A NO-REGRET GENERALIZATION OF HIERARCHICAL SOFTMAX TO EXTREME MULTI-LABEL CLASSIFICATION

Marek Wydmuch   Kalina Jasinska   Krzysztof Dembczyński
Institute of Computing Science, Poznań University of Technology, Poland

Mikhail Kuznetsov   Róbert Busa-Fekete
Yahoo! Research, New York, USA

Motivation

- In several machine learning applications, the label space can be enormous, containing even millions of different classes.
- Learning problems of this scale are referred to as **extreme classification**.
- Typical examples include:
  - Image and video annotation for multimedia search,
  - Tagging of text documents for categorization of Wikipedia articles,
  - Recommendation of bid words for online ads,
  - Prediction of the next word in a sentence.
- To tackle extreme classification problems in an efficient way, one can organize labels into a tree as in **hierarchical softmax** (HSM).
- To adapt HSM to **extreme multi-label classification** (XMLC), several very popular tools, such as fastText [1] and Learned Tree [6], apply the **pick-one-label** heuristic, which does not lead to a **consistent** solution.
- **Probabilistic label trees** are a **no-regret generalization** of HSM to XMLC.

XMLC under precision@k

- **Multi-label classification**:

$$\boldsymbol{x} = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d \xrightarrow{\boldsymbol{h}(\boldsymbol{x})} \boldsymbol{y} = (y_1, y_2, \ldots, y_m) \in \{0,1\}^m$$

- **Marginal probability** of a label: $\eta_j(\boldsymbol{x}) = \mathbf{P}(y_j = 1|\boldsymbol{x}) = \sum_{\boldsymbol{y}: y_j=1} \mathbf{P}(\boldsymbol{y}|\boldsymbol{x})$
- **Goal**: find a classifier $\boldsymbol{h}(\boldsymbol{x}) : \mathcal{X} \to \mathcal{R}^m$ **minimizing the expected loss**:

$$L_\ell(\boldsymbol{h}) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}) \sim \mathbf{P}(\boldsymbol{x},\boldsymbol{y})}(\ell(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})))$$

- The **regret** of a classifier $\boldsymbol{h}$ with respect to ℓ

$$\mathrm{reg}_\ell(\boldsymbol{h}) = L_\ell(\boldsymbol{h}) - L_\ell(\boldsymbol{h}^*_\ell) = L_\ell(\boldsymbol{h}) - L^*_\ell$$

- **Precision@k** is defined as:

$$\mathrm{p@}k(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{h}) = \frac{1}{k} \sum_{j \in \hat{\mathcal{Y}}_k} [\![y_j = 1]\!],$$

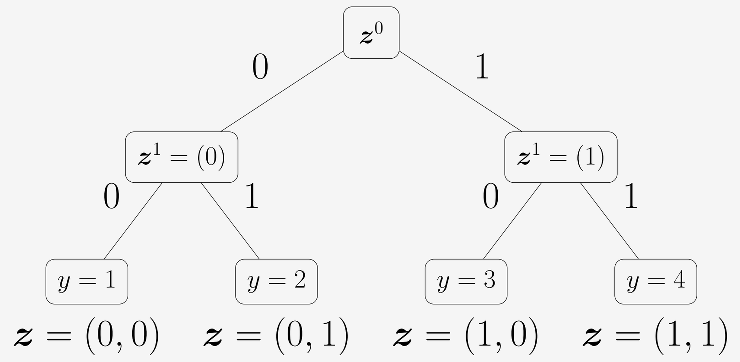
  where $\hat{\mathcal{Y}}_k$ is a set of k labels predicted by $\boldsymbol{h}$ for $\boldsymbol{x}$.
- **Conditional risk** for precision@k:

$$L_{p@k}(\boldsymbol{h}\,|\,\boldsymbol{x}) = \mathbb{E}\left(\ell_{p@k}(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{h})\right) = 1 - \frac{1}{k} \sum_{j \in \hat{\mathcal{Y}}_k} \eta_j(\boldsymbol{x}),$$

- The **optimal strategy**: predict **k labels with the highest $\eta_j(\boldsymbol{x})$**.

Hierarchical softmax

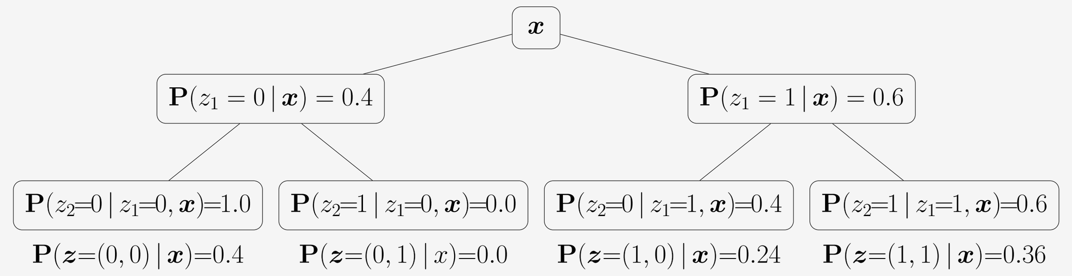
HSM [4] is a **multi-class** classification algorithm based on a **label tree**.



- Each label y **coded** by $\boldsymbol{z} = (z_1, \ldots, z_l) \in \mathcal{C}$
- An internal node identified by a **partial** code $\boldsymbol{z}^j = (z_1, \ldots, z_j)$
- The code does **not** have to be binary.

Factorize the marginal probabilities of labels using a **chain rule**.

$$\eta_j(\boldsymbol{x}) = \mathbf{P}(\boldsymbol{z}|\boldsymbol{x}) = \prod_{i=1}^{l} \mathbf{P}(z_i\,|\,\boldsymbol{z}^{i-1}, \boldsymbol{x})\,.$$



- HSM uses logistic loss and a linear model for estimating $\mathbf{P}(z_i\,|\,\boldsymbol{z}^{i-1}, \boldsymbol{x})$,
- For a multi-class distribution: $\sum_c \mathbf{P}(z_i = c\,|\,\boldsymbol{z}^{i-1}, \boldsymbol{x}) = 1$.

Multi-label data: Pick-one-label heuristic

- Tools like fastText [1] or Learned Trees [6], apply a **pick-one-label heuristic** to HSM to **transform multi-label instances to multi-class** ones.
- **Randomly picking a positive label** transforms the multi-label distribution to a multi-class distribution:

$$\eta'_j(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in \mathcal{Y}} \frac{y_j}{\sum_{j'=1}^{m} y_{j'}} \mathbf{P}(\boldsymbol{y}\,|\,\boldsymbol{x})$$

- **Inconsistent** (non-zero regret) for label-wise logistic loss and precision@k

$\boldsymbol{y}$	$\mathbf{P}(\boldsymbol{y}\,\vert\,\boldsymbol{x})$	True $\eta_j(\boldsymbol{x})$	Estimated $\eta'_j(\boldsymbol{x})$
$(1,0,0)$	0.1	$\eta_1(\boldsymbol{x}) = 0.6$	$\eta'_1(\boldsymbol{x}) = 0.35$
$(1,1,0)$	0.5	$\eta_2(\boldsymbol{x}) = 0.5$	$\eta'_2(\boldsymbol{x}) = 0.25$
$(0,0,1)$	0.4	$\eta_3(\boldsymbol{x}) = 0.4$	$\eta'_3(\boldsymbol{x}) = 0.4$

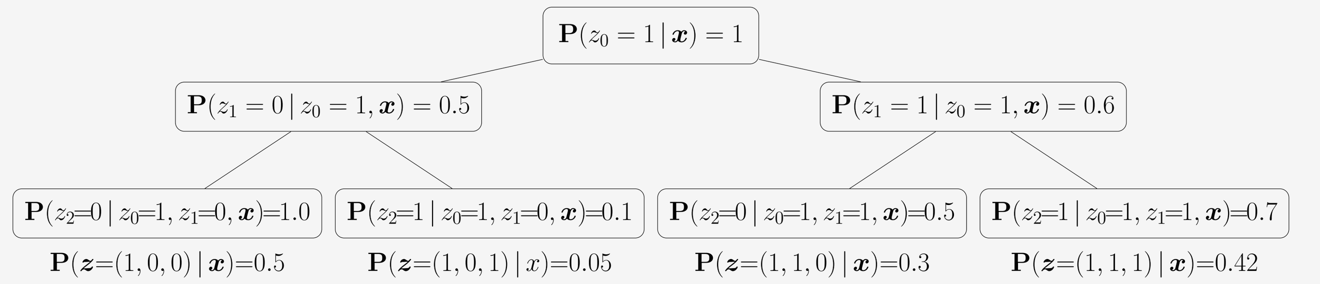
- Given **conditionally independent labels**, $\mathbf{P}(\boldsymbol{y}\,|\,\boldsymbol{x}) = \prod_{j=1}^{m} \mathbf{P}(y_i\,|\,\boldsymbol{x})$, HSM with pick-one-label heuristic is **consistent** for the precision@k loss.

Probabilistic label trees

PLTs [3] are a **no-regret** generalization of HSM to **multi-label** problems.

- Extended code $\boldsymbol{z} = (1, z_1, \ldots, z_l)$.
- **Factorization** of the marginal probability:

$$\eta_j(\boldsymbol{x}) = \mathbf{P}(\boldsymbol{z}\,|\,\boldsymbol{x}) = \prod_{i=0}^{l} \mathbf{P}(z_i\,|\,\boldsymbol{z}^{i-1}, \boldsymbol{x})\,.$$



- Different **normalization** than in HSM:

$$\sum_c \mathbf{P}(z_i = c\,|\,\boldsymbol{z}^{i-1}, \boldsymbol{x}) \geq 1\,.$$

- PLTs applied to a multi-class distribution boil down to HSM.

Regret bounds

- Bound for the **absolute difference** between the **true** and the **estimated marginal probability** for label j

$$|\eta_j(\boldsymbol{x}) - \hat{\eta}_j(\boldsymbol{x})| \leq \sum_{i=0}^{l} \mathbf{P}(\boldsymbol{z}^{i-1}\,|\,\boldsymbol{x}) \sqrt{\frac{2}{\lambda}} \sqrt{\mathrm{reg}_\ell(f_{\boldsymbol{z}^i}\,|\,\boldsymbol{z}^{i-1}, \boldsymbol{x})}\,,$$

- Bound for the **regret** with respect to **precision@k**

$$\mathrm{reg}_{p@k}(\boldsymbol{h}\,|\,\boldsymbol{x}) = \frac{1}{k} \sum_{i \in \mathcal{Y}_k} \eta_i(\boldsymbol{x}) - \frac{1}{k} \sum_{j \in \hat{\mathcal{Y}}_k} \eta_j(\boldsymbol{x}) \leq 2 \max_l |\eta_l(\boldsymbol{x}) - \hat{\eta}_l(\boldsymbol{x})|$$

Implementation (extremeText)

- Based on fastText.
- Tree structures: random, Huffman tree or **build via top-down hierarchical balanced clustering**.
- linear models in the nodes.
- **Online training with features embedding** (hidden, dense representation).
- **L2 regularization** for all parameters of the model (for embedding and internal node classifiers).
- Hidden representation obtained by **weighted average of the feature vectors** of proportion to the **tf-idf scores of features**.
- Depth first search prediction for **fast online prediction**.

Source code: **https://github.com/mwydmuch/extremeText**

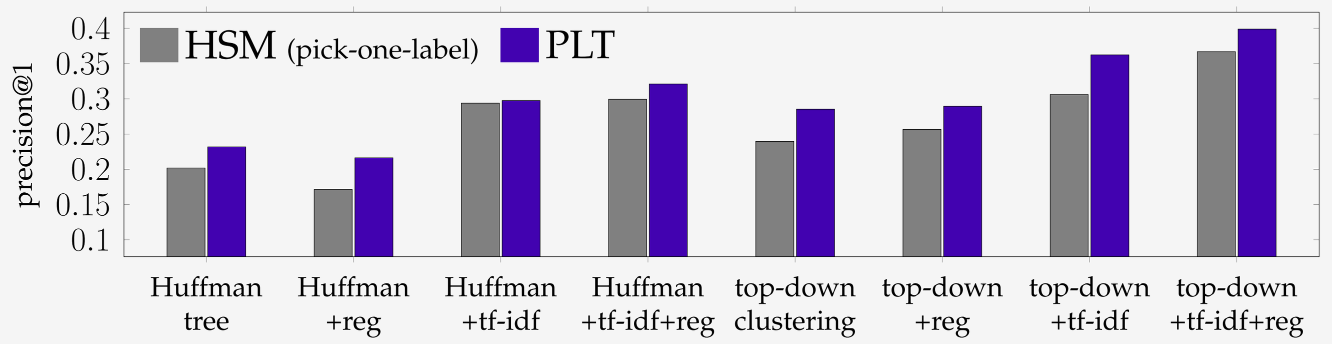
Experimental results

Results on WikiLSHTC, Wiki-500K and Amazon-670K

Dataset	Metrics	fastText	LearnedTrees	**extremeText**	Parabel [5]	XML-CNN [2]
WikiLSHTC	P@1	41.13	50.15	58.73	61.53	†
$N_{train} = 1778351$	P@3	24.09	31.95	39.24	40.07	†
$N_{test} = 587084$	P@5	17.44	23.59	29.26	29.25	†
$d = 617899$	T_{train}	207	212m	550m	**34m**	†
$m = 325056$	T_{test}/N_{test}	1.25ms	4.76ms	**0.81ms**	0.92ms$*$	†
	model size	6.5G	6.5G	**3.3G**	**1.1G$*$**	†
Wiki-500K	P@1	32.73	37.18	64.48	66.12	59.85
$N_{train} = 1813391$	P@3	19.02	21.62	45.84	47.02	39.28
$N_{test} = 783743$	P@5	14.46	16.01	35.46	36.45	29.81
$d = 2381304$	T_{train}	496m	531m	1253m	**168m**	7032m$\star$
$m = 501070$	T_{test}/N_{test}	2.05ms	6.43ms	**1.07ms**	4.68ms$*$	21.06ms$\star$
	model size	11G	11G	**5.5G**	**2.0G$*$**	**3.7G$\star$**
Amazon-670K	P@1	25.47	27.67	39.90	41.59	35.39
$N_{train} = 490449$	P@3	21.47	20.96	35.36	37.18	33.74
$N_{test} = 153025$	P@5	18.61	17.72	32.04	33.85	32.64
$d = 135909$	T_{train}	162m	182m	241m	**8m**	3134m$\star$
$m = 670091$	T_{test}/N_{test}	7.84ms	5.13ms	**1.72ms**	**0.68ms$*$**	16.18ms$\star$
	model size	3.2G	3.2G	**1.5G**	**0.7G$*$**	**1.5G$\star$**

N – number of samples, T – CPU time, m – number of labels, d – number of features, $*$ – result of offline prediction, $\star$ – calculated on GPU, † – cannot be calculated due to lack of a text version of a dataset

Ablation analysis for Amazon-670K



Acknowledgements: Kalina Jasinska is supported by the Polish National Science Center under grant no. 2017/25/N/ST6/00747. The work of Krzysztof Dembczyski was supported by the Polish Ministry of Science and Higher Education under grant no. 09/91/DSPB/0651. Computational experiments have been performed in Poznan Supercomputing and Networking Center.

Kalina Jasinska is supported by: allegro

Marek Wydmuch is supported by: OLX GROUP

[1] P. Bojanowski T. Mikolov A. Joulin, E. Grave. Bag of tricks for efficient text classification. *CoRR*, 2016

[2] Y. Wu Y. Yang J. Liu, W. Chang. Deep learning for extreme multi-label text classification. In *SIGIR*, 2018

[3] K. Jasinska, K. Dembczynski, R. Busa-Fekete, K. Pfannschmidt, T. Klerx, and E. Hüllermeier. Extreme F-measure maximization using sparse probability estimates. In *ICML*, 2016

[4] F Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In *AISTATS*, 2005

[5] Y. Prabhu, A. Kag, S. Harsola, R. Agrawal, and M. Varma. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *The Web Conf*, 2018

[6] D. Sontag Y. Jernite, A. Choromanska. Simultaneous learning of trees and representations for extreme classification and density estimation. In *ICML*, 2017

Neural Information Processing Systems 2018, Montreal, Canada.