Supplementary: Extreme Multi-label Learning with Label Features for Warm-start Tagging, Ranking & Recommendation

Anonymous Authors Anon University anon@email.com

ACM Reference format:

Anonymous Authors. 2018. Supplementary: Extreme Multi-label Learning with Label Features for Warm-start Tagging, Ranking & Recommendation. In *Proceedings of ACM WSDM Conference, Los Angeles, CA, USA, February 2018 (WSDM'2018),* 13 pages.

https://doi.org/10.475/123_4

Section 1 presents the pseudocodes for SwiftXML training and prediction algorithms. Section 2 reports complete set of experimental results comparing SwiftXML to various baselines in terms of both propensity-scored precisions (*PSP*1, *PSP*3, *PSP*5) as well as standard precisions (*P*1, *P*3, *P*5). Section 3 shows the derivations for individual steps of the alternating minimization algorithm used for node partitioning, as well as derivations of approximations for base classifier optimizations.

1 ALGORITHMS

```
Algorithm 1 SwiftXML-PREDICT(\{\mathcal{T}_1, ... \mathcal{T}_T\}, \{\mu_1, ..., \mu_L\}, \mathbf{x}, \mathbf{z})
```

```
for i = 1,...,T do
        n \leftarrow \mathcal{T}_i.\text{root}
        while n.isleaf \neq 1 do
                 \mathbf{w}_{x} \leftarrow n.\mathbf{w}_{x}
                  \mathbf{w}_z \leftarrow n.\mathbf{w}_z
                 if C_x \mathbf{w}_x^{\top} \mathbf{x} + C_z \mathbf{w}_z^{\top} \mathbf{z} > 0 then
                         n \leftarrow n.left child
                 else
                          n \leftarrow n.right\_child
                  end if
        end while
        P_i \leftarrow n.P
end for
\mathbf{P}_{\mathrm{pf}} = \frac{1}{T} \sum_{i=1}^{T} \mathbf{P}_{i}
P_{\text{tail},l}^{I} = 1/(1 + \exp(\frac{\gamma}{2} ||\mathbf{x} - \boldsymbol{\mu}_{l}||^{2})) \ \forall l \in \{1..L\}
\mathbf{r} = \operatorname{rank}_k \left( \alpha \log(\mathbf{P}_{pf}) + (1 - \alpha) \log(\mathbf{P}_{tail}) \right)
```

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

```
WSDM'2018, February 2018, Los Angeles, CA, USA
© 2018 Association for Computing Machinery.
ACM ISBN 123-4567-24-567/08/06...$15.00
https://doi.org/10.475/123_4
```

Algorithm 2 SwiftXML-TRAIN($\{\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i\}_{i=1}^N, \mathbf{p}, T$) for i = 1,...,N do for l = 1,...,L do $y_{il}^p = y_{il}/p_{il}$ end for end for parallel-for i = 1,..,T do $n^{root} \leftarrow \text{new node}$ $n^{root}.Id \leftarrow \{1,..,N\}$ GROW-NODE-RECURSIVE($\{\mathbf{x}_i, \mathbf{y}_i^p, \mathbf{z}_i\}_{i=1}^N, n^{root}$) $\mathcal{T}_i \leftarrow \text{new tree}$ $\mathcal{T}_i.\text{root} \leftarrow n^{root}$ end parallel-for for l = 1,...,L do $\mu_l = \frac{\sum_{i=1}^{N} y_{il} \mathbf{x}_i}{\sum_{i=1}^{N} y_{il}}$ end for return $\{\mathcal{T}_1,..,\mathcal{T}_T\},\{\mu_1,..,\mu_L\}$ $\textbf{procedure} \; \texttt{grow-node-recursive}(\{\mathbf{x}_i, \mathbf{y}_i^p, \mathbf{z}_i\}_{i=1}^N, \textit{n})$ **if** $|n.Id| \leq \text{MaxLeaf then}$ n.isleaf $\leftarrow 1$ $n.P \leftarrow \text{PROCESS-LEAF}(\{y_i^p\}_{i=1}^N, n)$ # Split node and grow child nodes recursively $\{n.\mathbf{w}_x, n.\mathbf{w}_z, n.\text{left_child}, n.\text{right_child}\}$ \leftarrow SPLIT-NODE $(\{\mathbf{x}_i, \mathbf{y}_i^p, \mathbf{z}_i\}_{i=1}^N, n)$ GROW-NODE-RECURSIVE($\{\mathbf{x}_i, \mathbf{y}_i^p, \mathbf{z}_i\}_{i=1}^N, n. \text{left_child}$)
GROW-NODE-RECURSIVE($\{\mathbf{x}_i, \mathbf{y}_i^p, \mathbf{z}_i\}_{i=1}^N, n. \text{right_child}$) end if end procedure **procedure** PROCESS-LEAF($\{y_i^p\}_{i=1}^N, n$) $\mathbf{P} \leftarrow \text{top-k} \left(\frac{\sum_{i \in n.Id} \mathbf{y}_i^p}{|n.Id|} \right)$ # Return scores for top k labels

end procedure

Algorithm 3 SPLIT-NODE($\{\mathbf{x}_i, \mathbf{y}_i^p, \mathbf{z}_i\}_{i=1}^N, n$)

```
Id \leftarrow n.Id
\delta_i[0] \sim \{-1,1\}, \forall i \in Id
                                                                                                                                              # Random coin tosses
\mathbf{w}_x[0] \leftarrow \mathbf{0}, \mathbf{w}_z[0] \leftarrow \mathbf{0}, t \leftarrow 0
                                                                                                                                                   # Various counters
repeat
          \mathbf{r}^{\pm}[t+1] \leftarrow \mathrm{rank}_L \left( \sum_{i \in Id} \tfrac{1}{2} (1 \pm \delta_i[t]) I_L(\mathbf{y}_i^p) \mathbf{y}_i^p \right)
          \delta[t+1] \leftarrow \text{FDELTA}(\mathbf{w}_x, \mathbf{w}_z, \mathbf{r}^{\pm}, \{\mathbf{x}_i, \mathbf{y}_i^p, \mathbf{z}_i, \delta_i[t]\}_{i=1}^N, Id)
\mathbf{w}_x[t+1] \leftarrow \underset{\mathbf{w}_x}{\text{argmin}} \|\mathbf{w}_x\|_1 + C_x \sum_{i \in Id} \log(1 + e^{-\delta_i[t]} \mathbf{w}_x^{\top \mathbf{x}_i})
          \begin{split} & \boldsymbol{\delta}[t+1] \leftarrow \text{FDELTA}(\mathbf{w}_x, \mathbf{w}_z, \mathbf{r}^{\pm}, \{\mathbf{x}_i, \mathbf{y}_i^p, \mathbf{z}_i, \delta_i[t+1]\}_{i=1}^N, Id) \\ & \mathbf{w}_z[t+1] \leftarrow \underset{\mathbf{w}_z}{\text{argmin}} \|\mathbf{w}_z\|_1 + C_z \sum_{i \in Id} \log(1 + e^{-\delta_i[t]\mathbf{w}_z \top \mathbf{z}_i}) \end{split}
           \boldsymbol{\delta}[t+1] \leftarrow \text{FDELTA}(\mathbf{w}_x, \mathbf{w}_z, \mathbf{r}^{\pm}, \{\mathbf{x}_i, \mathbf{y}_i^p, \mathbf{z}_i, \delta_i[t+1]\}_{i=1}^N, Id)
           t \leftarrow t + 1
until \delta[t] \neq \delta[t-1]
n^+ \leftarrow \text{new node}, n^- \leftarrow \text{new node}
n^{\pm}.Id \leftarrow \left\{ i \in Id : \operatorname{sign}\{C_{x}\mathbf{w}_{x}[t]^{\top}\mathbf{x}_{i} + C_{z}\mathbf{w}_{z}[t]^{\top}\mathbf{z}_{i}\} = \pm 1 \right\}
              return \mathbf{w}_{x}[t], \mathbf{w}_{z}[t], n^{+}, n^{-}
procedure FDELTA(\mathbf{w}_x, \mathbf{w}_z, \mathbf{r}^{\pm}, \{\mathbf{x}_i, \mathbf{y}_i^p, \mathbf{z}_i, \delta_i\}_{i=1}^N, Id)
           for i \in Id do
                     v^{\pm} \leftarrow C_X \log(1 + e^{\mp \mathbf{w}_X[t]^{\top} \mathbf{x}_i})
                                   +C_z \log(1 + e^{\mp \mathbf{w}_z[t]^{\top} \mathbf{z}_i})
                                   -C_r I_L(\mathbf{y}_i^p) \sum_{l=1}^L \left( \frac{y_{ir_l^{\pm}[t+1]}^p}{\log(1+l)} \right)
                     if v^+ = v^- then
                                \delta_i' = \delta_i
                                \delta_i' = \operatorname{sign}(v^- - v^+)
                      end if
           end for
return \delta'
 end procedure
```

Supplementary : Extreme Multi-label Learning with Label Features for Warm-start Applicated 2018, February 2018, Los Angeles, CA, USA

2 RESULTS

Table 1: The proposed SwiftXML makes significantly more accurate predictions as compared to both state-of-the-art extreme classifiers and classical recommendation algorithms. SwiftXML consistently improves as more and more test labels are revealed, and achieves accuracy gains of upto 14% as compared to the baselines. Performance is evaluated using unbiased propensity-scored Precision (PSP1,PSP3,PSP5).

EURLex-4K [K[N = 15K, D = 5K, L = 4K]											
A 1: 41		20%		ı	Revea 40%	led Lab	el Percei	ntages 60%		I	80%	
Algorithm	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5
WRMF	8.87	9.80	11.05	12.44	13.69	16.58	13.59	15.50	19.77	13.21	18.10	22.85
SVD++ BPR	0.17	$0.31 \\ 1.23$	$0.41 \\ 1.13$	0.17	$0.29 \\ 0.89$	$0.51 \\ 1.01$	0.18	$0.34 \\ 0.72$	$0.61 \\ 0.86$	0.14	0.29 1.65	$0.60 \\ 2.24$
PfastreXML	43.76	45.66	48.21	41.05	42.99	48.64	39.03	42.50	51.13	33.29	44.21	52.46
SLEEC PDSparse	34.14	39.14 40.32	42.72 43.79	36.16 34.52	40.52 39.80	46.31 45.72	36.01 32.97	40.79 37.81	$\frac{48.56}{46.02}$	34.64	44.63 42.33	51.64 49.87
DiSMEC	35.15	42.85	47.03	35.77	42.39	48.30	35.87	42.93	50.54	34.44	45.11	51.63
IMC Matchbox	10.28 0.25	10.73 0.48	11.23 0.50	9.26	9.90	11.45	7.94 0.59	9.02 0.65	11.45	6.11	9.30 0.79	$\frac{11.72}{1.09}$
SwiftXML	44.49	46.13	48.46	42.83	43.56	49.72	42.27	44.72	$\frac{1.00}{53.12}$	0.60 38.52	48.18	55.70
Wiki10-31K [N = 14	K,D=1	01K, L =	= 31 <i>K</i>]								
		200				led Lab	el Percei				00%	
Algorithm	PSP1	20% PSP3	PSP5	PSP1	40% PSP3	PSP5	PSP1	60% PSP3	PSP5	PSP1	80% PSP3	PSP5
WRMF	5.93	5.40	5.27	6.80	6.10	6.01	6.82	6.34	6.34	5.74	5.70	6.33
PfastreXML SLEEC	22.78 11.10	20.46 11.92	$19.80 \\ 12.42$	20.48	18.56 11.94	18.17 12.57	17.56 10.92	16.11 11.51	16.31 12.28	13.07 9.83	13.35 10.58	$14.77 \\ 12.14$
PDSparse	9.54	8.95	8.02	8.94	7.97	6.78	7.94	6.76	5.72	6.09	5.18	4.73
DISMEC	11.99	14.10	15.47	11.87	13.81	15.19	11.43	13.01	14.53	10.23	11.83	13.87
IMC SwiftXML	2.57 23.10	2.38 20.63	2.36 19.92	3.63	3.40 19.35	3.42 19.07	4.04 17.75	3.80 16.60	3.87 17.06	3.10 14.17	3.45 14.49	3.98 16.23
AmazonCat-	13K [<i>N</i> :	= 1.18 <i>M</i>	I.D = 20	3K.L =	13 <i>K</i>]							
			,		Revea	led Lab	el Percei					
Algorithm	PSP1	20% PSP3	PSP5	PSP1	40% PSP3	PSP5	PSP1	60% PSP3	PSP5	PSP1	80% PSP3	PSP5
PfastreXML	70.36	73.92	76.32	70.30	73.22	75.80	69.23	72.39	75.17	66.80	71.55	76.30
PDSparse SwiftXML	50.65 70.40	62.57 74.44	65.25 77.17	53.52 73.89	64.27 77.94	61.61 81.10	55.90 76.37	61.18 81.00	58.37 83.77	58.17 79.78	57.41 84.31	57.47 87.83
					_	01.10	10.57	01.00	03.77	1 7 7 . 7 0	04.31	07.03
CitationNetw	/OFK-30N 	[IN=62]	K, D=39.	K, L=301		led Lab	el Percei	ntages				
Algorithm		20%			40%			60%			80%	
	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5
PfastreXML SLEEC	11.10 7.41	13.31 9.43	15.39 11.35	10.26 7.65	12.75 10.08	15.19 12.43	9.04 7.37	12.59 10.80	15.30 13.36	7.70 6.40	12.46 10.91	15.24 13.79
PDSparse	10.14	12.71	14.65	9.31	12.27	14.48	8.36	12.02	14.05	7.18	12.06	14.21
DiSMEC	11.94	15.11	17.84	11.22	14.66	17.78	9.94	14.72	18.06	8.81	15.02	18.49
SwiftXML	11.84	14.57	16.92	11.50	14.86	17.84	11.48	16.12	19.44	9.97	15.79	19.34
Amazon-79K	N = 49	90K,D =	= 136K,I	L = 79K		led Lah	el Percei	ntages				
Algorithm		20%			40%			60%			80%	
	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5
PfastreXML SLEEC	25.92 15.43	31.56 19.98	36.39 23.83	25.39 17.91	31.52 23.87	36.14 28.30	24.19 18.88	32.50 27.58	36.61 32.24	22.92 18.26	31.82 27.32	35.40 31.33
PDSparse	23.60	29.51	34.12	23.11	29.55	33.57	21.88	30.12	33.54	20.59	29.82	32.85
DISMEC	27.13	35.15	41.89	26.67	35.34	41.94	25.53	36.54	42.54	24.43	36.31	41.86
SwiftXML	26.48	32.43	37.69	29.42	37.64	42.80	36.04	47.40	51.33	35.15	46.47	49.44
Wikipedia-50	00K [<i>N</i> =	= 1.81 <i>M</i>	,D = 2.3	58M,L =		led Lab	el Percei	ntages				
Algorithm		_20%		l	40%		1	60%		l	80%	
	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5
PfastreXML	33.76	32.00	33.34	32.17	31.19	33.35	29.38	30.27	33.22	26.26	31.39	35.22
SwiftXML	35.48	33.42	34.76	34.19	33.04	35.31	31.49	32.49	35.68	28.33	33.90	38.07

Table 2: The proposed SwiftXML makes significantly more accurate predictions as compared to both state-of-the-art extreme classifiers and classical recommendation algorithms. SwiftXML consistently improves as more and more test labels are revealed, and achieves accuracy gains of upto 14% as compared to the baselines. Performance is evaluated using unbiased propensity-scored nDCG (PSN1,PSN3,PSN5).

EURLex-4K [N = 15k	X,D = 5H	K, L = 4F	(]								
A1 '-1		20%		I	Revea 40%	aled Lab	el Percer	ntages 60%		ı	80%	
Algorithm	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5
WRMF	8.87	9.54	10.29	12.44	13.31	14.90	13.59	14.81	16.98	13.21	16.16	18.30
SVD++ BPR	0.17	$0.26 \\ 1.22$	$0.32 \\ 1.16$	0.17 1.18	$0.25 \\ 0.97$	0.37 1.03	0.18	$0.27 \\ 0.81$	$0.41 \\ 0.89$	0.14 1.09	$0.22 \\ 1.42$	0.36 1.68
PfastreXML SLEEC	43.76	45.16	46.67	41.05	42.43 39.31	45.53	39.03	41.19	45.57	33.29	39.81	43.54
PDSparse	34.14	37.82 38.73	40.05 40.93	36.16 34.52	38.34	$42.54 \\ 41.65$	36.01 32.97	39.16 36.16	43.11 40.33	34.64 31.05	40.63 37.89	43.82 41.29
DiSMEC IMC	35.15	40.75 10.65	43.45 10.94	35.77 9.26	40.53 9.75	43.87	35.87 7.94	40.63 8.67	44.51 9.90	34.44 6.11	40.95	43.91 9.15
Matchbox	10.28	10.03	10.94	_	9.73	10.58 -	_	-	9.90	-0	8.06	_
SwiftXML	44.49	45.67	47.04	42.83	43.40	46.75	42.27	43.77	48.02	38.52	44.33	47.73
Wiki10-31K	N = 14I	K,D=10	01K, L =	31K]								
A 1		20%		I	Revea 40%	aled Lab	el Percer	ntages 60%		ı	80%	
Algorithm	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5
WRMF	5.93	5.53	5.43	6.80	6.26	6.18	6.82	6.45	6.43	5.74	5.68	6.03
PfastreXML SLEEC	22.78	21.04 11.73	$20.50 \\ 12.08$	20.48	19.05 11.76	18.72 12.18	17.56 10.92	16.49 11.34	16.54 11.83	13.07 9.83	13.23 10.33	$\frac{14.01}{11.20}$
PDSparse	9.54	9.11	8.50	8.94	8.23	7.46	7.94	7.08	6.42	6.09	5.45	5.18
DiSMEC IMC	11.99	13.56 2.43	$\frac{14.53}{2.41}$	11.87	13.31 3.47	$\frac{14.24}{3.47}$	11.43	12.59 3.86	13.55 3.90	10.23 3.10	11.32 3.36	12.47 3.65
SwiftXML	23.10	21.23	20.66	21.30	19.84	19.57	17.75	16.87	17.10	14.17	14.36	15.32
AmazonCat-	13K [<i>N</i> =	= 1.18 <i>M</i>	D = 203	3K, L = 1	3K]							
A.1. :.1		20%		ı	Revea 40%	aled Lab	el Percer	ntages 60%		ı	80%	
Algorithm	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5
PfastreXML	70.36	72.94	74.44	70.30	72.37	73.89	69.23	71.30	72.78	66.80	69.68	71.86
PDSparse SwiftXML	50.65 70.40	59.28 73.35	61.35 75.05	53.52 73.89	61.28 76.78	60.08 78.65	55.90 76.37	59.87 79.48	58.46 80.98	58.17 79.78	57.65 82.60	57.68 84.22
CitationNetw												
<u>Charloth (ct)</u>		1 1 021	t, D 371	t, 12 301t		aled Lab	el Percer	ntages				
Algorithm	PSN1	20% PSN3	PSN5	PSN1	40% PSN3	PSN5	PSN1	60% PSN3	PSN5	PSN1	80% PSN3	PSN5
PfastreXML	11.10	12.61	13.76	10.26	11.89	13.16	9.04	11.29	12.59	7.70	10.54	11.74
SLEEC	7.41	8.78	9.85	7.65	9.25	10.48	7.37	9.55	10.78	6.40	9.07	10.32
PDSparse DiSMEC	10.14	11.89 14.09	12.98 15.61	9.31 11.22	11.26 13.48	12.27 15.11	8.36 9.94	10.67 12.97	11.66 14.58	7.18 8.81	10.08 12.50	$11.02 \\ 14.00$
SwiftXML	11.84	13.71	15.02	11.50	13.71	15.27	11.48	14.42	16.02	9.97	13.45	14.99
Amazon-79K	[N = 49]	0K,D =	136K,L	= 79K]								
						aled Lab	el Percer					
Algorithm	PSN1	20% PSN3	PSN5	PSN1	40% PSN3	PSN5	PSN1	60% PSN3	PSN5	PSN1	80% PSN3	PSN5
PfastreXML	-	_	_	-	_	_	<u> </u>	-	_	<u> </u>	_	_
SLEEC PDSparse	15.43 23.60	18.49 27.57	20.49 29.97	17.91 23.11	$21.75 \\ 27.24$	23.90 29.20	18.88 21.88	24.21 26.97	26.29 28.51	18.26 20.59	23.56 26.03	25.23 27.29
DiSMEC	27.13	32.52	36.02	26.67	32.26	35.45	25.53	32.30	34.97	24.43	31.39	33.68
SwiftXML	-	_	_	–			–	_		-		
Wikipedia-50	00K [N =	= 1.81 <i>M</i> ,	D = 2.3	8M, L =		1 1	1.0					
Algorithm		20%		I	Revea 40%	aled Lab	el Percer 	ntages 60%		l	80%	
	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5
PfastreXML	33.76	32.45	33.13	32.17	31.36	32.48	29.38	29.81	31.27	26.26	29.35	31.07
SwiftXML	35.48	33.95	34.63	34.19	33.25	34.43	31.49	31.98	33.57	28.33	31.69	33.55

Table 3: The proposed SwiftXML performs consistently better, across different revealed label percentages, as compared to baseline PfastreXML extensions: PfastreXML-early and PfastreXML-late. Performance is evaluated according to the unbiased propensity scored Precisions (PSP1,PSP3,PSP5).

EURLex-4K $[N = 15K, D = 5K, L = 4K]$												
					Revea	led Lab	el Percei	ntages				
4.1 1.1		20%			40%			60%			80%	
Algorithm	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5
DfootvoVMI oorly	43.67	45.11	47.29	41.77	43.54	49.82	39.30	43.08	52.04	35.31	46.10	54.40
PfastreXML-early PfastreXML-late	43.76	45.66	48.21	42.17	43.55	49.55	39.64	43.57	52.25	31.13	40.13	46.05
SwiftXML	43.03	44.49	46.65	42.83	43.56	49.72	42.27	44.72	53.12	38.52	48.18	55.70
	13.03	11.17	10.00	12.03	13.30	17.,2	12.27	11.,2	33.12	30.32	10.10	33.70
Wiki10-31K [N = 141	K,D=1	01K,L =	= 31 <i>K</i>]								
				i.		led Lab	el Percei	-		i.		
Algorithm		20%			40%		202	60%		202	80%	
	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5
PfastreXML-early	22.98	20.40	19.59	20.29	18.52	18.17	17.50	16.15	16.31	13.27	13.52	14.94
PfastreXML-late	22.78	20.46	19.80	20.63	18.47	18.03	17.56	16.11	16.31	13.20	13.28	14.65
SwiftXML	23.10	20.63	19.92	21.30	19.35	19.07	17.75	16.60	17.06	14.17	14.49	16.23
AmazonCat-1	13K [<i>N</i> :	= 1.18 <i>M</i>	I, D = 20	3K, L =	13 <i>K</i>]							
						led Lab	el Percei	ntages				
		20%		1	40%	пса дар		60%		1	80%	
Algorithm	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5	PSP1	PSP3	PSP5
DC + VAG 1	67.95	71.42	74.56	71.39	75.30	78.53	72.87	76.98	79.77	71.57	76.93	81.52
PfastreXML-early	69.83	73.70	76.23	70.86	74.72	78.20	72.65	77.78	81.43	73.60	80.74	85.40
PfastreXML-late												
SwiftXML 70.40 74.44 77.17 73.89 77.94 81.10 76.37 81.00 83.77 79.78 84.31 87.83												
CitationNetwork-36K [$N = 62K, D = 39K, L = 36K$]												
CitationNetw	ork-36k	K[N=6]	2K,D =	39K,L		led Lab	el Percei	ntages				
	ork-36k	$\frac{\sqrt{N} = 6}{20\%}$	52K, D =	39K,L		led Lab	el Percei	ntages 60%			80%	
Algorithm	ork-36k PSP1		PSP5	39K,L	Revea	led Labo	el Percer	-	PSP5	PSP1	80% PSP3	PSP5
Algorithm	PSP1	20% PSP3	PSP5	PSP1	Revea 40% PSP3	PSP5	PSP1	60% PSP3		1	PSP3	
		20%			Revea			60%	PSP5 15.43 19.11	PSP1 7.88 9.25		PSP5 15.54 19.86
Algorithm PfastreXML-early	PSP1 10.74	20% PSP3 12.98	PSP5 15.04	PSP1 10.08	Revea 40% PSP3 12.55	PSP5 15.08	PSP1 9.14	60% PSP3 12.65	15.43	7.88	PSP3 12.60	15.54
Algorithm PfastreXML-early PfastreXML-late SwiftXML	PSP1 10.74 11.11 11.84	20% PSP3 12.98 13.57 14.57	PSP5 15.04 15.73 16.92	PSP1 10.08 10.92 11.50	Revea 40% PSP3 12.55 14.16 14.86	PSP5 15.08 17.12	PSP1 9.14 10.53	60% PSP3 12.65 15.27	15.43 19.11	7.88 9.25	PSP3 12.60 15.73	15.54 19.86
Algorithm PfastreXML-early PfastreXML-late	PSP1 10.74 11.11 11.84	20% PSP3 12.98 13.57 14.57	PSP5 15.04 15.73 16.92	PSP1 10.08 10.92 11.50	Revea 40% PSP3 12.55 14.16 14.86	PSP5 15.08 17.12 17.84	PSP1 9.14 10.53 11.48	60% PSP3 12.65 15.27 16.12	15.43 19.11	7.88 9.25	PSP3 12.60 15.73	15.54 19.86
Algorithm PfastreXML-early PfastreXML-late SwiftXML	PSP1 10.74 11.11 11.84	20% PSP3 12.98 13.57 14.57	PSP5 15.04 15.73 16.92	PSP1 10.08 10.92 11.50	Revea 40% PSP3 12.55 14.16 14.86	PSP5 15.08 17.12 17.84	PSP1 9.14 10.53	60% PSP3 12.65 15.27 16.12	15.43 19.11	7.88 9.25	PSP3 12.60 15.73 15.79	15.54 19.86
Algorithm PfastreXML-early PfastreXML-late SwiftXML	PSP1 10.74 11.11 11.84	20% PSP3 12.98 13.57 14.57	PSP5 15.04 15.73 16.92	PSP1 10.08 10.92 11.50	Revea 40% PSP3 12.55 14.16 14.86	PSP5 15.08 17.12 17.84	PSP1 9.14 10.53 11.48	60% PSP3 12.65 15.27 16.12	15.43 19.11	7.88 9.25	PSP3 12.60 15.73	15.54 19.86
Algorithm PfastreXML-early PfastreXML-late SwiftXML Amazon-79K	PSP1 10.74 11.11 11.84 [N = 49]	20% PSP3 12.98 13.57 14.57 90K,D =	PSP5 15.04 15.73 16.92	PSP1 10.08 10.92 11.50 L = 79K	Revea 40% PSP3 12.55 14.16 14.86	PSP5 15.08 17.12 17.84	PSP1 9.14 10.53 11.48	60% PSP3 12.65 15.27 16.12 ntages 60%	15.43 19.11 19.44	7.88 9.25 9.97	PSP3 12.60 15.73 15.79	15.54 19.86 19.34
Algorithm PfastreXML-early PfastreXML-late SwiftXML Amazon-79K Algorithm	PSP1 10.74 11.11 11.84 [N = 49]	20% PSP3 12.98 13.57 14.57 90K, D =	PSP5 15.04 15.73 16.92 = 136K,1	PSP1 10.08 10.92 11.50 L = 79K	Revea 40% PSP3 12.55 14.16 14.86] Revea 40% PSP3	PSP5 15.08 17.12 17.84 led Labo	PSP1 9.14 10.53 11.48 PSP1 PSP1	60% PSP3 12.65 15.27 16.12 ntages 60% PSP3	15.43 19.11 19.44 PSP5	7.88 9.25 9.97	PSP3 12.60 15.73 15.79 80% PSP3	15.54 19.86 19.34 PSP5
Algorithm PfastreXML-early PfastreXML-late SwiftXML Amazon-79K Algorithm PfastreXML-early	PSP1 10.74 11.11 11.84 [N = 49]	20% PSP3 12.98 13.57 14.57 90K, D = 20% PSP3 31.45	PSP5 15.04 15.73 16.92 = 136K,1	PSP1 10.08 10.92 11.50 L = 79K PSP1 25.32	Revea 40% PSP3 12.55 14.16 14.86] Revea 40% PSP3 31.47	PSP5 15.08 17.12 17.84 led Laber PSP5 36.18	PSP1 9.14 10.53 11.48 PSP1 PSP1 24.08	60% PSP3 12.65 15.27 16.12 ntages 60% PSP3 32.42	15.43 19.11 19.44 PSP5 36.64	7.88 9.25 9.97 PSP1 22.76	PSP3 12.60 15.73 15.79 80% PSP3 31.75	15.54 19.86 19.34 PSP5 35.43
Algorithm PfastreXML-early PfastreXML-late SwiftXML Amazon-79K Algorithm PfastreXML-early PfastreXML-late	PSP1 10.74 11.11 11.84 [N = 49] PSP1 25.83 25.20 26.48	20% PSP3 12.98 13.57 14.57 90K, D = 20% PSP3 31.45 31.43 32.43	PSP5 15.04 15.73 16.92 = 136K,1 PSP5 36.45 36.71 37.69	PSP1 10.08 10.92 11.50 11.50 PSP1 25.32 27.18 29.42	Revea 40% PSP3 12.55 14.16 14.86 Revea 40% PSP3 31.47 34.80 37.64	PSP5 15.08 17.12 17.84 led Laber PSP5 36.18 40.42	PSP1 9.14 10.53 11.48 PSP1 PSP1 24.08 30.35	60% PSP3 12.65 15.27 16.12 ntages 60% PSP3 32.42 41.22	15.43 19.11 19.44 PSP5 36.64 46.71	7.88 9.25 9.97 PSP1 22.76 30.14	PSP3 12.60 15.73 15.79 80% PSP3 31.75 41.55	15.54 19.86 19.34 PSP5 35.43 45.98
Algorithm PfastreXML-early PfastreXML-late SwiftXML Amazon-79K Algorithm PfastreXML-early PfastreXML-late SwiftXML	PSP1 10.74 11.11 11.84 [N = 49] PSP1 25.83 25.20 26.48	20% PSP3 12.98 13.57 14.57 90K, D = 20% PSP3 31.45 31.43 32.43	PSP5 15.04 15.73 16.92 = 136K,1 PSP5 36.45 36.71 37.69	PSP1 10.08 10.92 11.50 11.50 PSP1 25.32 27.18 29.42	Revea 40% PSP3 12.55 14.16 14.86] Revea 40% PSP3 31.47 34.80 37.64	PSP5 15.08 17.12 17.84 led Labe PSP5 36.18 40.42 42.80	PSP1 9.14 10.53 11.48 PSP1 PSP1 24.08 30.35	60% PSP3 12.65 15.27 16.12 ntages 60% PSP3 32.42 41.22 47.40	15.43 19.11 19.44 PSP5 36.64 46.71	7.88 9.25 9.97 PSP1 22.76 30.14	PSP3 12.60 15.73 15.79 80% PSP3 31.75 41.55	15.54 19.86 19.34 PSP5 35.43 45.98
Algorithm PfastreXML-early PfastreXML-late SwiftXML Amazon-79K Algorithm PfastreXML-early PfastreXML-late SwiftXML Wikipedia-50	PSP1 10.74 11.11 11.84 [N = 49] PSP1 25.83 25.20 26.48	20% PSP3 12.98 13.57 14.57 90K, D = 20% PSP3 31.45 31.43 32.43	PSP5 15.04 15.73 16.92 = 136K,1 PSP5 36.45 36.71 37.69	PSP1 10.08 10.92 11.50 11.50 PSP1 25.32 27.18 29.42	Revea 40% PSP3 12.55 14.16 14.86] Revea 40% PSP3 31.47 34.80 37.64	PSP5 15.08 17.12 17.84 led Labe PSP5 36.18 40.42 42.80	PSP1 9.14 10.53 11.48 PSP1 PSP1 24.08 30.35 36.04	60% PSP3 12.65 15.27 16.12 ntages 60% PSP3 32.42 41.22 47.40	15.43 19.11 19.44 PSP5 36.64 46.71	7.88 9.25 9.97 PSP1 22.76 30.14	PSP3 12.60 15.73 15.79 80% PSP3 31.75 41.55	15.54 19.86 19.34 PSP5 35.43 45.98
Algorithm PfastreXML-early PfastreXML-late SwiftXML Amazon-79K Algorithm PfastreXML-early PfastreXML-late SwiftXML	PSP1 10.74 11.11 11.84 [N = 49] PSP1 25.83 25.20 26.48	20% PSP3 12.98 13.57 14.57 90K, D = 20% PSP3 31.45 31.43 32.43 = 1.81M	PSP5 15.04 15.73 16.92 = 136K,1 PSP5 36.45 36.71 37.69	PSP1 10.08 10.92 11.50 11.50 PSP1 25.32 27.18 29.42	Revea 40% PSP3 12.55 14.16 14.86] Revea 40% PSP3 31.47 34.80 37.64 = 501K] Revea	PSP5 15.08 17.12 17.84 led Labe PSP5 36.18 40.42 42.80	PSP1 9.14 10.53 11.48 PSP1 PSP1 24.08 30.35 36.04	60% PSP3 12.65 15.27 16.12 ntages 60% PSP3 32.42 41.22 47.40	15.43 19.11 19.44 PSP5 36.64 46.71	7.88 9.25 9.97 PSP1 22.76 30.14	PSP3 12.60 15.73 15.79 80% PSP3 31.75 41.55 46.47	15.54 19.86 19.34 PSP5 35.43 45.98
Algorithm PfastreXML-early PfastreXML-late SwiftXML Amazon-79K Algorithm PfastreXML-early PfastreXML-late SwiftXML Wikipedia-50	PSP1 10.74 11.11 11.84 [N = 49] PSP1 25.83 25.20 26.48 0K [N =	20% PSP3 12.98 13.57 14.57 90K, D = 20% PSP3 31.45 31.43 32.43 = 1.81M	PSP5 15.04 15.73 16.92 = 136K,1 PSP5 36.45 36.71 37.69 ,D = 2.3	PSP1 10.08 10.92 11.50 11.50 L = 79K PSP1 25.32 27.18 29.42 388M,L =	Revea 40% PSP3 12.55 14.16 14.86] Revea 40% PSP3 31.47 34.80 37.64 = 501K] Revea 40%	PSP5 15.08 17.12 17.84 led Labe PSP5 36.18 40.42 42.80	PSP1 9.14 10.53 11.48 PSP1 PSP1 24.08 30.35 36.04	60% PSP3 12.65 15.27 16.12 ntages 60% PSP3 32.42 41.22 47.40	15.43 19.11 19.44 PSP5 36.64 46.71 51.33	7.88 9.25 9.97 PSP1 22.76 30.14 35.15	PSP3 12.60 15.73 15.79 80% PSP3 31.75 41.55 46.47	15.54 19.86 19.34 PSP5 35.43 45.98 49.44
Algorithm PfastreXML-early PfastreXML-late SwiftXML Amazon-79K Algorithm PfastreXML-early PfastreXML-late SwiftXML Wikipedia-50 Algorithm	PSP1 10.74 11.11 11.84 [N = 49] PSP1 25.83 25.20 26.48 0K [N = 49]	20% PSP3 12.98 13.57 14.57 20% PSP3 31.45 31.43 32.43 = 1.81M 20% PSP3	PSP5 15.04 15.73 16.92 = 136K,1 PSP5 36.45 36.71 37.69 ,D = 2.3	PSP1 10.08 10.92 11.50	Revea 40% PSP3 12.55 14.16 14.86 Revea 40% PSP3 31.47 34.80 37.64 E501K] Revea 40% PSP3	PSP5 15.08 17.12 17.84 led Labo PSP5 36.18 40.42 42.80 led Labo PSP5	PSP1 9.14 10.53 11.48 PSP1 24.08 30.35 36.04 PSP1	60% PSP3 12.65 15.27 16.12 ntages 60% PSP3 32.42 41.22 47.40 ntages 60% PSP3	15.43 19.11 19.44 PSP5 36.64 46.71 51.33	7.88 9.25 9.97 PSP1 22.76 30.14 35.15	PSP3 12.60 15.73 15.79 80% PSP3 31.75 41.55 46.47	15.54 19.86 19.34 PSP5 35.43 45.98 49.44
Algorithm PfastreXML-early PfastreXML-late SwiftXML Amazon-79K Algorithm PfastreXML-early PfastreXML-late SwiftXML Wikipedia-50 Algorithm PfastreXML-early	PSP1 10.74 11.11 11.84 [N = 49] PSP1 25.83 25.20 26.48 0K [N = 49] PSP1 34.59	20% PSP3 12.98 13.57 14.57 20% PSP3 31.45 31.43 32.43 = 1.81M 20% PSP3 32.86	PSP5 15.04 15.73 16.92 = 136K,1 PSP5 36.45 36.71 37.69 ,D = 2.3	PSP1 10.08 10.92 11.50 11.50 PSP1 25.32 27.18 29.42 PSP1 33.14	Revea 40% PSP3 12.55 14.16 14.86 Revea 40% PSP3 31.47 34.80 37.64 E 501K] Revea 40% PSP3	PSP5 15.08 17.12 17.84 led Labo PSP5 36.18 40.42 42.80 PSP5 34.44	PSP1 9.14 10.53 11.48 PSP1 24.08 30.35 36.04 PSP1 30.37	60% PSP3 12.65 15.27 16.12 ntages 60% PSP3 32.42 41.22 47.40 ntages 60% PSP3	15.43 19.11 19.44 PSP5 36.64 46.71 51.33 PSP5 34.43	7.88 9.25 9.97 PSP1 22.76 30.14 35.15 PSP1 27.16	PSP3 12.60 15.73 15.79 80% PSP3 31.75 41.55 46.47 80% PSP3 32.51	15.54 19.86 19.34 PSP5 35.43 45.98 49.44 PSP5 36.47

Table 4: The proposed SwiftXML performs consistently better, across different revealed label percentages, as compared to baseline PfastreXML extensions: PfastreXML-early and PfastreXML-late. Performance is evaluated according to the unbiased propensity scored nDCGs (PSN1,PSN3,PSN5).

EURLex-4K	$\lceil N = 1$	5K D =	5K I :	= 4K
LUNLEX-41	11V — 1	ω	· JK.L ·	- 411

		Revealed Label Percentages												
Algorithm		20%			40%			60%			80%			
	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5		
PfastreXML-early	43.67	44.70	45.99	41.77	43.05	46.50	39.30	41.70	46.25	35.31	44.03	45.49		
PfastreXML-late	43.76	45.16	46.67	42.17	43.16	46.46	39.64	42.15	46.56	31.13	36.64	39.30		
SwiftXML	44.49	45.67	47.04	42.83	43.40	46.75	42.27	43.77	48.02	38.52	44.33	47.73		

Wiki10-31K [N = 14K, D = 101K, L = 31K]

					Revea	ıled Labe	el Percer	ntages				
Almonithm		20%			40%			60%			80%	
Algorithm	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5
PfastreXML-early	22.98	21.03	20.39	20.29	18.95	18.66	17.50	16.48	16.53	13.27	13.41	14.19
PfastreXML-late	22.78	21.04	20.50	20.63	19.01	18.64	17.56	16.49	16.54	13.20	13.21	13.96
SwiftXML	23.10	21.23	20.66	21.30	19.84	19.57	17.75	16.87	17.10	14.17	14.36	15.32

AmazonCat-13K [N = 1.18M, D = 203K, L = 13K]

					Revea	led Labe	el Percer	ntages				
A 1: 41		20%			40%			60%			80%	
Algorithm	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5
PfastreXML-early	67.95	70.45	72.38	71.39	74.16	76.07	72.87	75.62	77.12	71.57	76.08	76.97
PfastreXML-late	69.83	72.65	74.22	70.86	73.61	75.66	72.65	76.04	78.00	73.60	78.06	80.21
SwiftXML	70.40	73.35	75.05	73.89	76.78	78.65	76.37	79.48	80.98	79.78	82.60	84.22

CitationNetwork-36K [N = 62K, D = 39K, L = 36K]

					Revea	led Labe	el Percer	ntages				
A 1: 41		20%			40%			60%			80%	
Algorithm	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5
PfastreXML-early	10.74				11.71	13.03	9.14	11.38	12.71	7.88	10.72	11.98
PfastreXML-late	11.11	12.79	13.98	10.92	13.06	14.61	10.53	13.52	15.36	9.25	13.12	14.89
SwiftXML	11.84	13.71	15.02	11.50	13.71	15.27	11.48	14.42	16.02	9.97	13.45	14.99

Amazon-79K [N = 490K, D = 136K, L = 79K]

					Revea	aled Lab	el Percer	ntages				
Algorithm		20%			40%			60%			80%	
Algorithm	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5
PfastreXML-early	_	-	-	-	-	-	_	-	-	_	-	_
PfastreXML-late	_	_	_	_	_	_	_	_	_	_	_	_
SwiftXML	_	-	-	_	-	-	_	-	-	_	-	_

Wikipedia-500K [N = 1.81M, D = 2.38M, L = 501K]

					Revea	led Lab	el Percer	ntages				
A 1 : +1		20%			40%			60%			80%	
Algorithm	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5	PSN1	PSN3	PSN5
PfastreXML-early	34.59	33.29	33.99	33.14	32.38	33.53	30.37	30.88	32.39	27.16	30.39	32.16
PfastreXML-late	33.88	32.35	32.72	32.30	31.84	33.12	29.90	30.79	32.48	27.42	30.99	32.92
SwiftXML	35.48	33.95	34.63	34.19	33.25	34.43	31.49	31.98	33.57	28.33	31.69	33.55

Table 5: The proposed SwiftXML makes significantly more accurate predictions as compared to both state-of-the-art extreme classifiers as well as classical recommendation algorithms. SwiftXML consistently improves as more and more test labels are revealed, and achieves accuracy gains of upto 3% as compared to the baselines. Performance is evaluated using standard precisions (P1,P3,P5).

Algorithm	EURLex-4K [N = 15F	K,D=5	K, L = 4	<i>K</i>]								
MRMF 22.71 17.20 14.27 28.77 21.02 16.71 27.80 18.66 14.17 22.11 12.95 95 0.95							led Lab	el Percei					
SVD++	Algorithm	P1		P5	P1		P5	P1		P5	P1		P5
BPR													
PfastreXML					1								
SLEEC 72.72 57.63 46.01 70.73 52.31 39.16 62.77 41.66 29.52 49.93 27.36 18.73 SPOSPATS 71.44 57.13 45.84 65.42 49.52 37.39 55.12 37.32 62.69 43.50 42.84 17.34 DISMRC 78.10 64.27 51.17 72.59 55.81 40.99 64.19 44.05 30.39 50.85 27.72 18.55 Matchbox 0.58 20.59 15.86 24.26 16.78 12.64 18.54 12.21 91.4 11.76 75.2 55.4 Matchbox 0.58 50.02 45.40 64.53 49.95 38.78 58.15 40.90 29.87 47.02 26.74 18.65 Wiki10-31K N = 14K, D = 101K, L = 31K	_				1			1			1		
DisMEC 78.10 64.27 51.17 72.59 55.81 40.99 64.19 44.05 30.39 50.85 27.72 18.42 MacChbox 0.58 0.78 0.59 1.678 12.64 18.54 12.21 11.16 7.52 55.44 Matchbox 0.58 0.78 0.59 1.678 12.64 18.54 12.21 11.16 17.52 55.44 Matchbox 0.58 0.78 0.59 1.678 12.64 18.54 12.21 11.06 0.85 0.73 1.08 0.60 0.49 SwiftXML 67.58 55.02 45.40 64.53 49.59 38.78 58.15 40.90 29.87 47.02 26.74 18.65 Wiki10-31K N = 14K, D = 101K, L = 31K					1			l			1		
$ \begin{array}{c c c c c c c c c c c c c c c c c c c $					1			1			1		
$ \begin{array}{c c c c c c c c c c c c c c c c c c c $													
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$						-	-				1		
Algorithm P1 20% P5 P1 P3 P5 P1 P6 P1 P7 P8 P5 P1 P7 P7 P5 P1 P7 P5	SwiftXML	67.58	55.02	45.40	64.53	49.59	38.78	58.15	40.90	29.87	47.02	26.74	18.65
Algorithm	Wiki10-31K [N = 141	K,D=1	01K, L =	= 31 <i>K</i>]								
MRMF			0.00				led Lab	el Percei			ı	000	
PfastreXML	Algorithm	P1		P5	P1		P5	P1		P5	P1		P5
SLEEC 78.88 64.35 53.29 73.61 57.07 46.29 63.13 45.31 35.50 44.60 28.17 21.05	_				1			1			1		
PDSparse								1			1		
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$								1					
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$													
$\begin{array}{ c c c c c c c c c c c c c c c c c c c$					1			1					
$ \begin{array}{ c c c c c c c c } \hline Algorithm & \begin{array}{ c c c c c c c c } \hline & 20\% & P1 & P3 & P5 \\ \hline P1 & P3 & P5 \\ \hline & P2 & P3 & P5 \\ \hline & P3 & P5 \\ \hline & P1 & P3 & P5 \\ \hline & P1 & P3 & P5 \\ \hline & P2 & P3 & P5 \\ \hline & P3 & P5 \\ \hline & P1 & P3 & P5 \\ \hline & P3 & P5 \\ \hline & P1 & P3 & P5 \\ \hline & P1 & P3 & P5 \\ \hline & P1 & P3 & P5 \\ \hline & P2 & P3 & P5 \\ \hline & P2 & P3 & P5 \\ \hline & P3 & P5 & P1 & P3 & P5 \\ \hline & P4 & P3 & P5 & P1 & P3 \\ \hline & P4 & P3 & P5 & P1 & P3 \\ \hline & P4 & P3 & P5 & P1 & P3 \\ \hline & P4 & P3 & P5 & P1 & P3 \\ \hline & P4 & P3 & P5 & P1 & P3 \\ \hline & P4 & P3 & P5 & P1 & P3 \\ \hline & P4 & P3 & P5 & P1 & P3 \\ \hline & P5 & P1 & P3 & P5 & P1 \\ \hline & P5 & P1 & P3 & P5 & P1 \\ \hline & P5 & P1 & P3 & P5 \\$	SWIITANIL	60.85	51.15	45.48	55.09	44.97	39.27	47.85	30.83	30.96	30.83	22.32	18.06
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	AmazonCat-1	13K [<i>N</i> :	= 1.18M	I,D=20	3K, L =		1 1 7 1	1.0					
P1			20%		ı		led Lab	el Percei			ı	80%	
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	Algorithm	P1		P5	P1		P5	P1		P5	P1		P5
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$					1			1			1		
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$					1						1		
$ \begin{array}{ c c c c c c c c c } \hline Algorithm & P_1 & 20\% & P_5 & P_1 & P_3 & P_5 & P_1 & P_3 & P_5 & P_1 & P_3 & P_5 \\ \hline PfastreXML & 18.61 & 13.92 & 11.12 & 16.59 & 11.93 & 9.35 & 13.55 & 9.30 & 7.13 & 10.60 & 6.75 & 5.05 \\ SLEEC & 15.61 & 10.56 & 8.32 & 15.25 & 9.95 & 7.64 & 13.22 & 8.22 & 6.11 & 10.19 & 6.00 & 4.41 \\ PDSparse & 19.02 & 13.27 & 10.16 & 16.78 & 11.32 & 8.31 & 13.62 & 8.63 & 6.18 & 10.55 & 6.27 & 4.39 \\ SwiftXML & 20.23 & 15.11 & 12.06 & 19.11 & 13.75 & 10.76 & 17.35 & 11.53 & 8.68 & 13.88 & 8.24 & 6.11 \\ \hline Amazon-79K & & & & & & & & & & & & & & & & & & &$	SWILKINE	00.07	70.00	01.12	00.03	73.11	J2.74	00.73	37.12	10.17	04.01	37.20	24.70
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	CitationNetw	ork-36k	S[N=6]	52K,D =	39K,L								
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$			20%		ı		led Lab	el Percei			ı	8007	
SLEEC 15.61 10.56 8.32 15.25 9.95 7.64 13.22 8.22 6.11 10.19 6.00 4.41 PDSparse 19.02 13.27 10.16 16.78 11.32 8.31 13.62 8.63 6.18 10.55 6.27 4.39 SwiftXML 20.23 15.11 12.06 19.11 13.75 10.76 17.35 11.53 8.68 13.88 8.24 6.11 Amazon-79K $[N = 490K, D = 136K, L = 79K]$ Revealed Label Percentages 40% 60% 80%	Algorithm	P1		P5	P1		P5	P1		P5	P1		P5
$\begin{array}{ c c c c c c c c c c c c c c c c c c c$	PfastreXML	18.61	13.92	11.12	16.59	11.93	9.35	13.55	9.30	7.13	10.60	6.75	5.05
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$								l			1		
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$					1			1			1		
$\begin{array}{ c c c c c c c c c }\hline & & & & & & & & & & & & & & & & & & &$		_				_	10,70	17.00	11.00	0.00	10.00	0.21	0.11
$ \begin{array}{ c c c c c c c c c c c c c c c c c c c$	Amazon-79K	$\lfloor N = 49 \rfloor$	90K,D =	= 136K,1	L = 79K		led Lab	al Percer	ntages				
Algorithm P1 P3 P5 PfastreXML 32.32 22.70 16.55 31.19 20.39 14.15 28.44 15.96 10.77 25.99 12.26 8.18 SLEEC 20.85 14.72 10.89 23.43 15.62 11.02 23.22 13.52 9.32 21.33 10.43 7.07 PDSparse 30.06 20.95 15.17 28.78 18.78 12.83 25.87 14.50 9.65 23.39 11.24 7.40 DiSMEC 35.26 25.02 18.53 33.88 22.49 15.94 30.68 17.59 12.14 27.99 13.65 9.34 SwiftXML 33.21 27.70 17.05 35.88 23.86 16.35 39.27 21.61 14.06 36.90 16.52 10.	A.1 11		20%		I		neu Labt					80%	
SLEEC 20.85 14.72 10.89 23.43 15.62 11.02 23.22 13.52 9.32 21.33 10.43 7.07 PDSparse 30.06 20.95 15.17 28.78 18.78 12.83 25.87 14.50 9.65 23.39 11.24 7.40 DiSMEC 35.26 25.02 18.53 33.88 22.49 15.94 30.68 17.59 12.14 27.99 13.65 9.34 SwiftXML 33.21 27.70 17.05 35.88 23.86 16.35 39.27 21.61 14.06 36.90 16.52 10.56 Wikipedia-500K [N = 1.81M, D = 2.38M, L = 501K] Revealed Label Percentages 40% 60% 80%	Algorithm	P1		P5	P1		P5	P1		P5	P1		P5
PDSparse JS 20.95 15.17 28.78 18.78 12.83 25.87 14.50 9.65 23.39 11.24 7.40 DiSMEC 35.26 25.02 18.53 33.88 22.49 15.94 30.68 17.59 12.14 27.99 13.65 9.34 SwiftXML 33.21 27.70 17.05 35.88 23.86 16.35 39.27 21.61 14.06 36.90 16.52 10.56 Wikipedia-500K $[N = 1.81M, D = 2.38M, L = 501K]$ Revealed Label Percentages 40% 60% 80%	PfastreXML	32.32	22.70		31.19	20.39		28.44	15.96	10.77	25.99	12.26	8.18
DiSMEC 35.26 25.02 18.53 33.88 22.49 15.94 30.68 17.59 12.14 27.99 13.65 9.34 SwiftXML 33.21 27.70 17.05 35.88 23.86 16.35 39.27 21.61 14.06 36.90 16.52 10.56 Wikipedia-500K $[N = 1.81M, D = 2.38M, L = 501K]$ Revealed Label Percentages 40% 60% 80%													
SwiftXML 33.21 27.70 17.05 35.88 23.86 16.35 39.27 21.61 14.06 36.90 16.52 10.56 Wikipedia-500K $[N=1.81M, D=2.38M, L=501K]$ Revealed Label Percentages Absorbber 20% 40% 60% 80%													
Revealed Label Percentages 40% 60% 80%													
Revealed Label Percentages 40% 60% 80%	Wikipedia-50	0K [<i>N</i> =	= 1.81 <i>M</i>	D = 2.3	38M.L =	= 501 <i>K</i>]							
Algorithm 20% 40% 60% 80%				,	, , , , , , , , , , , ,		led Lab	el Percei	ntages				
P1 P3 P5	Algorithm	_		_	_	40%			60%	_	_		_
	- 11501111111	P1	P3	P5	P1		P5	P1	P3	P5	P1	P3	P5
PfastreXML 57.78 38.14 28.62 52.20 32.48 23.70 43.53 24.59 17.49 33.40 16.69 11.48													
SwiftXML 59.58 39.07 29.21 54.54 33.66 24.44 45.95 25.76 18.22 35.48 17.51 11.99	SWIITXML	59.58	39.07	29.21	54.54	33.66	24.44	45.95	25.76	18.22	35.48	17.51	11.99

Table 6: The proposed SwiftXML makes significantly more accurate predictions as compared to both state-of-the-art extreme classifiers as well as classical recommendation algorithms. SwiftXML consistently improves as more and more test labels are revealed, and achieves accuracy gains of upto 3% as compared to the baselines. Performance is evaluated using standard nDCG metrics (N1,N3,N5).

EURLex-4K [N = 15I	K,D=5	K, L = 4	<i>K</i>]								
						aled Lab	el Perce					
Algorithm	N1	20% N3	N5	N1	40% N3	N5	N1	60% N3	N5	N1	80% N3	N5
WDME	1			1								
WRMF SVD++	22.71 0.42	18.43 0.52	16.84 0.56	28.77	22.99 0.45	24.05 0.65	27.80 0.37	23.34 0.47	$26.27 \\ 0.72$	22.11 0.24	23.76 0.36	26.66 0.61
BPR	3.87	3.07	2.47	3.57	2.20	2.17	2.79	1.64	1.74	2.23	2.49	2.95
PfastreXML	67.52	59.72	57.08	61.41	53.63	55.69	53.08	48.35	53.97	38.86	44.71	49.22
SLEEC	72.72	61.48	56.87	70.73	57.68	58.45	62.77	53.06	57.42	49.93	52.25	55.47
PDSparse	71.44	60.74	56.61	65.42	54.37	55.52	55.21	47.54	52.01	43.50	47.14	50.76
DiSMEC	78.10	67.83	62.94	72.59	60.93	61.30	64.19	55.75	59.57	50.85	53.32	56.06
IMC Matchbox	29.38	22.71	20.40	24.26	18.80	18.99	18.54	15.47	17.28	11.76	13.57	15.20
SwiftXML	67.58	58.19	55.15	64.53	54.16	56.44	58.15	51.15	56.39	47.02	50.48	54.39
117:1:40 04IZ [<i>V</i> D 4	0117.1	0.17/1								
Wiki10-31K [N = 14	K,D=1	01K, L =	= 31K J	Darra	-1 - J T - L	al Danca					
		20%		ſ	40%	aled Lab	ei Perce	mages 60%		I.	80%	
Algorithm	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
WRMF	35.50	27.98	24.08	37.94	28.81	24.41	34.57	25.59	21.45	23.78	17.00	15.60
PfastreXML	62.59	54.77	49.87	55.20	46.80	42.11	44.68	36.31	32.49	29.55	23.45	22.98
SLEEC	78.88	67.81	59.27	73.61	60.94	52.55	63.13	49.38	42.18	44.60	32.58	30.11
PDSparse	75.47	59.54	48.83	65.90	49.10	38.51	52.75	36.56	28.15	32.68	121.25	17.98
DiSMEC	80.52	71.22	63.78	73.34	62.27	54.68	62.14	49.65	42.94	42.99	32.27	30.02
IMC SwiftXML	5.65	5.15 53.32	4.88	6.18	5.41 47.27	4.94	6.03	4.97 39.30	4.45	3.72 30.83	3.20	3.10
SWIIIANIL	60.85	33.34	48.85	55.09	47.47	42.81	47.85	39.30	35.21	30.63	24.91	24.32
AmazonCat-	13K [<i>N</i> :	= 1.18 <i>N</i>	I,D=20	3K, L =								
						aled Lab	el Perce					
Algorithm	2.74	20%	N.T.	2.74	40%	> T=	274	60%	N.T.	NT4	80%	N.T.
	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
PfastreXML	85.42	82.35	82.49	82.93	79.65	81.12	78.37	77.59	79.41	71.02	75.06	77.26
PDSparse	87.91	80.72	77.94	84.46	76.75	74.74	78.18	72.28	71.49	67.21	67.90	67.97
SwiftXML	86.69	83.53	83.35	88.03	84.40	85.73	86.73	85.77	87.17	84.81	87.23	88.64
CitationNetw	ork-36F	$K[N=\epsilon$	52K,D =	39K,L	= 36K]							
	1				Reve	aled Lab	el Perce	ntages				
Algorithm		20%		1	40%			60%			80%	
Algorithm	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
PfastreXML	18.61	18.39	19.33	16.59	16.95	18.41	13.55	15.64	17.36	10.60	14.57	16.34
SLEEC	15.61	13.94	14.25	15.25	14.00	14.84	13.22	13.85	15.05	10.19	13.04	14.43
PDSparse	19.02	17.84	18.35	16.78	16.40	17.30	13.62	14.97	16.12	10.55	13.89	15.08
SwiftXML	20.23	19.87	20.78	19.11	19.23	20.76	17.35	19.15	20.95	13.88	17.86	19.83
Amazon-79K	N = 4	90K,D =	= 136K,	L = 79K	1							
	Ī		,,			aled Lab	el Perce	ntages				
Algorithma		20%			40%			60%			80%	
Algorithm	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
PfastreXML	-	_	_	l –	_	_	-	_	_	-	_	_
SLEEC	20.85	21.19	22.31	23.43	23.85	25.58	23.22	25.93	27.77	21.33	26.54	28.18
PDSparse	30.06	31.04	32.87	28.78	30.37	32.31	25.87	29.56	31.18	23.39	29.04	30.39
DiSMEC	35.26	36.92	39.58	33.88	36.23	39.19	30.68	35.54	38.14	27.99	35.03	37.36
SwiftXML	_	_	_	_	_	_	_	_	_	_	_	_
Wikipedia-50	00K [<i>N</i> =	= 1.81 <i>M</i>	D = 2.3	38M,L =	= 501 <i>K</i>]							
			,	,—		aled Lab	el Perce	ntages				
A.1. **1		20%		1	40%	area Dab		60%		1	80%	
Algorithm	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
PfastreXML	57.78	48.61	47.23	52.20	44.41	44.59	43.53	40.25	41.56	33.40	36.45	38.50
SwiftXML	59.58	49.75	48.11	54.54	45.93	45.91	45.95	41.95	43.17	35.48	38.19	40.23
	1			1			1			1		

Table 7: The proposed SwiftXML performs consistently better, across different revealed label percentages, as compared to baseline PfastreXML extensions which make use of label features. Performance is evaluated according to the standard Precisions (P1,P3,P5).

EURLex-4K [N = 15F	K,D=5	K, L = 4	K]								
	Revealed Label Percentages											
A 1 '.1		20%			40%			60%			80%	
Algorithm	P1	P3	P5	P1	P3	P5	P1	P3	P5	P1	P3	P5
PfastreXML-early	66.79	55.23	46.14	61.54	49.31	38.98	53.71	39.27	29.23	41.56	25.45	18.18
PfastreXML-late	67.52	56.94	47.50	62.09	49.51	38.78	53.66	39.52	29.16	37.36	22.09	15.42
SwiftXML	67.58	55.02	45.40	64.53	49.59	38.78	58.15	40.90	29.87	47.02	26.74	18.65
Wiki10-31K [N = 14	K,D=1	01K,L =	= 31 <i>K</i>]								
	Revealed Label Percentages											
Algorithm		20%			40%			60%			80%	
	P1	P3	P5	P1	P3	P5	P1	P3	P5	P1	P3	P5
PfastreXML-early	61.59	51.19	44.96	54.94	44.51	38.46	44.53	34.10	28.58	30.74	21.55	17.36
PfastreXML-late	62.59	52.47	46.23	53.69	43.13	37.16	44.68	33.84	28.43	30.18	20.93	17.01
SwiftXML	60.85	51.15	45.48	55.09	44.97	39.27	47.85	36.83	30.96	30.83	22.32	18.06
AmazonCat-1	13K [<i>N</i> :	= 1.18 <i>M</i>	I, D = 20	3K, L =	13 <i>K</i>]							
					Revea	led Labe	el Percei	ıtages				
۸ 1 مد مانداه سم		20%			40%			60%			80%	
Algorithm	P1	P3	P5	P1	P3	P5	P1	P3	P5	P1	P3	P5
PfastreXML-early	80.36	70.99	58.12	82.82	70.01	51.09	82.13	56.20	38.36	76.00	36.00	23.34
PfastreXML-late	85.92	75.21	60.43	85.39	70.50	51.30	83.61	56.97	39.14	78.50	37.67	24.31
SwiftXML	86.69	76.08	61.12	88.03	73.11	52.94	86.73	59.12	40.17	84.81	39.28	24.96
CitationNetw	ork-36k	X[N=6]	52K,D =	39K,L	= 36 <i>K</i>]							
						led Labe	el Percei	ntages				
Algorithm	D4	20%	D.=	704	40%	70.5	704	60%	70.5	704	80%	D.5
	P1	P3	P5	P1	P3	P5	P1	P3	P5	P1	P3	P5
PfastreXML-early	18.36	13.72	10.97	16.62	11.87	9.36	14.02	9.44	7.23	11.00	6.88	5.17
PfastreXML-late	19.31	14.12	11.14	17.31				10.24				
SwiftXML	20.23		10.06		12.54	9.85	14.79		8.02	11.87	7.66	5.84
Amazon-79K	20.20	15.11	12.06	19.11	12.54 13.75	9.85 10.76	14.79 17.35	11.53	8.02 8.68	11.87 13.88	7.66 8.24	
- 1111a2O11- / 9IX	I			19.11	13.75							5.84
7 111102O11-7 7K	I			19.11	13.75		17.35	11.53				5.84
	[N=49]	90K,D =	= 136 <i>K</i> , i	19.11 $L = 79K$	13.75 Revea 40%	10.76 led Labe	17.35	11.53 ntages 60%	8.68	13.88	8.24	5.84 6.11
Algorithm	I	90K,D =		19.11	13.75 Revea	10.76	17.35	11.53			8.24	5.84
Algorithm PfastreXML-early	[N = 49]	90K,D = 20% P3 22.68	P5	19.11 $L = 79K$ P1 31.19	13.75 Revea 40% P3 20.41	10.76 led Labe P5 14.18	17.35 el Percei P1 28.42	11.53 ntages 60% P3 15.94	P5 10.79	P1 25.90	8.24 80% P3 12.26	5.84 6.11 P5 8.20
Algorithm PfastreXML-early PfastreXML-late	[N = 49] P1 32.30 31.44	90K, D = 20% P3 22.68 22.48	P5 16.58 16.54	19.11 $L = 79K$ P1 31.19 32.92	13.75 Revea 40% P3 20.41 22.08	10.76 led Labe P5 14.18 15.47	17.35 el Percer P1 28.42 34.08	11.53 ntages 60% P3 15.94 19.33	P5 10.79 13.10	P1 25.90 32.46	80% P3 12.26 15.19	5.84 6.11 P5 8.20 10.07
Algorithm PfastreXML-early	[N = 49]	90K,D = 20% P3 22.68	P5	19.11 $L = 79K$ P1 31.19	13.75 Revea 40% P3 20.41	10.76 led Labe P5 14.18	17.35 el Percei P1 28.42	11.53 ntages 60% P3 15.94	P5 10.79	P1 25.90	8.24 80% P3 12.26	5.84 6.11 P5 8.20
Algorithm PfastreXML-early PfastreXML-late	$ \begin{bmatrix} N = 49 \\ & \end{bmatrix} $ $ \begin{bmatrix} N = 49 \\ & \end{bmatrix} $ $ \begin{bmatrix} 32.30 \\ 31.44 \\ 33.21 \end{bmatrix} $	90K, D = 20% P3 22.68 22.48 27.70	P5 16.58 16.54 17.05	P1 31.19 32.92 35.88	13.75 Revea 40% P3 20.41 22.08 23.86	10.76 led Labe P5 14.18 15.47	17.35 el Percer P1 28.42 34.08	11.53 ntages 60% P3 15.94 19.33	P5 10.79 13.10	P1 25.90 32.46	80% P3 12.26 15.19	5.84 6.11 P5 8.20 10.07
Algorithm PfastreXML-early PfastreXML-late SwiftXML	$ \begin{bmatrix} N = 49 \\ & \end{bmatrix} $ $ \begin{bmatrix} N = 49 \\ & \end{bmatrix} $ $ \begin{bmatrix} 32.30 \\ 31.44 \\ 33.21 \end{bmatrix} $	90K, D = 20% P3 22.68 22.48 27.70 = 1.81M	P5 16.58 16.54 17.05	P1 31.19 32.92 35.88	Revea 40% P3 20.41 22.08 23.86 501K]	10.76 led Labe P5 14.18 15.47	17.35 el Percer P1 28.42 34.08 39.27	11.53 ntages 60% P3 15.94 19.33 21.61	P5 10.79 13.10	P1 25.90 32.46	80% P3 12.26 15.19 16.52	5.84 6.11 P5 8.20 10.07
Algorithm PfastreXML-early PfastreXML-late SwiftXML Wikipedia-50	P1 32.30 31.44 33.21 00K [N =	90K, D = 20% P3 22.68 22.48 27.70 = 1.81M	P5 16.58 16.54 17.05 , D = 2.3	P1 31.19 32.92 35.88 38M,L =	Revea 40% P3 20.41 22.08 23.86 : 501K] Revea 40%	10.76 P5 14.18 15.47 16.35	17.35 el Percer P1 28.42 34.08 39.27	11.53 ntages 60% P3 15.94 19.33 21.61 ntages 60%	P5 10.79 13.10 14.06	P1 25.90 32.46 36.90	8.24 80% P3 12.26 15.19 16.52	5.84 6.11 P5 8.20 10.07 10.56
Algorithm PfastreXML-early PfastreXML-late SwiftXML	$ \begin{bmatrix} N = 49 \\ & \end{bmatrix} $ $ \begin{bmatrix} N = 49 \\ & \end{bmatrix} $ $ \begin{bmatrix} 32.30 \\ 31.44 \\ 33.21 \end{bmatrix} $	90K, D = 20% P3 22.68 22.48 27.70 = 1.81M	P5 16.58 16.54 17.05 , D = 2.3	P1 31.19 32.92 35.88	Revea 40% P3 20.41 22.08 23.86 • 501K] Revea 40% P3	10.76 lled Labo P5 14.18 15.47 16.35	17.35 el Percer P1 28.42 34.08 39.27	11.53 ntages 60% P3 15.94 19.33 21.61 ntages 60% P3	P5 10.79 13.10	P1 25.90 32.46	80% P3 12.26 15.19 16.52	5.84 6.11 P5 8.20 10.07
Algorithm PfastreXML-early PfastreXML-late SwiftXML Wikipedia-50	P1 32.30 31.44 33.21 00K [N =	90K, D = 20% P3 22.68 22.48 27.70 = 1.81M 20% P3 38.63	P5 16.58 16.54 17.05 , D = 2.3	P1 31.19 32.92 35.88 P1 F1 52.62	Revea 40% P3 20.41 22.08 23.86 Footh J Revea 40% P3 33.19	10.76 P5 14.18 15.47 16.35 lled Labe	17.35 el Percei P1 28.42 34.08 39.27 el Percei P1 44.25	11.53 ntages 60% P3 15.94 19.33 21.61 ntages 60% P3 25.26	P5 10.79 13.10 14.06 P5 17.95	P1 25.90 32.46 36.90 P1 34.11	8.24 80% P3 12.26 15.19 16.52 80% P3 17.13	P5 8.20 10.07 10.56 P5 11.78
Algorithm PfastreXML-early PfastreXML-late SwiftXML Wikipedia-50 Algorithm	P1 32.30 31.44 33.21 00K [N =	90K, D = 20% P3 22.68 22.48 27.70 = 1.81M 20% P3	P5 16.58 16.54 17.05 , D = 2.3	19.11 L = 79K P1 31.19 32.92 35.88 88M,L =	Revea 40% P3 20.41 22.08 23.86 • 501K] Revea 40% P3	10.76 P5 14.18 15.47 16.35	17.35 el Percei P1 28.42 34.08 39.27 el Percei	11.53 ntages 60% P3 15.94 19.33 21.61 ntages 60% P3	P5 10.79 13.10 14.06	P1 25.90 32.46 36.90	8.24 80% P3 12.26 15.19 16.52 80% P3	P5 8.20 10.07 10.56

Table 8: The proposed SwiftXML performs consistently better, across different revealed label percentages, as compared to baseline PfastreXML extensions which make use of label features. Performance is evaluated according to the standard nDCG metrics (N1,N3,N5).

•												
EURLex-4K [N = 15I	K,D=5	K, L = 4	<i>K</i>]								
					Revea	led Lab	el Perce	ntages				
		20%			40%			60%		I	80%	
Algorithm	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
DftVMIl	66.79	58.15	55.64	61.54	53.28	56.05	53.71	48.70	54.37	41.56	49.62	51.35
PfastreXML-early	67.52	59.72	57.08	62.09	53.55	56.03	53.66	49.00	54.43	37.36	42.89	44.31
PfastreXML-late SwiftXML	67.58	58.19	55.15	64.53	54.16	56.44	58.15	51.15	56.39	47.02	50.48	54.39
JWIIIAML	07.30	30.17	33.13	04.33	34.10	30.44	30.13	31.13	30.37	47.02	30.40	34.37
Wiki10-31K [N = 14	K,D=1	.01K,L =	= 31 <i>K</i>]								
Algorithm	Revealed Label Percentages											
		20%			40%			60%			80%	
7 Hgoritimi	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
PfastreXML-early	61.59	53.53	48.63	54.94	46.86	42.16	44.53	36.45	32.59	30.74	24.22	23.56
PfastreXML-late	62.59	54.77	49.87	53.69	45.52	40.86	44.68	36.31	32.49	30.18	23.58	23.05
SwiftXML	60.85	53.32	48.85	55.09	47.27	42.81	47.85	39.30	35.21	30.83	24.91	24.32
	1017 []	4.401	(D 00	017.1	1077]					'		
AmazonCat-	13K [N :	= 1.18//	1, D = 20	3K,L =								
		20~		ı		lled Lab	el Percei	_		ı	22~	
Algorithm	274	20%		274	40%		274	60%		274	80%	
	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
PfastreXML-early	80.36	77.68	78.26	82.82	80.28	82.09	82.13	81.82	83.54	76.00	80.25	82.31
PfastreXML-late	85.92	82.76	82.78	85.39	81.72	83.26	83.61	82.66	84.41	78.50	82.57	84.48
SwiftXML	86.69	83.53	83.35	88.03	84.40	85.73	86.73	85.77	87.17	84.81	87.23	88.64
CitationNetw	ork-36k	$X[N=\epsilon]$	52K, D =	: 39K,L	= 36K]							
					Revea	led Lab	el Perce	ntages				
		20%			40%			60%		I	80%	
Algorithm	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
PfastreXML-early	18.36	18.03	18.88	16.62	16.78	18.23	14.02	15.89	17.60	11.00	14.90	16.73
PfastreXML-late	19.31	18.91	19.72	17.31	17.83	19.32	14.79	17.08	19.12	11.87	16.33	18.50
SwiftXML	20.23	19.87	20.78	19.11	19.23	20.76	17.35	19.15	20.95	13.88	17.86	19.83
	1						1 -,			1		
Amazon-79K	[N=49]	90K,D =	= 136 <i>K</i> , i	L = 79K]							
					Revea	led Lab	el Perce	ntages				
Algorithm		20%			40%			60%			80%	
Algorithm	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
PfastreXML-early	l –	_	_	l –	_	_	_	_	_	_	_	_
PfastreXML-late	_	_	_	_	_	_	_	_	_	_	_	_
SwiftXML	_	_	_	_	_	_	_	_	_	_	_	_
Wikipedia-50	00K [<i>N</i> =	= 1.81 <i>M</i>	D = 2.3	38M,L =	501 <i>K</i>]							
1						led Lab	el Perce	ntages				
		20%			40%	2 2		60%		I	80%	
Algorithm	N1	N3	N5	N1	N3	N5	N1	N3	N5	N1	N3	N5
PfastreXML-early	57.81	49.98	47.62	52.62	45.08	45.27	44.25	41.11	42.46	34.11	37.32	39.41
PfastreXML-late	57.11	48.61	47.35	52.24	44.95	45.12	44.43	41.28	42.63	34.94	38.02	40.18
min											_ 5.50	
SwiftXML	59.58	49.75	48.11	54.54	45.93	45.91	45.95	41.95	43.17	35.48	38.19	40.23

3 DERIVATIONS OF OPTIMIZATION ALGORITHMS

3.1 Node Partitioning Objective

Objective: SwiftXML uses the following node partitioning objective:

$$\begin{aligned} & \mathbf{Min} \ \mathcal{F}(\{\mathbf{x}_i, \mathbf{y}_i^r, \mathbf{z}_i\} | \mathbf{w}_X, \mathbf{w}_Z, \mathbf{r}^{\pm}, \boldsymbol{\delta}) \\ & = \mathbf{Min} \ \| \mathbf{w}_X \|_1 + C_X \sum_i \mathcal{L}_{\text{reg}}(\delta_i \mathbf{w}_X^{\top} \mathbf{x}_i) + \| \mathbf{w}_Z \|_1 + C_Z \sum_i \mathcal{L}_{\text{reg}}(\delta_i \mathbf{w}_Z^{\top} \mathbf{z}_i) \\ & + C_r \sum_i \left(\frac{1 + \delta_i}{2} \mathcal{L}_{\text{rank}}(\mathbf{r}^+, \mathbf{y}_i^r) + \frac{1 - \delta_i}{2} \mathcal{L}_{\text{rank}}(\mathbf{r}^-, \mathbf{y}_i^r) \right) \\ & \mathbf{w.r.t.} \ \mathbf{w}_X \in \mathcal{R}^D, \mathbf{w}_Z \in \mathcal{R}^{D'}, \boldsymbol{\delta} \in \{-1, +1\}^L, \mathbf{r}^+, \mathbf{r}^- \in \Pi(1, L) \end{aligned}$$

where
$$\mathcal{L}_{reg}(x) = \log(1 + e^{-x})$$
, $\mathcal{L}_{rank}(r, y) = -\frac{\sum_{l=1}^{L} \frac{y_l}{p_l \log(r_l + 1)}}{\sum_{l=1}^{L} \frac{1}{\log(l + 1)}}$ (1)

where, i enumerates the training users; $\delta_i \in \{-1,+1\}$ indicates the user assignment to either negative (right) or positive (left) partition; $\mathbf{w}_x, \mathbf{w}_z$ represent the separating hyperplanes learned in the user and item-set feature spaces; \mathbf{r}^+ and \mathbf{r}^- represent the item ranking variables for positive and negative partitions; $\Pi(1,L)$ denotes the space of all possible rankings over the L items; C_x, C_z, C_r are SwiftXML hyper-parameters; p_I are the item propensity scores.

The above objective is optimized through an alternating minimization algorithm which alternately optimizes over one of the four classes of variables $(\mathbf{w}_x, \mathbf{w}_z, \mathbf{r}^\pm, \delta)$ at a time with the others held constant.

For the following discussions, let:

$$F(\mathbf{x}_{i}, \mathbf{y}_{i}, \mathbf{z}_{i} | \mathbf{w}_{x}, \mathbf{w}_{z}, \mathbf{r}^{\pm}, \delta_{i}) = C_{x} \mathcal{L}_{\text{reg}}(\delta_{i} \mathbf{w}_{x}^{\top} \mathbf{x}_{i}) + C_{z} \mathcal{L}_{\text{reg}}(\delta_{i} \mathbf{w}_{z}^{\top} \mathbf{z}_{i})$$

$$+ C_{r} \left(\frac{1 + \delta_{i}}{2} \mathcal{L}_{\text{rank}}(\mathbf{r}^{+}, \mathbf{y}_{i}^{r}) + \frac{1 - \delta_{i}}{2} \mathcal{L}_{\text{rank}}(\mathbf{r}^{-}, \mathbf{y}_{i}^{r}) \right)$$

$$(2)$$

Hence:

$$\mathbf{Min}_{\mathbf{w}_{x},\mathbf{w}_{z},\boldsymbol{\delta},\mathbf{r}^{\pm}} \mathcal{F}(\{\mathbf{x}_{i},\mathbf{y}_{i}^{r},\mathbf{z}_{i}\}|\mathbf{w}_{x},\mathbf{w}_{z},\mathbf{r}^{\pm},\boldsymbol{\delta})$$

$$= \mathbf{Min}_{\mathbf{w}_{x},\mathbf{w}_{z},\boldsymbol{\delta},\mathbf{r}^{\pm}} \|\mathbf{w}_{x}\|_{1} + \|\mathbf{w}_{z}\|_{1} + \sum_{i} F(\mathbf{x}_{i},\mathbf{y}_{i},\mathbf{z}_{i}|\mathbf{w}_{x},\mathbf{w}_{z},\mathbf{r}^{\pm},\boldsymbol{\delta}_{i})$$
(3)

Minimization w.r.t δ : Keeping \mathbf{r}^{\pm} , \mathbf{w}_x , \mathbf{w}_z constant and optimizing w.r.t δ reduces (3) to:

$$\begin{aligned} & \boldsymbol{\delta}^* = \mathbf{Argmin}_{\boldsymbol{\delta} \in \{-1,+1\}^L} \ \mathcal{F}(\{\mathbf{x}_i, \mathbf{y}_i^r, \mathbf{z}_i\} | \mathbf{w}_X, \mathbf{w}_z, \mathbf{r}^{\pm}, \boldsymbol{\delta}) \\ & \equiv \boldsymbol{\delta}^* = \mathbf{Argmin}_{\boldsymbol{\delta} \in \{-1,+1\}^L} \ \sum_i F(\mathbf{x}_i, \mathbf{y}_i^r, \mathbf{z}_i | \mathbf{w}_X, \mathbf{w}_z, \mathbf{r}^{\pm}, \delta_i) \end{aligned}$$

Since δ_i are separable:

$$\begin{split} &\equiv \boldsymbol{\delta}_{i}^{*} = \mathbf{Argmin}_{\boldsymbol{\delta}_{i} \in \{-1,+1\}} \ F(\mathbf{x}_{i}, \mathbf{y}_{i}^{r}, \mathbf{z}_{i} | \mathbf{w}_{x}, \mathbf{w}_{z}, \mathbf{r}^{\pm}, \boldsymbol{\delta}_{i}) \\ &\equiv \boldsymbol{\delta}_{i}^{*} = \mathbf{Sign} \left(F(\mathbf{x}_{i}, \mathbf{y}_{i}^{r}, \mathbf{z}_{i} | \mathbf{w}_{x}, \mathbf{w}_{z}, \mathbf{r}^{\pm}, +1) - F(\mathbf{x}_{i}, \mathbf{y}_{i}^{r}, \mathbf{z}_{i} | \mathbf{w}_{x}, \mathbf{w}_{z}, \mathbf{r}^{\pm}, -1) \right) \\ &\equiv \boldsymbol{\delta}_{i}^{*} = \mathbf{Sign} \left(C_{x} \log \left(\frac{1 + e^{-\mathbf{w}_{x}^{\top} \mathbf{x}_{i}}}{1 + e^{+\mathbf{w}_{x}^{\top} \mathbf{x}_{i}}} \right) + C_{z} \log \left(\frac{1 + e^{-\mathbf{w}_{z}^{\top} \mathbf{z}_{i}}}{1 + e^{+\mathbf{w}_{z}^{\top} \mathbf{z}_{i}}} \right) \right. \\ &+ C_{r} (\mathcal{L}_{\text{rank}}(\mathbf{r}^{+}, \mathbf{y}) - \mathcal{L}_{\text{rank}}(\mathbf{r}^{-}, \mathbf{y})) \right) \end{split}$$

$$\equiv \delta_i^* = \mathbf{Sign} \left(C_X \mathbf{w}_X^{\top} \mathbf{x}_i + C_Z \mathbf{w}_Z^{\top} \mathbf{z}_i + C_r (\mathcal{L}_{rank}(\mathbf{r}^+, \mathbf{y}) - \mathcal{L}_{rank}(\mathbf{r}^-, \mathbf{y})) \right)$$
where each δ_i^* can be derived by solving the above trivial equation.

Minimization w.r.t \mathbf{r}^{\pm} : Keeping δ , \mathbf{w}_x , \mathbf{w}_z constant and optimizing w.r.t \mathbf{r}^{\pm} reduces (3) to:

$$\mathbf{r}^{\pm *} = \mathbf{Argmin}_{\mathbf{r}^{\pm} \in \Pi(1,L)} \mathcal{F}(\{\mathbf{x}_{i}, \mathbf{y}_{i}^{r}, \mathbf{z}_{i}\} | \mathbf{w}_{x}, \mathbf{w}_{z}, \mathbf{r}^{\pm}, \boldsymbol{\delta})$$

$$\equiv \mathbf{r}^{\pm *} = \mathbf{Argmin}_{\mathbf{r}^{\pm} \in \Pi(1,L)} \sum_{i} F(\mathbf{x}_{i}, \mathbf{y}_{i}^{r}, \mathbf{z}_{i} | \mathbf{w}_{x}, \mathbf{w}_{z}, \mathbf{r}^{\pm}, \delta_{i})$$

After ignoring r[±] independent terms:

$$\equiv \mathbf{r}^{\pm*} = \mathbf{Argmin}_{\mathbf{r}^{\pm} \in \Pi(1,L)} \sum_{i} \left(\frac{1 + \delta_{i}}{2} \mathcal{L}_{rank}(\mathbf{r}^{+}, \mathbf{y}_{i}^{r}) + \frac{1 - \delta_{i}}{2} \mathcal{L}_{rank}(\mathbf{r}^{-}, \mathbf{y}_{i}^{r}) \right)$$

Since \mathbf{r}^+ and \mathbf{r}^- terms are separable:

$$\mathbf{r}^{+*} = \mathbf{Argmin}_{\mathbf{r}^+ \in \Pi(1,L)} \sum_{i} \left(\frac{1+\delta_i}{2} \mathcal{L}_{\text{rank}}(\mathbf{r}^+, \mathbf{y}_i^r) \right)$$

and

$$\mathbf{r}^{-*} = \mathbf{Argmin}_{\mathbf{r}^- \in \Pi(1,L)} \sum_{i} \left(\frac{1 - \delta_i}{2} \mathcal{L}_{\text{rank}}(\mathbf{r}^-, \mathbf{y}_i^r) \right)$$

Now,

$$\mathbf{r}^{+*} = \mathbf{Argmin}_{\mathbf{r}^{+} \in \Pi(1,L)} \sum_{i} \left(\frac{1 + \delta_{i}}{2} \mathcal{L}_{rank}(\mathbf{r}^{+}, \mathbf{y}_{i}^{r}) \right)$$

$$\equiv \mathbf{r}^{+*} = \mathbf{Argmin}_{\mathbf{r}^{+} \in \Pi(1,L)} \sum_{i:\delta_{i}=1} \mathcal{L}_{rank}(\mathbf{r}^{+}, \mathbf{y}_{i}^{r})$$

$$\equiv \mathbf{r}^{+*} = \mathbf{Argmin}_{\mathbf{r}^{+} \in \Pi(1,L)} \sum_{i:\delta_{i}=1} - \frac{\sum_{l=1}^{L} \frac{y_{il}^{r}}{p_{l} \log(r_{l}+1)}}{\sum_{l=1}^{L} \frac{1}{\log(l+1)}}$$

$$\equiv \mathbf{r}^{+*} = \mathbf{Argmax}_{\mathbf{r}^{+} \in \Pi(1,L)} \sum_{i:\delta_{i}=1} \sum_{l=1}^{L} \frac{y_{il}^{r}}{p_{l} \log(r_{l}+1)}$$

$$\equiv \mathbf{r}^{+*} = \mathbf{Argmax}_{\mathbf{r}^{+} \in \Pi(1,L)} \sum_{l=1}^{L} \frac{\sum_{i:\delta_{i}=1} y_{il}^{r}}{p_{l} \log(r_{l}+1)}$$

$$\equiv \mathbf{r}^{+*} = \mathbf{Argmax}_{\mathbf{r}^{+} \in \Pi(1,L)} \left(\sum_{s} \tilde{\mathbf{y}}_{i}^{r} \right)^{\mathsf{T}} \mathbf{d}$$

$$(4)$$

where $\tilde{y}_{il}^r = \frac{y_{il}^r}{p_l}$ and **d** is an *L*-vector such that $d_l = 1/\log(1 + r_l^+)$. Since \mathbf{r}^+ are permutations of $1, 2, \ldots, L$ it is clear that (4) will be maximized if r_l is chosen as the index of the l^{th} largest value in the vector $\sum_{i:\delta_i=11} \tilde{\mathbf{y}}_i$. Thus:

$$\mathbf{r}^{+^*} = \operatorname{rank}_L \left(\sum_{i:\delta_i = +1} \tilde{\mathbf{y}}_i^r \right)$$

and through similar derivations:

$$\mathbf{r}^{-*} = \operatorname{rank}_{L} \left(\sum_{i:\delta_{i}=-1} \tilde{\mathbf{y}}_{i}^{r} \right)$$

Minimization w.r.t \mathbf{w}_x : Keeping $\boldsymbol{\delta}, \mathbf{w}_z, \mathbf{r}^{\pm}$ constant and optimizing w.r.t \mathbf{w}_x reduces (3) to:

$$\begin{aligned} \mathbf{w}_{x}^{*} &= \mathbf{Argmin}_{\mathbf{w}_{x}} \mathcal{F}(\{\mathbf{x}_{i}, \mathbf{y}_{i}^{r}, \mathbf{z}_{i}\} | \mathbf{w}_{x}, \mathbf{w}_{z}, \mathbf{r}^{\pm}, \boldsymbol{\delta}) \\ &\equiv \mathbf{w}_{x}^{*} &= \mathbf{Argmin}_{\mathbf{w}_{x}} \|\mathbf{w}_{x}\|_{1} + \sum_{i} F(\mathbf{x}_{i}, \mathbf{y}_{i}^{r}, \mathbf{z}_{i} | \mathbf{w}_{x}, \mathbf{w}_{z}, \mathbf{r}^{\pm}, \boldsymbol{\delta}_{i}) \end{aligned}$$

After ignoring terms independent of \mathbf{w}_x :

$$\equiv \mathbf{w}_{x}^{*} = \mathbf{Argmin}_{\mathbf{w}_{x}} \|\mathbf{w}_{x}\|_{1} + \sum_{i} C_{x} \log(1 + e^{-\delta_{i} \mathbf{w}_{x}^{\top} \mathbf{x}_{i}})$$
 (5)

(5) is a standard L1 regularized logistic regression problem and can be efficiently solved using Liblinear package.

Minimization w.r.t \mathbf{w}_z : Keeping δ , \mathbf{w}_x , \mathbf{r}^{\pm} constant and optimizing w.r.t \mathbf{w}_z reduces (3) to:

$$\mathbf{w}_{z}^{*} = \mathbf{Argmin}_{\mathbf{w}_{z}} \mathcal{F}(\{\mathbf{x}_{i}, \mathbf{y}_{i}^{r}, \mathbf{z}_{i}\} | \mathbf{w}_{x}, \mathbf{w}_{z}, \mathbf{r}^{\pm}, \boldsymbol{\delta})$$

$$\equiv \mathbf{w}_{z}^{*} = \mathbf{Argmin}_{\mathbf{w}_{z}} \|\mathbf{w}_{z}\|_{1} + \sum_{i} F(\mathbf{x}_{i}, \mathbf{y}_{i}^{r}, \mathbf{z}_{i} | \mathbf{w}_{x}, \mathbf{w}_{z}, \mathbf{r}^{\pm}, \delta_{i})$$

After ignoring terms independent of \mathbf{w}_z :

$$\equiv \mathbf{w}_z^* = \mathbf{Argmin}_{\mathbf{w}_z} \|\mathbf{w}_z\|_1 + \sum_i C_z \log(1 + e^{-\delta_i \mathbf{w}_z^{\mathsf{T}} \mathbf{z}_i})$$
 (6)

(6) is a standard L1 regularized logistic regression problem and can be efficiently solved using Liblinear package.

3.2 Base Classifiers Optimization and Approximation

We learn compact hyperspherical decision boundaries for each label j independently, according to:

$$B_{j}(\mathbf{x}_{i}) = 1/\left(1 + v_{ij}^{2y_{ij}^{r}-1}\right)$$
where $v_{ij} = e^{\left(\frac{\lambda_{x}}{2} \|\mathbf{x}_{i} - \boldsymbol{\mu}_{j}^{x}\|_{2}^{2} + \frac{\lambda_{z}}{2} \|\mathbf{z}_{i} - \boldsymbol{\mu}_{j}^{z}\|_{2}^{2}\right)}$
(7)

where, μ_j^x, μ_j^z are the centroids of the hyperspherical regressors and λ_x, λ_z are the algorithm's hyperparameters.

For *i*th label, the optimization problem is as follows:

$$\mathbf{Min}_{\mu_x,\mu_z} \prod_{i=1}^{N} B_j(\mathbf{x}_i)$$

$$\equiv \mathbf{Min}_{\mu_x,\mu_z} \prod_{i=1}^{N} 1/(1 + v_{ij}^{2y_{ij}^r - 1})$$

By taking negative logarithm:

$$\equiv \mathbf{Max}_{\mu_x, \mu_z} O = \sum_{i=1}^{N} \log \left(1 + v_{ij}^{2y_{ij}^r - 1} \right)$$
 (8)

Since (8) is continuous and unconstrained, at the optimum the following conditions hold:

$$\Delta \mu_x O = 0$$
 and $\Delta \mu_z O = 0$

where

$$\Delta_{\boldsymbol{\mu}_{j}^{x}}O = \sum_{i:y_{ij}^{r}=1} \frac{\lambda_{x}v_{ij}}{1+v_{ij}}(\boldsymbol{\mu}_{j}^{x}-\mathbf{x}_{i}) - \sum_{i:y_{ij}^{r}=0} \frac{\lambda_{x}}{1+v_{ij}}(\boldsymbol{\mu}_{j}^{x}-\mathbf{x}_{i}) \quad (9)$$

and

$$\Delta_{\mu_{j}^{z}}O = \sum_{i:y_{ij}^{r}=1} \frac{\lambda_{z}v_{ij}}{1 + v_{ij}} (\mu_{j}^{z} - \mathbf{z}_{i}) - \sum_{i:y_{ij}^{r}=0} \frac{\lambda_{z}}{1 + v_{ij}} (\mu_{j}^{z} - \mathbf{z}_{i}) \quad (10)$$

We assume the following:

$$\exists \Delta \in \mathcal{R}, \ \|\mathbf{x}_i - \boldsymbol{\mu}_j^x\|, \|\mathbf{z}_i - \boldsymbol{\mu}_j^z\| \ge \Delta > 0 \ \forall i \in \{1, .., N\}$$
 and
$$\lambda_x, \lambda_z \gg 0$$

Above assumptions imply that:

$$\lambda_{X} \|\mathbf{x}_{i} - \boldsymbol{\mu}_{j}^{x}\|^{2} \ge \lambda_{X} \Delta^{2} \gg 0 \text{ and } \lambda_{z} \|\mathbf{z}_{i} - \boldsymbol{\mu}_{j}^{z}\|^{2} \ge \lambda_{z} \Delta^{2} \gg 0$$

$$\implies v_{ij} \gg 1$$

$$\implies \Delta_{\boldsymbol{\mu}_{j}^{x}} O \approx \sum_{i: y_{ij}^{r} = 1} \lambda_{X} (\boldsymbol{\mu}_{j}^{x} - \mathbf{x}_{i}) = 0$$
(12)

and

$$\Delta_{\boldsymbol{\mu}_{j}^{z}} O \approx \sum_{i: \boldsymbol{y}_{i}^{r} = 1} \lambda_{z} (\boldsymbol{\mu}_{j}^{z} - \mathbf{z}_{i}) = 0$$
 (13)

$$\implies \mu_j^x \approx \frac{\sum_{i=1}^N y_{ij}^r \mathbf{x}_i}{\sum_{i=1}^N y_{ij}^r} \text{ and } \mu_j^z \approx \frac{\sum_{i=1}^N y_{ij}^r \mathbf{z}_i}{\sum_{i=1}^N y_{ij}^r}$$
(14)

The above approximate values of μ_I^x and μ_I^z are not only efficient to calculate, but also provide good prediction performance as observed experimentally.