TWO-SIDED SAFE SCREENING FOR SCALABLE TVERSKY OPTIMIZATION IN SEGMENTATION DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Image segmentation is a fundamental task in computer vision. Segmentation networks typically output pixel- or voxel-level probabilities, whereas overlap measures evaluate discrete prediction masks. Metric-aware decoding uses these probabilities to optimize overlap measures, but exact optimization of the resulting decoding objective can be computationally demanding for large images and volumes. We study scalable decoding for a broad family of Tversky-type objectives, including Dice, intersection over union, and F-scores, under a plug-in expected linear-fractional (ELF) criterion defined by a ratio of expected confusion counts. We characterize the ranking-induced structure of its maximizers and derive a pair of complementary, closed-form safe-screening certificates: an upper-threshold certificate identifies coordinates that must be selected, whereas a data-adaptive lower-threshold certificate identifies coordinates that must be excluded before downstream exact optimization. Together, these two-sided certificates reduce the effective problem dimension while preserving a global optimizer of this criterion. We instantiate the screening rules in an efficient GPU implementation with a sorting-based downstream backend. Experiments on synthetic probability maps and native medical volumes demonstrate substantial reductions in runtime and working memory at large scales.

1 INTRODUCTION

Image segmentation assigns a semantic label to each pixel or voxel and is a fundamental dense-prediction task in computer vision. The resulting masks delineate objects, regions, or anatomical structures and support scene understanding and downstream decision making across applications. This broad importance has motivated influential architectures, including FCN (Long et al., 2015) and DeepLab (Chen et al., 2018). Medical imaging has also motivated specialized volumetric and self-configuring systems such as U-Net (Ronneberger et al., 2015) and nnU-Net (Isensee et al., 2021). Modern segmentation networks typically output a foreground probability at each pixel or voxel, but performance is usually assessed after discretization using overlap measures such as Dice (Dice, 1945) and intersection over union (IoU), or their asymmetric generalization, the Tversky index (Tversky, 1977; Taha & Hanbury, 2015; Salehi et al., 2017; Wang et al., 2023).

Converting the probability map into a mask remains part of the statistical decision rule rather than a purely cosmetic post-processing step (Bartlett et al., 2006; Gneiting & Raftery, 2007). The conventional cutoff at $1/2$ is aligned with coordinatewise zero-one loss, but generally does not maximize a nondecomposable overlap measure. For Dice and IoU, RankSEG addresses this mismatch through a statistically consistent, metric-aware decoder (Dai & Li, 2023; Wang & Dai, 2025): decoding ranks the d foreground probabilities and selects an image-specific top-τ prefix. This replaces a search over 2^d binary masks by a search over $d+1$ nested candidates, but requires a full sort before the optimal prediction size is known. The full sort can become a material bottleneck in high-dimensional segmentation. For example, a typical 512^3 volume contains more than 130 million coordinates, and a native medical volume may be even larger. At this scale, sorting can dominate decoder runtime and, particularly on a graphics processing unit, require several gigabytes of working memory. This motivates a broader computational question: can coordinates on both sides of the unknown optimal cutoff be certified before the metric-aware decoding backend processes them? We pursue this ques-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

tion through two-sided safe screening. Complementary certificates fix high-probability coordinates as selected and low-probability coordinates as excluded while preserving an optimizer of the target objective, leaving only the unresolved middle set for the downstream solver. When instantiated with sort-prefix optimization, only this unresolved set needs to be sorted.

1.1 Literature Review

Ranking and prefix decision rules. For Tversky-type index (TI) metrics and several related confusion-matrix metrics, metric-aware plug-in rules often take a ranking-and-thresholding form. For calibrated binary probabilities, Lipton et al. (2014) show that an optimal F_1 cutoff is one half of the optimal F_1 value. Koyejo et al. (2014) establish threshold characterizations and consistent learning algorithms for a broader family of linear-fractional population metrics. In multilabel and structured prediction, expected F-measure maximization can be organized as a search over prediction cardinality with a ranking step for each candidate size. Under general label dependence, these ranking scores involve summaries of the joint distribution, so a single ordering of marginal probabilities need not suffice (Waegeman et al., 2014). Related work addresses general multiclass confusion-matrix objectives (Narasimhan et al., 2024) and multilabel metrics with a fixed prediction budget (Schultheis et al., 2024).

Metric-aware segmentation decoding. In segmentation, the prediction size is adapted to each image and class. RankSEG characterizes Dice/IoU Bayes rules under conditional independence: one ordering of the foreground probabilities generates nested candidate masks, whose scores involve expectations over Poisson-binomial counts (Dai & Li, 2023). RankSEG-RMA approximates the reciprocal moments in these scores, allowing all candidate sizes to be evaluated from cumulative probability sums in linear time after sorting (Wang & Dai, 2025). This simplifies score evaluation while retaining the image-specific prefix search and its initial probability ordering. Repeated sorting therefore remains a computational concern for these decoders at large spatial scales.

Linear-fractional optimization. With probabilities fixed, the ELF criterion is a linear-fractional function of the mask, connecting decoding to classical fractional programming. Dinkelbach's method solves fractional objectives through a sequence of parametric subproblems (Dinkelbach, 1967; Schaible, 1976). For a linear-fractional objective over unconstrained binary masks, these subproblems separate across coordinates, providing a sorting-free alternative to prefix search.

Safe screening. Safe screening formalizes a computational principle: an inexpensive certificate can remove coordinates without changing the solution. SAFE rules use optimality conditions and bounds on the dual solution to discard predictors guaranteed to have zero coefficients in Lasso-type problems (El Ghaoui et al., 2010; Liu et al., 2014). Strong rules screen more aggressively but require optimality checks because they may discard active predictors (Tibshirani et al., 2012). Dynamic refinements such as Gap Safe rules (Ndiaye et al., 2017) and Dynamic Sasvi (Yamada & Yamada, 2021) use dual regions updated as the solver progresses, while related working-set and dual methods restrict the variables visited by coordinate descent (Massias et al., 2020; Bertrand et al., 2022). Safe screening also extends to grouped, non-separable penalties (Bao et al., 2025). Beyond certifying inactivity, Shibagaki et al. (2016) develop *safe keeping* rules that certify active features and samples using primal and dual regions. These certificates avoid further screening tests on variables already known to be active, although certifying a nonzero coefficient does not determine its value. Screen & Relax combines screening with safe identification of nonzero coefficients to reduce the dimension of Elastic-Net problems (Guyard et al., 2022). Together, these approaches illustrate how complementary certificates can reduce computation by resolving both active and inactive status.

1.2 Motivation and Contribution

The general principle from the safe-screening literature suggests a way to reduce the cost of metric-aware decoding. Exact computational backends, including sort-prefix search and Dinkelbach iteration, typically process all d coordinates. Our central methodological idea is to construct complementary certificates on both sides of the unknown optimal cutoff, independently of the subsequent optimization backend. We implement this insight with a data-adaptive exclusion threshold from computable objective bounds, yielding an up-front partition into included, excluded, and unresolved coordinates that preserves a global optimizer while reducing runtime and workspace.

Our contributions are threefold. First, and most centrally, for the plug-in Tversky-type criterion formalized in Section 2, we derive complementary inclusion and exclusion certificates that jointly screen both tails of the probability ranking before exact optimization. These closed-form certificates preserve a global optimizer and are independent of the downstream exact backend. Second, we establish the supporting characterization of the maximizers of the general Tversky-type family and the ranking-induced prefix structure underlying both certificates. Third, we instantiate the certificates with vectorized full-sort optimization and assess their gains in simulations and full-volume biomedical segmentation; solver comparisons are reported in Section 4.3.

2 Tversky-Type Evaluation Metrics

Let $\mathbf{X} \in \mathbb{R}^d$ be an image, let $\mathbf{Y} \in \{0,1\}^d$ be the ground-truth foreground mask, and let $\mathbf{m}(\mathbf{X}) \in \{0,1\}^d$ be the corresponding predicted binary mask, where d is the number of pixels or voxels in a single image or volume. Here $d = \prod_i d_i$ is the product of the spatial dimensions and can range from millions of pixels in 2D images to billions of voxels in 3D volumes. The same formulation applies more generally to any set prediction unit with d coordinates, such as a positive set in binary classification or a label space in multilabel classification.

We focus on TI metrics, which cover the main evaluation criteria in segmentation. Segmentation commonly forms a TI metric within each image and then averages the resulting scores across images. Accordingly, metric-aware decoding is performed conditionally for each input image. For TI parameter $\theta = (\alpha, \beta, \gamma_N, \gamma_D)^\intercal$, define the image-level Tversky index

$$\mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m}) = \frac{\mathrm{TP}(\mathbf{Y}, \mathbf{m}) + \gamma_N}{\mathrm{TP}(\mathbf{Y}, \mathbf{m}) + \alpha\,\mathrm{FP}(\mathbf{Y}, \mathbf{m}) + \beta\,\mathrm{FN}(\mathbf{Y}, \mathbf{m}) + \gamma_D}. \tag{1}$$

Here α and β penalize false positives and false negatives, respectively. The constants γ_N and γ_D handle degenerate $0/0$ cases with an empty prediction and ground truth. With $\gamma_N = \gamma_D > 0$, the empty–empty case scores one (Tilborghs et al., 2022). The family in (1) recovers standard unsmoothed TIs when $\gamma_N = \gamma_D = 0$. Important cases are Dice/F_1 with $\theta = (1/2, 1/2, 0, 0)^\intercal$ and IoU/Jaccard with $\theta = (1, 1, 0, 0)^\intercal$. TP, FP, and FN are the image-level true positive, false positive, and false negative counts:

$$\mathrm{TP}(\mathbf{Y}, \mathbf{m}) = \mathbf{m}^\intercal \mathbf{Y}, \quad \mathrm{FP}(\mathbf{Y}, \mathbf{m}) = \mathbf{m}^\intercal (\mathbf{1} - \mathbf{Y}), \quad \mathrm{FN}(\mathbf{Y}, \mathbf{m}) = (\mathbf{1} - \mathbf{m})^\intercal \mathbf{Y}.$$

For each image $\mathbf{x}$, we assume that the conditional foreground probabilities are well estimated and are denoted by $\mathbf{p}(\mathbf{x}) = [p_1(\mathbf{x}), \ldots, p_d(\mathbf{x})]^\intercal$, where $p_j(\mathbf{x}) = \mathbb{P}(Y_j = 1 \mid \mathbf{X} = \mathbf{x})$; we seek a metric-aware prediction mask $\mathbf{m}(\mathbf{x})$ maximizing the conditional criterion from $\mathbf{p}(\mathbf{x})$:

$$\mathbf{m}(\mathbf{x}) \in \operatorname*{arg\,max}_{\mathbf{m} \in \{0,1\}^d} R_\theta(\mathbf{m} \mid \mathbf{x}). \tag{2}$$

We henceforth suppress the dependence on $\mathbf{x}$ in all notation, for example, writing $R_\theta(\mathbf{m}) = R_\theta(\mathbf{m}|\mathbf{x})$, $\mathbf{m} = \mathbf{m}(\mathbf{x})$ and $\mathbf{p} = \mathbf{p}(\mathbf{x})$. All subsequent results hold separately for each individual input $\mathbf{x}$. In our main text, we focus on the plug-in expected linear-fractional (ELF) criterion, which takes the ratio of the expected numerator and denominator:

$$\begin{aligned} R_\theta(\mathbf{m}) = \mathsf{ELF}[\mathrm{TI}_\theta](\mathbf{m}) &:= \frac{\mathbb{E}\big[\mathrm{TP}(\mathbf{Y}, \mathbf{m}) + \gamma_N\big]}{\mathbb{E}\big[\mathrm{TP}(\mathbf{Y}, \mathbf{m}) + \alpha\,\mathrm{FP}(\mathbf{Y}, \mathbf{m}) + \beta\,\mathrm{FN}(\mathbf{Y}, \mathbf{m}) + \gamma_D\big]} \\ &= \frac{s(\mathbf{m}) + \gamma_N}{s(\mathbf{m}) + \alpha\big[\|\mathbf{m}\|_1 - s(\mathbf{m})\big] + \beta\big[\mu - s(\mathbf{m})\big] + \gamma_D}, \end{aligned} \tag{3}$$

where $\mu = \|\mathbf{p}\|_1$ and $s(\mathbf{m}) = \mathbf{m}^\intercal \mathbf{p}$. Throughout the main text, exact decoding refers to exact optimization of R_θ, not to optimization of the expected random TI ratio. For Dice and IoU, practical reciprocal-moment approximations to the expectation of the random ratio produce objectives within the same linear-fractional family (Wang & Dai, 2025); in particular, Dice-RMA corresponds to $\theta = (1/2, 1/2, 0, 1/2)^\intercal$. Thus R_θ can be specified directly as a linear-fractional decision criterion and also includes tractable approximations to expected image-level metrics. Appendix D develops a separate, theoretical analysis for exact expected samplewise TI under conditional independence.

We impose the following conditions to ensure that the Tversky-type criterion (3) is well defined.

Assumption 1 (Tversky parameters)**.** *The parameters in (3) satisfy (i) $\alpha, \beta \geq 0$ and $\alpha + \beta > 0$; (ii) $0 \leq \gamma_N \leq \gamma_D$; and (iii) $\beta\mu + \gamma_D > 0$.*

Condition (i) gives the usual Tversky parameter restrictions: false-positive and false-negative penalties are nonnegative, and at least one is active. Condition (ii) ensures that the smoothed reward assigned to a $0/0$ case is at most one. Condition (iii) is a mild nondegeneracy requirement that makes the empty-mask score well defined under the expected linear-fractional criterion. For unsmoothed metrics with $\gamma_D = 0$, it requires $\beta\mu > 0$; in particular, for Dice, F-scores, and IoU, this reduces to $\mu > 0$. Thus, all-background prediction units require either positive denominator smoothing or a separate empty-mask convention. Accordingly, when $\mu > 0$, the metrics described above satisfy Assumption 1.

We next characterize the maximizers of (3). Algebraically, the results hold for any supplied $\mathbf{p} \in [0,1]^d$ without requiring calibration. For estimated $\mathbf{p}$, exactness concerns the corresponding estimated plug-in objective and does not address estimation error.

Proposition 1 (Ranking-induced prefix structure)**.** *Under Assumption 1, let $p_{j_1} \geq \cdots \geq p_{j_d}$ be the probabilities arranged in non-increasing order, where $j_1, \ldots, j_d$ is the corresponding permutation of $\{1, \ldots, d\}$. Define $s_\tau = \sum_{r=1}^{\tau} p_{j_r}$ for $\tau = 1, \ldots, d$, with $s_0 = 0$, and let*

$$\tau^* \in \operatorname*{arg\,max}_{\tau \in \{0,\ldots,d\}} \pi_\tau, \qquad \pi_\tau = \frac{s_\tau + \gamma_N}{\alpha\tau + \beta\mu + \gamma_D + (1-\alpha-\beta)s_\tau}, \quad \tau = 0, \ldots, d. \tag{4}$$

Then there exists an optimal prediction vector of the form

$$m_j^* = 1 \text{ if } p_j \text{ ranks among the top } \tau^*, \qquad m_j^* = 0 \text{ otherwise}.$$

Proposition 1 yields an exact sort-prefix backend for optimizing R_θ: $\mathbf{p}$ is sorted and the objective is searched over the prefix size $\tau = \|\mathbf{m}\|_1$. Related probability-ranking and cardinality-reduction arguments have appeared for F-measure optimization in binary, sequence, and multilabel prediction. The proposition consolidates these ideas for the general Tversky-type family considered here; see Section 1.1 for more detailed discussion.

3 Safe Screening for TI Metrics

The core construction is two-sided: an upper-threshold certificate fixes coordinates that must be included, while a complementary lower-threshold certificate removes coordinates that must be excluded. The pair is independent of the downstream optimization backend and preserves a global optimizer. Both full-sort search and Dinkelbach iteration are exact methods for optimizing R_θ. We use sort-prefix search in the main implementation and compare both backends, with and without screening, in Section 4.3.

Specifically, after sorting the probabilities, compute $\boldsymbol{\tau} = (0, \ldots, d)^\intercal$ and $\mathbf{s} = (s_0, \ldots, s_d)^\intercal$, where $s_0 = 0$ and $s_\tau = \sum_{r=1}^{\tau} p_{j_r}$. All prefix scores can be computed in vectorized form in a parallel GPU implementation, and τ^* is obtained by a one-dimensional argmax:

$$\boldsymbol{\pi} = \frac{\mathbf{s} + \gamma_N \mathbf{1}}{\alpha\boldsymbol{\tau} + \beta\mu\mathbf{1} + \gamma_D\mathbf{1} + (1-\alpha-\beta)\mathbf{s}}, \qquad \tau^* \in \operatorname*{arg\,max}_{\tau \in \{0,\ldots,d\}} \pi_\tau. \tag{5}$$

For large segmentation tensors, however, sorting can dominate runtime and create substantial GPU memory pressure. This motivates screening both ends of the probability ranking before sorting. We seek complementary certificates for an optimal prefix:

$$p_j < p_L \Rightarrow j \text{ is excluded}, \qquad p_j > p_H \Rightarrow j \text{ is included}.$$

Only the unresolved middle set then enters the sort and prefix search. We next derive the upper inclusion threshold p_H and the lower exclusion threshold p_L for the TI prefix scores.

Lemma 1 (Marginal improvement for TI prefixes)**.** *Under Assumption 1, if $p_{j_1} \geq \cdots \geq p_{j_d}$, then for any $\tau = 0, \ldots, d-1$,*

$$\pi_{\tau+1} > \pi_\tau \iff p_{j_{\tau+1}} > \frac{\alpha\pi_\tau}{1-(1-\alpha-\beta)\pi_\tau} \iff p_{j_{\tau+1}} > \frac{\alpha\pi_{\tau+1}}{1-(1-\alpha-\beta)\pi_{\tau+1}}.$$

The same equivalence holds with $>$ replaced by $\geq$.

The function $u \mapsto \alpha u/\{1-(1-\alpha-\beta)u\}$ is the specialization of the optimal-threshold formula for linear-fractional metrics in Koyejo et al. (2014). Our use of it is computational: bounds on the unknown optimal value produce certificates before the probabilities are sorted.

Lemma 2 (Unimodality). *Under Assumption 1, $\tau \mapsto \pi_\tau$ is unimodal on $\{0,\dots,d\}$. That is, if $\pi_{\tau+1} \le \pi_\tau$ for some $\tau = 0,\dots,d-1$, then $\pi_{\tau'+1} \le \pi_{\tau'}$, for all $\tau' = \tau,\dots,d-1$.*

Corollary 1 (Upper inclusion threshold). *Under Assumption 1, every coordinate in*

$$H = \left\{ j : p_j > p_H := \frac{\alpha}{\alpha+\beta} \right\}$$

is included in the optimal mask $\mathbf{m}^*$ of Proposition 1, that is, $m_j^* = 1$.

The bound $\pi_{\tau^*} \le 1$ yields the inclusion threshold p_H through Lemma 1. For exclusion, the largest score among the maximum-probability singleton, the mask selecting H, and the full mask provides a lower bound on the optimal score, computable without sorting.

Corollary 2 (Lower exclusion threshold). *Under Assumption 1, denote*

$$s_H = \sum_{j\in H} p_j, \quad p_{\max} = \max_{1\le j\le d} p_j, \quad D(\tau) = \alpha\tau + \beta\mu + \gamma_D - (1-\alpha-\beta)\gamma_N.$$

Then every coordinate in

$$L = \left\{ j : p_j < p_L := \min\left[p_{\max}, \max\left\{ \frac{\alpha(p_{\max}+\gamma_N)}{D(1)}, \frac{\alpha(s_H+\gamma_N)}{D(|H|)}, \frac{\alpha(\mu+\gamma_N)}{D(d)} \right\} \right] \right\}$$

is excluded from the optimal mask $\mathbf{m}^*$ of Proposition 1; that is, $m_j^* = 0$.

Corollary 1 supplies the inclusion side, while Corollary 2 supplies the complementary exclusion side. Together they certify H as included in and L as excluded from an optimal prefix. Only the remaining coordinates are sorted, with H treated as a fixed offset.

Safe-screened sort-prefix backend. Let $M = [d] \setminus (H \cup L)$ be the unresolved set. After forming H, L, M, sort the unresolved coordinates as $p_{k_1} \ge \cdots \ge p_{k_{|M|}}$ for $k_\ell \in M$. Denote $r_0 = 0$ and $r_\ell = \sum_{v=1}^{\ell} p_{k_v}$. Selecting H and the first ℓ unresolved coordinates gives the same score as the corresponding full-sort prefix:

$$\pi_{|H|+\ell} = \frac{s_H + r_\ell + \gamma_N}{\alpha(|H|+\ell) + \beta\mu + \gamma_D + (1-\alpha-\beta)(s_H + r_\ell)}, \qquad \ell = 0,\dots,|M|. \tag{6}$$

A GPU cumulative sum computes all r_ℓ, after which the remaining prefix scores are evaluated through elementwise vector operations.

Algorithm 1: Sort-prefix optimization with two-sided safe screening for TI metrics

Input: Estimated conditional probabilities $\mathbf{p}$ and parameters $(\alpha, \beta, \gamma_N, \gamma_D)$.
Compute μ, $p_{\max} = \max_{j=1,\dots,d} p_j$, and p_H from Corollary 1;
Form $H = \{j : p_j > p_H\}$ and compute $s_H = \sum_{j\in H} p_j$;
Compute p_L from Corollary 2, then form $L = \{j : p_j < p_L\}$ and $M = [d] \setminus (H \cup L)$;
Sort only the unresolved coordinates: $p_{k_1} \ge \cdots \ge p_{k_{|M|}}$ for $k_\ell \in M$;
Compute the cumulative sums r_ℓ and the remaining prefix scores by (6), then choose $\ell^* \in \arg\max_{\ell\in\{0,\dots,|M|\}} \pi_{|H|+\ell}$;
Output: The mask selecting $H \cup \{k_1,\dots,k_{\ell^*}\}$.

Computational cost and workspace. By Corollaries 1 and 2, Algorithm 1 returns a global maximizer of R_θ: two-sided screening fixes only certified coordinate decisions, and the remaining exact prefix search therefore preserves an optimizer of the original problem. Scanning and forming H, L, and M take $O(d)$ time. With comparison-based sorting, the total cost is $O\{d+|M|\log|M|\}$, compared with $O(d\log d)$ without screening. The auxiliary sorting workspace is $O(|M|)$ rather than $O(d)$, beyond the $O(d)$ storage required for the input probabilities and dense output mask. PyTorch's large-vector CUDA radix sort still requires linear-size index and temporary buffers. The same certificates restrict Dinkelbach iteration to the unresolved coordinates while treating H as fixed; Section 4.3 compares the two backends before and after screening.

4 EXPERIMENTS

4.1 SIMULATION STUDIES

Simulation design. We use oracle conditional probabilities to isolate criterion optimization from estimation error. Each replication draws d independent probabilities from one of six distributions in Hong (2013), spanning sparse/dense and balanced/imbalanced settings; and the corresponding densities are shown in Fig. 1.

(a) $\mathrm{Unif}(0,1)$, (b) $\mathrm{Beta}(0.1,3)$, (c) $\mathrm{Beta}(3,0.1)$,
(d) $\frac{1}{2}\mathrm{Beta}(3,0.1)+\frac{1}{2}\mathrm{Beta}(0.1,3)$, (e) $\mathrm{Beta}(3,3)$, (f) $\frac{1}{2}\mathrm{Beta}(3,10)+\frac{1}{2}\mathrm{Beta}(10,3)$.

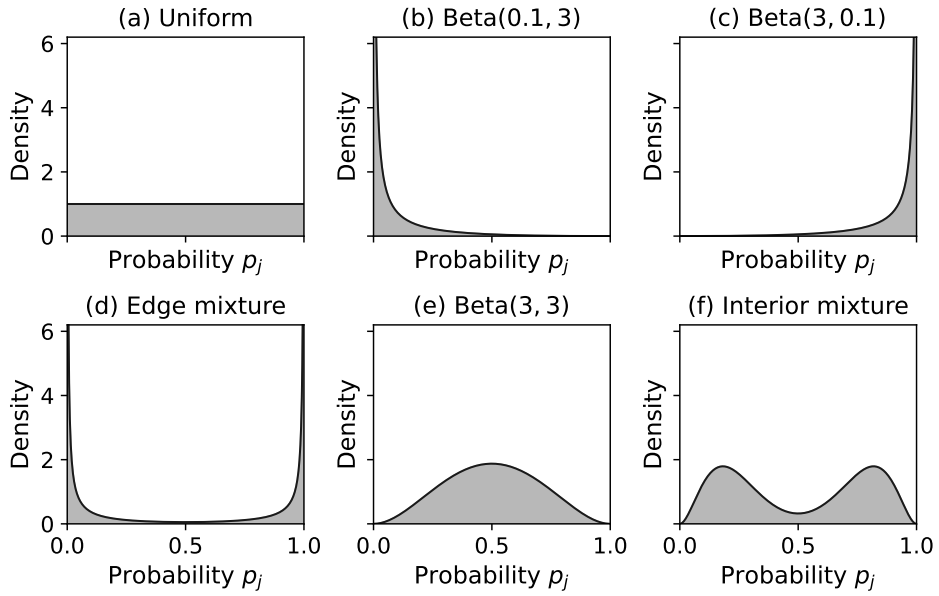

Figure 1: Probability densities used in the simulation study.

Given $\mathbf{p}$, the labels are generated independently as $Y_j \sim \mathrm{Bernoulli}(p_j)$ $(j = 1,\ldots,d)$. The main GPU scaling benchmarks optimize the unsmoothed Dice/F_1 instance of R_θ, corresponding to $(\alpha,\beta,\gamma_N,\gamma_D) = (1/2,1/2,0,0)$. In every comparison, the full-sort and safe-screened implementations receive the same $\mathbf{p}$ and optimize the same criterion; screening changes only the computation. The main benchmarks use the screened sort-prefix implementation. Appendix B.3 further shows runtime and working-memory reductions across asymmetric TI settings, with objective values matching full sort to recorded precision. For completeness, Appendix B.4 also compares the TI-optimal rule with the conventional 0.5 cutoff.

Results. We compare full-sort and safe-screened implementations on an NVIDIA RTX 3090 GPU for $d \in \{512^2, 1024^2, 512^3\}$ under the six probability distributions defined above. In the screened implementation, the same downstream sorting step is applied only to the unresolved coordinates. Each setting uses 20 paired replications. GPU working memory is measured as the peak additional GPU allocation during prediction, excluding the resident probability tensor and CUDA context. Further benchmarking details are provided in Appendix B.5. Table 1 reports runtime and working memory; Section 4.3 examines the certificate decomposition and screening across exact solvers.

Conclusion. The optimized screened implementation yields 1.02–1.32-fold speedups for 512^2 inputs, 2.28–3.65-fold speedups for 1024^2 inputs, and 4.04–14.85-fold speedups for 512^3 inputs, with memory savings of 76.27%–94.44%. These are end-to-end implementation gains; the matched-backend ablation in Appendix B.2 attributes 2.51–6.49-fold speedups at 512^3 to screening alone. Gains are greatest when few coordinates remain to be sorted.

Table 1: Detailed GPU runtime and peak working-memory comparison between full-sort optimization and safe screening. Entries are means over 20 paired replications; runtime is reported as mean ± standard deviation. The safe-screening columns report the optimized implementation; Appendix B.2 separates screening and implementation gains. Scr. is $100(1 - |M|/d)$.

		Full sort		Safe screening (ours)			
Shape	Case	Time	Mem.	Time	Mem.	Scr. (%)	**Speedup**
512^2	(a)	0.32 ± 0.00 ms	18.00 MiB	0.30 ± 0.00 ms	2.28 MiB	87.51	1.07
	(b)	0.32 ± 0.00 ms	18.00 MiB	0.30 ± 0.00 ms	1.75 MiB	91.89	1.09
	(c)	0.32 ± 0.00 ms	18.00 MiB	0.25 ± 0.01 ms	1.00 MiB	99.96	1.28
	(d)	0.32 ± 0.00 ms	18.00 MiB	0.24 ± 0.00 ms	1.00 MiB	99.92	1.32
	(e)	0.32 ± 0.00 ms	18.00 MiB	0.31 ± 0.00 ms	4.27 MiB	70.99	1.02
	(f)	0.32 ± 0.00 ms	18.00 MiB	0.30 ± 0.00 ms	1.36 MiB	95.12	1.08
1024^2	(a)	0.97 ± 0.00 ms	72.00 MiB	0.36 ± 0.00 ms	9.05 MiB	87.50	2.73
	(b)	0.98 ± 0.00 ms	72.00 MiB	0.35 ± 0.00 ms	6.92 MiB	91.91	2.82
	(c)	0.95 ± 0.00 ms	72.00 MiB	0.27 ± 0.00 ms	4.00 MiB	99.96	3.56
	(d)	0.96 ± 0.00 ms	72.00 MiB	0.26 ± 0.00 ms	4.00 MiB	99.92	3.65
	(e)	0.97 ± 0.00 ms	72.00 MiB	0.42 ± 0.00 ms	17.02 MiB	70.98	2.28
	(f)	0.97 ± 0.00 ms	72.00 MiB	0.34 ± 0.00 ms	5.38 MiB	95.11	2.89
512^3	(a)	115.13 ± 0.10 ms	9.00 GiB	16.61 ± 0.04 ms	1.13 GiB	87.50	6.93
	(b)	115.90 ± 0.07 ms	9.00 GiB	13.69 ± 0.04 ms	882.98 MiB	91.93	8.46
	(c)	114.32 ± 0.08 ms	9.00 GiB	7.76 ± 0.01 ms	512.00 MiB	99.96	14.74
	(d)	115.55 ± 0.07 ms	9.00 GiB	7.78 ± 0.02 ms	512.00 MiB	99.92	14.85
	(e)	115.26 ± 0.07 ms	9.00 GiB	28.51 ± 0.03 ms	2.13 GiB	70.99	4.04
	(f)	115.23 ± 0.08 ms	9.00 GiB	11.40 ± 0.02 ms	686.41 MiB	95.11	10.11

4.2 Application to Full-Volume Biomedical Image Segmentation

Benchmark design. We evaluate our methods on Medical Segmentation Decathlon (MSD) Lung and Pancreas data (Antonelli et al., 2022), where the largest call contains around 197 million coordinates. The benchmark comprises 344 volumes and 625 binary decoding calls, with each foreground class optimized independently. We use the nnU-Net v1 `3d_lowres` and `3d_fullres` checkpoints with default test-time mirroring and probability averaging (Isensee et al., 2021). Volume dimensions and benchmark details are in Appendix B.5.

Figure 2 summarizes the out-of-fold probability distributions, which suggest that most probabilities are close to zero or one, and few fall in between. Safe screening can therefore exclude most voxels. We apply full-volume RankSEG under the Dice-RMA instance of R_θ (Wang & Dai, 2025), with $(\alpha, \beta, \gamma_N, \gamma_D) = (1/2, 1/2, 0, 1/2)$, and compare its full-sort and safe-screened implementations while holding the decoder and objective fixed. A contextual comparison with native nnU-Net argmax decoding is reported in Table 10 of the Appendix.

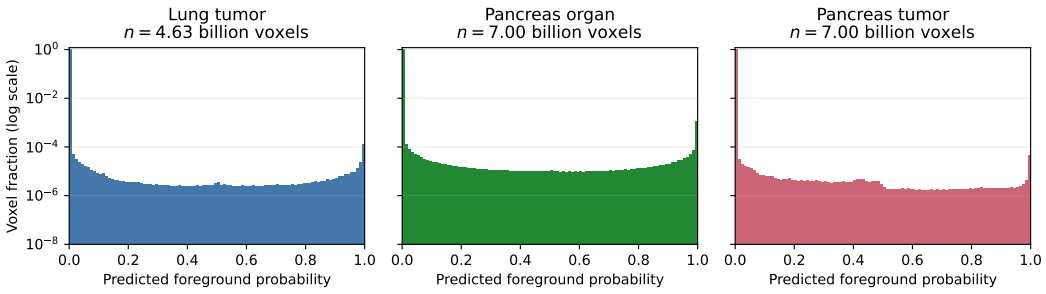

Figure 2: Distributions of out-of-fold foreground probabilities pooled across cases in the full-volume nnU-Net lung and pancreas benchmarks; log-scaled vertical axes show voxel fractions.

Table 2: Full-volume nnU-Net decoding efficiency. Runtime and memory are means ± standard errors across cases; memory is peak additional GPU memory. Scr. is the percentage certified before sorting. Speedups are end-to-end gains from screening with the optimized implementation.

	Full sort		Safe screening (ours)			
Dataset/class	Time	Mem.	Time	Mem.	Scr. (%)	**Speedup**
Lung/tumour	20.86 ± 1.04 ms	2.60 ± 0.13 GiB	2.92 ± 0.14 ms	560.60 ± 27.90 MiB	99.998	7.14x
Pancreas/organ	7.81 ± 0.25 ms	903.60 ± 27.50 MiB	1.08 ± 0.03 ms	190.50 ± 5.80 MiB	99.995	7.20x
Pancreas/tumour	7.14 ± 0.21 ms	903.60 ± 27.50 MiB	1.07 ± 0.03 ms	190.50 ± 5.80 MiB	99.990	6.67x

Results. Table 2 reports results for the three dataset classes. Holding the Dice-RMA decoder fixed, the optimized safe-screened implementation matches the full-sort R_θ value. It certifies 99.990%–99.998% of coordinates before sorting while yielding 6.67–7.20-fold speedups and about 79% lower GPU working memory. These computational gains demonstrate scalability to native three-dimensional medical volumes. Figure 3 shows how the unresolved problem size, runtime speedup, and working-memory saving vary across the individual decoder calls.

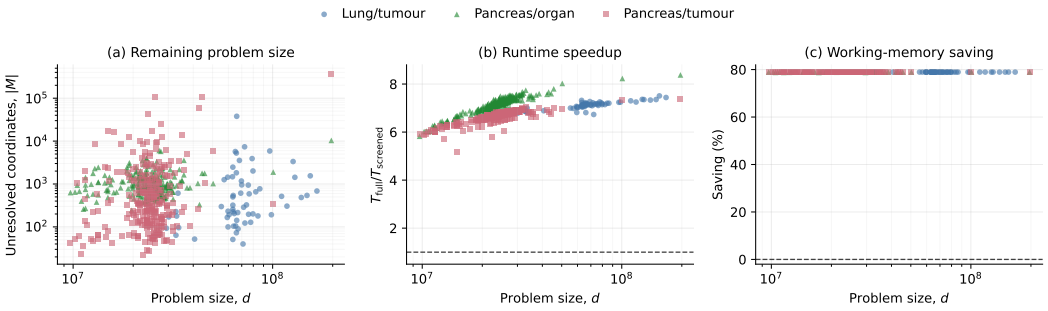

Figure 3: Per-case computational performance over 625 classwise full-volume decoder calls. (a) Number of unresolved coordinates $|M|$ after safe screening; both axes are logarithmic. (b) Runtime speedup $T_{\text{full}}/T_{\text{screened}}$. (c) Working-memory saving relative to full sort. Each point represents one case–class call; dashed lines indicate no runtime or memory improvement.

4.3 Ablation Studies

Among the six simulation settings, case (d) most closely captures the bimodal shape of the application probability distributions (Fig. 2); we therefore focus the synthetic ablations on this case.

Two-sided certificate decomposition. Table 3 partitions the coordinates into the certified-included set H, the certified-excluded set L, and the unresolved set M. In simulation case (d), H and L each certify approximately half of the coordinates, whereas L accounts for nearly all screening in the sparse application maps. The certificates thus play complementary roles across probability regimes, leaving only a small M for exact optimization.

Table 3: Two-sided screening decomposition (%). Simulation case (d): mean (SE) over 20 replications; applications: mean (SE) across cases.

Setting	$\lvert H\rvert/d\,\%$	$\lvert L\rvert/d\,\%$	$\lvert M\rvert/d\,\%$
Sim. (d), 512^2	49.991 (0.022)	49.926 (0.022)	0.083 (0.001)
Sim. (d), 1024^2	50.003 (0.011)	49.914 (0.011)	0.083 (0.001)
Sim. (d), 512^3	50.000 (0.001)	49.918 (0.001)	0.082 (0.000)
Lung/tumour	0.033 (0.007)	99.965 (0.008)	0.002 (0.001)
Pancreas/organ	0.196 (0.005)	99.799 (0.005)	0.005 (0.000)
Pancreas/tumour	0.014 (0.001)	99.975 (0.002)	0.010 (0.002)

Screening across exact solvers. We compare sort-prefix search and Dinkelbach iteration, each with and without screening, using standard PyTorch operations under the paired GPU protocol of Section 4.1; solver details are given in Appendix B.1.

Table 4: GPU comparison of sort-prefix and Dinkelbach backends, with and without safe screening, for simulation case (d). All methods use standard PyTorch operations. Runtime is mean ± standard deviation over 20 replications; memory is mean peak additional GPU allocation. Speedup and memory saving are relative to full sort.

Shape	Method	Time	Mem.	Speedup	Saving (%)
512^2	Full sort	$\mathbf{0.32 \pm 0.00}$ ms	18.00 MiB	1.00	0.00
	Pre-screened sort	0.46 ± 0.01 ms	**2.25 MiB**	0.71	87.48
	Plain Dinkelbach	0.84 ± 0.01 ms	6.18 MiB	0.38	65.66
	Pre-screened Dinkelbach	0.65 ± 0.01 ms	5.25 MiB	0.50	70.84
1024^2	Full sort	0.96 ± 0.00 ms	72.00 MiB	1.00	0.00
	Pre-screened sort	$\mathbf{0.48 \pm 0.00}$ ms	**9.00 MiB**	2.00	87.50
	Plain Dinkelbach	1.03 ± 0.01 ms	24.77 MiB	0.94	65.60
	Pre-screened Dinkelbach	0.79 ± 0.07 ms	21.00 MiB	1.22	70.83
512^3	Full sort	115.62 ± 0.06 ms	9.00 GiB	1.00	0.00
	Pre-screened sort	$\mathbf{17.82 \pm 0.03}$ ms	**1.13 GiB**	6.49	87.50
	Plain Dinkelbach	37.44 ± 1.58 ms	3.09 GiB	3.09	65.71
	Pre-screened Dinkelbach	20.52 ± 0.03 ms	2.63 GiB	5.63	70.83

Table 4 shows that screening benefits both solvers at large scale. At 512^3, it reduces sort-prefix runtime from 115.62 to 17.82 ms and Dinkelbach runtime from 37.44 to 20.52 ms, with screened sort also using the least working memory (1.13 GiB). At 512^2, full sort is fastest because screening overhead exceeds the sorting work avoided; screened sort becomes fastest at 1024^2 and 512^3.

5 Discussion

Reduced GPU working memory is the most consistent benefit of safe screening across the tested input sizes. By certifying coordinates before exact optimization, the two-sided rules restrict downstream computation to the unresolved set while preserving a global optimizer of R_θ. This makes screening useful even when full sorting is too inexpensive for substantial runtime savings. The comparison in Section 4.3 further shows that these benefits extend to both sort-prefix and Dinkelbach solvers.

Runtime gains, however, depend on problem size. For small 2D inputs, computing the certificates and forming the reduced problem can offset or exceed the sorting work avoided. Safe screening therefore need not accelerate 2D segmentation decoding; its clearer advantage in this regime is reduced working memory. Higher-resolution 2D inputs can benefit in runtime, as the 1024^2 results illustrate, so this distinction reflects computational scale rather than spatial dimensionality alone.

The strongest combined benefits arise for large 3D volumes, where sorting creates substantial runtime and memory pressure. Our synthetic 3D and full-volume application benchmarks show that screening can reduce both decoder runtime and peak working memory, making it particularly attractive for high-resolution volumetric decoding on memory-constrained GPUs. The gains also depend on the probability distribution: if few coordinates can be certified, the reduction is limited, and in the worst case $|M| = d$, screening adds a linear pass without reducing the sorting problem.

AI use statement

Generative AI tools were used to provide feedback on experimental design and assist with interpretation of results, creating or editing software code, literature searches, language editing, manuscript organization, and LaTeX formatting. All AI-assisted material was reviewed and verified by the authors, who take responsibility for the final content, including all claims, proofs, experimental results, and artifacts.

Reproducibility statement

The assumptions and complete proofs are provided in the appendices. The main experimental design and solver comparisons are described in Section 4, with implementation details and benchmarking protocols provided in Appendix B. The accompanying code and configurations are available in an anonymous reproducibility repository.

References

Michela Antonelli, Annika Reinke, Spyridon Bakas, Keyvan Farahani, Annette Kopp-Schneider, Bennett A Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M Summers, et al. The medical segmentation decathlon. *Nature Communications*, 13(1):4128, 2022.

Runxue Bao, Quanchao Lu, and Yanfu Zhang. Safe screening rules for group slope. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 3–19. Springer, 2025.

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Quentin Bertrand, Quentin Klopfenstein, Pierre-Antoine Bannier, Gauthier Gidel, and Mathurin Massias. Beyond l1: Faster and better sparse models with skglm. *Advances in Neural Information Processing Systems*, 35:38950–38965, 2022.

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *European Conference on Computer Vision*, pp. 833–851. Springer, 2018.

Ben Dai and Chunlin Li. RankSEG: a consistent ranking-based framework for segmentation. *Journal of Machine Learning Research*, 24(224):1–50, 2023.

Lee R Dice. Measures of the amount of ecologic association between species. *Ecology*, 26(3): 297–302, 1945.

Werner Dinkelbach. On nonlinear fractional programming. *Management Science*, 13(7):492–498, 1967.

Laurent El Ghaoui, Vivian Viallon, and Tarek Rabbani. Safe feature elimination in sparse supervised learning. Technical Report UC/EECS-2010-126, EECS Dept., University of California at Berkeley, September 2010.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Théo Guyard, Cédric Herzet, and Clément Elvira. Screen & relax: accelerating the resolution of elastic-net by safe identification of the solution support. In *Icassp 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 5443–5447. IEEE, 2022.

Yili Hong. On computing the distribution function for the Poisson binomial distribution. *Computational Statistics & Data Analysis*, 59:41–51, 2013.

Fabian Isensee, Paul F Jaeger, Simon AA Kohl, Jens Petersen, and Klaus H Maier-Hein. nnU-Net: A self-configuring method for deep learning-based biomedical image segmentation. *Nature Methods*, 18(2):203–211, 2021. doi: 10.1038/s41592-020-01008-z.

Oluwasanmi Koyejo, Nagarajan Natarajan, Pradeep Ravikumar, and Inderjit S Dhillon. Consistent binary classification with generalized performance metrics. *Advances in Neural Information Processing Systems*, 27, 2014.

Zachary C Lipton, Charles Elkan, and Balakrishnan Naryanaswamy. Optimal thresholding of classifiers to maximize F1 measure. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 225–239. Springer, 2014.

Jun Liu, Zheng Zhao, Jie Wang, and Jieping Ye. Safe screening with variational inequalities and its application to lasso. In *International Conference on Machine Learning*, pp. 289–297. PMLR, 2014.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

Mathurin Massias, Samuel Vaiter, Alexandre Gramfort, and Joseph Salmon. Dual extrapolation for sparse glms. *Journal of Machine Learning Research*, 21(234):1–33, 2020.

Harikrishna Narasimhan, Harish G. Ramaswamy, Shiv Kumar Tavker, Drona Khurana, Praneeth Netrapalli, and Shivani Agarwal. Consistent multiclass algorithms for complex metrics and constraints. *Journal of Machine Learning Research*, 25(367):1–81, 2024.

Eugene Ndiaye, Olivier Fercoq, Alexandre Gramfort, and Joseph Salmon. Gap Safe screening rules for sparsity enforcing penalties. *Journal of Machine Learning Research*, 18(128):1–33, 2017.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241. Springer, 2015.

Seyed Sadegh Mohseni Salehi, Deniz Erdogmus, and Ali Gholipour. Tversky loss function for image segmentation using 3d fully convolutional deep networks. In *International Workshop on Machine Learning in Medical Imaging*, pp. 379–387. Springer, 2017.

Siegfried Schaible. Fractional programming. II, on Dinkelbach's algorithm. *Management Science*, 22(8):868–873, 1976.

Erik Schultheis, Wojciech Kotlowski, Marek Wydmuch, Rohit Babbar, Strom Borman, and Krzysztof Dembczynski. Consistent algorithms for multi-label classification with macro-at-k metrics. In *International Conference on Learning Representations*, volume 2024, pp. 13440–13469, 2024.

Atsushi Shibagaki, Masayuki Karasuyama, Kohei Hatano, and Ichiro Takeuchi. Simultaneous safe screening of features and samples in doubly sparse modeling. In *International Conference on Machine Learning*, pp. 1577–1586. PMLR, 2016.

Abdel Aziz Taha and Allan Hanbury. Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. *BMC Medical Imaging*, 15(1):29, 2015.

Robert Tibshirani, Jacob Bien, Jerome Friedman, Trevor Hastie, Noah Simon, Jonathan Taylor, and Ryan J Tibshirani. Strong rules for discarding predictors in lasso-type problems. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 74(2):245–266, 2012.

Sofie Tilborghs, Jeroen Bertels, David Robben, Dirk Vandermeulen, and Frederik Maes. The dice loss in the context of missing or empty labels: introducing Φ and ϵ. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 527–537. Springer, 2022.

Amos Tversky. Features of similarity. *Psychological Review*, 84(4):327, 1977.

Willem Waegeman, Krzysztof Dembczyński, Arkadiusz Jachnik, Weiwei Cheng, and Eyke Hüllermeier. On the Bayes-optimality of F-measure maximizers. *Journal of Machine Learning Research*, 15:3513–3568, 2014.

Zifu Wang, Xuefei Ning, and Matthew Blaschko. Jaccard metric losses: Optimizing the jaccard index with soft labels. *Advances in Neural Information Processing Systems*, 36:75259–75285, 2023.

Zixun Wang and Ben Dai. RankSEG-RMA: An efficient segmentation algorithm via reciprocal moment approximation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 163310–163333, 2025.

Hiroaki Yamada and Makoto Yamada. Dynamic sasvi: Strong safe screening for norm-regularized least squares. In *Advances in Neural Information Processing Systems*, volume 34, pp. 14645–14655. Curran Associates, Inc., 2021.

Appendix Contents

A Rank-Inducible Evaluation Metrics **12**

B Additional Experimental and Application Details **13**

B.1 Exact solver implementation details 13

B.2 Implementation ablation: PyTorch versus Triton 13

B.3 Sensitivity to asymmetric TI parameters 14

B.4 Predictive performance comparison 15

B.5 GPU benchmarking and full-volume application protocol 16

B.6 Illustrative Lung case 17

C Technical Proofs **18**

C.1 Proof of Proposition 1 18

C.2 Proof of Lemma 1 18

C.3 Proof of Lemma 2 19

C.4 Proof of Corollary 1 19

C.5 Proof of Corollary 2 19

D Theoretical Analysis of Exact Expected Samplewise TI Metrics **19**

A Rank-Inducible Evaluation Metrics

This section gives the general structural condition underlying Proposition 1. As in the main text, write $s(\mathbf{m}) = \mathbf{m}^\intercal \mathbf{p}$ and $\mathrm{TN}(\mathbf{Y}, \mathbf{m}) = (\mathbf{1} - \mathbf{m})^\intercal (\mathbf{1} - \mathbf{Y})$. The conditional expected confusion components can then be expressed as

$$\mathbb{E}\big(\mathrm{TP}(\mathbf{Y}, \mathbf{m})\big) = s(\mathbf{m}), \qquad \mathbb{E}\big(\mathrm{FP}(\mathbf{Y}, \mathbf{m})\big) = \|\mathbf{m}\|_1 - s(\mathbf{m}),$$
$$\mathbb{E}\big(\mathrm{FN}(\mathbf{Y}, \mathbf{m})\big) = \mu - s(\mathbf{m}), \qquad \mathbb{E}\big(\mathrm{TN}(\mathbf{Y}, \mathbf{m})\big) = d - \|\mathbf{m}\|_1 - \mu + s(\mathbf{m}).$$

Thus any metric constructed from these expected components depends on the mask through $\|\mathbf{m}\|_1$ and $s(\mathbf{m})$, although it need not be monotone in the latter. For a fixed prediction size, increasing $s(\mathbf{m})$ simultaneously increases the expected numbers of true positives and true negatives and decreases the expected numbers of false positives and false negatives. This observation motivates the following definition.

Definition 1 (Rank-inducible metric)**.** *An evaluation metric $R(\mathbf{m})$ is rank-inducible if there exists a function ψ such that $R(\mathbf{m}) = \psi\big(\|\mathbf{m}\|_1, s(\mathbf{m})\big)$ and, for every fixed τ, the map $s \mapsto \psi(\tau, s)$ is nondecreasing on its feasible domain.*

Rank-inducibility is more general than the TI family. Several familiar criteria are rank-inducible without being TI overlap indices; one example is the Matthews correlation coefficient (MCC). Write $\mathbb{E}(\mathrm{TP})$ for $\mathbb{E}\{\mathrm{TP}(\mathbf{Y}, \mathbf{m})\}$ and analogously for the other components. Applied to the conditional expected components above, MCC simplifies to

$$R_{\mathrm{MCC}}(\mathbf{m}) = \frac{\mathbb{E}(\mathrm{TP})\mathbb{E}(\mathrm{TN}) - \mathbb{E}(\mathrm{FP})\mathbb{E}(\mathrm{FN})}{[\{\mathbb{E}(\mathrm{TP}) + \mathbb{E}(\mathrm{FP})\}\{\mathbb{E}(\mathrm{TP}) + \mathbb{E}(\mathrm{FN})\}\{\mathbb{E}(\mathrm{TN}) + \mathbb{E}(\mathrm{FP})\}\{\mathbb{E}(\mathrm{TN}) + \mathbb{E}(\mathrm{FN})\}]^{1/2}}$$
$$= \frac{ds(\mathbf{m}) - \|\mathbf{m}\|_1 \mu}{\{\|\mathbf{m}\|_1 \mu (d - \|\mathbf{m}\|_1)(d - \mu)\}^{1/2}}.$$

It depends on the mask only through $\|\mathbf{m}\|_1$ and $s(\mathbf{m})$, and its slope in s for fixed $\tau = \|\mathbf{m}\|_1$ is $d/\{\tau\mu(d - \tau)(d - \mu)\}^{1/2}$, which is positive on the nondegenerate domain. Thus Definition 1

applies. MCC requires $0 < \mu < d$ and $0 < \|\mathbf{m}\|_1 < d$, with the degenerate sizes $\|\mathbf{m}\|_1 \in \{0, d\}$ handled separately according to the chosen convention.

Proposition 2 (Generic ranking-induced prefix structure)**.** *Let $R(\mathbf{m})$ be rank-inducible in the sense of Definition 1. Let $j_1, \ldots, j_d$ be a permutation of $1, \ldots, d$ such that $p_{j_1} \geq \cdots \geq p_{j_d}$, and define $s_\tau = \sum_{r=1}^{\tau} p_{j_r}$ for $\tau = 1, \ldots, d$, with $s_0 = 0$. Then there exists an optimal prediction vector satisfying*

$$m^*_{j_r} = \mathbf{1}\{r \leq \tau^*\}, \qquad r = 1, \ldots, d,$$

where

$$\tau^* \in \operatorname*{arg\,max}_{\tau \in \{0,\ldots,d\}} \psi(\tau, s_\tau).$$

Proof. For any fixed τ, the prefix selecting $j_1, \ldots, j_\tau$ maximizes $s(\mathbf{m})$ over all masks with $\|\mathbf{m}\|_1 = \tau$. Because $s \mapsto \psi(\tau, s)$ is nondecreasing, this prefix also maximizes $R(\mathbf{m})$ at volume τ. Maximizing its score over $\tau = 0, \ldots, d$ proves the result. □

Proposition 2 is the general counterpart of Proposition 1: the same two-stage procedure applies to any rank-inducible metric, so one global ranking serves not only Dice, IoU, and F-scores but also the non-TI example above. The result asserts existence rather than uniqueness: ties may produce additional optima, but at least one optimal mask is a prefix of the displayed ordering.

B Additional Experimental and Application Details

B.1 Exact solver implementation details

This subsection specifies the implementations compared in Section 4.3. All four methods use standard PyTorch operations without custom Triton kernels. Full sort uses `torch.sort`, a cumulative sum, and a vectorized prefix search. For a selected set $S \subseteq [d]$, write $R(S)$ for the score of its indicator mask under the unsmoothed Dice objective used in this comparison, and let $h(\lambda) = \lambda/2$. Plain Dinkelbach initializes $S_0 = [d]$ and $\lambda_0 = R(S_0)$, then repeatedly updates

$$S_{t+1} = \{j \in [d] : p_j > h(\lambda_t)\}, \qquad \lambda_{t+1} = R(S_{t+1}),$$

from the full active set until the selection stabilizes. The screened variant first forms H, L, and M, keeps H fixed, and discards L. Starting from $T_0 = M$ and $\lambda_0 = R(H \cup T_0)$, it updates

$$T_{t+1} = \{j \in M : p_j > h(\lambda_t)\}, \qquad \lambda_{t+1} = R(H \cup T_{t+1}),$$

until the selection stabilizes. In both variants, the thresholded active sets are nested, so unchanged cardinality certifies stabilization; no numerical stopping tolerance is used. The iteration limits are $d + 1$ and $|M| + 1$ for the plain and screened variants, respectively.

The comparison uses the unsmoothed Dice setting of Section 4.1 and the paired GPU protocol in Appendix B.5. Peak additional working memory excludes the resident input tensor and CUDA context.

B.2 Implementation ablation: PyTorch versus Triton

We separate the computational benefit of screening from that of the fused Triton implementation. Using the same probability vectors, GPU platform, and paired protocol as the main simulation benchmark, we compare (i) full sort in PyTorch, (ii) safe screening implemented entirely with portable PyTorch operations, and (iii) safe screening with fused Triton kernels. Both screened implementations use `torch.sort` for the unresolved coordinates. For compactness, the tables denote these three implementations by *Full PT*, *Screen PT*, and *Screen Triton*, respectively; PT abbreviates PyTorch. The ratio of Full PT runtime to Screen PT runtime is therefore a matched-backend measure of the screening gain; the ratio of Screen PT runtime to Screen Triton runtime measures the additional Triton gain; and their product is the overall gain reported in the main text.

The Triton kernels fuse screening-statistics reductions with lower-threshold computation, prefix scoring with argmax reduction, and threshold comparisons for final mask reconstruction. Extraction of the probabilities in M still uses `torch.masked_select`, and the implementation retains length-d Boolean buffers, hence $O(d)$ temporary storage.

Table 5: GPU runtime decomposition between safe screening and the Triton implementation. Runtime entries are means ± sample standard deviations over 20 paired replications. Screening, Triton, and Overall denote the three runtime ratios defined in the text.

Shape	Case	Full PT	Screen PT	Screen Triton	Scr. (%)	Screening	Triton	**Overall**
512^2	(a)	0.32 ± 0.00 ms	0.55 ± 0.00 ms	0.30 ± 0.00 ms	87.51	0.59	1.82	**1.07**
	(b)	0.32 ± 0.00 ms	0.53 ± 0.00 ms	0.30 ± 0.00 ms	91.89	0.61	1.79	**1.09**
	(c)	0.32 ± 0.00 ms	0.44 ± 0.01 ms	0.25 ± 0.01 ms	99.96	0.71	1.79	**1.28**
	(d)	0.32 ± 0.00 ms	0.43 ± 0.01 ms	0.24 ± 0.00 ms	99.92	0.74	1.77	**1.32**
	(e)	0.32 ± 0.00 ms	0.62 ± 0.01 ms	0.31 ± 0.00 ms	70.99	0.52	1.97	**1.02**
	(f)	0.32 ± 0.00 ms	0.52 ± 0.00 ms	0.30 ± 0.00 ms	95.12	0.61	1.76	**1.08**
1024^2	(a)	0.97 ± 0.00 ms	0.62 ± 0.04 ms	0.36 ± 0.00 ms	87.50	1.57	1.74	**2.73**
	(b)	0.98 ± 0.00 ms	0.67 ± 0.00 ms	0.35 ± 0.00 ms	91.91	1.46	1.93	**2.82**
	(c)	0.95 ± 0.00 ms	0.48 ± 0.00 ms	0.27 ± 0.00 ms	99.96	1.99	1.79	**3.56**
	(d)	0.96 ± 0.00 ms	0.48 ± 0.00 ms	0.26 ± 0.00 ms	99.92	2.01	1.81	**3.65**
	(e)	0.97 ± 0.00 ms	0.70 ± 0.00 ms	0.42 ± 0.00 ms	70.98	1.39	1.65	**2.28**
	(f)	0.97 ± 0.00 ms	0.62 ± 0.00 ms	0.34 ± 0.00 ms	95.11	1.57	1.84	**2.89**
512^3	(a)	115.13 ± 0.10 ms	29.84 ± 0.02 ms	16.61 ± 0.04 ms	87.50	3.86	1.80	**6.93**
	(b)	115.90 ± 0.07 ms	25.72 ± 0.04 ms	13.69 ± 0.04 ms	91.93	4.51	1.88	**8.46**
	(c)	114.32 ± 0.08 ms	17.73 ± 0.02 ms	7.76 ± 0.01 ms	99.96	6.45	2.29	**14.74**
	(d)	115.55 ± 0.07 ms	17.81 ± 0.01 ms	7.78 ± 0.02 ms	99.92	6.49	2.29	**14.85**
	(e)	115.26 ± 0.07 ms	46.00 ± 0.02 ms	28.51 ± 0.03 ms	70.99	2.51	1.61	**4.04**
	(f)	115.23 ± 0.08 ms	22.61 ± 0.01 ms	11.40 ± 0.02 ms	95.11	5.10	1.98	**10.11**

Safe screening is not cost-free: it requires additional passes over the probability vector to compute the certificates, partition the coordinates, and construct the unresolved problem. At 512^2, full sorting is already inexpensive and fits comfortably in GPU memory, so under matched PyTorch backends this overhead outweighs the saving from sorting fewer coordinates, giving full-sort-to-screened runtime ratios of 0.52–0.74. As the input grows, the reduction in sorting work and associated memory traffic becomes dominant: screening alone yields 1.39–2.01-fold speedups at 1024^2 and 2.51–6.49-fold speedups at 512^3. The Triton implementation further reduces the screening overhead, improving screened runtime by 1.61–2.29-fold across the tested settings.

Table 6: Peak additional GPU working memory for the implementation ablation.

Shape	Implementation	(a)	(b)	(c)	(d)	(e)	(f)
512^2	Full PT	18.00 MiB	18.00 MiB	18.00 MiB	18.00 MiB	18.00 MiB	18.00 MiB
	Screen PT	3.25 MiB	2.38 MiB	2.25 MiB	2.25 MiB	6.56 MiB	2.25 MiB
	Screen Triton	2.28 MiB	1.75 MiB	1.00 MiB	1.00 MiB	4.27 MiB	1.36 MiB
1024^2	Full PT	72.00 MiB	72.00 MiB	72.00 MiB	72.00 MiB	72.00 MiB	72.00 MiB
	Screen PT	13.01 MiB	9.47 MiB	9.00 MiB	9.00 MiB	26.60 MiB	9.00 MiB
	Screen Triton	9.05 MiB	6.92 MiB	4.00 MiB	4.00 MiB	17.02 MiB	5.38 MiB
512^3	Full PT	9.00 GiB	9.00 GiB	9.00 GiB	9.00 GiB	9.00 GiB	9.00 GiB
	Screen PT	1.63 GiB	1.18 GiB	1.13 GiB	1.13 GiB	3.28 GiB	1.13 GiB
	Screen Triton	1.13 GiB	882.98 MiB	512.00 MiB	512.00 MiB	2.13 GiB	686.41 MiB

Matched-backend screening reduces working memory by 63.05%–87.50%; Triton reduces the screened working memory by a further 26.42%–55.56%. The maximum absolute objective discrepancy is 2.22×10^{-16} in the runtime runs and 3.33×10^{-16} in the separate memory runs, confirming agreement to floating-point precision.

B.3 Sensitivity to asymmetric TI parameters

We next assess whether the computational behavior extends beyond the symmetric Dice setting used in the main experiments. We consider

$$(\alpha, \beta) \in \{(0.8, 0.2), (0.5, 0.5), (0.2, 0.8)\}, \qquad \gamma_N = \gamma_D = 0.$$

Since $\alpha = 1/(1+\rho^2)$ and $\beta = \rho^2/(1+\rho^2)$, these settings correspond to F_ρ with $\rho = 0.5$, 1, and 2, respectively. They therefore range from heavier penalization of false positives to heavier

penalization of false negatives, with Dice as the symmetric case. Keeping $\alpha + \beta = 1$ normalizes the total weight assigned to false-positive and false-negative errors, so that the experiment varies their relative emphasis.

The experiment otherwise follows the GPU benchmark in Section 4.1, restricted to $d = 1024^2 = 1{,}048{,}576$. It uses the same six probability distributions, computational environment, and definition of working memory. Within each distribution, a common set of 20 probability vectors is reused for all three parameter settings. This gives $6 \times 3 \times 20 = 360$ paired full-sort and safe-screening comparisons. Each distribution–parameter setting is run in a fresh process. Runtime is summarized by the mean $\pm$ sample standard deviation of the 20 medians. Working memory is measured on the same vectors, with the two methods run in separate CUDA processes. Screening uses the refined threshold in Corollary 2, with strict inequalities so that threshold ties remain in M.

Table 7: GPU runtime and peak working-memory sensitivity to asymmetric TI parameters.

		Full sort		Safe screening (ours)			
Case	(α, β)	Time	Mem.	Time	Mem.	Scr. (%)	**Speedup**
(a)	$(0.8, 0.2)$	0.97 ± 0.00 ms	72.00 MiB	0.40 ± 0.00 ms	15.00 MiB	75.39	2.44
	$(0.5, 0.5)$	0.97 ± 0.00 ms	72.00 MiB	0.35 ± 0.00 ms	9.05 MiB	87.50	2.75
	$(0.2, 0.8)$	0.98 ± 0.00 ms	72.00 MiB	0.34 ± 0.00 ms	4.40 MiB	97.14	2.91
(b)	$(0.8, 0.2)$	0.98 ± 0.00 ms	72.00 MiB	0.37 ± 0.00 ms	11.51 MiB	82.40	2.63
	$(0.5, 0.5)$	0.98 ± 0.00 ms	72.00 MiB	0.34 ± 0.00 ms	6.92 MiB	91.92	2.85
	$(0.2, 0.8)$	0.98 ± 0.00 ms	72.00 MiB	0.34 ± 0.01 ms	5.37 MiB	95.14	2.92
(c)	$(0.8, 0.2)$	0.95 ± 0.00 ms	72.00 MiB	0.31 ± 0.00 ms	4.00 MiB	99.54	3.09
	$(0.5, 0.5)$	0.95 ± 0.00 ms	72.00 MiB	0.26 ± 0.00 ms	4.00 MiB	99.96	3.73
	$(0.2, 0.8)$	0.95 ± 0.00 ms	72.00 MiB	0.26 ± 0.01 ms	4.00 MiB	100.00	3.66
(d)	$(0.8, 0.2)$	0.96 ± 0.00 ms	72.00 MiB	0.27 ± 0.00 ms	4.00 MiB	99.69	3.59
	$(0.5, 0.5)$	0.96 ± 0.00 ms	72.00 MiB	0.26 ± 0.00 ms	4.00 MiB	99.92	3.74
	$(0.2, 0.8)$	0.96 ± 0.00 ms	72.00 MiB	0.26 ± 0.00 ms	4.00 MiB	99.92	3.73
(e)	$(0.8, 0.2)$	0.97 ± 0.00 ms	72.00 MiB	0.58 ± 0.00 ms	29.35 MiB	45.46	1.68
	$(0.5, 0.5)$	0.97 ± 0.00 ms	72.00 MiB	0.42 ± 0.00 ms	17.02 MiB	70.99	2.31
	$(0.2, 0.8)$	0.97 ± 0.00 ms	72.00 MiB	0.32 ± 0.00 ms	4.09 MiB	97.78	3.00
(f)	$(0.8, 0.2)$	0.97 ± 0.00 ms	72.00 MiB	0.41 ± 0.00 ms	15.59 MiB	73.97	2.38
	$(0.5, 0.5)$	0.97 ± 0.00 ms	72.00 MiB	0.34 ± 0.00 ms	5.37 MiB	95.13	2.91
	$(0.2, 0.8)$	0.97 ± 0.00 ms	72.00 MiB	0.34 ± 0.00 ms	5.66 MiB	94.54	2.90

Runtime entries are means $\pm$ sample standard deviations of 20 within-vector medians, each based on ten timing repetitions per method. Speedup is the ratio of the mean full-sort runtime to the mean safe-screening runtime. Scr. is the mean $100(1 - |M|/d)$. Memory is the mean peak additional GPU allocation during prediction, excluding the resident probability tensor and CUDA context. Experiments use an NVIDIA GeForce RTX 3090 (24 GB), PyTorch 2.13.0, CUDA 13.0.

The screened percentage varies with both the probability distribution and the asymmetric weights, and generally decreases as α/β increases. The largest change occurs in case (e), where screening decreases from 97.78% at $(\alpha, \beta) = (0.2, 0.8)$ to 45.46% at $(0.8, 0.2)$, while the speedup decreases from 3.00 to 1.68.

Across all settings, speedups range from 1.68 to 3.74, and working-memory savings range from 59.24% to 94.44%. The full-sort and safe-screening objectives agree to recorded numerical precision in all 360 parameter–vector comparisons, with an absolute difference of zero in every comparison. The computational benefits therefore extend across the tested asymmetric TI objectives.

B.4 Predictive performance comparison

We compare the TI-optimal rule with the conventional fixed cutoff $m_j = \mathbf{1}(p_j \geq 0.5)$ under the six simulation scenarios in Section 4.1. Because safe screening returns the same optimizer as the full-sort search, this comparison concerns the decision rule rather than its computational implementation. The plug-in Dice and IoU objectives obey $R_{\mathrm{Dice}} = 2R_{\mathrm{IoU}}/(1 + R_{\mathrm{IoU}})$ and therefore produce the

same optimal mask. For each realized label vector, we record F_1, IoU, and accuracy using the binary metric implementations in `scikit-learn`, with an undefined $0/0$ score set to zero. Each replication uses $d = 1000$ coordinates. We use 1000 paired replications per scenario, evaluating both rules on the same $\mathbf{p}$ and $\mathbf{Y}$.

The optimized rule has mean F_1 and IoU values at least as high as those of the fixed cutoff in every scenario. The largest gain occurs under the sparse $\mathrm{Beta}(0.1, 3)$ distribution: mean F_1 increases from 0.229 to 0.431, and mean IoU increases from 0.133 to 0.277. The two rules are nearly indistinguishable in scenarios (c) and (d), where the probabilities are already concentrated near zero and one. Accuracy is lower for the optimized rule in scenarios (a), (b), (e), and (f), and nearly unchanged in scenarios (c) and (d). This contrast is expected: the 0.5 cutoff targets coordinatewise zero–one loss, whereas the TI-optimal rule targets a set-valued overlap criterion. Thus metric-aligned optimization improves the target overlap measures but need not improve accuracy.

Table 8: Predictive performance of the optimized and fixed-cutoff rules over 1000 replications. Entries are means with standard deviations in parentheses.

Distribution	Rule	F_1	IoU	Accuracy
(a)	TI-optimal	0.77 (0.01)	0.62 (0.02)	0.74 (0.01)
	0.5 cutoff	0.75 (0.02)	0.60 (0.02)	0.75 (0.01)
(b)	TI-optimal	0.43 (0.07)	0.28 (0.06)	0.96 (0.01)
	0.5 cutoff	0.23 (0.09)	0.13 (0.06)	0.97 (0.01)
(c)	TI-optimal	0.98 (0.00)	0.97 (0.01)	0.97 (0.01)
	0.5 cutoff	0.98 (0.00)	0.97 (0.01)	0.97 (0.01)
(d)	TI-optimal	0.97 (0.01)	0.94 (0.01)	0.97 (0.01)
	0.5 cutoff	0.97 (0.01)	0.94 (0.01)	0.97 (0.01)
(e)	TI-optimal	0.70 (0.02)	0.54 (0.02)	0.62 (0.02)
	0.5 cutoff	0.66 (0.02)	0.49 (0.02)	0.66 (0.02)
(f)	TI-optimal	0.78 (0.01)	0.63 (0.02)	0.77 (0.01)
	0.5 cutoff	0.77 (0.02)	0.63 (0.02)	0.77 (0.01)

B.5 GPU BENCHMARKING AND FULL-VOLUME APPLICATION PROTOCOL

GPU runtimes were measured with synchronized CUDA events. Each synthetic configuration was run in a fresh subprocess using 20 paired replications, five warm-up runs, and rotating method order. Compilation, autotuning, probability generation, and host-to-device transfer were excluded from timing. Further implementation and environment details are provided in the accompanying code repository. The screening guarantees are exact-arithmetic statements. In the finite-precision GPU implementation, strict threshold comparisons retain exact boundary ties in M, and numerical equivalence is assessed against full-sort optimization at the prespecified tolerance reported below rather than claimed as an additional formal guarantee. For $\mu = 0$ and $\gamma_N = \gamma_D = 0$, the implementation returns the empty mask, assigning its undefined $0/0$ score the default value one (`zero_division=1`).

The application uses all labelled cases from Medical Segmentation Decathlon Task006 Lung (63 cases) and Task007 Pancreas (281 cases) (Antonelli et al., 2022). Probability maps are generated by strict five-fold out-of-fold inference using the official nnU-Net v1 splits and checkpoints (Isensee et al., 2021), so every case is predicted by a model that excluded it from training. Inference uses default three-axis test-time mirroring. The `3d_lowres` and `3d_fullres` probability maps are combined by the arithmetic average implemented in `nnUNet_ensemble`. Lung requires one binary optimization per case, whereas the two Pancreas foreground classes are optimized separately. These are independent classwise binary decoding calls rather than a joint decoder enforcing mutually exclusive multiclass predictions. RankSEG uses the Dice-RMA objective with $(\alpha, \beta, \gamma_N, \gamma_D) = (1/2, 1/2, 0, 1/2)$, and Dice is reported classwise. Each $512 \times 512 \times z$ foreground map is flattened into $d = 512^2 z$ coordinates; Table 9 summarizes the call sizes.

For context, Table 10 compares the predictive performance of the Dice-RMA decoder with native nnU-Net argmax decoding. This comparison motivates the downstream decoder; the screening experiment itself holds the decoder fixed.

Table 9: Native volume dimensions and binary optimization sizes.

Dataset	Cases	z: median [min, max]	d (million): mean [min, max]
Lung	63	252 [112, 636]	73.47 [29.36, 166.72]
Pancreas	281	93 [37, 751]	24.93 [9.70, 196.87]

Table 10: Contextual full-volume Dice comparison. Entries are percentages reported as means $\pm$ standard errors across cases.

Dataset/class	Native argmax	RankSEG
Lung/tumour	72.29 ± 2.85	73.79 ± 2.59
Pancreas/organ	82.24 ± 0.45	82.29 ± 0.45
Pancreas/tumour	52.56 ± 1.93	59.48 ± 1.63

On an NVIDIA GeForce RTX 3090, the full-sort and safe-screened implementations are alternated over three repetitions per case and class, and the within-case median is retained. Timing includes decoding and mask reconstruction but excludes model inference, file input/output, and host-to-device transfer. Working memory is the peak additional GPU allocation beyond the resident probability tensor and CUDA context. Table 2 reports across-case means and standard errors, with speedup computed from the mean runtimes.

Full-sort and safe-screened objective values agree within the prespecified 10^{-6} tolerance for every case and class. Decoded masks may differ at cutoff ties while attaining the same objective to this tolerance.

B.6 Illustrative Lung case

Figure 4 compares the native argmax and RankSEG decoders for one Lung case. On the displayed axial slice, argmax misses a substantial part of the tumour, whereas RankSEG recovers most of the lesion with few false-positive voxels. The slice Dice increases from 0.367 to 0.814, and the full-volume Dice increases from 0.647 to 0.814.

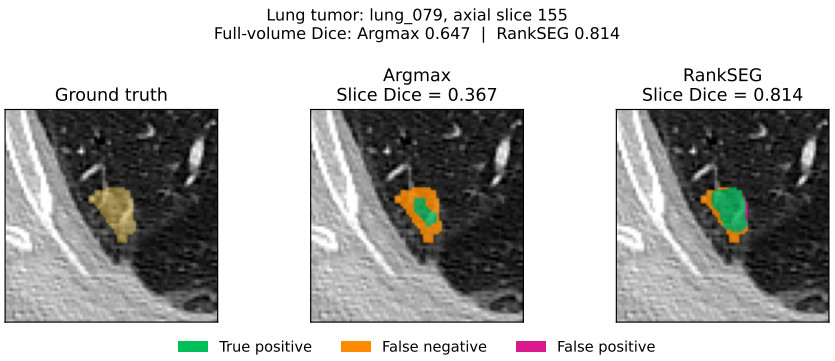

Figure 4: Illustrative Lung result for case `lung_079`, axial slice 155. The ground-truth mask is shown on the left; the middle and right panels overlay the argmax and RankSEG predictions, respectively. Green, orange, and magenta denote true-positive, false-negative, and false-positive voxels. The displayed slice Dice values are 0.367 for argmax and 0.814 for RankSEG; their full-volume Dice values are 0.647 and 0.814, respectively.

C TECHNICAL PROOFS

This section collects the proofs of the formal results stated in the main text.

Lemma 3 (Monotonicity)**.** *Under Assumption 1, for each $\tau = 1, \dots, d$, the map*

$$s \mapsto \frac{s + \gamma_N}{\alpha\tau + \beta\mu + \gamma_D + (1 - \alpha - \beta)s}$$

is nondecreasing on its feasible domain.

Proof. For fixed $\tau \in \{1, \dots, d\}$, write

$$\phi_\theta(\tau, s) = \frac{s + \gamma_N}{\alpha\tau + \beta\mu + \gamma_D + (1 - \alpha - \beta)s}.$$

Its denominator is

$$s + \alpha(\tau - s) + \beta(\mu - s) + \gamma_D,$$

a sum of nonnegative terms. It is positive when $s > 0$. When $s = 0$, it equals $\alpha\tau + \beta\mu + \gamma_D$, which is positive either because $\alpha > 0$ or, if $\alpha = 0$, by Assumption 1. Therefore

$$\frac{\partial}{\partial s}\phi_\theta(\tau, s) = \frac{\alpha\tau + \beta\mu + \gamma_D - (1 - \alpha - \beta)\gamma_N}{\{\alpha\tau + \beta\mu + \gamma_D + (1 - \alpha - \beta)s\}^2}.$$

The numerator can be written as

$$\alpha(\tau + \gamma_N) + \beta(\mu + \gamma_N) + \gamma_D - \gamma_N,$$

which is nonnegative under Assumption 1. Hence $s \mapsto \phi_\theta(\tau, s)$ is nondecreasing for every fixed τ. □

C.1 PROOF OF PROPOSITION 1

Proof. Fix $\tau \in \{1, \dots, d\}$. Among all prediction vectors $\mathbf{m}$ with $\|\mathbf{m}\|_1 = \tau$, the mask selecting the top-τ probabilities maximizes $s(\mathbf{m}) = \mathbf{m}^\intercal \mathbf{p}$. By Lemma 3, the TI objective is nondecreasing in $s(\mathbf{m})$ for fixed τ. Thus the top-τ mask is optimal among all prediction vectors of size τ. The empty mask is the sole candidate at $\tau = 0$. Maximizing the best mask at each volume over $\tau = 0, \dots, d$ gives the result. □

C.2 PROOF OF LEMMA 1

Proof. Let

$$N_\tau = s_\tau + \gamma_N, \qquad D_\tau = \alpha\tau + \beta\mu + \gamma_D + (1 - \alpha - \beta)s_\tau,$$

so that $\pi_\tau = N_\tau / D_\tau$. Note that $N_{\tau+1} = N_\tau + p_{j_{\tau+1}}$ and $D_{\tau+1} = D_\tau + \alpha + (1 - \alpha - \beta)p_{j_{\tau+1}}$. Therefore

$$\pi_{\tau+1} - \pi_\tau = \frac{p_{j_{\tau+1}}\{D_\tau - (1 - \alpha - \beta)N_\tau\} - \alpha N_\tau}{D_\tau D_{\tau+1}} = \frac{p_{j_{\tau+1}}\{1 - (1 - \alpha - \beta)\pi_\tau\} - \alpha\pi_\tau}{D_{\tau+1}}.$$

Here $D_{\tau+1} > 0$ by Assumption 1. Moreover,

$$D_\tau - N_\tau = \alpha(\tau - s_\tau) + \beta(\mu - s_\tau) + \gamma_D - \gamma_N \geq 0,$$

so $0 \leq \pi_\tau \leq 1$ and

$$1 - (1 - \alpha - \beta)\pi_\tau \geq \min\{1, \alpha + \beta\} > 0.$$

The first equivalence follows immediately:

$$\pi_{\tau+1} > \pi_\tau \quad \Longleftrightarrow \quad p_{j_{\tau+1}} > \frac{\alpha\pi_\tau}{1 - (1 - \alpha - \beta)\pi_\tau}.$$

For the second equivalence, direct substitution gives

$$\begin{aligned} &p_{j_{\tau+1}}\{D_{\tau+1} - (1 - \alpha - \beta)N_{\tau+1}\} - \alpha N_{\tau+1} \\ &= p_{j_{\tau+1}}\{D_\tau - (1 - \alpha - \beta)N_\tau\} - \alpha N_\tau, \end{aligned}$$

because the additional terms cancel. Thus the same sign condition is obtained with $\pi_{\tau+1}$ in place of π_τ. The argument is unchanged when all strict inequalities are replaced by non-strict inequalities. □

C.3 Proof of Lemma 2

Proof. Suppose $\pi_{\tau+1} \le \pi_\tau$. By Lemma 1, using the equivalent threshold written with $\pi_{\tau+1}$,

$$p_{j_{\tau+1}} \le \frac{\alpha\pi_{\tau+1}}{1-(1-\alpha-\beta)\pi_{\tau+1}}.$$

If $\tau \le d-2$, then $p_{j_{\tau+2}} \le p_{j_{\tau+1}}$ by the ordering, so $p_{j_{\tau+2}} \le \alpha\pi_{\tau+1}/\{1-(1-\alpha-\beta)\pi_{\tau+1}\}$. Applying Lemma 1 again at index $\tau+1$ gives $\pi_{\tau+2} \le \pi_{\tau+1}$. Repeating this argument proves the claim. □

C.4 Proof of Corollary 1

Proof. If $\tau^* = d$, the claim is immediate. Otherwise, $\pi_{\tau^*} \ge \pi_{\tau^*+1}$. By Lemma 1,

$$p_{j_{\tau^*+1}} \le \frac{\alpha\pi_{\tau^*}}{1-(1-\alpha-\beta)\pi_{\tau^*}} \le \frac{\alpha}{\alpha+\beta}.$$

The last inequality follows from $0 \le \pi_{\tau^*} \le 1$: the denominator of the τ^* component in (5) exceeds its numerator by $\alpha(\tau^* - s_{\tau^*}) + \beta(\mu - s_{\tau^*}) + \gamma_D - \gamma_N \ge 0$. Thus any coordinate not included in the top-τ^* mask has probability at most p_H, which proves the claim. □

C.5 Proof of Corollary 2

Proof. Let H and s_H be as defined in Corollary 2. This set is the top-$|H|$ prefix, whose score is

$$\pi_{|H|} = \frac{s_H + \gamma_N}{\alpha|H| + \beta\mu + \gamma_D + (1-\alpha-\beta)s_H}.$$

Let $\tau^* \in \arg\max_{\tau\in\{0,\dots,d\}} \pi_\tau$. If $\tau^* = 0$, the claim is immediate. If $\tau^* = 1$, then $p_{j_{\tau^*}} = p_{\max} \ge p_L$. If $\tau^* \ge 2$, then $\pi_{\tau^*} \ge \pi_{\tau^*-1}$. By Lemma 1,

$$p_{j_{\tau^*}} \ge \frac{\alpha\pi_{\tau^*}}{1-(1-\alpha-\beta)\pi_{\tau^*}}.$$

Since τ^* maximizes the prefix score, $\pi_{\tau^*} \ge \max\{\pi_1, \pi_{|H|}, \pi_d\}$. The map $u \mapsto \alpha u/\{1-(1-\alpha-\beta)u\}$ is nondecreasing on $[0,1]$. Substituting $(s_1, s_{|H|}, s_d) = (p_{\max}, s_H, \mu)$ gives

$$\frac{\alpha\pi_1}{1-(1-\alpha-\beta)\pi_1} = \frac{\alpha(p_{\max}+\gamma_N)}{D(1)},$$
$$\frac{\alpha\pi_{|H|}}{1-(1-\alpha-\beta)\pi_{|H|}} = \frac{\alpha(s_H+\gamma_N)}{D(|H|)},$$
$$\frac{\alpha\pi_d}{1-(1-\alpha-\beta)\pi_d} = \frac{\alpha(\mu+\gamma_N)}{D(d)}.$$

It follows from the definition of p_L that $p_{j_{\tau^*}} \ge p_L$. Therefore no coordinate with $p_j < p_L$ can appear in an optimal prefix. □

D Theoretical Analysis of Exact Expected Samplewise TI Metrics

Unlike the ELF objective in (3), samplewise aggregation forms the metric within each prediction unit and then averages the resulting random ratios. Its population objective is

$$R_\theta^{\mathrm{sw}}(\mathbf{m}) := \mathbb{E}\{\mathrm{TI}_\theta(\mathbf{Y},\mathbf{m})\} = \mathbb{E}\left[\frac{\mathrm{TP}(\mathbf{Y},\mathbf{m})+\gamma_N}{\mathrm{TP}(\mathbf{Y},\mathbf{m})+\alpha\,\mathrm{FP}(\mathbf{Y},\mathbf{m})+\beta\,\mathrm{FN}(\mathbf{Y},\mathbf{m})+\gamma_D}\right]. \tag{7}$$

This includes image-level Dice and samplewise F-measures, for which the expectation remains outside the random ratio (Waegeman et al., 2014; Dai & Li, 2023).

Throughout this section, we impose Assumption 1 and assume

$$Y_j \mid \mathbf{X} = \mathbf{x} \sim \mathrm{Bernoulli}(p_j), \qquad j = 1,\dots,d, \tag{8}$$

independently, and suppress conditioning on $\mathbf{x}$. We assign a score of zero when the ratio is $0/0$.

This section is deliberately separate from the plug-in method analyzed and evaluated in the main text. The results below concern only the exact expected samplewise objective under conditional independence. They do not imply that Algorithm 1, its linear-time pre-sort screening pass, or its complexity and empirical guarantees carry over to this objective.

The samplewise and ELF objectives need not coincide for finite d. In large prediction units, random confusion counts can concentrate around their conditional means, motivating the linear-fractional approximation used in the main text; for Dice, this connection can also be expressed through reciprocal moment approximations (Dai & Li, 2023; Wang & Dai, 2025). Exact samplewise evaluation is more costly because it involves expectations of random ratios. The results below establish a prefix structure and conservative inclusion and exclusion certificates under (8). These are structural results: in particular, the exclusion threshold depends on ordered prefix quantities and is not the closed-form linear-time pre-sort rule used by the evaluated plug-in method.

The argument has three parts. First, an exchange argument shows that, at any fixed prediction volume, selecting the largest probabilities is optimal. Second, the effect of adding one coordinate is written as its probability of being positive times a conditional gain, minus its probability of being negative times a conditional loss. Finally, uniform bounds on this marginal comparison yield safe inclusion and exclusion thresholds.

The next proposition extends the ranking-induced Bayes structure established for samplewise Dice in Theorem 1 of Dai & Li (2023), together with its IoU analogue, to the general samplewise TI family. Although (7) is not a linear-fractional function of $s(\mathbf{m})$, it retains the same prefix structure.

Proposition 3 (Prefix structure for samplewise TI metrics)**.** *Under conditions (i)–(ii) of Assumption 1 and (8), let $j_1, \dots, j_d$ be a permutation of $1, \dots, d$ such that $p_{j_1} \geq \cdots \geq p_{j_d}$. For every fixed prediction volume τ, the prefix selecting $j_1, \dots, j_\tau$ maximizes (7) among all masks of that volume. Consequently, at least one global maximizer is a prefix of this ordering.*

Proof. The argument is inspired by the proof of Theorem 1 in Dai & Li (2023). The case $\tau = 0$ is immediate, so fix $\tau \in \{1, \dots, d\}$ and let $B = \sum_{j=1}^{d} Y_j$. We show that any inversion in a volume-τ mask can be removed without decreasing its expected score.

Suppose that $m_i = 1$ and $m_j = 0$ for some $p_j \geq p_i$, and let $\mathbf{m}' = \mathbf{m} - \mathbf{e}_i + \mathbf{e}_j$. Condition on $\mathbf{Y}_{-ij} = (Y_k : k \notin \{i, j\})$, and write

$$B_{-ij} = \sum_{k \notin \{i,j\}} Y_k, \qquad S_{-ij} = \sum_{k \notin \{i,j\}} m_k Y_k.$$

Since $B = B_{-ij} + Y_i + Y_j$, the confusion counts under $\mathbf{m}$ are

$$\mathrm{TP}(\mathbf{Y}, \mathbf{m}) = S_{-ij} + Y_i, \quad \mathrm{FP}(\mathbf{Y}, \mathbf{m}) = \tau - S_{-ij} - Y_i, \quad \mathrm{FN}(\mathbf{Y}, \mathbf{m}) = B_{-ij} + Y_j - S_{-ij}.$$

If $Y_i = Y_j$, the two masks have the same confusion counts and hence the same score. If exactly one of Y_i and Y_j is positive, then $B = B_{-ij} + 1$. A mask selecting the positive coordinate has $S_{-ij} + 1$ true positives, whereas a mask selecting the negative coordinate has S_{-ij}. The score advantage of the former is therefore

$$\begin{aligned}\Delta &= \frac{S_{-ij} + 1 + \gamma_N}{\alpha\tau + \beta B + \gamma_D + (1 - \alpha - \beta)(S_{-ij} + 1)} - \frac{S_{-ij} + \gamma_N}{\alpha\tau + \beta B + \gamma_D + (1 - \alpha - \beta) S_{-ij}} \\ &= \frac{\alpha(\tau + \gamma_N) + \beta(B + \gamma_N) + \gamma_D - \gamma_N}{\{\alpha\tau + \beta B + \gamma_D + (1 - \alpha - \beta) S_{-ij}\}\{\alpha\tau + \beta B + \gamma_D + (1 - \alpha - \beta)(S_{-ij} + 1)\}} > 0.\end{aligned}$$

By conditions (i)–(ii) of Assumption 1, all terms in the numerator are nonnegative. Moreover, $\tau \geq 1$ and, in either mixed-label case, $B \geq 1$. If $\alpha > 0$, then $\alpha\tau > 0$; if $\alpha = 0$, then $\alpha + \beta > 0$ implies $\beta > 0$, so $\beta B > 0$. Hence the numerator is strictly positive. For $u \in \{S_{-ij}, S_{-ij} + 1\}$, each denominator factor can be written as $u + \alpha(\tau - u) + \beta(B - u) + \gamma_D$. Feasibility gives $0 \leq u \leq \tau$ and $u \leq B$, so all four terms are nonnegative. If $u > 0$, the first term is positive. If $u = 0$, then $\tau, B \geq 1$ and $\alpha + \beta > 0$ imply $\alpha\tau + \beta B > 0$. Thus both denominator factors are strictly positive. Conditional independence now gives

$$\mathbb{E}\left[\mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m}') - \mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m}) \mid \mathbf{Y}_{-ij}\right] = \big\{(1 - p_i)p_j - p_i(1 - p_j)\big\}\Delta = (p_j - p_i)\Delta \geq 0.$$

Averaging over $\mathbf{Y}_{-ij}$ shows that the exchange does not decrease R_θ^{sw}. Repeating such exchanges yields the top-τ prefix; this also covers ties, for which the score difference is zero. Finally, maximizing over $\tau = 0, \ldots, d$ yields a global prefix maximizer. □

We next compare a mask with the mask obtained by adding one coordinate. Let $\mathbf{e}_j$ denote the jth standard basis vector. Conditional on $Y_j = 1$, $U_j(\mathbf{m})$ is the expected gain from adding j; conditional on $Y_j = 0$, $V_j(\mathbf{m})$ is the expected loss. Thus both quantities have a direct decision-theoretic interpretation.

Lemma 4 (Marginal threshold for samplewise TI metrics)**.** *Under Assumption 1, for any mask* $\mathbf{m}$ *with* $m_j = 0$*, define*

$$U_j(\mathbf{m}) = \mathbb{E}\left[\mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m} + \mathbf{e}_j) - \mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m}) \mid Y_j = 1\right],$$
$$V_j(\mathbf{m}) = \mathbb{E}\left[\mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m}) - \mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m} + \mathbf{e}_j) \mid Y_j = 0\right]. \tag{9}$$

Here each conditional expectation is understood by fixing Y_j at the stated value and averaging over the product distribution of the remaining labels. Then

$$R_\theta^{\mathrm{sw}}(\mathbf{m} + \mathbf{e}_j) - R_\theta^{\mathrm{sw}}(\mathbf{m}) = p_j U_j(\mathbf{m}) - (1 - p_j) V_j(\mathbf{m}). \tag{10}$$

Moreover, $U_j(\mathbf{m}) \geq 0$, $V_j(\mathbf{m}) \geq 0$, and $U_j(\mathbf{m}) + V_j(\mathbf{m}) > 0$. Consequently, adding j is non-worsening if and only if

$$p_j \geq \frac{V_j(\mathbf{m})}{U_j(\mathbf{m}) + V_j(\mathbf{m})}. \tag{11}$$

Proof. We first show (10). Note that

$$\begin{aligned} &R_\theta^{\mathrm{sw}}(\mathbf{m} + \mathbf{e}_j) - R_\theta^{\mathrm{sw}}(\mathbf{m}) = \mathbb{E}\left[\mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m} + \mathbf{e}_j) - \mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m})\right] \\ &= p_j \mathbb{E}\left[\mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m} + \mathbf{e}_j) - \mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m}) \mid Y_j = 1\right] \\ &\quad + (1 - p_j)\mathbb{E}\left[\mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m} + \mathbf{e}_j) - \mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m}) \mid Y_j = 0\right] = p_j U_j(\mathbf{m}) - (1 - p_j) V_j(\mathbf{m}). \end{aligned}$$

Next, we determine the signs of $U_j(\mathbf{m})$ and $V_j(\mathbf{m})$. Suppress the common arguments $(\mathbf{Y}, \mathbf{m})$ of TP, FP, and FN in the following calculation. When $Y_j = 1$, adding j increases TP by one and decreases FN by one. Directly simplifying the definition of $U_j(\mathbf{m})$ in (9) gives

$$\begin{aligned} U_j(\mathbf{m}) &= \mathbb{E}\left[\frac{\mathrm{TP} + 1 + \gamma_N}{\mathrm{TP} + 1 + \alpha\,\mathrm{FP} + \beta(\mathrm{FN} - 1) + \gamma_D} - \frac{\mathrm{TP} + \gamma_N}{\mathrm{TP} + \alpha\,\mathrm{FP} + \beta\,\mathrm{FN} + \gamma_D} \,\middle|\, Y_j = 1\right] \\ &= \mathbb{E}\left[\frac{\alpha\,\mathrm{FP} + \beta\{\mathrm{TP} + \mathrm{FN} + \gamma_N\} + \gamma_D - \gamma_N}{\{\mathrm{TP} + 1 + \alpha\,\mathrm{FP} + \beta(\mathrm{FN} - 1) + \gamma_D\}\{\mathrm{TP} + \alpha\,\mathrm{FP} + \beta\,\mathrm{FN} + \gamma_D\}} \,\middle|\, Y_j = 1\right] \geq 0. \end{aligned}$$

Indeed, $m_j = 0$ and $Y_j = 1$ imply $\mathrm{FN} \geq 1$. If $\beta > 0$, the numerator and the second denominator factor contain the positive term $\beta\,\mathrm{FN}$, while the first denominator factor contains $\mathrm{TP} + 1 > 0$; the pointwise gain is then strictly positive. If $\beta = 0$, Assumption 1 implies $\alpha > 0$ and $\gamma_D > 0$. Both denominator factors are therefore positive, and the numerator reduces to $\alpha\,\mathrm{FP} + \gamma_D - \gamma_N \geq 0$. Thus $U_j(\mathbf{m}) \geq 0$ in all cases.

When $Y_j = 0$, adding j increases FP by one while leaving TP and FN unchanged. Thus, by its definition,

$$\begin{aligned} V_j(\mathbf{m}) &= \mathbb{E}\left[\frac{\mathrm{TP} + \gamma_N}{\mathrm{TP} + \alpha\,\mathrm{FP} + \beta\,\mathrm{FN} + \gamma_D} - \frac{\mathrm{TP} + \gamma_N}{\mathrm{TP} + \alpha(\mathrm{FP} + 1) + \beta\,\mathrm{FN} + \gamma_D} \,\middle|\, Y_j = 0\right] \\ &= \mathbb{E}\left[\frac{\alpha(\mathrm{TP} + \gamma_N)}{\{\mathrm{TP} + \alpha\,\mathrm{FP} + \beta\,\mathrm{FN} + \gamma_D\}\{\mathrm{TP} + \alpha(\mathrm{FP} + 1) + \beta\,\mathrm{FN} + \gamma_D\}} \,\middle|\, Y_j = 0\right]. \end{aligned}$$

The integrand in the second line is defined to be zero when $\mathrm{TP} + \alpha\,\mathrm{FP} + \beta\,\mathrm{FN} + \gamma_D = 0$. We now justify both the equality and the sign. First suppose that $\mathrm{TP} + \alpha\,\mathrm{FP} + \beta\,\mathrm{FN} + \gamma_D > 0$. Then

$$\mathrm{TP} + \alpha(\mathrm{FP} + 1) + \beta\,\mathrm{FN} + \gamma_D = \mathrm{TP} + \alpha\,\mathrm{FP} + \beta\,\mathrm{FN} + \gamma_D + \alpha > 0.$$

Then the integrand is nonnegative because $\alpha \geq 0$ and $\mathrm{TP} + \gamma_N \geq 0$.

Now suppose $\mathrm{TP} + \alpha\,\mathrm{FP} + \beta\,\mathrm{FN} + \gamma_D = 0$. The nonnegativity of its terms implies $\mathrm{TP} = \gamma_D = 0$, and $\alpha\,\mathrm{FP} = \beta\,\mathrm{FN} = 0$. Because $0 \leq \gamma_N \leq \gamma_D$, we also have $\gamma_N = 0$. Hence $\mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m}) = 0$ by

the zero-score convention. Since $Y_j = 0$, adding j creates one false positive but leaves the numerator equal to zero. The denominator of $\mathrm{TI}_\theta(\mathbf{Y}, \mathbf{m} + \mathbf{e}_j)$ equals α. If $\alpha > 0$, this score is $0/\alpha = 0$; if $\alpha = 0$, it is $0/0$ and is again zero by convention. Thus both scores are zero, and the pointwise loss is zero in this degenerate case. This agrees with the zero value assigned to the second-line integrand. Hence the displayed equality is valid in all cases and $V_j(\mathbf{m}) \geq 0$.

It suffices to verify that $U_j(\mathbf{m})$ and $V_j(\mathbf{m})$ are not both zero. If $\beta > 0$, the argument above gives $U_j(\mathbf{m}) > 0$. If $\beta = 0$, then $\alpha > 0$ and $\gamma_D > 0$. When $\gamma_D > \gamma_N$, the pointwise gain defining $U_j(\mathbf{m})$ is strictly positive. When $\gamma_D = \gamma_N > 0$, the pointwise loss defining $V_j(\mathbf{m})$ is strictly positive because its numerator contains $\alpha\gamma_N > 0$. Hence $U_j(\mathbf{m}) + V_j(\mathbf{m}) > 0$ in every case. Therefore,

$$p_j U_j(\mathbf{m}) - (1 - p_j)V_j(\mathbf{m}) = \{U_j(\mathbf{m}) + V_j(\mathbf{m})\} \left\{ p_j - \frac{V_j(\mathbf{m})}{U_j(\mathbf{m}) + V_j(\mathbf{m})} \right\}.$$

Since $U_j(\mathbf{m}) + V_j(\mathbf{m}) > 0$, this expression is nonnegative if and only if (11) holds. □

The threshold in (11) depends on the current mask. A safe inclusion certificate requires an upper bound that is valid for every mask; the following bound depends only on α and β.

Corollary 3 (Upper inclusion threshold for samplewise TI metrics)**.** *Under Assumption 1 and (8), define*

$$p_H = \begin{cases} 1/2, & \alpha \leq \beta, \\ \alpha/(\alpha + \beta), & \alpha > \beta. \end{cases} \tag{12}$$

Every coordinate with $p_j > p_H$ is included in every maximizer of (7).

Proof. Fix any mask $\mathbf{m}$ with $m_j = 0$. By Lemma 4, adding j is non-worsening if and only if

$$p_j \geq \frac{V_j(\mathbf{m})}{U_j(\mathbf{m}) + V_j(\mathbf{m})}.$$

If p_j exceeds a uniform upper bound on the right-hand side over all masks that exclude j, then adding j strictly improves every such mask. Hence no maximizer can exclude j. It therefore remains to derive a mask-independent upper bound.

We obtain the bound through $U_j(\mathbf{m}) \geq cV_j(\mathbf{m})$. Indeed, if $V_j(\mathbf{m}) = 0$, the marginal threshold is zero, which is always less than or equal to p_H. If $V_j(\mathbf{m}) > 0$, then

$$U_j(\mathbf{m}) \geq cV_j(\mathbf{m}) \quad \Longrightarrow \quad \frac{V_j(\mathbf{m})}{U_j(\mathbf{m}) + V_j(\mathbf{m})} \leq \frac{1}{1 + c}.$$

We show below that

$$c = 1 \quad (\alpha \leq \beta), \qquad c = \frac{\beta}{\alpha} \quad (\alpha > \beta).$$

To proceed, we first rewrite the pointwise gain and loss in a more convenient form. For $\delta \in \{0, 1\}$,

$$\mathbf{Y}^{(\delta)} = (Y_1, \ldots, Y_{j-1}, \delta, Y_{j+1}, \ldots, Y_d)^\intercal,$$

and define the pointwise gain and loss

$$u_j(\mathbf{Y}, \mathbf{m}) = \mathrm{TI}_\theta(\mathbf{Y}^{(1)}, \mathbf{m} + \mathbf{e}_j) - \mathrm{TI}_\theta(\mathbf{Y}^{(1)}, \mathbf{m}),$$
$$v_j(\mathbf{Y}, \mathbf{m}) = \mathrm{TI}_\theta(\mathbf{Y}^{(0)}, \mathbf{m}) - \mathrm{TI}_\theta(\mathbf{Y}^{(0)}, \mathbf{m} + \mathbf{e}_j).$$

Both quantities depend only on $\mathbf{Y}_{-j}$, and conditional independence gives

$$U_j(\mathbf{m}) = \mathbb{E}_{\mathbf{Y}_{-j}}\{u_j(\mathbf{Y}, \mathbf{m})\}, \qquad V_j(\mathbf{m}) = \mathbb{E}_{\mathbf{Y}_{-j}}\{v_j(\mathbf{Y}, \mathbf{m})\}.$$

For fixed $\mathbf{Y}_{-j}$, set

$$\mathrm{TP}_{-j} = \sum_{k \neq j} m_k Y_k, \quad \mathrm{FP}_{-j} = \sum_{k \neq j} m_k (1 - Y_k), \quad \mathrm{FN}_{-j} = \sum_{k \neq j} (1 - m_k) Y_k,$$
$$D_0 = \mathrm{TP}_{-j} + \alpha\, \mathrm{FP}_{-j} + \beta\, \mathrm{FN}_{-j} + \gamma_D.$$

First suppose that $D_0 > 0$. The pointwise gain has the exact decomposition

$$u_j(\mathbf{Y}, \mathbf{m}) = \frac{\mathrm{TP}_{-j} + 1 + \gamma_N}{D_0 + 1} - \frac{\mathrm{TP}_{-j} + \gamma_N}{D_0 + \beta}$$
$$= \frac{\beta(\mathrm{TP}_{-j} + \gamma_N)}{D_0(D_0 + \beta)} + \frac{D_0 - (\mathrm{TP}_{-j} + \gamma_N)}{D_0(D_0 + 1)} \geq \frac{\beta(\mathrm{TP}_{-j} + \gamma_N)}{D_0(D_0 + \beta)},$$

where the last inequality follows from

$$D_0 - (\mathrm{TP}_{-j} + \gamma_N) = \alpha\,\mathrm{FP}_{-j} + \beta\,\mathrm{FN}_{-j} + \gamma_D - \gamma_N \geq 0$$

by conditions (i)–(ii) of Assumption 1.

The pointwise loss of $V_j(\mathbf{m})$ is

$$v_j(\mathbf{Y}, \mathbf{m}) = \frac{\mathrm{TP}_{-j} + \gamma_N}{D_0} - \frac{\mathrm{TP}_{-j} + \gamma_N}{D_0 + \alpha} = \frac{\alpha(\mathrm{TP}_{-j} + \gamma_N)}{D_0(D_0 + \alpha)}.$$

If $D_0 = 0$, its nonnegative terms imply $\mathrm{TP}_{-j} = \gamma_D = 0$ and $\alpha\,\mathrm{FP}_{-j} = \beta\,\mathrm{FN}_{-j} = 0$, and hence $\gamma_N = 0$. In this case the pointwise gain is one and the pointwise loss is zero. We now derive the upper bound separately for the two parameter regimes.

If $\alpha \leq \beta$ and $D_0 > 0$, then

$$\frac{\beta}{D_0 + \beta} - \frac{\alpha}{D_0 + \alpha} = \frac{(\beta - \alpha)D_0}{(D_0 + \beta)(D_0 + \alpha)} \geq 0.$$

Consequently,

$$u_j(\mathbf{Y}, \mathbf{m}) \geq \frac{\beta(\mathrm{TP}_{-j} + \gamma_N)}{D_0(D_0 + \beta)} \geq \frac{\alpha(\mathrm{TP}_{-j} + \gamma_N)}{D_0(D_0 + \alpha)} = v_j(\mathbf{Y}, \mathbf{m}).$$

The same inequality holds in the zero-denominator case because then $u_j(\mathbf{Y}, \mathbf{m}) = 1$ and $v_j(\mathbf{Y}, \mathbf{m}) = 0$. Therefore

$$U_j(\mathbf{m}) = \mathbb{E}_{\mathbf{Y}_{-j}}\{u_j(\mathbf{Y}, \mathbf{m})\} \geq \mathbb{E}_{\mathbf{Y}_{-j}}\{v_j(\mathbf{Y}, \mathbf{m})\} = V_j(\mathbf{m}),$$

so $c = 1$ when $\alpha \leq \beta$.

If $\alpha > \beta$ and $D_0 > 0$, then

$$\alpha u_j(\mathbf{Y}, \mathbf{m}) \geq \frac{\alpha\beta(\mathrm{TP}_{-j} + \gamma_N)}{D_0(D_0 + \beta)} \geq \frac{\alpha\beta(\mathrm{TP}_{-j} + \gamma_N)}{D_0(D_0 + \alpha)} = \beta v_j(\mathbf{Y}, \mathbf{m}),$$

where the second inequality follows from $\alpha > \beta$. In the zero-denominator case, the same inequality follows from $u_j(\mathbf{Y}, \mathbf{m}) = 1$ and $v_j(\mathbf{Y}, \mathbf{m}) = 0$. Hence

$$\alpha U_j(\mathbf{m}) = \alpha\mathbb{E}_{\mathbf{Y}_{-j}}\{u_j(\mathbf{Y}, \mathbf{m})\} \geq \beta\mathbb{E}_{\mathbf{Y}_{-j}}\{v_j(\mathbf{Y}, \mathbf{m})\} = \beta V_j(\mathbf{m}).$$

Equivalently, $U_j(\mathbf{m}) \geq (\beta/\alpha)V_j(\mathbf{m})$, so $c = \beta/\alpha$ when $\alpha > \beta$.

Substituting these two choices of c into the bound at the beginning of the proof gives

$$\frac{V_j(\mathbf{m})}{U_j(\mathbf{m}) + V_j(\mathbf{m})} \leq \begin{cases} 1/2, & \alpha \leq \beta, \\ \alpha/(\alpha + \beta), & \alpha > \beta. \end{cases}$$

This bound holds for every mask excluding j. If $p_j > p_H$, Lemma 4 shows that adding j strictly improves any such mask. Hence every maximizer must include j. □

When false negatives receive at least as much weight as false positives ($\alpha \leq \beta$), probabilities above $1/2$ are always selected. When false positives receive more weight, the inclusion certificate becomes the more conservative threshold $\alpha/(\alpha + \beta) > 1/2$.

For exclusion, Proposition 3 allows us to consider only prefix masks. The proof removes the least-probable coordinate from a putative optimal prefix and shows that adding it back would decrease the objective whenever its probability is below the following threshold. Recall that $p_{\max} = \max_{1 \leq j \leq d} p_j$ and that $s_\tau = \sum_{r=1}^{\tau} p_{j_r}$ is the probability mass of the top-τ prefix.

Proposition 4 (Lower exclusion threshold for samplewise TI metrics)**.** *Under Assumption 1 and (8), suppose $d \geq 2$. Set*

$$p_L = \min\left\{p_{\max},\ \min_{1\leq\tau\leq d-1} \frac{\alpha(s_\tau+\gamma_N)}{\bar{z}_\tau(\bar{z}_\tau+\alpha)+\alpha(s_\tau+\gamma_N)}\right\}, \tag{13}$$

where $\bar{z}_\tau = \tau\max\{1,\alpha\}+\beta(d-\tau-1)+\gamma_D$ for $\tau = 1,\dots,d-1$. Then no optimal prefix mask for (7) contains a coordinate with $p_j < p_L$.

Proof. Suppose, for contradiction, that an optimal prefix $\mathbf{m}^*$ contains a coordinate with probability below p_L, and let j be its least-probable selected coordinate. Because $p_L \leq p_{\max}$, a one-coordinate prefix cannot provide such a counterexample. Hence $\|\mathbf{m}^*\|_1 \geq 2$. Set

$$\mathbf{m} = \mathbf{m}^* - \mathbf{e}_j, \qquad \tau = \|\mathbf{m}\|_1.$$

Then $\mathbf{m}$ is the top-τ prefix and $\mathbf{m}^\intercal\mathbf{p} = s_\tau$.

Because $\mathbf{m}^* = \mathbf{m}+\mathbf{e}_j$ is optimal, adding j to $\mathbf{m}$ cannot decrease the objective. Lemma 4 therefore requires

$$p_j \geq \frac{V_j(\mathbf{m})}{U_j(\mathbf{m})+V_j(\mathbf{m})}.$$

We derive a lower bound for the fraction on the right-hand side.

First consider $V_j(\mathbf{m})$. Conditional on $Y_j = 0$, let

$$D_0 = \mathrm{TP}(\mathbf{Y},\mathbf{m}) + \alpha\,\mathrm{FP}(\mathbf{Y},\mathbf{m}) + \beta\,\mathrm{FN}(\mathbf{Y},\mathbf{m}) + \gamma_D.$$

Since $\mathrm{TP}(\mathbf{Y},\mathbf{m})+\mathrm{FP}(\mathbf{Y},\mathbf{m}) = \|\mathbf{m}\|_1 = \tau$,

$$\mathrm{TP}(\mathbf{Y},\mathbf{m}) + \alpha\,\mathrm{FP}(\mathbf{Y},\mathbf{m}) \leq \tau\max\{1,\alpha\}.$$

Moreover, because $m_j = 0$ and $Y_j = 0$,

$$\mathrm{FN}(\mathbf{Y},\mathbf{m}) = \sum_{\substack{k\neq j\\ m_k=0}} Y_k \leq \sum_{\substack{k\neq j\\ m_k=0}} 1 = \big|\{k\neq j : m_k = 0\}\big| = d-\tau-1.$$

Consequently,

$$0 \leq D_0 \leq \tau\max\{1,\alpha\}+\beta(d-\tau-1)+\gamma_D = \bar{z}_\tau.$$

If $D_0 > 0$, the pointwise loss satisfies

$$\frac{\alpha(\mathrm{TP}+\gamma_N)}{D_0(D_0+\alpha)} \geq \frac{\alpha(\mathrm{TP}+\gamma_N)}{\bar{z}_\tau(\bar{z}_\tau+\alpha)}.$$

If $D_0 = 0$, then $\mathrm{TP} = \gamma_D = \gamma_N = 0$. Under $Y_j = 0$, both scores in the pointwise loss are therefore zero, using the zero-score convention when a denominator vanishes. The pointwise loss and the right-hand side of the last display are both zero, so the inequality holds in this case as well. Taking the conditional expectation over the remaining labels gives

$$V_j(\mathbf{m}) \geq \frac{\alpha(s_\tau+\gamma_N)}{\bar{z}_\tau(\bar{z}_\tau+\alpha)}.$$

Here independence gives $\mathbb{E}\{\mathrm{TP}(\mathbf{Y},\mathbf{m}) \mid Y_j = 0\} = \mathbf{m}^\intercal\mathbf{p} = s_\tau$.

Next, every TI score lies in $[0,1]$: whenever its denominator is positive, the denominator minus the numerator equals $\alpha\,\mathrm{FP}+\beta\,\mathrm{FN}+\gamma_D-\gamma_N \geq 0$. Hence the difference between the score after and before adding j is at most one. The same conclusion holds in the $0/0$ case under the zero-score convention, so $U_j(\mathbf{m}) \leq 1$. Since $x \mapsto x/(1+x)$ is nondecreasing on $[0,\infty)$,

$$\frac{V_j(\mathbf{m})}{U_j(\mathbf{m})+V_j(\mathbf{m})} \geq \frac{V_j(\mathbf{m})}{1+V_j(\mathbf{m})} \geq \frac{\alpha(s_\tau+\gamma_N)}{\bar{z}_\tau(\bar{z}_\tau+\alpha)+\alpha(s_\tau+\gamma_N)} \geq p_L.$$

This contradicts $p_j < p_L$ and proves the result. □

The threshold is conservative because it replaces the random denominator by a deterministic worst-case upper bound $\bar{z}_\tau$ and bounds the possible conditional gain by one. Its role is certification: coordinates below p_L can be discarded without evaluating the exact expected ratio for every candidate prefix.