay, $\beta = 1.01,1.05,1.10$ for exponential decay, and $\beta = 1.2,4$ for linear decay. For each case, four different dimensions are considered: $W = H = 28,64,128,256$, and therefore $d$ increases from 784 to 65,536. Then, the proposed RankDice framework and the thresholding-based framework (at 0.5) are conducted on the true probability matrix $P$. Both decay scenarios are replicated 100 times, and the averaged Dice metrics and its standard errors are summarized in Table 9.

**Example 2.** As indicated in Theorem 1 and Remark 2 the optimal segmentation volume (or the optimal threshold) varies significantly across different inputs. This example aims to illustrate the suboptimality of (tuning a threshold of) a fixed thresholding based on step decay in Example 1. To this end, the conditional probability matrices $(P_i)_{i=1,...,n}$ of inputs are generated as follows. First, $\rho_i \sim U(0,1)$ to mimic the different segmentation scales/patterns over images in real applications. Next, $P_i$ is generated by step decay based on $\beta = 0.1$ and $\rho = \rho_i$. Then, the proposed RankDice framework and the fixed thresholding-based framework (at 0.1, ..., 0.9) are applied as the same manner in Example 1. All methods are applied with $n = 2000$ and $W = H = 64$, and the averaged Dice metrics and its standard errors are summarized in Table 10. In this example, the heterogeneous conditional probabilities yield different optimal segmentation volumes (or thresholds) for images, thus (tuning a threshold on) a fixed thresholding leads to a suboptimal solution. To better illustrate the adaptiveness over optimal thresholds, we also present the optimal thresholds for different images with various generating parameters $(\beta,\rho)$ on segmentation patterns of Example 2 in Figure 7, and similar results are demonstrated in Figure 4 for Pascal VOC 2012 dataset.

The major conclusions on the simulated examples are listed as follows.

- It is evident that RankDice significantly outperforms the thresholding-based (at 0.5) framework in both decay scenarios with various dimensions (Table 9). The substantial improvement is consistent with the findings of Lemma 1, which indicates that segmentation and classification are entirely distinct problems.

- Interestingly, the amount of improvement is gradually increased when the decay of probabilities becomes progressively faster (for exponential decay and linear decay). This suggests that the proposed RankDice might be even more advantageous in well-separated segmentation.

- As suggested in Table 10, the proposed RankDice generally outperforms a fixed thresholding framework (with any threshold). This because that the optimal threshold can vary greatly across inputs, as indicated in Figure 7. Moreover, tuning the threshold may improve the performance of the thresholding-based framework, yet it still leads to a suboptimal solution compared with the proposed RankDice.

**5.2 Real datasets**

This section examines the performance of the proposed RankDice framework in the PASCAL VOC 2012 (Everingham et al., 2012), the fine-annotated CityScapes semantic segmentation benchmark.
Figure 4: The heatmap for averaged (over all validation samples) optimal thresholding probabilities (i.e. the probability cutpoint $q_j$) provided by the proposed RankDice, see Remark 2 for more details. Here, the x-axis indicates the class, and the results are provided by a PSPNet trained with the cross-entropy loss.

(Cordts et al. 2016), and Kvasir SEG dataset polyp segmentation dataset (Ajtai et al. 1983). Three different neural network architectures are considered: DeepLab-V3+ (Chen et al. 2018) with resnet101 as the backbone, PSPNet with resnet50 as the backbone, and FCN8 with resnet101 as the backbone. We report both mDice and mIoU metrics of the segmentation produced by Threshold, Argmax and RankDice on the same estimated network/probability. Note that only overlapping segmentation is considered.

Fine-annotated Cityscapes dataset contains 5,000 high quality pixel-level annotated images. For all methods, we employ SGD on the learning rate (lr) schedule lr_schedule='poly', and the initial learning rate initial lr=0.01, weight decay=100, momentum=0.9, crop size 512x512, batch size 6, and 300 epochs. The performance on validation set is measured in terms of the mDice and mIoU averaged across 19 object classes (Table 2).

Pascal VOC 2012 dataset contains 20 foreground object classes and one background class. The dataset contains 1,464 training and 1,449 validation pixel-level annotated images. We augment the dataset by using the additional annotations provided by Hariharan et al. (2011). For all methods, we employ SGD on lr_schedule='poly', and the initial learning rate initial lr=0.01, weight decay=100, momentum=0.9, crop size 480x480, batch size 8, and an early stop with patient 10 based on validation loss. The performance on validation set is measured in terms of the mDice and mIoU averaged across the 20 object classes (Table 3). In this dataset, we also present a heatmap (Figure 4) for averaged minimal estimated probabilities for segmented features (i.e. $q_j$) by the proposed RankDice, to highlight its difference to the thresholding-based framework.

Kvasir SEG dataset contains 1000 polyp images and their ground truth segmentation (a single class) from the Kvasir dataset. The scale of the images varies from 332x487 to 1920x1072 pixels. For all methods, we employ SGD on lr_schedule='poly', and the initial learning rate initial lr=0.01, weight decay=100, momentum=0.9, crop size 320x320, batch size 8, and 140 epochs. The performance on testing set is measured in terms of the Dice and IoU (Table 4).

Multiclass/multilabel loss. In both datasets, we use six loss functions (including multiclass and multilabel losses) in the implementation, including the cross-entropy (CE), the focal loss (Focal), the binary cross-entropy (BCE), the soft-Dice loss (Soft-Dice), the binary soft-Dice loss (B-Soft-Dice), and the LovaszSoftmax loss (LovaszSoftmax). For multiclass losses, including CE, Focal, Soft-Dice, and LovaszSoftmax, we use the softmax function as the output layer activation function of a neural network. For multilabel losses, including BCE and B-Soft-Dice, we use the sigmoid function as the output layer activation function of a neural network.

Dice-segmentation based on a single class. For the first two datasets, when we focus on a single object class, it reduces to a Dice-segmentation with binary outcomes. To examine the performance for the proposed RankDice in binary segmentation, we also report the Dice/IoU metric for each label separately. In principle, we need to train a model only on the binary label for an object class, then produce the segmentation prediction by thresholding (or argmax) and RankDice. However, based
on our empirical study, a model trained from full labels is significantly better than the one trained from a single binary label. We thus train a model based the same procedure in the aforementioned segmentation with multiclass/multilabel losses on full labels, and then produce the prediction based on the estimated probability for each object class separately. The best two performances (“PSPNet + CE” and “PSPNet + BCE”) for both datasets are summarized in Tables 5 and 6. The performance for other models and losses can be found in the supplementary. In Figure 5, we present the segmentation results on illustrative examples for all methods to demonstrate the difference between the proposed RankDice and the existing methods.

**The fixed-thresholding framework with different thresholds.** As indicated in Remark 2 and Example 2 in Section 5.1, the optimal segmentation volume (or the optimal thresholding) varies significantly across different images, thus (tuning on) a fixed-thresholding yields a suboptimal solution. In Tables 7 and 10, we also report the numerical performance based on different thresholds for real datasets and Example 2, respectively.

**Probability calibration via Temperature scaling (TS).** According to Theorem 11, the consistency of the proposed method holds if the estimated conditional probabilities are calibrated. Alternatively, improving the probability calibration may improve the segmentation performance for the proposed framework. Therefore, in Table 8, we examine the numerical results of the proposed method via TS (with different tuning temperatures), which is one of the most effective probability calibration methods as suggested by Guo et al. (2017).

<table>
<thead>
<tr>
<th>Model</th>
<th>Loss</th>
<th>Threshold (at 0.5)</th>
<th>Argmax</th>
<th>mRankDice (our)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(mDice, mIoU)</td>
<td>(mDice, mIoU)</td>
<td>(mDice, mIoU)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(×.01)</td>
<td>(×.01)</td>
<td>(×.01)</td>
</tr>
<tr>
<td>DeepLab-V3+</td>
<td>CE</td>
<td>(56.00, 48.40)</td>
<td>(54.20, 46.60)</td>
<td>(57.80, 49.80)</td>
</tr>
<tr>
<td>(resnet101)</td>
<td>Focal</td>
<td>(54.10, 46.60)</td>
<td>(53.30, 45.60)</td>
<td>(56.50, 48.70)</td>
</tr>
<tr>
<td></td>
<td>BCE</td>
<td>(49.80, 24.90)</td>
<td>(44.20, 22.10)</td>
<td>(54.00, 27.00)</td>
</tr>
<tr>
<td></td>
<td>Soft-Dice</td>
<td>(39.50, 35.90)</td>
<td>(39.50, 35.90)</td>
<td>(39.50, 35.90)</td>
</tr>
<tr>
<td></td>
<td>B-Soft-Dice</td>
<td>(41.00, 20.50)</td>
<td>(27.60, 13.80)</td>
<td>(41.10, 20.50)</td>
</tr>
<tr>
<td></td>
<td>LovaszSoftmax</td>
<td>(55.20, 47.60)</td>
<td>(52.30, 45.10)</td>
<td>(55.50, 47.80)</td>
</tr>
<tr>
<td>PSPNet</td>
<td>CE</td>
<td>(57.50, 49.60)</td>
<td>(56.50, 48.50)</td>
<td>(59.30, 51.00)</td>
</tr>
<tr>
<td>(resnet50)</td>
<td>Focal</td>
<td>(56.00, 48.20)</td>
<td>(55.80, 47.70)</td>
<td>(58.20, 50.00)</td>
</tr>
<tr>
<td></td>
<td>BCE</td>
<td>(51.40, 25.70)</td>
<td>(47.60, 23.80)</td>
<td>(55.10, 27.60)</td>
</tr>
<tr>
<td></td>
<td>Soft-Dice</td>
<td>(49.10, 43.50)</td>
<td>(48.70, 43.20)</td>
<td>(49.30, 43.60)</td>
</tr>
<tr>
<td></td>
<td>B-Soft-Dice</td>
<td>(46.30, 23.10)</td>
<td>(32.70, 16.40)</td>
<td>(46.20, 23.10)</td>
</tr>
<tr>
<td></td>
<td>LovaszSoftmax</td>
<td>(56.80, 48.90)</td>
<td>(55.40, 47.70)</td>
<td>(56.70, 49.10)</td>
</tr>
<tr>
<td>FCN8</td>
<td>CE</td>
<td>(51.40, 43.70)</td>
<td>(50.50, 42.60)</td>
<td>(53.50, 45.30)</td>
</tr>
<tr>
<td>(resnet101)</td>
<td>Focal</td>
<td>(48.50, 41.20)</td>
<td>(49.60, 41.60)</td>
<td>(51.00, 43.70)</td>
</tr>
<tr>
<td></td>
<td>BCE</td>
<td>(39.40, 19.70)</td>
<td>(39.40, 19.70)</td>
<td>(41.30, 20.60)</td>
</tr>
<tr>
<td></td>
<td>B-Soft-Dice</td>
<td>(29.10, 14.60)</td>
<td>(29.10, 14.60)</td>
<td>(29.10, 14.60)</td>
</tr>
<tr>
<td></td>
<td>LovaszSoftmax</td>
<td>(48.10, 40.40)</td>
<td>(42.90, 35.80)</td>
<td>(48.90, 40.90)</td>
</tr>
</tbody>
</table>

Table 2: Averaged mDice and mIoU metrics of Threshold, Argmax, and the proposed mRankDice based on state-of-the-art models/losses on Fine-annotated CityScapes val set. Gray color indicates that RankDice/mRankDice is inappropriately applied to a loss function which is not strictly proper. The best performance in each model is bold-faced.

Overall, the empirical results show that the proposed RankDice/mRankDice framework yields good performance in three segmentation benchmarks. The major empirical conclusions on the proposed RankDice are listed as follows.
Table 3: Averaged mDice and mIoU of threshold, argmax, and the proposed mRankDice based on state-of-the-art models/losses on PASCAL VOC 2012 val set. “—” indicates that either the performance is significantly worse or the training is unstable. Gray color indicates that RankDice/mRankDice is inappropriately applied to a loss function which is not strictly proper. The best performance in each model is bold-faced.

<table>
<thead>
<tr>
<th>Model</th>
<th>Loss</th>
<th>Threshold (at 0.5) (mDice, mIoU) (×.01)</th>
<th>Argmax (mDice, mIoU) (×.01)</th>
<th>mRankDice (our) (mDice, mIoU) (×.01)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepLab-V3+</td>
<td>CE</td>
<td>(63.60, 56.70)</td>
<td>(61.90, 55.30)</td>
<td>(64.01, 57.01)</td>
</tr>
<tr>
<td>(resnet101)</td>
<td>Focal</td>
<td>(62.70, 55.01)</td>
<td>(60.50, 53.20)</td>
<td>(62.90, 55.10)</td>
</tr>
<tr>
<td>BCE</td>
<td></td>
<td>(63.30, 31.70)</td>
<td>(59.90, 29.90)</td>
<td>(64.60, 32.30)</td>
</tr>
<tr>
<td>Soft-Dice</td>
<td></td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>B-Soft-Dice</td>
<td></td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>LovaszSoftmax</td>
<td></td>
<td>(57.70, 51.60)</td>
<td>(56.20, 50.30)</td>
<td>(57.80, 51.60)</td>
</tr>
<tr>
<td>PSPNet</td>
<td>CE</td>
<td>(64.60, 57.10)</td>
<td>(63.20, 55.90)</td>
<td>(65.40, 57.80)</td>
</tr>
<tr>
<td>(resnet50)</td>
<td>Focal</td>
<td>(64.00, 56.10)</td>
<td>(63.90, 56.10)</td>
<td>(66.60, 58.50)</td>
</tr>
<tr>
<td>BCE</td>
<td></td>
<td>(64.20, 32.10)</td>
<td>(65.20, 32.60)</td>
<td>(67.10, 33.50)</td>
</tr>
<tr>
<td>Soft-Dice</td>
<td></td>
<td>(59.60, 54.00)</td>
<td>(58.80, 53.20)</td>
<td>(60.00, 54.30)</td>
</tr>
<tr>
<td>B-Soft-Dice</td>
<td></td>
<td>(63.30, 31.60)</td>
<td>(54.00, 27.00)</td>
<td>(64.30, 32.20)</td>
</tr>
<tr>
<td>LovaszSoftmax</td>
<td></td>
<td>(62.00, 55.20)</td>
<td>(60.80, 54.10)</td>
<td>(62.20, 55.40)</td>
</tr>
<tr>
<td>FCN8</td>
<td>CE</td>
<td>(49.50, 41.90)</td>
<td>(45.30, 38.40)</td>
<td>(50.40, 42.70)</td>
</tr>
<tr>
<td>(resnet101)</td>
<td>Focal</td>
<td>(50.40, 41.80)</td>
<td>(47.20, 39.30)</td>
<td>(51.50, 42.50)</td>
</tr>
<tr>
<td>BCE</td>
<td></td>
<td>(46.20, 23.10)</td>
<td>(44.20, 22.10)</td>
<td>(47.70, 23.80)</td>
</tr>
<tr>
<td>Soft-Dice</td>
<td></td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>B-Soft-Dice</td>
<td></td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>LovaszSoftmax</td>
<td></td>
<td>(39.80, 34.30)</td>
<td>(37.30, 32.20)</td>
<td>(40.00, 34.40)</td>
</tr>
</tbody>
</table>

Table 4: Averaged mDice and mIoU of threshold/argmax, and the proposed mRankDice based on state-of-the-art models/losses on Kvasir SEG dataset (with a single class segmentation). “—” indicates that either the performance is significantly worse. Gray color indicates that RankDice/mRankDice is inappropriately applied to a loss function which is not strictly proper. The best performance in each model is bold-faced.

<table>
<thead>
<tr>
<th>Model</th>
<th>Loss</th>
<th>Threshold/Argmax (Dice, IoU) (×.01)</th>
<th>mRankDice (our) (Dice, IoU) (×.01)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepLab-V3+</td>
<td>CE</td>
<td>(87.9, 80.7)</td>
<td>(88.3, 80.9)</td>
</tr>
<tr>
<td>(resnet101)</td>
<td>Focal</td>
<td>(86.5, 87.3)</td>
<td>(83.1, 73.2)</td>
</tr>
<tr>
<td>BCE</td>
<td></td>
<td>(85.7, 77.8)</td>
<td>(85.8, 77.9)</td>
</tr>
<tr>
<td>LovaszSoftmax</td>
<td></td>
<td>(84.3, 77.3)</td>
<td>(84.5, 77.4)</td>
</tr>
<tr>
<td>PSPNet</td>
<td>CE</td>
<td>(86.3, 79.2)</td>
<td>(87.1, 79.8)</td>
</tr>
<tr>
<td>(resnet50)</td>
<td>Focal</td>
<td>(83.8, 75.4)</td>
<td>(81.8, 72.4)</td>
</tr>
<tr>
<td>BCE</td>
<td></td>
<td>(83.5, 75.9)</td>
<td>(83.7, 76.1)</td>
</tr>
<tr>
<td>LovaszSoftmax</td>
<td></td>
<td>(86.0, 79.2)</td>
<td>(86.0, 79.2)</td>
</tr>
<tr>
<td>FCN8</td>
<td>CE</td>
<td>(81.9, 73.5)</td>
<td>(82.1, 73.6)</td>
</tr>
<tr>
<td>(resnet101)</td>
<td>Focal</td>
<td>(78.5, 69.0)</td>
<td>(70.3, 58.3)</td>
</tr>
<tr>
<td>BCE</td>
<td></td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>LovaszSoftmax</td>
<td></td>
<td>(82.0, 73.4)</td>
<td>(82.0, 73.4)</td>
</tr>
</tbody>
</table>

- As suggested in Tables 2 and 3, the proposed RankDice framework consistently outperforms the threshold-based and argmax-based frameworks based on the same trained model/network. The percentage of improvement on the best performance (for each framework) are 3.13% (over threshold) and 4.96% (over argmax) for CityScapes dataset (PSPNet + CE), and 3.87% (over threshold) and 2.91% (over argmax) for Pascal VOC 2012 dataset (PSPNet + CE/BCE), and 0.926% for Kvasir SEG dataset (PSPNet + CE).
For Dice-segmentation based on a single class, as suggested in Tables 5 and 6, the proposed RankDice framework consistently outperforms the threshold-argmax-based framework for each class. The largest percentage of class-specific improvement in terms of the Dice metric on the best performance (for each framework) is 23.6% for CityScapes dataset, and 26.9% for Pascal VOC 2021 dataset.

The proposed RankDice works significantly better in “difficult” segmentation. As indicated in Tables 5 and 6, the improvements by RankDice are negatively correlated with the resulting Dice/IoU metrics. It is also suggested by Figure 5, where we illustrate three images from classes cat (no improvement) and chair (26.9% improvement). As presented in all examples of chair and the last example of cat, the improvement is significant when segmentation is difficult (the target object is either occluded or similar with other objects).

As suggested in Table 7, the empirical results are in line with Remark 2 (theoretically) and Table 10 (numerically), suggesting that the proposed RankDice generally outperforms a fixed thresholding framework (with any threshold). This because that the optimal threshold can vary greatly across images, as indicated in Figure 4. Moreover, tuning the threshold may improve the performance of the fixed-thresholding framework, yet it still leads to a suboptimal solution compared with the proposed RankDice.

As suggested in Table 8, the segmentation performance can be potentially improved by TS probability calibration method, especially 4.59% improvement for CE loss and 3.28% improvement for BCE in Pascal VOC 2021 dataset. Yet, the TS demands an additional validation dataset to tune the optimal temperature, thus more numerical experiments are required to suggest the effectiveness of this promising method. We leave pursuing this topic as future work.

As expected, the proposed RankDice/mRankDice performs well for strictly proper loss functions, including CE and BCE. In addition, we show that the performance is continuously improved compared with existing frameworks for some classification calibrated (only) losses, such as the focal loss. It is possible that this phenomenon is due to the relationship between the estimated scores (from focal loss) and the true conditional probabilities (cf. Charoenphakdee et al. (2021); Liu et al. (2021)). We leave this topic as future work.

Although the RankDice/mRankDice framework is developed for Dice/mDice optimization, the performance in terms of the IoU/mIoU metric is also consistently improved.

Moreover, we also present some important observations based on our experiments about losses, frameworks, and models.

CE, Focal and BCE are the top three losses for Dice-segmentation. While some Dice approximating losses, such as Soft-Dice and binary Soft-Dice, usually lead to suboptimal solutions.

It seems that the multiclass modeling and multiclass losses are more preferred for both datasets. Moreover, the threshold-based framework usually outperforms the argmax-based framework for both multiclass and multilabel losses.
For multiclass losses, including CE Focal, Soft-Dice, and LovaszSoftmax, the Dice and IoU metrics are consistent, i.e., a higher Dice yields a higher IoU score. For multilabel losses, including BCE and B-Soft-Dice, there is a significant difference between Dice and IoU metrics.

<table>
<thead>
<tr>
<th>Object class</th>
<th>Threshold/Argmax (Dice, IoU) (×0.1)</th>
<th>RankDice (our) (Dice, IoU) (×0.1)</th>
<th>imps. (best vs. best)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CE Focal BCE</td>
<td>CE Focal BCE</td>
<td></td>
</tr>
<tr>
<td>road</td>
<td>(85.7, 77.2) (86.1, 77.9) (92.2, 46.1)</td>
<td>(85.6, 77.1) (86.0, 77.7) (92.2, 46.1)</td>
<td>~</td>
</tr>
<tr>
<td>sidewalk</td>
<td>(57.3, 47.8) (53.6, 43.8) (43.8, 21.9)</td>
<td>(60.8, 50.8) (58.7, 48.4) (54.0, 27.0)</td>
<td>6.1%</td>
</tr>
<tr>
<td>building</td>
<td>(84.6, 76.2) (83.4, 74.8) (79.4, 39.7)</td>
<td>(85.1, 76.7) (83.6, 74.7) (82.1, 41.0)</td>
<td>~</td>
</tr>
<tr>
<td>wall</td>
<td>(17.4, 13.6) (16.1, 12.4) (04.9, 02.4)</td>
<td>(21.0, 16.4) (21.5, 16.8) (08.3, 04.2)</td>
<td>23.6%</td>
</tr>
<tr>
<td>fence</td>
<td>(14.7, 10.9) (12.4, 08.9) (15.2, 07.6)</td>
<td>(15.8, 11.7) (13.7, 09.8) (19.1, 09.5)</td>
<td>25.7%</td>
</tr>
<tr>
<td>pole</td>
<td>(41.9, 29.0) (34.7, 23.4) (27.1, 13.5)</td>
<td>(46.0, 31.7) (35.6, 23.1) (36.4, 18.2)</td>
<td>9.8%</td>
</tr>
<tr>
<td>traffic light</td>
<td>(34.9, 26.5) (31.5, 24.0) (18.7, 09.4)</td>
<td>(37.4, 28.3) (33.5, 24.7) (21.3, 10.6)</td>
<td>7.2%</td>
</tr>
<tr>
<td>traffic sign</td>
<td>(49.9, 39.0) (45.9, 35.1) (35.3, 17.6)</td>
<td>(51.4, 40.1) (46.6, 35.1) (39.6, 19.8)</td>
<td>3.0%</td>
</tr>
<tr>
<td>vegetation</td>
<td>(90.2, 84.1) (90.2, 83.8) (80.0, 44.5)</td>
<td>(90.3, 84.1) (89.6, 82.8) (89.4, 44.7)</td>
<td>~</td>
</tr>
<tr>
<td>terrain</td>
<td>(25.7, 20.1) (24.1, 18.5) (19.8, 09.9)</td>
<td>(29.4, 23.1) (28.7, 22.7) (25.3, 12.7)</td>
<td>14.4%</td>
</tr>
<tr>
<td>sky</td>
<td>(83.6, 77.0) (82.0, 75.2) (80.1, 40.0)</td>
<td>(84.5, 77.8) (83.1, 76.2) (80.7, 40.3)</td>
<td>1.1%</td>
</tr>
<tr>
<td>person</td>
<td>(45.1, 36.3) (42.6, 34.1) (32.8, 16.4)</td>
<td>(49.5, 40.0) (47.6, 38.2) (38.6, 19.3)</td>
<td>9.8%</td>
</tr>
<tr>
<td>rider</td>
<td>(35.1, 27.3) (31.2, 24.0) (18.6, 09.3)</td>
<td>(37.2, 29.2) (33.9, 26.3) (24.0, 12.0)</td>
<td>6.0%</td>
</tr>
<tr>
<td>car</td>
<td>(84.1, 76.9) (83.4, 76.2) (80.8, 40.4)</td>
<td>(84.0, 76.6) (81.8, 74.0) (81.2, 40.6)</td>
<td>~</td>
</tr>
<tr>
<td>truck</td>
<td>(24.7, 21.9) (25.6, 22.7) (21.8, 10.9)</td>
<td>(26.6, 23.3) (28.1, 24.8) (26.8, 13.4)</td>
<td>9.8%</td>
</tr>
<tr>
<td>bus</td>
<td>(46.8, 42.2) (48.8, 43.8) (36.3, 18.2)</td>
<td>(51.3, 46.5) (51.5, 46.8) (39.2, 19.6)</td>
<td>5.5%</td>
</tr>
<tr>
<td>train</td>
<td>(34.9, 30.7) (36.3, 31.0) (33.8, 16.9)</td>
<td>(35.8, 31.5) (37.3, 32.2) (34.7, 17.4)</td>
<td>2.8%</td>
</tr>
<tr>
<td>motorcycle</td>
<td>(19.7, 15.8) (20.4, 16.1) (07.0, 03.5)</td>
<td>(22.2, 17.7) (21.1, 16.8) (08.7, 04.4)</td>
<td>8.8%</td>
</tr>
<tr>
<td>bicycle</td>
<td>(41.4, 32.5) (42.1, 32.9) (32.9, 16.5)</td>
<td>(41.9, 32.6) (42.0, 32.5) (36.7, 18.4)</td>
<td>~</td>
</tr>
</tbody>
</table>

Table 5: Averaged class-specific Dice and IoU metrics of Threshold/Argmax, and the proposed RankDice based on various losses of PSPNet + resnet50 on Fine-annotated CityScapes val set. The class-specific improvement in terms of the Dice metric of the proposed RankDice framework is computed in “imps.”, where ~ indicates that the difference between Threshold/Argmax and RankDice is smaller than 1.0%.

6. Conclusions and future work

Summary. In this paper, we proposed a ranking-based framework for segmentation called RankSEG that comprises three steps: conditional probability estimation, ranking, and volume estimation. Specifically, we have focused on the Dice metric and developed RankDice, a version of RankSEG for optimal Dice-segmentation. We introduced a key concept “Dice-calibrated” and demonstrated that RankDice is able to recover the optimal segmentation rule, as opposed to the existing fixed-thresholding frameworks that are suboptimal with respect to the Dice metric. Computationally, we have developed efficient exact/approximate numerical methods, including GPU-enabled algorithms, to carry out RankDice. Moreover, we established general theoretical results, including excess risk bounds and a rate of convergence for RankDice, showing that RankDice is consistent when the conditional probability estimation is well-calibrated. Empirical experiments suggested that the proposed framework performs consistently well on a variety of segmentation benchmarks and state-of-the-art deep learning architectures. In parallel to RankDice, we also developed the framework RankIoU for the IoU metric. The theoretical results are similar, while the computation for the optimal IoU-segmentation could be more expensive in high-dimensional situation.

Limitation and future work. (i) For multiclass/multilabel segmentation, our results in this paper cover the overlapping (allowing) case; however, computing the optimal segmentation for the non-
Figure 5: Comparison of segmentation results between the proposed method and existing methods for classes cat (upper panel) and chair (lower panel). Column 1 indicates original images, Column 2 indicates ground truths, and Columns 3-5 indicate the predicted segmentation produced by argmax, thresholding, and the proposed RankDice, respectively. The results are provided by a PSPNet trained with the cross-entropy loss.
Table 6: Class-specific Dice and IoU of Threshold/Argmax, and the proposed RankDice based on various losses of PSPNet + resnet50 on PASCAL VOC 2012 val set. The class-specific improvement in terms of the Dice metric of the proposed RankDice framework is computed in “imps.”, where ∼ indicates that the difference between Threshold/Argmax and the proposed RankDice is smaller than 1.0%.

Table 7: The averaged Dice and IoU metrics and their standard errors (in parentheses) of the proposed RankDice framework and the fixed-thresholding (with different thresholds) framework in Fine-annotated CityScapes (left) and PASCAL VOC 2012 (right) datasets. The performance is reported based on PSPNet + resnet50 with the BCE loss.

overlapping case is NP-hard. Thus, it would be interesting to develop a scalable approximating algorithm to utilize the proposed framework in the non-overlapping setting. (ii) The conditional independence, $Y_i \perp Y_j | X$ for any $i \neq j$, is crucial for Theorem 1 and subsequent theorems in Section 4. In some segmentation applications, it is of interest to extend the proposed frameworks and theorems with locally dependent outcomes. (iii) When given the testing features, the proposed method can be extended to maximize the F1-score in classification tasks.
Table 8: The averaged Dice and IoU metrics and their standard errors (in parentheses) of the proposed RankDice framework and the fixed-thresholding framework based on temperature-scaling calibration methods with different temperature tuning parameters in Fine-annotated CityScapes and PASCAL VOC 2012 datasets. '/' indicates that the performance of the thresholding-based framework over different temperatures are all the same under BCE loss. The performance is reported based on PSPNet + resnet50.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Temp</th>
<th>Threshold (Dice, IoU) (×.01)</th>
<th>RankDice (our) (Dice, IoU) (×.01)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>CE (57.50, 49.60) BCE</td>
<td>CE (55.10, 27.60) BCE</td>
</tr>
<tr>
<td>CityScapes</td>
<td>1.0</td>
<td>(51.40, 25.70)</td>
<td>(59.30, 51.00)</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>(57.40, 49.60) /</td>
<td>(59.40, 51.20) BCE</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>(56.90, 49.20) /</td>
<td>(59.40, 51.20) BCE</td>
</tr>
<tr>
<td></td>
<td>1.7</td>
<td>(56.20, 48.50) /</td>
<td>(59.10, 51.00) BCE</td>
</tr>
<tr>
<td></td>
<td>2.0</td>
<td>(54.40, 46.90) /</td>
<td>(58.00, 50.10) BCE</td>
</tr>
<tr>
<td></td>
<td>2.2</td>
<td>(52.80, 45.40) /</td>
<td>(57.10, 49.30) BCE</td>
</tr>
<tr>
<td></td>
<td>2.5</td>
<td>(49.80, 42.50) /</td>
<td>(55.40, 48.00) BCE</td>
</tr>
<tr>
<td>VOC 2012</td>
<td>1.0</td>
<td>(64.60, 57.10) BCE</td>
<td>(65.40, 57.80) BCE</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>(64.50, 57.50) /</td>
<td>(66.10, 58.30) BCE</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>(65.30, 57.70) / BCE</td>
<td>(66.90, 59.10) BCE (69.30, 34.60)</td>
</tr>
<tr>
<td></td>
<td>1.7</td>
<td>(65.30, 57.70) / BCE</td>
<td>(67.60, 59.80) BCE (69.00, 34.50)</td>
</tr>
<tr>
<td></td>
<td>2.0</td>
<td>(64.80, 57.10) / BCE</td>
<td>(68.30, 60.50) BCE (68.00, 34.00)</td>
</tr>
<tr>
<td></td>
<td>2.2</td>
<td>(63.80, 56.00) / BCE</td>
<td>(68.40, 60.60) BCE (67.00, 33.50)</td>
</tr>
<tr>
<td></td>
<td>2.5</td>
<td>(61.30, 53.40) / BCE</td>
<td>(67.90, 60.20) BCE (65.10, 32.50)</td>
</tr>
</tbody>
</table>

Acknowledgments

We would like to acknowledge support for this project from the HK RGC-ECS 24302422 and the CUHK direct fund. Ben Dai developed the main framework (RankDice), theory, algorithms and experiments. Chunlin Li mainly extended RankDice to RankIoU (Section 4.2), and partly contributed to the proof of Lemma 3 and Theorem 11. We would like to thank the referees and the Action Editor for constructive feedback which greatly improved this work.
Appendix A. Empirical evaluation of the Dice metric

Recall the definition of the Dice and IoU metrics in (1), their empirical evaluation based on a validation/testing dataset \((\tilde{x}_i, \tilde{y}_i)_{i=1,\ldots,m}\), can be written as:

\[
\hat{\text{Dice}}_\gamma(\delta) = \frac{1}{m} \sum_{i=1}^{m} \frac{2\tilde{y}_i^\top \delta(\tilde{x}_i) + \gamma}{\|\tilde{y}_i\|_1 + \|\delta(\tilde{x}_i)\|_1 + \gamma} = \frac{1}{m} \sum_{i=1}^{m} \frac{2TP_i + \gamma}{2TP_i + FP_i + FN_i + \gamma},
\]

\[
\hat{\text{IoU}}_\gamma(\delta) = \frac{1}{m} \sum_{i=1}^{m} \left( \frac{\tilde{y}_i^\top \delta(\tilde{x}_i) + \gamma}{\|\tilde{y}_i\|_1 + \|\delta(\tilde{x}_i)\|_1 - \tilde{y}_i^\top \delta(\tilde{x}_i) + \gamma} \right) = \frac{1}{m} \sum_{i=1}^{m} \frac{TP_i + \gamma}{TP_i + FP_i + FN_i + \gamma},
\]

(22)

where \(TP_i\), \(FP_i\) and \(FN_i\) are defined at the instance level. In general, the empirical Dice and IoU metrics are not equal to the evaluation criteria used in some literature:

\[
\hat{\text{Dice}}_\gamma(\delta) \neq \bar{\text{Dice}}_\gamma(\delta) := \frac{1}{m} \sum_{i=1}^{m} \frac{2\tilde{y}_i^\top \delta(\tilde{x}_i) + \gamma}{\|\tilde{y}_i\|_1 + \frac{1}{m} \sum_{i=1}^{m} \|\delta(\tilde{x}_i)\|_1 + \gamma} \xrightarrow{P} \frac{\mathbb{E}(2\tilde{Y}^\top \delta(\tilde{X})) + \gamma}{\mathbb{E}(\|\tilde{Y}\|_1) + \mathbb{E}(\|\delta(\tilde{X})\|_1) + \gamma},
\]

\[
\hat{\text{IoU}}_\gamma(\delta) \neq \bar{\text{IoU}}_\gamma(\delta) := \frac{1}{m} \sum_{i=1}^{m} \frac{\tilde{y}_i^\top \delta(\tilde{x}_i) + \gamma}{\|\tilde{y}_i\|_1 + \frac{1}{m} \sum_{i=1}^{m} \|\delta(\tilde{x}_i)\|_1 - \frac{1}{m} \sum_{i=1}^{m} \tilde{y}_i^\top \delta(\tilde{x}_i) + \gamma} \xrightarrow{P} \frac{\mathbb{E}(2\tilde{Y}^\top \delta(\tilde{X})) + \gamma}{\mathbb{E}(\|\tilde{Y}\|_1) + \mathbb{E}(\|\delta(\tilde{X})\|_1) - \mathbb{E}(\tilde{Y}^\top \delta(\tilde{X})) + \gamma}.
\]

(23)

Here \(\xrightarrow{P}\) denotes convergence in probability following from the law of large numbers and Slutsky’s theorem. Clearly, both empirical and population evaluations in (23) do not match with the empirical verisons in (22) and the population Dice in (1). Although the empirical evaluation in (23) is widely used, it inherently discounts the effects of instances with small segmented features/pixels, leading to bias in the empirical evaluation. The issues of \(\bar{\text{Dice}}_\gamma(\delta)\) and \(\bar{\text{IoU}}_\gamma(\delta)\) are also indicated in some recent literature, including Cordts et al. (2016) and Berman et al. (2018).

Therefore, it is highly recommended using the empirical Dice in (22) in implementation, and our numerical results in Section 5 are reported based on (22).

Appendix B. Auxiliary definitions

B.1 Population RankSEG

In this section, we present the definition of population RankSEG, including the proposed frameworks RankDice and RankIoU. In other words, we work with population of \((X, Y) \in \mathbb{R}^d \times \{0, 1\}^d\). Denote \(\mathcal{Q}\) as the class of all measurable functions \(q : x \in \mathbb{R}^d \rightarrow q(x) = (q_1(x), \cdots, q_d(x))^T \in [0, 1]^d\).

**Step 1 (Conditional probability estimation):** Estimate the conditional probability based on a strictly proper loss \(l(\cdot, \cdot)\):

\[
\hat{q} = \arg\min_{q \in \mathcal{Q}} \mathbb{E}(l(Y, q(X))).
\]

(24)

**Step 2 (Ranking):** Given a new instance \(x\), sort its estimated conditional probabilities decreasingly, and denote the corresponding indices as \(j_1, \cdots, j_d\), that is, \(\hat{q}_{j_1}(x) \geq \hat{q}_{j_2}(x) \geq \cdots \geq \hat{q}_{j_d}(x)\).
Step 3 (Volume estimation): From (4), we estimate the volume \( \hat{\tau}(x) \) by replacing the true conditional probability \( p(x) \) by the estimated one \( \hat{q}(x) \):

\[
\text{(RankDice)} \quad \hat{\tau}(x) = \arg\max_{\tau \in \{0, \ldots, d\}} \sum_{s=1}^{d} \sum_{i=0}^{d-1} \frac{2}{\tau + l + \gamma + 1} \hat{q}_{j_i}(x) \mathbb{P}(\hat{\tau}_{-j_i}(x) = l) + \sum_{l=0}^{d} \frac{\gamma}{\tau + l + \gamma} \mathbb{P}(\hat{\tau}(x) = l),
\]

\[
\text{(RankIoU)} \quad \hat{\tau}(x) = \arg\max_{\tau \in \{0.1 \ldots d\}} \left( \sum_{j \in J(x)} \hat{q}_j(x) + \gamma \right) \gamma \sum_{l=0}^{d-\tau} \frac{1}{\tau + l + \gamma} \mathbb{P}(\hat{\tau}_{-j}(x) = l),
\]

where \( \hat{\tau}(x) = \sum_{j=1}^{d} \hat{B}_j(x), \hat{\tau}_{-j_i}(x) = \sum_{j \neq j_i} \hat{B}_j(x), \) and \( \hat{B}_j(x) \) denote Poisson-binomial random variables, and \( \hat{B}_j(x) \) is a Bernoulli random variable with the success probability \( \hat{q}_j(x) \); for \( j = 1, \ldots, d \).

B.2 Poisson-binomial distribution

The Poisson binomial distribution is the discrete probability distribution of a sum of independent non-identical Bernoulli trials. Specifically, suppose \( B_1, \ldots, B_d \) are independent Bernoulli random variables, with probabilities of success \( p = (p_1, \ldots, p_d)^T \), then \( \Gamma = \sum_{j=1}^{d} B_j \) is a Poisson-Binomial random variable with parameter \( p \), denoted as \( \Gamma \sim \text{PB}(p) \), and its probability mass function is:

\[
\mathbb{P}(\Gamma = l) = \sum_{\mathbf{b}: |\mathbf{b}|_1 = l} \prod_{j=1}^{d} (b_j p_j + (1 - b_j)(1 - p_j)),
\]

where \( \mathbf{b} = (b_1, \ldots, b_d)^T \in \{0, 1\}^d \). Moreover, the mean, variance, and skewness for \( \Gamma \sim \text{PB}(p) \) are listed as follows.

\[
\mu := \mathbb{E}(\Gamma) = \sum_{j=1}^{d} p_j, \quad \sigma^2 := \text{Var}(\Gamma) = \sum_{j=1}^{d} p_j(1 - p_j), \quad \eta := \text{Skew}(\Gamma) = \frac{1}{\sigma^3} \sum_{j=1}^{d} p_j(1 - p_j)(1 - 2p_j).
\]

B.3 Conditional independence in segmentation

In this section, we adopt a probabilistic perspective on the likelihood of the segmentation task, to suggest that the conditional independence \( Y_j \perp Y_{j'} \mid X \) for \( j \neq j' \) is implicitly assumed to ensure the validity of the cross-entropy (CE) loss, and widely accepted due to the high dimensional nature of segmentation data.

Suppose \( (X_i, Y_i) \overset{iid}{\sim} \mathbb{P}_{X,Y} \), the negative conditional log-likelihood function of \( q \) for the probabilistic model is:

\[
\mathcal{L}_n(q) := -\log \left( \prod_{i=1}^{n} \mathbb{P}_q(Y_i = y_i \mid X = x_i) \right) = -\log \left( \prod_{i=1}^{n} \prod_{j=1}^{d} (q_j(x_i))^{y_{ij}} (1 - q_j(x_i))^{1-y_{ij}} \right) \\
= -\sum_{i=1}^{n} \sum_{j=1}^{d} (y_{ij} \log(q_j(x_i)) + (1 - y_{ij}) \log(1 - q_j(x_i))) = \sum_{i=1}^{n} l_{CE}(y_i, q(x_i)),
\]

where the second equality follows from the conditional independence assumption \( Y_j \perp Y_{j'} \mid X \) for \( j \neq j' \), which connects the CE loss to the negative conditional log-likelihood function. In this sense, the conditional independence is naturally (and implicitly) assumed by CE to presuppose the probabilistic interpretation of its estimator.
Figure 6: Simulation setting in Section 5.1 with different decay patterns and dimensions/shapes (28x28 - 256x256). **Upper panel.** Heatmaps for the simulated probabilities with step decay ($\beta = 0.1, 0.3, 0.5$). **Middle panel.** Heatmaps for the simulated probabilities with exponential decay (base=1.01, 1.05, 1.10). **Lower panel.** Heatmaps for the simulated probabilities with linear decay (slope=1, 2, 4). The performance for the proposed RankDice and thresholding-based frameworks is summarized in Table 9.

Moreover, the conditional independence is widely accepted due to the high dimensional nature of segmentation data. For example, given a 512x512 image, it is infeasible to consider $512^4$ pairs of label-dependence, which can even be adaptive with respect to x.

Appendix C. Simulation results and Implementation details

C.1 Simulation setting and results

This subsection includes the simulation setting demonstration (Figure 6) and the numerical results (Tables 9 and 10 and Figure 7).
Decay | Shape | Threshold (at 0.5) | RankDice (our)  
--- | --- | --- | ---  
step(0.1) | 28x28 | 0.049(.000) | 0.274(.001)  
64x64 | 0.083(.000) | 0.279(.000)  
128x128 | 0.081(.000) | 0.278(.000)  
256x256 | 0.089(.000) | 0.279(.000)  
step(0.3) | 28x28 | 0.022(.001) | 0.499(.001)  
64x64 | 0.038(.000) | 0.517(.001)  
128x128 | 0.036(.000) | 0.518(.000)  
256x256 | 0.040(.000) | 0.518(.000)  
step(0.5) | 28x28 | 0.708(.000) | 0.708(.000)  
64x64 | 0.707(.000) | 0.707(.000)  
128x128 | 0.708(.000) | 0.708(.000)  
256x256 | 0.708(.000) | 0.708(.000)  
exp(1.01) | 28x28 | 0.870(.000) | 0.870(.000)  
64x64 | 0.669(.000) | 0.714(.000)  
128x128 | 0.410(.000) | 0.551(.000)  
256x256 | 0.286(.000) | 0.450(.000)  
exp(1.05) | 28x28 | 0.427(.001) | 0.551(.001)  
64x64 | 0.296(.001) | 0.446(.001)  
128x128 | 0.276(.001) | 0.427(.001)  
256x256 | 0.274(.001) | 0.427(.001)  
exp(1.10) | 28x28 | 0.332(.002) | 0.467(.002)  
64x64 | 0.301(.001) | 0.439(.002)  
128x128 | 0.300(.002) | 0.438(.002)  
256x256 | 0.298(.002) | 0.436(.002)  

Table 9: The averaged Dice metrics and its standard errors (in parentheses) of the proposed RankDice framework and the thresholding-based (or argmax-based) framework in Example 1 (see Fig 6) in Section 5.1.

| Framework | Threshold | Dice  
--- | --- | ---  
threshold-based | 0.1 | 0.481(.005)  
| 0.2 | 0.560(.005)  
| 0.3 | 0.560(.005)  
| 0.4 | 0.560(.005)  
| 0.5 | 0.560(.005)  
| 0.6 | 0.528(.005)  
| 0.7 | 0.471(.005)  
| 0.8 | 0.377(.005)  
| 0.9 | 0.230(.004)  
RankDice(our) | — | 0.601(.005)  

Table 10: The averaged Dice metrics and its standard errors (in parentheses) of the proposed RankDice framework and the thresholding-based (with different thresholds) framework in Example 2 in Section 5.1.

Figure 7: The optimal thresholds for different images with various generating parameters ($\beta, \rho$) in Example 2 in Section 5.1.

C.2 Implementation details

The experiment protocol of our numerical sections basically follows a well-developed Github repository PYTORCH-SEGMENTATION (Ouali, 2022). The major difference lies in the empirical evaluation of the Dice and IoU metrics. In our experiments, we report the unbiased evaluations $\hat{m}_{\text{Dice}, \gamma}(\cdot)$.
and \( \overline{\text{mIoU}}(\cdot) \), yet the biased evaluations \( \overline{\text{mDice}}(\cdot) \) and \( \overline{\text{mIoU}}(\cdot) \) are usually used for existing literature, see more discussion and definitions in Appendix A.

To justify the effectiveness of our experiment protocol, we also report \( \overline{\text{mDice}}(\cdot) \) and \( \overline{\text{mIoU}}(\cdot) \) under our setting based on the \( \text{argmax} \)-based framework and compare the performance with the existing benchmarks. Specifically, in our setting, the performance is, DeepLab: (i) CityScapes: \( \text{mIoU} \) 63.20%; \( \text{mDice} \) 76.10%; (ii) VOC: \( \text{mIoU} \) 74.40%; \( \text{mDice} \) 84.30%; (i) CityScapes: \( \text{mIoU} \) 65.20%; \( \text{mDice} \) 77.60%; (ii) VOC: \( \text{mIoU} \) 79% (which is provided by Ouali (2022) with the same configuration expect \( \text{batch size}=16 \)); FCN8: VOC: \( \text{mIoU} \) 55.60%; \( \text{mDice} \) 70.40%.

The experiment protocol, including learning rate, crop size, backbone, and batch size, on the existing networks are summarized as follows.

**DeepLab** (Chen et al., 2018). The experiment on the Fine-annotated CityScapes dataset is set as follows: backbone is “Xception-65”. The final \( \overline{\text{mIoU}} \) is 79.14%; The experiment on the PASCAL VOC 2012 dataset is set as follows: learning rate is 0.007 with poly schedule; crop size is 513x513, backbone is “resnet101”, batch size is 16. The final \( \overline{\text{mIoU}} \) is 78.21%;

**PSPNet** (Zhao et al., 2017). The experiment on the Fine-annotated CityScapes dataset is set as follows: learning rate is 0.01. The final \( \overline{\text{mIoU}} \) is 78.4%; The experiment on the PASCAL VOC 2012 dataset is set as follows: learning rate is 0.01, batch size is 16. The final \( \overline{\text{mIoU}} \) based on the VOC test dataset is 82.6%;

**FCN8** (Long et al., 2015). The experiment on the PASCAL VOC 2012 dataset is set as follows: learning rate is 0.0001, batch size is 20, backbone is “VGG16”. The final \( \overline{\text{mIoU}} \) is 62.2%.

Note that the suboptimal performance of our experiment may be caused by a small batch/crop size, different specified backbone models, and other fitting hyperparameters. The current experiment can be further improved by carefully tuning the hyperparameters, yet it provides a fair numerical comparison of all frameworks (threshold-based, argmax-based, and the proposed RankDice).

**Appendix D. Technical proofs**

**D.1 Proof of Theorem 1**

**Proof** It suffices to consider the point-wise maximization:

\[
\delta^*(x) = \arg \max_{v \in \{0,1\}^d} \text{Dice}_\gamma(v|x), \quad \text{Dice}_\gamma(v|x) = \mathbb{E}\left( \frac{2Y^Tv + \gamma}{\|Y\|_1 + \|v\|_1 + \gamma} \mid X = x \right).
\]

Let \( y_j = (y_1, \cdots, y_{j-1}, y_{j+1}, \cdots, y_d)^T, I(v) = I(\delta(x)) = \{ j : v_j = 1 \} \) be the index set of segmented features by \( \delta(x) \), and \( \|v\|_1 = \tau \), we have

\[
\text{Dice}_\gamma(v|x) = \mathbb{E}\left( \frac{2Y^Tv}{\|Y\|_1 + \tau + \gamma} \mid X = x \right) + \mathbb{E}\left( \frac{\gamma}{\|Y\|_1 + \tau + \gamma} \mid X = x \right).
\]

Note that the second term is only related to \( \tau \), and the first term can be rewritten as:

\[
\mathbb{E}\left( \frac{2Y^Tv}{\|Y\|_1 + \tau + \gamma} \mid X = x \right) = \sum_{y \in \{0,1\}^d} \frac{2y^TP(Y = y|x)}{\tau + \|y\|_1 + \gamma} = \sum_{y \in \{0,1\}^d} \sum_{j=1}^d \frac{2y_jP(Y = y|x)}{\tau + \|y\|_1 + \gamma}
\]

\[
= \sum_{j \in I(v)} \sum_{y \in \{0,1\}^d} \frac{2y_jP(Y = y|x)}{\tau + \|y\|_1 + \gamma} = \sum_{j \in I(v)} \sum_{y \in \{0,1\}^d-1} \frac{2P(Y = y|x)}{\tau + \|y\|_1 + \gamma} = \sum_{j \in I(v)} s_j(\tau).
\]

(25)
As indicated in (25), when \( \tau \) is given, Dice\(_{\gamma}(v|x) \) is an additive function with respect to \( j \in I(v) \). Therefore, maximizing Dice\(_{\gamma}(v|x) \) suffices to find the indices of top \( \tau \) largest \( s_j(\tau) \). Toward this end, we consider the differenced score function:

\[
D_{j'}(\tau) = s_j(\tau) - s_{j'}(\tau) = \sum_{y_j \in \{0,1\}^{d-1}} 2\mathbb{P}(Y = y|x) \frac{2\mathbb{P}(Y = y|x)}{\tau + ||y||_1 + \gamma} - \sum_{y_{j'} \in \{0,1\}^{d-2}} 2\mathbb{P}(Y = y|x) \frac{2\mathbb{P}(Y = y|x)}{\tau + ||y||_1 + \gamma}
\]

\[
= \sum_{y,j' \in \{0,1\}^{d-2} \ y_j = 1 \ y_{j'} = 0} 2\mathbb{P}(Y = y|x) \frac{2\mathbb{P}(Y = y|x)}{\tau + ||y||_1 + \gamma}
\]

\[
= (\mathbb{P}(Y_j = 1|x) - \mathbb{P}(Y_{j'} = 1|x)) \sum_{y,j' \in \{0,1\}^{d-2}} 2\mathbb{P}(Y = y|x) \frac{2\mathbb{P}(Y = y|x)}{\tau + ||y||_1 + \gamma},
\]

where \( y,j' \) is \( y \) removing \( j \) and \( j' \), the second last equality follows from the conditional independence of \( Y_j \) and \( Y_{j'} \) given \( X \) for any pair \( j \) and \( j' \). According to (26), we have

\[
D_{j'}(\tau) \geq 0 \iff \mathbb{P}(Y_j = 1|x) - \mathbb{P}(Y_{j'} = 1|x) \geq 0,
\]

thus sorting \( s_j(\tau) \) is equivalent to sorting \( \mathbb{P}(Y_j = 1|x) \) for any given \( \tau \). Let \( J_\tau = \{ j : \sum_{j' = 1}^d 1(\mathbb{P}(Y_{j'} = 1|x) \geq \mathbb{P}(Y_j = 1|x)) \leq \tau \} \) be the index set of the \( \tau \)-largest conditional probabilities, it suffices to solve

\[
\tau^* = \arg\max_{\tau = 0, \ldots, d} \sum_{j \in J_\tau} \mathbb{E}(\frac{2Y_j}{||y||_1 + \tau + \gamma}) + \mathbb{E}(\frac{\gamma}{||y||_1 + \tau + \gamma})
\]

\[
= \arg\max_{\tau = 0, \ldots, d} \sum_{j \in J_\tau} \sum_{l=0}^{d-1} \frac{2p_j(x)}{\tau + l + \gamma + 1} \mathbb{P}(||Y_j||_1 = l|X = x) + \sum_{l=0}^{d-1} \frac{\gamma}{\tau + l + \gamma} \mathbb{P}(||Y||_1 = l|X = x),
\]

where \( ||Y_{-j}||_1 = \sum_{j' \neq j} Y_{j'} \) is a Poisson-binomial random variable with success probabilities \( p_j(x) \), since \( Y_j (j = 1, \ldots, d) \) is an independent Bernoulli random variable given \( X = x \). The desirable result then follows.

### D.2 Proof of Lemma 3

**Proof** To proceed, we denote \( \xi_l(x) = \mathbb{P}(\hat{\Gamma}(x) = l) \), and \( \xi_{j'}(x) = \mathbb{P}(\hat{\Gamma}_{-j}(x) = l) \), and \( \mathbb{P}_\tau = \mathbb{P}_\tau(x) + \mathbb{P}_\tau(x) \). For simplicity in presentation, we assume that \( \hat{\xi}_{l}(x) \geq \cdots \geq \hat{\xi}_{d}(x) \). Then, for any \( \tau' > \tau \),
since $v_\tau \geq v_{\tau'}$, and

\[
\frac{\Pi_\tau(x) - \Pi_{\tau'}(x)}{2(\tau' - \tau)} \geq \sum_{j=1}^{\tau} \eta_j(x) \sum_{l=0}^{d-1} \frac{\xi_j(x)}{(\tau + l + 1 + \gamma)(\tau' + l + 1 + \gamma)} - \frac{1}{\tau' - \tau} \sum_{j=\tau+1}^{\tau'} \eta_j(x) \sum_{l=0}^{d-1} \frac{\xi_j(x)}{\tau' + l + 1 + \gamma + 1} \\
\geq \sum_{j=1}^{\tau} \eta_j(x) \sum_{l=0}^{d-1} \frac{\xi_j(x)}{(\tau + l + \gamma + 1)(\tau' + l + 1 + \gamma + 1)} - \eta_{\tau+1}(x) \sum_{l=0}^{d-1} \frac{\xi_{\tau+1}(x)}{\tau' + l + 1 + \gamma + 1} \\
\geq \eta_{\tau+1}(x) \sum_{l=0}^{d-1} \frac{(\tau + \gamma + \nu)\xi_{\tau+1}(x)}{(\tau + l + \gamma + 1)(\tau' + l + 1 + \gamma + 1)} - \eta_{\tau+1}(x) \sum_{l=0}^{d-1} \frac{\xi_{\tau+1}(x)}{\tau' + l + 1 + \gamma + 1} \geq 0,
\]

where the second inequality follows from Lemma 17 with $\zeta_l = (\tau + l + \gamma + 1)(\tau' + l + \gamma + 1)$ and $\zeta_l = \tau' + l + \gamma + 1$, and $\eta_1(x) \geq \cdots \geq \eta_\tau(x) \geq \cdots \geq \eta_{\tau'}(x)$, and the third inequality follows from the condition that $\sum_{j=1}^{\tau} \eta_j(x)/\eta_{\tau+1}(x) \geq \tau + \gamma + \nu$. The desirable result then follows. The upper bound provided by Lemma 3 is illustrated in Figure 8 based on a random example of Example 1.

\[\text{Figure 8: Dice score vs. } \tau \text{ based on a random example of Example 1. Note that Lemma 3 indicates the optimal } \tau (\text{red line}) \text{ is always obtained before the upper bound (green line). Thus, the searching region of } \tau \text{ can be shrunk.}\]

**D.3 Proof of Lemma 4**

**Proof** Without loss of generality, assume $L(\epsilon) \subset \{0, \cdots, d - 1\}$. Denote $l_L = |\hat{\sigma}(x)\Psi^{-1}(\epsilon) + \hat{\mu}(x)| - 1$ and $l_U = [-\hat{\sigma}(x)\Psi^{-1}(\epsilon) + \hat{\mu}(x)]$, $\xi_l = \mathbb{P}(\hat{\Gamma}(x) = l)$, $\xi_{\tau} = \mathbb{P}(\hat{\Gamma}(x) = \tau)$, $\xi_{\tau,l} = \mathbb{P}(\hat{\Gamma}_{\tau,l}(x) = l)$. 

35
where the last inequality follows from Lemma 16. For this completes the proof.

Similarly, for $|\tilde{\nu}_l - \nu_l|$, we have

$$|\tilde{\nu}_l - \nu_l| \leq \frac{2\gamma}{\tau + \gamma - 1 + \gamma} \left(\log \frac{1 + \frac{d}{\tau + \gamma}}{1 + \frac{1}{\tau + \gamma}} + 1 - \frac{1}{\tau + \gamma - 1}\right).$$

This completes the proof.
D.4 Proof of Lemma 5

Proof With the same argument in the proof of Lemma 4, it suffices to consider

$$|\tilde{P}(\Gamma(x) = l) - \tilde{P}(\Gamma_j(x) = l)| \leq \sum_{l' = l-1}^l |\tilde{P}(\Gamma(x) \leq l') - \tilde{P}(\Gamma_j(x) \leq l')|.$$  

Denote $I = \tilde{\sigma}(x)^{-1}(l/2 - \tilde{\mu}(x))$ and $I_j = \tilde{\sigma}_j(x)^{-1}(l/2 - \tilde{\mu}_j(x))$, we have

$$|\tilde{P}(\Gamma(x) \leq l) - \tilde{P}(\Gamma_j(x) \leq l)| = |\Psi(I) - \Psi_j(I_j)|$$

$$\leq |\Phi(I) - \Phi(I_j)| + \frac{1}{6} |\tilde{\eta}(x)(1 - I^2)\phi(I) - \tilde{\eta}_j(x)(1 - I_j^2)\phi(I_j)| =: T_1 + \frac{1}{6} T_2.$$

Next, we turn to treat $T_1$ and $T_2$ separately. Without loss generalization, we assume $|I_j| \geq |I|$, then

$$T_1 \leq \left| \int_I^{I_j} \phi(x)dx \right| \leq |I_j - I|\phi(I)$$

$$\leq \left( |\tilde{\sigma}^{-1}(x) - \tilde{\sigma}_j^{-1}(x)| |I_j + 1/2 - \tilde{\mu}(x)| + \tilde{\sigma}_j^{-1}(x)|\tilde{\mu}(x) - \tilde{\mu}_j(x)| \right)\phi(I)$$

$$= \tilde{\sigma}_j^{-1}(x)|\tilde{\sigma}(x) - \tilde{\sigma}_j(x)||I|\phi(I) + \tilde{\sigma}_j^{-1}(x)\tilde{\eta}_j(x)\phi(I)$$

$$\leq \frac{\tilde{\eta}_j(x)(1 - \tilde{\eta}_j(x))}{\tilde{\sigma}_j(x)(\tilde{\sigma}_j(x) + \tilde{\sigma}(x))}|I|\phi(I) + \frac{1}{\sqrt{2\pi}\tilde{\sigma}_j(x)}$$

$$\leq \frac{1}{4\sqrt{2\pi}} \left( \frac{1}{2\sqrt{\tilde{\sigma}^2(x) - 1/4}} + \frac{4}{\sqrt{\tilde{\sigma}^2(x) - 1/4}} \right),$$

where the last inequality follows the fact that $|u|\phi(u) \leq 1/\sqrt{2\pi}$ and $0 \leq \phi(u) \leq 1/\sqrt{2\pi}$. For $T_2$, let $g(u) = (1 - u^2)\phi(u)$, we have

$$T_2 \leq |\tilde{\eta}(x)(1 - I^2)\phi(I)| + |\tilde{\eta}_j(x)(1 - I_j^2)\phi(I_j)| \leq \frac{\tilde{m}_3(x)}{\sqrt{2\pi}(\tilde{\sigma}^2(x) - 1/4)^{3/2}},$$

where the last inequality follows from the fact that $|g(u)| \leq 1/\sqrt{2\pi}$, and $\tilde{m}_3(x) = \sum_{j=1}^d \tilde{p}_j(x)(1 - \tilde{p}_j(x))(1 - 2 \tilde{p}_j(x))$. Taken together, using the same argument in the proof of Lemma 4, we have

$$|\tilde{\omega}^b - \tilde{\omega}| \leq \frac{1}{4\sqrt{2\pi}} \left( \frac{1}{\tilde{\sigma}^2(x) - 1/4} + \frac{4}{\sqrt{\tilde{\sigma}^2(x) - 1/4}} + \frac{4\tilde{m}_3(x)}{(\tilde{\sigma}^2(x) - 1/4)^{3/2}} \right) (\log(1 + \frac{d}{r + \gamma}) + 1).$$

Combining Lemma 4, the desirable result then follows. □

D.5 Proof of Lemma 6

Proof Denote $K_+ = \{1 \leq k \leq K : \alpha_k > 0\}$. We first prove the necessity. Suppose $\Delta^*_k$ is a global minimizer of Dice$_k(\cdot)$, for $k \in K_+$. Then for any $\Delta = (\Delta_1, \cdots, \Delta_K)$, we have

$$\text{mDice}_\gamma(\Delta^*) = \sum_{k \in K_+} \alpha_k \text{Dice}_{\gamma,k}(\Delta^*_k) \leq \sum_{k \in K_+} \alpha_k \text{Dice}_{\gamma,k}(\Delta_k),$$

$$\sum_{k \in K_+} \alpha_k$$
yields that $\Delta^*$ is a global minimizer of $m\text{Dice}_\gamma(\cdot)$. We next prove the sufficiency by contradiction. Suppose $\Delta^*$ is a global minimizer of $m\text{Dice}_\gamma(\cdot)$, yet there exists $k_0 \in K_+$ such that $\Delta^*_{k_0}$ is not a minimizer of $\text{Dice}_{\gamma,k_0}(\cdot)$. Thus, there exists a segmentation rule $\tilde{\Delta}$ such that $\text{Dice}_{\gamma,k_0}(\tilde{\Delta}) < \text{Dice}_{\gamma,k_0}(\Delta^*_{k_0})$, then let $\tilde{\Delta} = (\Delta^*_1, \ldots, \Delta^*_{k_0-1}, \Delta^*_{k_0}, \Delta^*_{k_0+1}, \ldots, \Delta^*_K)$

$$m\text{Dice}_\gamma(\Delta^*) = \sum_{k \in K_+} \alpha_k \text{Dice}_{\gamma,k}(\Delta^*_k) > \sum_{k \in K_+} \alpha_k \text{Dice}_{\gamma,k}(\tilde{\Delta}) = m\text{Dice}_\gamma(\tilde{\Delta}),$$

which leads to contradiction of that $\Delta^*$ is a global minimizer of $m\text{Dice}_\gamma(\cdot)$. The desirable result then follows.

**D.6 Proof of Lemma 7**

**Proof** Given a segmentation rule $\Delta(X) = (\Delta_1(X), \ldots, \Delta_K(X))$, $m\text{Dice}_\gamma(\cdot)$ can be rewritten as

$$m\text{Dice}_\gamma(\Delta) = \sum_{k=1}^K \alpha_k \text{Dice}_{\gamma,k}(\Delta_k(X)).$$

Similarly, it is equivalent to consider the point-wise minimization conditional on $X = x$:

$$\Delta^*(x) = \arg\max_{V \in \{0,1\}^{d \times K}} m\text{Dice}_\gamma(V|x), \quad \text{s.t.} \sum_{k=1}^K V_k = 1_d,
\text{Dice}_\gamma(V|x) = \sum_{k=1}^K \alpha_k \text{Dice}_{\gamma,k}(V_k|x),$$

where $V_k$ is the $k$-th column of $V$. According to (25), we have

$$m\text{Dice}_\gamma(V|x) = \sum_{k=1}^K \alpha_k \text{Dice}_{\gamma,k}(V_k|x) = \sum_{k=1}^K \alpha_k \sum_{y \in \{0,1\}} \frac{2\mathbb{P}(Y_k = y_k|x) \tau_k + \|y_k\|_1 + \gamma}{\gamma + 1 + \tau_k + \gamma} + \sum_{k=1}^K \alpha_k \mathbb{E}
\left(\frac{\gamma}{\|y_k\|_1 + \tau_k + \gamma}\right)$$

$$= \sum_{k=1}^K \sum_{j=1}^d R_{jk}(\tau_k) v_{jk} + \sum_{k=1}^K \alpha_k \bar{v}(\tau_k),$$

where $v_{jk} \in \{0,1\}$ is the segmentation indicator of the $j$-th feature under the class-$k$, and $R_{jk}(\cdot)$ is a reward function defined as:

$$R_{jk}(\tau_k) = \alpha_k \sum_{y \in \{0,1\}^{d-1}} \frac{2\mathbb{P}(Y_k = y_k|x) \tau_k + \|y_k\|_1}{\gamma + 1 + \tau_k + \gamma}.$$
Now, suppose the optimal volume function \( \tau^*(x) = (\tau^*_1(x), \ldots, \tau^*_K(x))^T \) is given, then \( \nabla(\tau^*_k) \) becomes a constant, and the point-wise maximization on \( \text{mDice}_\gamma \) is equivalent to:

\[
\begin{align*}
\max_{\forall \in \{0,1\}^{d \times K}} & \sum_{k=1}^K \sum_{j=1}^d R^*_{jk} v_{jk}, \\
\text{subject to} & \sum_{j=1}^d v_{jk} = \tau_k^*(x), \quad \sum_{k=1}^K v_{jk} = 1, \\
& \text{for } k = 1, \ldots, K; \quad \text{for } j = 1, \ldots, d,
\end{align*}
\]

where \( R^*_{jk} = R_{jk}(\tau_k^*(x)) \) is the reward under \( \tau_k^*(x) \). Note that (28) is the formulation for the assignment problem (Kuhn, 1955). This completes the proof.

D.7 Proof of Lemma 9

**Proof** For simplicity, we construct a counter example based on \( \gamma = 0 \) and \( d = 2 \), that is, \( Y = (Y_1, Y_2)^T \) and \( p(x) = (p_1(x), p_2(x))^T \). Without loss generality, we assume \( p_1(x) > p_2(x) \).

First, we derive the Bayes rule in Theorem 1 for this case. Note that it suffices to compare the scores for \( \tau = 1 \ (v = (1,0)^T) \) and \( \tau = 2 \ (v = (1,1)^T) \).

\[
\begin{align*}
\text{Dice}((1,0)^T|x) &= p_1(x) - \frac{1}{3} p_1(x)p_2(x), \\
\text{Dice}((1,1)^T|x) &= \frac{2}{3} (p_1(x) + p_2(x)) - \frac{1}{3} p_1(x)p_2(x).
\end{align*}
\]

Therefore, the Bayes rule for Dice optimal segmentation is:

\[
\delta^*(x) = (1,0)^T, \text{ if } \frac{1}{2} p_1(x) > p_2(x), \quad \delta^*(x) = (1,1)^T, \text{ otherwise.}
\]

Now, we check the Dice-calibrated for classification-calibrated losses. For example, \( p_1(x) = 0.45 \) and \( p_2(x) = 0.44 \), then

\[
\tilde{\delta}(x) = 1(p(x) \geq 0.5) = (0,0)^T \neq (1,1)^T = \delta^*(x),
\]

where the first equality follows from the definition of a classification-calibrated loss: the decision rule must agree with the conditional probabilities. Therefore, \( \text{Dice}(\tilde{\delta}) < \text{Dice}(\delta^*) \) yields that a classification-calibrated loss with thresholding at 0.5 is not Dice-calibrated.

D.8 Proof of Lemma 10

**Proof** By the definition of a strictly proper loss, see Gneiting and Raftery (2007) and references herein, and the formulation in (24), we have \( \hat{q}(x) = p(x) = (\mathbb{P}(Y_1 = 1|X = x), \ldots, \mathbb{P}(Y_d = 1|X = x))^T \). Then the estimation of \( \hat{\tau}(x) \) and \( \hat{\delta}(x) \) agrees with the definition of \( \tau^*(x) \) and \( \delta^*(x) \). This completes the proof.
D.9 Proof of Theorem 11

**Proof** First, we consider point-wise approximation of the Dice metric under probabilities \( p \) and \( \hat{q} \). For any \( \delta \), such that \( \delta_j(x) = 1 \) for \( j = j_1, \ldots, j_r \), and \( \delta_j(x) = 0 \) otherwise. Define

\[
\hat{\text{Dice}}_\gamma(\delta(x) \mid X = x) := \sum_{s=1}^\tau \sum_{l=0}^{d-1} 2\hat{q}_{js}(x)\mathbb{P}(\hat{\Gamma}_{-js}(x) = l) + \sum_{l=0}^{d} \gamma^l \mathbb{P}(\hat{\Gamma}(X) = l)
\]

\[
= \sum_{s=1}^\tau 2\hat{q}_{js}(x)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \hat{\Gamma}_{-js}(x)}\right) + \gamma \mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \hat{\Gamma}(x)}\right)
\]

\[
\text{Dice}_\gamma(\delta(x) \mid X = x) := \sum_{s=1}^\tau \sum_{l=0}^{d-1} 2p_{js}(x)\mathbb{P}(\Gamma_{-js}(x) = l) + \sum_{l=0}^{d} \gamma^l \mathbb{P}(\Gamma(X) = l)
\]

\[
= \sum_{s=1}^\tau 2p_{js}(x)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \Gamma_{-js}(x)}\right) + \gamma \mathbb{E}\left(\frac{1}{\tau + \gamma + \Gamma(x)}\right).
\]

Now, we have

\[
\left| \hat{\text{Dice}}_\gamma(\delta(x) \mid X = x) - \text{Dice}_\gamma(\delta(x) \mid X = x) \right|
\]

\[
\leq 2 \sum_{s=1}^\tau \left( \hat{q}_{js}(x)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \hat{\Gamma}_{-js}(x)}\right) - p_{js}(x)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \Gamma_{-js}(x)}\right) \right)
\]

\[
+ \gamma \left| \mathbb{E}\left(\frac{1}{\tau + \gamma + \hat{\Gamma}(x)}\right) - \mathbb{E}\left(\frac{1}{\tau + \gamma + \Gamma(x)}\right) \right|
\]

\[
\leq 2 \sum_{s=1}^\tau \left( \hat{q}_{js}(x)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \hat{\Gamma}_{-js}(x)}\right) - \mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \hat{\Gamma}_{-js}(x)}\right) \right)
\]

\[
+ 2 \sum_{s=1}^\tau \left( \hat{q}_{js}(x) - p_{js}(x) \right)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \hat{\Gamma}_{-js}(x)}\right)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \hat{\Gamma}_{-js}(x)}\right)
\]

\[
\leq 2 \sum_{s=1}^\tau \left( \hat{q}_{js}(x) - p_{js}(x) \right)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \Gamma_{-js}(x)}\right)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \Gamma_{-js}(x)}\right)
\]

\[
+ \gamma \left| \mathbb{E}\left(\frac{1}{\tau + \gamma + \hat{\Gamma}(x)}\right) - \mathbb{E}\left(\frac{1}{\tau + \gamma + \Gamma(x)}\right) \right|
\]

\[
\leq 2 \sum_{s=1}^\tau \left( \hat{q}_{js}(x) - p_{js}(x) \right)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \Gamma_{-js}(x)}\right)\mathbb{E}\left(\frac{1}{\tau + \gamma + 1 + \Gamma_{-js}(x)}\right)
\]

\[
+ \gamma \left| \mathbb{E}\left(\frac{1}{\tau + \gamma + \hat{\Gamma}(x)}\right) - \mathbb{E}\left(\frac{1}{\tau + \gamma + \Gamma(x)}\right) \right|
\]

\[
\leq \left( \frac{3}{2(1+\gamma)} + c_1 \right) \|q(x) - p(x)\|_1,
\]

where the second inequality follows from the triangle inequality, \( c_1 = 0 \) if \( \gamma = 0 \), \( c_1 = 1/\gamma \) if \( \gamma > 0 \). Therefore,

\[
\text{Dice}_\gamma(\delta^*) - \text{Dice}_\gamma(\hat{\delta}) = \text{Dice}_\gamma(\delta^*) - \text{Dice}_\gamma(\hat{\delta}^*) + \text{Dice}_\gamma(\hat{\delta}^*) - \text{Dice}_\gamma(\hat{\delta}) - \text{Dice}_\gamma(\hat{\delta}) - \text{Dice}_\gamma(\hat{\delta})
\]

\[
\leq \text{Dice}_\gamma(\delta^*) - \text{Dice}_\gamma(\hat{\delta}^*) + \text{Dice}_\gamma(\hat{\delta}) - \text{Dice}_\gamma(\hat{\delta})
\]

\[
\leq \mathbb{E}_x\left| \text{Dice}_\gamma(\delta^*(X) \mid X) - \text{Dice}_\gamma(\delta^*(X) \mid X) \right| + \mathbb{E}_x\left| \text{Dice}_\gamma(\delta(X) \mid X) - \text{Dice}_\gamma(\delta(X) \mid X) \right|
\]

\[
\leq \left( \frac{3}{1+\gamma} + 2c_1 \right) \mathbb{E}\|q(X) - p(X)\|_1,
\]
where the first inequality follows from the definition of $\delta$ such that $\hat{\text{Dice}}_j(\delta^*) - \hat{\text{Dice}}_j(\delta) \leq 0$. This completes the proof.

### D.10 Proof of Corollary 12

**Proof** According to the Pinsker’s inequality, we have

$$
\mathbb{E}_x \|q(x) - p(x)\|_1 = \sum_{j=1}^{d} \mathbb{E}_x |\hat{q}_j(X) - p_j(X)| \leq \sum_{j=1}^{d} \mathbb{E}_x \sqrt{\frac{1}{2} \text{KL}(p_j(X), \hat{p}(Y_j|X))}
$$

$$
\leq \sqrt{\frac{d}{2} \sum_{j=1}^{d} \mathbb{E}_x \text{KL}(P(Y_j|X), \hat{P}(Y_j|X))}
$$

$$
= \sqrt{\frac{d}{2} \mathbb{E} \left(l_{CE}(Y, \hat{q}(X))\right) - \mathbb{E} \left(l_{CE}(Y, p(X))\right)},
$$

where the last inequality follows from the Cauchy-Schwarz inequality and the Jensen’s inequality, and $\text{KL}(\hat{P}(Y_j|X), P(Y_j|X)) := p_j(x) \log(p_j(x)/\hat{q}_j(x)) + (1 - p_j(x)) \log((1 - p_j(x))/(1 - \hat{q}_j(x)))$ is the KL divergence between $\hat{P}(Y_j|X)$ under $q$ and $P(Y_j|X)$ under $\hat{q}$. The desirable result then follows by combining (19) in Theorem 11. This completes the proof.

### D.11 Proof of Lemma 13

**Proof** Denote $y_I = (y_j : j \in I)^T$, $y_{\bar{I}} = (y_j : j \notin I)^T$, and let $I(v) = I(\delta(x)) = \{j : v_j = 1\}$ be a segmentation index set, and $\|v\|_1 = \tau$. Again, consider the point-wise maximization:

$$
\delta^*(x) = \arg\max_{v \in \{0,1\}^d} \text{IoU}_y(v|x),
$$

where $\text{IoU}_y(v|x)$ is defined as

$$
\text{IoU}_y(v|x) = \mathbb{E} \left( \frac{\|Y_{I(v)}\|_1 + \gamma}{\|Y_{\bar{I}(v)}\|_1 + \|v\|_1 + \gamma} \bigg| X = x \right)
$$

$$
= \mathbb{E} \left( \frac{\|Y_{I(v)}\|_1}{\|Y_{\bar{I}(v)}\|_1 + \tau + \gamma} \bigg| X = x \right) + \mathbb{E} \left( \frac{\gamma}{\|Y_{\bar{I}(v)}\|_1 + \tau + \gamma} \bigg| X = x \right)
$$

$$
= \left( \mathbb{E}(\|Y_{I(v)}\|_1 | X = x) + \gamma \right) \mathbb{E} \left( \frac{1}{\|Y_{\bar{I}(v)}\|_1 + \tau + \gamma} \bigg| X = x \right),
$$

where the last equality follows from the fact that $Y_{I(v)} \perp Y_{\bar{I}(v)} | X$. Fix $\tau = \|v\|_1$. The first term is maximized at $v^* = (v_1^*, \cdots, v_d^*)$ with

$$
v_j^* = \begin{cases} 
1 & \text{if } p_j(x) \text{ ranks top } \tau, \\
0 & \text{otherwise},
\end{cases}
$$

where $\tau$ is the top-$\tau$ segmentation index set.
and the maximum value is $\sum_{j \in J_t(x)} p_j(x) + \gamma$. The second term is

$$\mathbb{E}\left(\frac{1}{\|Y_{\mathcal{J}(\tau)}\|_1 + \tau + \gamma} \mid X = x\right) = \mathbb{E}\left(\frac{1}{\Gamma_{-\mathcal{J}(\tau)}(x) + \tau + \gamma}\right).$$

Given $I(\nu) \neq I(\nu^*)$, we have $\Gamma_{-\mathcal{J}(\tau)}(x) = \sum_{j=1}^{d-\tau} B_j$ and $\Gamma_{-\mathcal{J}(\nu^*)}(x) = \sum_{j=1}^{d-\tau} B_j^*$, where $B_j$ and $B_j^*$ are independent Bernoulli variables with success probability $p_j \geq p_j^*$, respectively, for $j = 1, \ldots, d - \tau$. By Theorem 2.2.9 of Belzunce et al. (2015), $\Gamma_{-\mathcal{J}(\nu)}(x)$ is stochastically greater than $\Gamma_{-\mathcal{J}(\nu^*)}(x)$, namely

$$\mathbb{P}(\Gamma_{-\mathcal{J}(\nu)}(x) \leq \zeta) \leq \mathbb{P}(\Gamma_{-\mathcal{J}(\nu^*)}(x) \leq \zeta)$$

for any $\zeta \in \mathbb{R}$. Thus,

$$\mathbb{E}\left(\frac{1}{\Gamma_{-\mathcal{J}(\nu)}(x) + \tau + \gamma}\right) \leq \mathbb{E}\left(\frac{1}{\Gamma_{-\mathcal{J}(\nu^*)}(x) + \tau + \gamma}\right) = \mathbb{E}\left(\frac{1}{\Gamma_{-\mathcal{J}(\nu^*)}(x) + \tau + \gamma}\right).$$

As a result, the second term is also maximized at $\nu^*$. Therefore, $\nu^*$ maximizes $\text{IoU}_\gamma(\nu|x)$ for a fixed $\tau$. The desired result follows.

**D.12 Proof of Lemma 14**

**Proof** Without loss of generality, assuming $\tilde{q}_1(x) \geq \cdots \geq \tilde{q}_d(x)$ are fixed, then for any $\tau' > \tau$, $\sigma_\tau(x) - \sigma_{\tau'}(x)$

$$= \left(\sum_{j=1}^{\tau} \tilde{q}_j(x) + \gamma\right) \mathbb{E}\left(\frac{1}{\Gamma_{-\mathcal{J}(\tau)}(x) + \tau + \gamma}\right) - \left(\sum_{j=1}^{\tau'} \tilde{q}_j(x) + \gamma\right) \mathbb{E}\left(\frac{1}{\Gamma_{-\mathcal{J}(\tau')} (x) + \tau' + \gamma}\right)$$

$$= \left(\sum_{j=1}^{\tau} \tilde{q}_j(x) + \gamma\right) \mathbb{E}\left(\frac{1}{\Gamma_{-\mathcal{J}(\tau)}(x) + \sum_{j=\tau+1}^{\tau'} B_j(x) + \tau + \gamma}\right) - \left(\sum_{j=1}^{\tau'} \tilde{q}_j(x) + \gamma\right) \mathbb{E}\left(\frac{1}{\Gamma_{-\mathcal{J}(\tau')} (x) + \tau' + \gamma}\right)$$

$$= \mathbb{E}\left(\frac{\sum_{j=1}^{\tau} \tilde{q}_j(x) + \gamma(\tau' - \tau - \sum_{j=\tau+1}^{\tau'} B_j(x))}{\Gamma_{-\mathcal{J}(\tau)}(x) + \sum_{j=\tau+1}^{\tau'} B_j(x) + \tau + \gamma}\right) \geq \mathbb{E}\left(\frac{\sum_{j=1}^{\tau} \tilde{q}_j(x) + \gamma(\tau' - \tau - \sum_{j=\tau+1}^{\tau'} \tilde{q}_j(x))}{\Gamma_{-\mathcal{J}(\tau)}(x) + \sum_{j=\tau+1}^{\tau'} \tilde{q}_j(x) + \tau + \gamma}\right) - \frac{(d - \tau') \tilde{q}_{\tau+1}(x) + \gamma}{\tau' + \gamma} \geq 0,$$

whenever

$$\sum_{j=1}^{\tau} \tilde{q}_j(x) + \gamma \geq \frac{\tilde{q}_{\tau+1}(x)}{1 - \tilde{q}_{\tau+1}(x)} \max_{\tau' > \tau} \frac{(d - \tau') \tilde{q}_{\tau+1}(x) + \tau' + \gamma}{\tau' + \gamma} = \frac{\tilde{q}_{\tau+1}(x)}{1 - \tilde{q}_{\tau+1}(x)} \max\left(d + \gamma, \frac{(d - \tau') \tilde{q}_{\tau+1}(x) + \tau + \gamma}{\tau + \gamma}\right).$$
This completes the proof.

D.13 Proof of Theorem 15

Proof Similar to the proof of Theorem 11, we consider point-wise approximation of IoU metric under \( p \) and \( \hat{q} \). For any \( \delta \) such that \( \delta_j(x) = 1 \) for \( j \in J_t(x) \) and \( \delta_j(x) = 0 \) otherwise. Define

\[
\hat{\text{IoU}}_\gamma(\delta(x)|X = x) := \left( \sum_{s=1}^{\tau} \tilde{q}_j(x) + \gamma \right) \frac{\sum_{l=0}^{d-\tau} p(\tilde{r}_j(x)) = l}{\tau + l + \gamma}
\]

\[
= \left( \sum_{s=1}^{\tau} \tilde{q}_j(x) + \gamma \right) \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right),
\]

\[
\text{IoU}_\gamma(\delta(x)|X = x) := \left( \sum_{s=1}^{\tau} p_j(x) + \gamma \right) \frac{\sum_{l=0}^{d-\tau} p(\tilde{r}_j(x)) = l}{\tau + l + \gamma}
\]

\[
= \left( \sum_{s=1}^{\tau} p_j(x) + \gamma \right) \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right).
\]

We have

\[
\left| \text{IoU}_\gamma(\delta(x)|X = x) - \text{IoU}_\gamma(\delta(x)|X = x) \right|
\]

\[
\leq \left| \sum_{s=1}^{\tau} \left( \tilde{q}_j(x) \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right) - p_j(x) \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right) \right) \right|
\]

\[
+ \gamma \left| \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right) - \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right) \right|
\]

\[
\leq \sum_{s=1}^{\tau} \left( \tilde{q}_j(x) \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right) \right)
\]

\[
+ \sum_{s=1}^{\tau} \left( \tilde{q}_j(x) - p_j(x) \right) \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right)
\]

\[
+ \gamma \left| \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right) - \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right) \right|
\]

\[
\leq \sum_{s=1}^{\tau} \left( \tilde{q}_j(x) - p_j(x) \right) \mathbb{E}\left( \frac{1}{\tau + \gamma + 1} \right)
\]

\[
+ \sum_{s=1}^{\tau} \left( \tilde{q}_j(x) - p_j(x) \right) \mathbb{E}\left( \frac{1}{\tau + \gamma + 1} \right)
\]

\[
+ \gamma \left| \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right) - \mathbb{E}\left( \frac{1}{\tau + \gamma + \tilde{r}_j(x)} \right) \right|
\]

\[
\leq C_2 \left\| \tilde{q}(x) - p(x) \right\|_1,
\]

where \( C_2 \) is a constant that may depend on \( \gamma \). Then the rest of the proof follows from the same arguments for Theorems 11 and 12 with Dice replaced by IoU, and is omitted.
Appendix E. Auxiliary Lemmas

Lemma 16 Suppose $\Gamma = \sum_{j=1}^{d} B_j$ is a Poisson binomial random variable, and $B_j (j = 1, \cdots, d)$ are independent Bernoulli trials with success probabilities $p_1, \cdots, p_d$. Then, for any $j = 1, \cdots, d$, we have

$$\mathbb{P}(\Gamma_j \leq l - 1) \leq \mathbb{P}(\Gamma \leq l) \leq \mathbb{P}(\Gamma_j \leq l),$$

where $\Gamma_j = \sum_{j \neq j^*} B_j$.

Proof Note that $\Gamma = \Gamma_j + B_j$, then

$$\mathbb{P}(\Gamma \leq l) = \mathbb{P}(\Gamma_j + B_j \leq l) = \mathbb{P}(\Gamma_j \leq l \mid B_j = 0)(1 - p_j) + \mathbb{P}(\Gamma_j \leq l - 1 \mid B_j = 1)p_j$$

$$= \mathbb{P}(\Gamma_j \leq l)(1 - p_j) + \mathbb{P}(\Gamma_j \leq l - 1)p_j,$$

where the last equality follows from the fact that $B_j \perp B_{j^*}$ for $j \neq j^*$. The desirable result then follows since $\mathbb{P}(\Gamma_j \leq l) \geq \mathbb{P}(\Gamma_j \leq l - 1)$.

Lemma 17 Suppose $\Gamma = \sum_{j=1}^{d} B_j$ is a Poisson binomial random variable, and $B_j (j = 1, \cdots, d)$ are independent Bernoulli trials with success probabilities $p_1 \geq p_2 \geq \cdots \geq p_d$. Then, for an arbitrary positive non-decreasing sequence $\xi_l (l = 0, \cdots, d - 1)$,

$$Q_j = \sum_{l=0}^{d-1} \frac{1}{\xi_l} \mathbb{P}(\Gamma_j = l)$$

is a non-increasing function with respect to $j = 1, \cdots, d$.

Proof Note that for any $j \neq j^*$, $\Gamma_j = \sum_{i \neq j^*} B_i = \sum_{i \neq j^*} B_i + B_j = B_j$ + $\Gamma_j$. Denote $\xi_{j, j^*, l} = \mathbb{P}(\Gamma_{j^*, l} = l)$, we have

$$\mathbb{P}(\Gamma_j \leq l) = (1 - p_j)\xi_{j, j^*, l} + p_j\xi_{j, j^*, l - 1}.$$

Then,

$$Q_j - Q_{j^*} = \sum_{l=0}^{d} \frac{1}{\xi_l}((1 - p_j)\xi_{j, j^*, l} - \sum_{l=0}^{d} \frac{1}{\xi_l} \xi_{j, j^*, l} - (1 - p_j)\xi_{j, j^*, l} - p_j\xi_{j, j^*, l - 1})$$

$$= (p_j - p_{j^*})\left(\sum_{l=0}^{d} \frac{1}{\xi_l} \xi_{j, j^*, l} - \sum_{l=0}^{d} \frac{1}{\xi_l} \xi_{j, j^*, l - 1}\right) = (p_j - p_{j^*})\left(\sum_{l=0}^{d} \left(\frac{1}{\xi_l} - \frac{1}{\xi_{l+1}}\right) \xi_{j, j^*, l}\right),$$

where the first equality follows from the fact that $\mathbb{P}(\Gamma_j = d) = 0$, and the last equality follows from the fact that $\xi_{j, j^*, l} = 0$ for $l < 0$ or $l = d$. Hence, we have $Q_j - Q_{j^*}$ has the same sign with $p_j - p_{j^*}$, and the desirable result then follows.

Lemma 18 Let $\Gamma$ be a Poisson-binomial random variable with the success probability $(p_j)_{j=1, \cdots, d}$, then for any $\tau \geq 1$, we have

$$\left(\sum_{j=1}^{d} p_j + \tau\right)^{-1} \leq \mathbb{E}\left(\frac{1}{\Gamma + \tau}\right) \leq \left(\frac{d + 1}{d}\right) \sum_{j=1}^{d} p_j + \tau - 1^{-1}.$$
**Proof** According to the corollary in Chao and Strawderman (1972), we have

\[
\mathbb{E}\left(\frac{1}{\Gamma + \tau}\right) = \int_0^1 t^\tau - 1 P_1(t) dt = \int_0^1 t^{\tau - 1} \left( \prod_{j=1}^d (1 - p_j + p_j t) \right) dt \leq \int_0^1 t^{\tau - 1} (1 - \bar{p} + \bar{p} t)^d dt
\]

\[
= \mathbb{E}\left(\frac{1}{\Lambda + \tau}\right) \leq \left( \frac{d + 1}{d} \sum_{j=1}^d p_j + \tau - 1 \right)^{-1},
\]

where \( \bar{p} = d^{-1} \sum_{j=1}^d p_j \), the first inequality follows from the inequality of arithmetic and geometric means, the last equality follows from Section 3.1 of Chao and Strawderman (1972), and the last inequality follows from (25) in Wooff (1985). This completes the proof. \( \blacksquare \)
References


RANKSEG: A CONSISTENT RANKING-BASED FRAMEWORK FOR SEGMENTATION


47


