Convex Calibrated Surrogates for the Multi-Label F-Measure
Supplementary Material

Implementation of 'decode'

In order to solve the combinatorial optimization problem involved in the mapping decode : $\mathbb{R}^{s^2+1} \to \{0,1\}^s$ as defined in Eq. (7) efficiently, we make use of an $O(s^3)$-time procedure due to Dembczynski et al. (2011). Specifically, Dembczynski et al. (2011) gave a procedure that, given a certain set of s^2+1 statistics of the true conditional distribution $p(\mathbf{y}|x)$ at a point $x \in \mathcal{X}$, computes in $O(s^3)$ time a Bayes optimal multi-label prediction $h^*(x) \in \{0,1\}^s$ at that point with respect to the F_1-measure by solving a similar combinatorial optimization problem (the approach generalizes easily to the F_β-measure for general β). Our algorithm (Algorithm 2) can be viewed as effectively estimating the same s^2+1 statistics from the training sample S; in particular, once a scoring function $\mathbf{f}_S : \mathcal{X} \to \mathbb{R}^{s^2+1}$ is learned by minimizing our surrogate loss ψ, the estimated statistics at a point $x \in \mathcal{X}$ are given by $\gamma^{-1}(\mathbf{f}_S(x))$ (where γ^{-1} is the inverse of the link function $\gamma : [0,1] \to \mathbb{R}$ associated with the strictly proper composite binary loss ϕ used in our surrogate, and is applied element-wise to $\mathbf{f}_S(x)$). Our 'decode' mapping effectively corresponds to estimating a Bayes optimal prediction at x using these estimated statistics; we can therefore apply the procedure of Dembczynski et al. (2011) to these estimated statistics.

The implementation below is described for a general input vector $\mathbf{u} \in \mathbb{R}^{s^2+1}$ (see Eq. (7)); in our F_β learning algorithm, to make a prediction at $x \in \mathcal{X}$, it would be applied to $\mathbf{u} = \mathbf{f}_S(x)$. The overall idea is that the combinatorial search over $\widehat{\mathbf{y}} \in \{0,1\}^s$ is stratified over the $s+1$ sets $\widehat{\mathcal{Y}}_l = \{\widehat{\mathbf{y}} \in \{0,1\}^s : \|\widehat{\mathbf{y}}\|_1 = l\}$, $l \in \{0,1,\ldots,s\}$; to find an optimal element $\widehat{\mathbf{y}}^{l,*}$ within each set $\widehat{\mathcal{Y}}_l$, one need only solve a problem of the form $\widehat{\mathbf{y}}^{l,*} \in \operatorname{argmin}_{\widehat{\mathbf{y}} \in \widehat{\mathcal{Y}}_l} \sum_{j=1}^s \widehat{y}_j T_{jl}$ for certain numbers T_{jl}, which can be done simply by finding the smallest l numbers among $\{T_{jl} : j \in [s]\}$ and setting the corresponding l entries of $\widehat{\mathbf{y}}^{l,*}$ to 1 (and remaining entries to 0). Solving these $s+1$ subproblems and picking the best solution among them takes a total of $O(s^2 \ln(s))$ time; computing the s^2 numbers T_{jl} involves a matrix multiplication that takes a total of $O(s^3)$ time.[7]

Algorithm 2 Decode

1: **Input:** Vector $\mathbf{u} = (u_0, (u_{jk})_{j,k=1}^s))^\top \in \mathbb{R}^{s^2+1}$
2: **Parameters:** Link function $\gamma : [0,1] \to \mathbb{R}$
3: Define matrices $\mathbf{Q} \in [0,1]^{s \times s}$ and $\mathbf{V} \in \mathbb{R}^{s \times s}$ as follows:

$$\begin{aligned} Q_{jk} &= \gamma^{-1}(u_{jk}) \\ V_{kl} &= \frac{-(1+\beta)^2}{\beta^2 k + l} \end{aligned}$$

4: Compute $\mathbf{T} = \mathbf{QV}$ // matrix multiplication, $O(s^3)$ time
5: **For** $l = 1 \ldots s$: // for loop takes total $O(s^2 \ln(s))$ time
6: Find the l smallest numbers among $\{T_{jl} : j \in [s]\}$; call the corresponding indices $j_1^l, \ldots, j_l^l$
7: Define $\widehat{\mathbf{y}}^{l,*} \in \{0,1\}^s$ as follows:

$$\widehat{y}_j^{l,*} = \begin{cases} 1 & \text{if } j \in \{j_1^l, \ldots, j_l^l\} \\ 0 & \text{otherwise.} \end{cases} \qquad \text{// this solves } \widehat{\mathbf{y}}^{l,*} \in \operatorname{argmin}_{\widehat{\mathbf{y}} \in \widehat{\mathcal{Y}}_l} \sum_{j=1}^s \widehat{y}_j T_{jl}$$

8: Set $z_l^* = \sum_{j=1}^s \widehat{y}_j^{l,*} T_{jl}$
9: **End for**
10: Pick $\widehat{\mathbf{y}}^* \in \{0,1\}^s$ as follows:

$$\widehat{\mathbf{y}}^* \in \operatorname*{argmin}_{\widehat{\mathbf{y}} \in \{\mathbf{0}, \widehat{\mathbf{y}}^{1,*}, \ldots, \widehat{\mathbf{y}}^{s,*}\}} -\mathbf{1}(\widehat{\mathbf{y}} = \mathbf{0}) \cdot \gamma^{-1}(u_0) + \mathbf{1}(\widehat{\mathbf{y}} \neq \mathbf{0}) \cdot z^*_{\|\widehat{\mathbf{y}}\|_1}$$

11: **Output:** $\widehat{\mathbf{y}}^* \in \{0,1\}^s$

[7]One could in principle use faster matrix multiplication methods that take $o(s^3)$ time, but in practice, this would be helpful for only extremely large values of s.