r-softmax: Generalized Softmax with Controllable Sparsity Rate

Klaudia Bałazy, Łukasz Struski, Marek Śmieja, and Jacek Tabor

Jagiellonian University
Corresponding author: klaudia.balazy@doctoral.uj.edu.pl

Abstract. Nowadays artificial neural network models achieve remarkable results in many disciplines. Functions mapping the representation provided by the model to the probability distribution are the inseparable aspect of deep learning solutions. Although softmax is a commonly accepted probability mapping function in the machine learning community, it cannot return sparse outputs and always spreads the positive probability to all positions. In this paper, we propose r-softmax, a modification of the softmax, outputting sparse probability distribution with controllable sparsity rate. In contrast to the existing sparse probability mapping functions, we provide an intuitive mechanism for controlling the output sparsity level. We show on several multi-label datasets that r-softmax outperforms other sparse alternatives to softmax and is highly competitive with the original softmax. We also apply r-softmax to the self-attention module of a pre-trained transformer language model and demonstrate that it leads to improved performance when fine-tuning the model on different natural language processing tasks.

Keywords: Sparse probability function · Controlling sparsity level · Softmax alternative.

1 Introduction

Deep learning models excel in various domains, including computer vision, natural language processing (NLP), among others. Mapping the numerical output of a neural network into a probability distribution on a discrete set is crucial for many machine learning models. In classification, it describes the probability over classes, while in the attention mechanism for NLP, it indicates which words in a text are contextually relevant to other words. Softmax [4,11] is the well accepted standard for probability mapping, as it is easily evaluated and differentiated.

Although softmax [4,11] is the most widely applied probability mapping function in machine learning it cannot return sparse outputs. This means it assigns a non-zero probability to every component, making it difficult to identify insignificant elements. As a result, it does not return the number of relevant labels, making it necessary to define a threshold below which the label is considered negative. This threshold often necessitates a hyperparameter selection process, which adds computational overhead, especially in multi-label classification.

ICCS Camera Ready Version 2023
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-36021-3_11

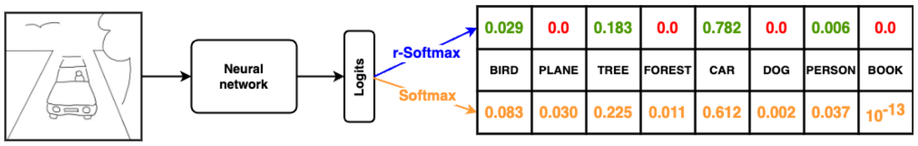


Fig. 1: Comparison of softmax and r-softmax for multi-label classification. Our r-softmax can produce zero probabilities indicating negative classes, making it more intuitive and interpretable than softmax.

Some of the noteworthy alternatives to softmax include the spherical softmax [3], the multinomial probit [1], softmax approximations [2] and Gumbel-Softmax [8]. As this paper introduces a novel sparse alternative to softmax, we focus on existing sparse probability mapping functions. Sparsemax [12] projects an input vector onto the probability simplex, producing sparse outputs. Unfortunately, it generally performs worse than softmax. In [9], a general family of probability mapping functions, including softmax and sparsemax, was defined, and a strategy for designing convex loss functions was proposed, including an alternative loss for sparsemax that improved its experimental performance.

We introduce r-softmax, a sparse alternative to the softmax function, that solves the issue of non-zero probabilities and provides intuitive control of the sparsity rate. Users can specify the sparsity rate r, representing the desired fraction of zero values, or train the model to determine its value using gradient descent. This eliminates the need for an additional mechanism like a threshold to identify positive labels in multi-label classification, as shown in Figure 1.

We evaluate r-softmax as a function determining probabilities of classes in a multi-label classification problem and as a function determining the significance probability of elements in the attention mechanism. In multi-label classification, we benchmark r-softmax on real and synthetic datasets and find it outperforms other sparse alternatives to softmax, like sparsemax [12] and sparsehourglass [9], and competes with the original softmax using an optimal threshold for positive labels. For the attention mechanism, we replace the pre-trained transformer language model's softmax with r-softmax and show that our modification improves the fine-tuned model's performance on various NLP tasks.

2 Sparse version of softmax

We introduce r-softmax, a sparse probability mapping with controllable sparsity.

Problem motivation Probability mapping functions transform a real-valued response $x = (x_1, \dots, x_n) \in \mathbb{R}^n$ of the neural network into a probability vector $p = (p_1, \dots, p_n)$, where $p_i \geq 0$ and $\sum_{i=1}^{n} p_i = 1$. The most commonly used function to parameterize this probability is softmax:

$$\text{softmax}(x) = \left(\frac{\exp(x_1)}{\sum_{i=1}^{n} \exp(x_i)}, \dots, \frac{\exp(x_n)}{\sum_{i=1}^{n} \exp(x_i)} \right).$$

The limitation of softmax is its inability to return sparse outputs with zero probabilities. Sparse outputs are very useful for example in (i) multi-label classification, where zero-probabilities indicate absence of labels, or (ii) self-attention layers, where they allow to ignore irrelevant keys.

The weighted softmax With the motivation of constructing a probability mapping function capable of producing sparse output vectors, we introduce the weighted softmax as a generalization of the traditional softmax. By appropriately parameterizing its weights, the weighted softmax can reduce to a typical softmax or binary one-hot vector form where one coordinate contains a value of 1 and the rest are set to 0. It can also enable sparse probability mapping functions that fall between the two extremes.

Let $x = (x_1, \dots, x_n) \in \mathbb{R}^n$ be a point, associated with vector of weights $w = (w_1, \dots, w_n) \in \mathbb{R}_+^n$, where $\sum_{i=1}^n w_i > 0$. We define a weighted softmax by the following formula:

$$\text{softmax}(x, w) = \left(\frac{w_1 \exp(x_1)}{\sum_{i=1}^{n} w_i \exp(x_i)}, \dots, \frac{w_n \exp(x_n)}{\sum_{i=1}^{n} w_i \exp(x_i)} \right).$$

The weighted softmax is a proper probability distribution (all components are non-negative and sum to 1) that reduces to classical softmax for a constant weight vector. Unlike softmax, it can return zero probability at some coordinates by setting the corresponding weight to zero, and produce one-hot vectors by setting exactly one non-zero weight.

To achieve a smooth transition between softmax and binary one-hot vectors, we construct t-softmax, in which all weights depend on a single parameter $t > 0$:

$$\text{t-softmax}(x, t) = \text{softmax}(x, w_t), \tag{1}$$

where $w_t = (w_t^1, \dots, w_t^n)$ and $w_t^i = \text{ReLU}(x_i + t - \max(x))$. All weights w_i are nonnegative with at least one positive weight, satisfying the definition of weighted softmax. We can observe that w_i equals zero when the absolute difference between x_i and the maximum value $\max(x)$ is greater than or equal to t.

The following examines how t-softmax changes with varying values of t:

Theorem 1. *Let $x \in \mathbb{R}^n$ be a data point and let $t \in (0, \infty)$. Then*

- *the limit of* t-softmax(x,t) *is* softmax(x) *as t approaches infinity,*
- *if x reaches unique max at index k, then*

$$\text{t-softmax}(x, t) = \text{onehot}(\arg\max_i(x)), \tag{2}$$

for $t \in (0, x_k - \max_{i \neq k}(x)]$, where onehot$(i) \in \mathbb{R}^n$ *is a vector consisting of zeros everywhere except k-th position where 1 is located.*

Proof. The first property is a consequence of $\text{t-softmax}(x, t) = \text{softmax}(x, \frac{w_t}{t})$, and if t approaches infinity then $\frac{w_t}{t}$ goes to 1, leading to $\text{softmax}(x, 1) = \text{softmax}(x)$. The last property follows directly from the definition of t-softmax.

Controlling the number of non-zero values using r-softmax Instead of learning the optimal value of t as discussed above, there are situations in which we would like to have the ability to explicitly decide how many components returned by t-softmax should be zero. For this purpose, we introduce a parameter $r \in [0, 1]$ that we call a *sparsity rate*. The sparsity rate r represents the fraction of zero components we would like to obtain in the output probability distribution.

Recall that $w_i^t = 0$ for $i = 1, \ldots, n$ if $|x_i - \max(x)| \geq t$, as defined in Equation (1). To control the number of non-zero weights, we can inspect the range $[\min(x), \max(x)]$ and select t such that $x_i < t < x_j$, where x_i and x_j are two distinct elements in $x_1, \ldots, x_n$, in increasing order. This will zero out the i-th component while keeping the j-th component non-zero. We can use the quantile of the set of x's coordinates $x_1, \ldots, x_n$ to implement this rule. The q-quantile $\text{quantile}(x, q)$ outputs the value v in $[\min(x), \max(x)]$ such that the probability of $x_i : x_i \leq v$ equals q. If the quantile lies between x_i and x_j with indices i and j in the sorted order, we use linear interpolation to compute the result as $x_i + \alpha \cdot (x_j - x_i)$, where α is the fractional part of the computed quantile index. Setting $q = 0$ or $q = 1$ in $\text{quantile}(x, q)$ will return the lowest or highest value of x, respectively.

We define r-softmax as a probability mapping function with a fixed sparsity rate $r \in [0, 1]$ as shown in Equation (3), where $t_r = -\text{quantile}(x, r) + \max(x)$.

$$\text{r-softmax}\,(x, r) = \text{t-softmax}(x, t_r). \tag{3}$$

The parameterization of t_r ensures that a fraction of r components will be zero. When $\text{r-softmax}(x, r)$ is applied to $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ and $r = \frac{k}{n}$, for $k \leq n$, the output will be a probability distribution with k zero coordinates. The r-softmax reduces model complexity by eliminating less probable components. Our experiments demonstrate the benefits of this approach, particularly in the self-attention mechanism for NLP tasks.

3 Experiments

We benchmarked r-softmax against basic softmax and other sparse probability mapping functions (such as sparsemax and sparsehouglass) in multi-label classification and self-attention blocks of pre-trained language models. Our results demonstrate that r-softmax outperforms the other functions in most cases. [1].

[1] Code with r-softmax is available at `https://github.com/gmum/rsoftmax`

3.1 Alternative to softmax in multi-label classification

Multi-label classification is a crucial problem in various domains, including image classification, where a single class description may not suffice due to multiple object classes in an image. In these models, the final element is a function that maps network output to a probability vector representing class membership probabilities. Softmax is often used, but other functions, such as those introducing sparse probability distributions, have also been investigated [9,12].

R-softmax for multi-label classification To use r-softmax in multi-label classification, we need a proper loss function. We cannot directly apply cross-entropy loss as r-softmax can return zeros, which makes the logarithm undefined. We follow the approach used in [9] to overcome this. Given input x, logits z and a probability distribution $\eta = y/|y|_1$ over labels y, we define our loss function:

$$\mathcal{L}(z, y) = \|y \cdot (\text{r-softmax}(z, r) - \eta)\|_2^2 + \sum_{y_i=1, y_j=0} \max\left(0, \eta_i - (z_i - z_j)\right), \quad (4)$$

where y_i is i-th coordinate of the vector y (similarly for z and η). The first term approximates the positive label probability given by r-softmax$(z, r)_i$, while the second term pushes negative label logits away from positive ones by η_i.

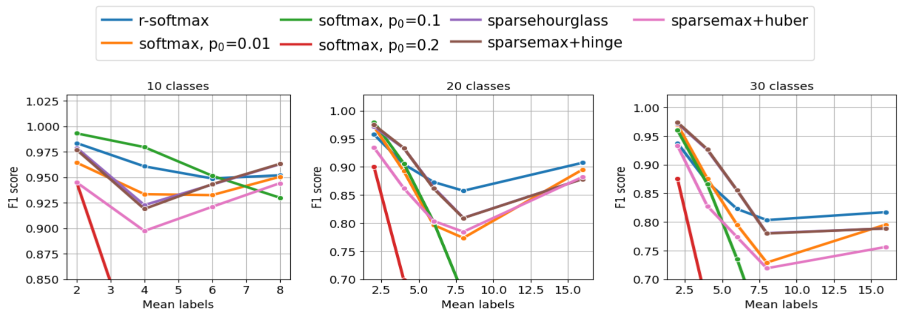
Datasets We tested different probability mapping functions on multi-label classification using synthetic data (similarly to [9]), with various possible output classes, average number of labels per sample and document length. We also evaluated considered functions on two real datasets (VOC 2007 [6] and COCO [10]).

Experimental setting We compare our method with sparsemax, sparsehourglass, and softmax. To handle the issue of obtaining zero values with softmax, we use various thresholds p_0 to consider a class as negative. For softmax we use cross-entropy loss, for sparsehourglass we use function proposed by [9] and for sparsemax we test functions: sparsemax+huber [12] and sparsemax+hinge [9].

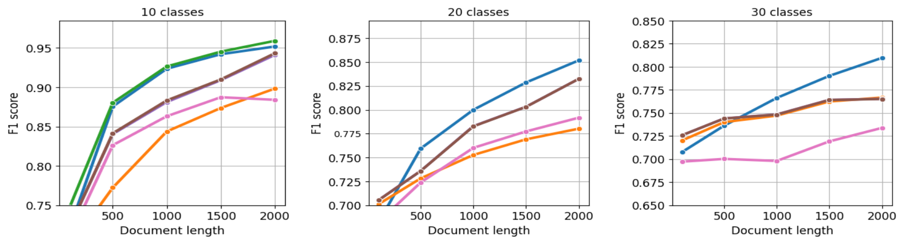
We use a two-layer neural network for synthetic datasets and pre-trained ResNet models [7] with an additional linear layer for classification for real datasets. The models are trained with different learning rates. Our r-softmax is parameterized by the sparsity rate r that is found using an additional classification layer. The multi-label classification cost function for r-softmax includes cross-entropy loss to evaluate the correctness of the number of labels.

We report the best validation score after the models reach stability. The F1 score is used as quality metric for multi-label classification models, as it is based on the returned classes rather than target scores (e.g., mean average precision).

Results on synthetic datasets Figure 2 compares using r-softmax and other functions for multi-label classification on the synthetic data validation set. In Figure 2a, we show that r-softmax consistently outperforms other functions, particularly when the dataset has a large number of possible output classes. In Figure 2b,



(a) Varying average number of positive labels.



(b) Varying document length.

Fig. 2: Different probability mapping functions for multi-label classification on synthetic datasets with varying output class numbers. For large output class numbers, r-softmax seems to be the most beneficial choice.

we demonstrate the impact of the average document length on model performance and show that r-softmax achieves the best results for most configurations, especially for a larger number of possible output classes. Our method, r-softmax, is the preferred choice as it provides the most benefits in the investigated scenarios.

Results on real datasets We also evaluate r-softmax on real multi-label datasets VOC and COCO, see Table 1. Our r-softmax outperforms other sparse softmax alternatives and is competitive with the original softmax.

3.2 Alternative to softmax in the self-attention block in transformer-based model

Transformer-based [13] models are widely used in NLP and rely on attention mechanisms to identify important information. Self-attention modules in each layer apply softmax to generate weights for all tokens, but this assigns non-zero weight to even insignificant tokens. We propose replacing softmax with r-softmax function to obtain sparse probability distributions that allow the model to ignore irrelevant tokens, which we show to be beneficial in the following section.

Table 1: Performance of different probability mappings for multi-label classification on VOC and COCO datasets. Our r-softmax outperforms other sparse mapping functions and is competitive with softmax (with threshold selection).

Experimental setup	VOC (F1)	COCO (F1)
Softmax (p_0=0.05)	75.05	71.38
Softmax (p_0=0.10)	78.87	72.29
Softmax (p_0=0.15)	**79.43**	69.22
Softmax (p_0=0.20)	79.07	64.88
Softmax (p_0=0.30)	75.88	54.76
Sparsemax+huber	66.84	52.30
Sparsemax+hinge	71.91	65.67
Sparsehourglass	71.35	64.85
r-softmax	77.90	**72.56**

Table 2: Comparing different probability mapping functions in pretrained BERT$_{BASE}$ self-attention blocks for finetuning on GLUE benchmarks. Our r-softmax with a specific sparsity level outperforms other approaches.

Experiment setup	MRPC (Acc)	RTE (Acc)	SST-2 (Acc)	QNLI (Acc)	QQP (Acc)
Softmax	84.56	68.95	92.32	**91.76**	91.12
Sparsemax	68.38	52.71	79.82	55.57	77.18
Sparsehourglass	68.38	52.71	79.24	70.99	76.04
r-softmax	**85.54**	**71.84**	**92.89**	91.73	**91.13**

Experimental setting We experiment with BERT$_{BASE}$ [5] language model and focus on probability mapping function in each self-attention block during fine-tuning. We compare the performance of baseline softmax with sparsemax, sparsehourglass, and r-softmax replacements on GLUE benchmark classification tasks [14]. The final best validation score is reported after fine-tuning for 5 epochs for MRPC and 3 epochs for others, with different learning rates. We linearly increase hyperparameter r for r-softmax from 0 to desired sparsity $r \in 0.05, 0.1, 0.15, 0.2, 0.5$ during training.

Results The results for GLUE tasks obtained by the best run in the grid search are summarized in Table 2. Using r-softmax instead of softmax generally improves the performance of the fine-tuned transformer-based model. Sparsemax and sparsehourglass do not perform very well in this application. We found that introducing a small sparsity rate produced the best results for r-softmax. The optimal sparsity rates for the QQP, MRPC, QNLI, RTE, and SST-2 tasks were $r = 0.1, 0.15, 0.15, 0.2, 0.2$ respectively. These results suggest that eliminating irrelevant elements is beneficial, but excluding too many can cause the model to lose important context or hinder gradient flow during learning.

4 Conclusions

We proposed r-softmax, a generalization of softmax that produces sparse probability distributions with a controllable sparsity rate. We applied r-softmax to multi-label classification and self-attention tasks and showed that it outperforms or is highly competitive with baseline models.

Acknowledgements The work of Klaudia Bałazy[2] and Łukasz Struski was supported by the National Centre of Science (Poland) Grant No. 2020/39/D/ST6/01332. The research of Jacek Tabor was carried out within the research project "Bio-inspired artificial neural network" (grant no. POIR.04.04.00-00-14DE/18-00) within the Team-Net program of the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund. The work of Marek Śmieja was supported by the National Centre of Science (Poland) Grant No. 2022/45/B/ST6/01117.

References

1. Albert, J.H., Chib, S.: Bayesian analysis of binary and polychotomous response data. Journal of the American statistical Association **88**(422), 669–679 (1993)
2. Bouchard, G.: Efficient bounds for the softmax function and applications to approximate inference in hybrid models. In: NIPS Workshop For Approximate Bayesian Inference in Continuous/Hybrid Systems (2007)
3. de Brébisson, A., Vincent, P.: An exploration of softmax alternatives belonging to the spherical loss family. arXiv preprint arXiv:1511.05042 (2015)
4. Bridle, J.S.: Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In: Neurocomputing, pp. 227–236. Springer (1990)
5. Devlin, J., et al.: BERT: Pre-training of deep bidirectional transformers for language understanding. vol. 1, pp. 4171–4186. ACL (2019)
6. Everingham, M., et al.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results (2007)
7. He, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
8. Jang, E., et al.: Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144 (2016)
9. Laha, A., et al.: On controllable sparse alternatives to softmax. NeurIPS (2018)
10. Lin, T.Y., et al.: Microsoft coco: Common objects in context. In: ECCV. pp. 740–755. Springer International Publishing (2014)
11. Luce, R.D.: Individual choice behavior: A theoretical analysis. Courier Corporation (2012)
12. Martins, A., et al.: From softmax to sparsemax: A sparse model of attention and multi-label classification. ICML'16, vol. 48, p. 1614–1623. JMLR.org (2016)
13. Vaswani, A., et al.s: Attention is all you need. In: NeurIPS. pp. 5998–6008 (2017)
14. Wang, A., et al.: GLUE: A multi-task benchmark and analysis platform for natural language understanding (2018)

[2] Doctoral School of Exact and Natural Sciences at the Jagiellonian University.