An Exact Algorithm for F-Measure Maximization

Krzysztof Dembczyński[1], Willem Waegeman[2], Weiwei Cheng[3], and Eyke Hüllermeier[3]

[1] *Intelligent Decision Support Systems Laboratory (IDSS), Poznań University of Technology, Poland*

[2] *Research Unit Knowledge-Based Systems (KERMIT), Ghent University, Belgium*

[3] *Knowledge Engineering & Bioinformatics Lab (KEBI), University of Marburg, Germany*

Motivation

- **The F-measure** is routinely used as a performance metric for different types of prediction problems, including **binary classification**, **multi-label classification**, and certain applications of **structured output prediction**.
- Given a prediction $\boldsymbol{h} = (h_1, \ldots, h_m) \in \{0,1\}^m$ of a binary label vector $\boldsymbol{y} = (y_1, \ldots, y_m)$, the **F-measure** is defined as:

$$F(\boldsymbol{y}, \boldsymbol{h}) = \frac{2\sum_{i=1}^m y_i h_i}{\sum_{i=1}^m y_i + \sum_{i=1}^m h_i} \in [0,1] \quad , \tag{1}$$

  where $0/0 = 1$ by definition.
- Compared to measures like error rate in binary classification, it enforces a better **balance** between performance on the **minority** and the **majority** class, therefore it is more suitable in the case of **imbalanced** data.
- Despite its **popularity** in experimental settings, only a **few** methods for training classifiers that directly **optimize** the F-measure have been proposed so far.

Optimal Inference for F-Measure Maximization

- Let $\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_m)$ be a random variable that follows a joint distribution $p(\boldsymbol{Y})$ on $\{0,1\}^m$. The prediction $\boldsymbol{h}^*$ that **maximizes** the expected **F-measure** is given by

$$\boldsymbol{h}^*_F = \arg\max_{\boldsymbol{h} \in \{0,1\}^m} \mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{Y})} \left[F(\boldsymbol{y}, \boldsymbol{h})\right] = \arg\max_{\boldsymbol{h} \in \{0,1\}^m} \sum_{\boldsymbol{y} \in \{0,1\}^m} p(\boldsymbol{y})\, F(\boldsymbol{y}, \boldsymbol{h}). \tag{2}$$

- Unfortunately, **no closed form** solution exists for this optimization problem.
- This problem **cannot** be solved **naively** by brute-force search, since this would require to check **all** possible combinations of labels (2^m) and to sum over an **exponential** (2^m) number of elements for computing the expected value.

Existing Algorithms

Label Independence:

- The optimal solution **always** contains the labels with the **highest** marginal probabilities or no label.
- The **maximum expected utility framework** (MEUF) introduced by Jansche (2007) takes **marginal** probabilities $p_1, p_2, \ldots p_m$ as inputs and **solves** (2) in $\mathcal{O}(m^4)$ time.
- If the **independence** assumption is **violated**, this method may produce predictions being **far away** from the optimal one: **the worst-case regret converges to 1 in the limit of** m.

Multinomial Distribution:

- Maximizer $\boldsymbol{h}^*_F$ of (2) consists of the k labels with the **highest** marginal probabilities, where k is the first integer for which $\sum_{j=1}^k p_j \geq (1+k)p_{k+1}$; if there is no such integer, then $\boldsymbol{h} = \mathbf{1}$ (Del Coz et al., 2009).

Thresholding on Ordered Marginal Probabilities:

- The F-maximizer is **not** necessarily **consistent** with the **order** of **marginal** label probabilities:

$\boldsymbol{y}$	$p(\boldsymbol{y})$
1 0 0 0 0 0 0 0 0 0	0.48
0 1 1 1 1 1 0 0 0 0	0.26
0 1 0 0 0 0 1 1 1 1	0.26

  The F-measure maximizer is given by $(1\,0\,0\,0\,0\,0\,0\,0\,0\,0)$; yet, not the first label but the second one exhibits the highest marginal probability.

An Exact Algorithm for F-Measure Maximization

- The algorithm follows the same decomposition of the problem to **inner** and **outer** maximization as in Jansche (2007):

$$\boldsymbol{h}^{(k)^*} = \arg\max_{\boldsymbol{h} \in H_k} \mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{Y})} \left[F(\boldsymbol{y}, \boldsymbol{h})\right], \tag{3}$$

  where $H_k = \{\boldsymbol{h} \in \{0,1\}^m \mid \sum_{i=1}^m h_i = k\}$,

$$\boldsymbol{h}^*_F = \arg\max_{\boldsymbol{h} \in \{\boldsymbol{h}^{(0)^*}, \ldots, \boldsymbol{h}^{(m)^*}\}} \mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{Y})} \left[F(\boldsymbol{y}, \boldsymbol{h})\right]. \tag{4}$$

- **Main result**: an algorithm that needs $m^2 + 1$ parameters and runs in time $o(m^3)$ to compute the F-measure maximizer exactly.

General F-Measure Maximizer

INPUT: matrix P of elements

$$p_{is} = p(Y_i = 1\,,\, s_{\boldsymbol{y}} = s), \quad i, s \in \{1, \ldots, m\}\,,$$

where $s_{\boldsymbol{y}} = \sum_{i=1}^m y_i$, and probability $p(\boldsymbol{Y} = \mathbf{0})$;

define matrix W of elements $w_{sk} = (s+k)^{-1}$, $\;s, k \in \{1, \ldots, m\}$;
compute $\mathrm{F} = \mathrm{PW}$;
for $k = 0$ **take** $\boldsymbol{h}^{(k)^*} = \mathbf{0}$, and $\mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{Y})}\left[F(\boldsymbol{y}, \mathbf{0})\right] = p(\boldsymbol{Y} = \mathbf{0})$;
for $k = 1$ to m **do**
  solve the inner optimization problem (3) that can be reformulated as:

$$\boldsymbol{h}^{(k)^*} = \arg\max_{\boldsymbol{h} \in H_k} \; 2\sum_{i=1}^m h_i \mathrm{f}_{ik}$$

  by setting $h_i = 1$ for top k elements in the k-th column of matrix F;
  store a value of

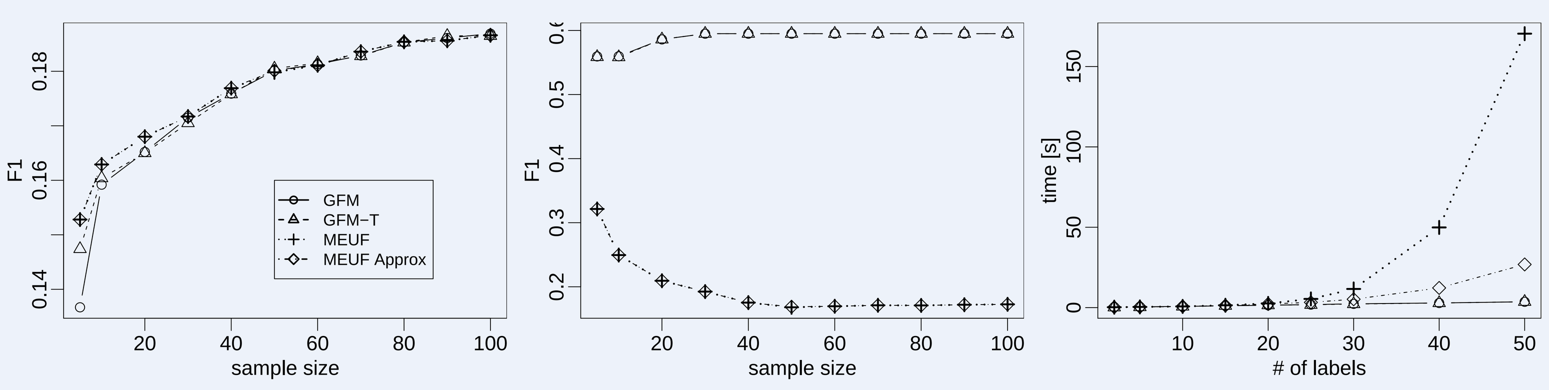
$$\mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{Y})}\left[F(\boldsymbol{y}, \boldsymbol{h}^{(k)^*})\right] = 2\sum_{i=1}^m h_i^{(k)^*} \mathrm{f}_{ik};$$

end for
solve the outer optimization problem (4):

$$\boldsymbol{h}^*_F = \arg\max_{\boldsymbol{h} \in \{\boldsymbol{h}^{(0)^*}, \ldots, \boldsymbol{h}^{(m)^*}\}} \mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{Y})}\left[F(\boldsymbol{y}, \boldsymbol{h})\right];$$

return $\boldsymbol{h}^*_F$ and $\mathbb{E}_{\boldsymbol{y} \sim p(\boldsymbol{Y})}\left[F(\boldsymbol{y}, \boldsymbol{h}^*_F)\right]$;

Application of the Algorithm



Performance under the F-measure on synthetic data of four inference methods: GFM, its thresholding variant GFM-T, MEUF, and its approximate version MEUF Approx. Left: performance as a function of sample size generated from independent random variables with $p_i = 0.12$ and $m = 25$ labels. Center: similar as above, but the distribution is defined by $p(\boldsymbol{Y} = \boldsymbol{y} \,|\, \boldsymbol{x}) = \prod_{k=1}^m p(Y_k = y_k \,|\, \boldsymbol{x}, y_1, \ldots, y_{k-1})$, where all $p(Y_i = y_i \,|\, y_1, \ldots, y_{i-1})$ are given by logistic models with a linear part $-\frac{1}{2}(i-1) + \sum_{j=1}^{i-1} y_j$. Right: running times as a function of the number of labels with a sample size of 200. All the results are averaged over 50 trials.

Method	Hamming loss	macro-F	micro-F	F	Inference time [s]	Hamming loss	macro-F	micro-F	F	Inference time [s]
	Scene: $m = 6$ (1211/1169)					Yeast: $m = 14$ (1500/917)				
PCC H	0.1030	0.6673	0.6675	0.5779	0.969	0.2046	0.3633	0.6391	0.6160	3.704
PCC GFM	0.1341	**0.7159**	0.6915	**0.7101**	0.985	0.2322	0.4034	0.6554	0.6479	3.796
PCC GFM-T	0.1343	0.7154	0.6908	0.7094	1.031	0.2324	0.4039	0.6553	0.6476	3.907
PCC MEUF approx.	0.1323	0.7131	0.6910	0.6977	1.406	0.2295	0.4030	0.6551	0.6469	10.000
PCC MEUF	0.1323	0.7131	0.6910	0.6977	1.297	0.2292	0.4034	0.6557	0.6477	11.453
BR	**0.1023**	0.6591	0.6602	0.5542	1.125	**0.1987**	0.3349	0.6299	0.6039	0.640
BR MEUF approx.	0.1140	0.7048	**0.6948**	0.6468	1.579	0.2248	**0.4098**	**0.6601**	**0.6527**	7.110
BR MEUF	0.1140	0.7048	0.6948	0.6468	2.094	0.2263	0.4096	0.6591	0.6523	10.031
	Enron: $m = 53$ (1123/579)					Mediamill: $m = 101$ (30999/12914)				
PCC H	0.0471	0.1141	0.5185	0.4892	195.061	**0.0304**	0.0931	0.5577	0.5429	1405.772
PCC GFM	0.0521	0.1618	0.5943	0.6006	194.889	0.0348	0.1491	0.5849	0.5734	1420.663
PCC GFM-T	0.0521	**0.1619**	0.5948	**0.6011**	196.030	0.0348	0.1499	0.5854	0.5737	1464.147
PCC MEUF approx.	0.0523	0.1612	0.5932	0.6007	1081.837	0.0350	0.1504	0.5871	0.5740	308582.019
PCC MEUF	0.0523	0.1612	0.5932	0.6007	6676.145	-	-	-	-	-
BR	**0.0468**	0.1049	0.5223	0.4821	8.594	**0.0304**	0.1429	0.5623	0.5462	207.655
BR MEUF approx.	0.0513	0.1554	**0.5969**	0.5947	850.494	0.3508	**0.1917**	**0.5889**	**0.5744**	258431.125
BR MEUF	0.0513	0.1554	0.5969	0.5947	7014.453	-	-	-	-	-

Experimental results on four multi-label benchmark datasets. Inference algorithms are used with Probabilistic Classifier Chains (PCC) (Dembczynski et al. 2010) and Binary Relevance (BR). Main statistics for each dataset are given: the number of labels (m), the size of training and test sets (training/test set). Symbol "-" indicates that an algorithm did not complete the computation in a reasonable time (several days). In bold: the best results for a given dataset and given performance measure.

Neural Information Processing Systems 2011, Granada, December 2011.