Review : SoftNDCG

Taeyoung Lee

June 2022

Abstract

The first obstacle that you might encounter in Learning-To-Rank is that you cannot directly apply gradient-based learning methods because sorting the documents makes everything discrete. In this paper, the authors introduce SoftNDCG to avoid this problem. They interpret a score not as a deterministic value but as a smooth distribution which enables gradient methods. Then they produce a rank distribution by comparing the documents pairwise. Using this rank distribution, they define SoftNDCG by replacing the discount by the 'expected' discounts.

1 NDCG

Before we dive into the paper, we start with the notion of NDCG.

Definition 1.1 (Cumulative Gain)**.** Cumulative Gain(CG) is the sum of the graded relevance values of all results in a search result list.i.e.,

$$CG_p = \sum_{i=1}^{p} rel_i$$

Definition 1.2 (Standard and industry discounted CG)**.**

1. Traditional DCG

$$\text{Standard } DCG_p = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i+1)} \tag{1.3}$$

2. Alternative formulation(used in industries and DS competition like Kaggle)

$$\text{Industry } DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{1.4}$$

Previously there was no theoretically sound justification for using a logarithmic reduction factor other than the fact that it produces a smooth reduction.

Definition 1.5 (Normalized DCG)**.** For a query, the *normalized discounted cumulative gain*(nDCG) is computed as

$$nDCG_p = \frac{DCG_p}{IDCG_p} \tag{1.6}$$

where $IDCG_p$ is ideal DCG_p, defined as,

$$IDCG_p = \sum_{i=1}^{|REL_p|} \frac{rel_i}{\log_2(i+1)} \tag{1.7}$$

and REL_p represents the list of relevant documents ordered by its relevance in the corpus up to position p.

Remark 1.8.

1. In a perfect ranking system, the DCG_p will be the same as the $IDCG_p$ producing an $nDCG$ of 1.0.

2 Smoothing Scores

The key idea is treating scores as smoothed score distribution. A simple approach to do this, is to give every score the equal variance Gaussian distributions. Hence, the score $s_j = f(w, x_j)$ becomes the mean of the Gaussian distribution, with a shared smoothing variance σ_s:

$$p(s_j) = \mathcal{N}(s_j|\bar{s_j}, \sigma_s^2) = \mathcal{N}(s_j|f(w, x_j), \sigma_s^2) \tag{2.1}$$

Remark 2.2. Note that the following equation holds :

$$\mathcal{N}(x|\mu, \sigma^2) = (2\pi\sigma^2)^{-0.5}(-(x-\mu)^2/2\sigma^2)$$

2.1 From Score to Rank Distribution

When we have deterministic scores, we have deterministic rank distribution. The rank distribution may be simulated by the following exact generative process:

1. Sample a vector of N scores, one from each distribution,
2. *sort* the score samples
3. accumulate histograms of the resulting ranks for each documents.

For a given doc_j, consider the probability that another doc_i will rank above doc_j. Denoting S_j as a draw from $p(s_j)$, we require the probability that $S_i > S_j$, or equivalently $Pr(S_i - S_j > 0)$.

The probability π_{ij} that document i beats documnet j,is

$$\pi_{ij} := Pr(S_i - S_j > 0) = \int_0^\infty \mathcal{N}(s|\bar{s}_i - \bar{s}_j, 2\sigma_s^2)ds$$
$$\int_0^\infty = (2\pi\sigma^2)^{-0.5}\exp[-(s-(f(w,x_i) - f(w,x_j)))^2/2\sigma^2]ds$$

To generate ranks from this quantity π_{ij}, we simply integrate them and get the *expected rank* as follows:

$$E[r_j] = \sum_{i=1, i\neq j}^{N} \pi_{ij} \tag{2.3}$$

The actual *distribution* of the rank r_j of a document j under the pairwise contest approximation is obtained by considering th erank r_j as a Binomial-like random variable. But this random variable is bit more complicated than the Binomial. The authors have defined it as the *Rank-Binomial distribution.* If we define the initial rank distribution for document j as $p_j^{(1)}(r)$, where we have just the document j, then the rank can only have value 0 (the best rank) with probability 1:

$$p_j^{(1)}(r) = \delta_{r0}$$

where δ_{r0} is the Kronecker-delta.

Suppose we got $N-1$ *other* documents that contribute to the rank distribution that we will index with $i = 2, \ldots, N$. Each time we add a new document i, the event space of the rank distribution gets one larger, taking the r variable to a maximum of $N-1$ on the last iteration.

The new distribution over the ranks is as follows:

$$p_j^{(i)}(r) = p_j^{(i-1)}(r-1)\pi_{ij} + p_j^{(i-1)}(r)(1-\pi_{ij}). \tag{2.4}$$

The left summand of the right hand side of the equation is the probability that the document get shifted from the best rank to the rank r and the right summand is exactly the same thing from the worst rank.

2.2 SoftNDCG

This section shows how we can use rank distribution to smooth traditional IR metrics. The expression for deterministic NDCG was given in (2) as $G = G_{max}^{-1} \sum_{r=0}^{N-1} g(r)D(r)$. We set out to compute the *expected* NDCG given the rank distributions described above. Rewriting NDCG as a sum over document indices rather than document ranks we get:

$$G \quad = \quad G_{max}^{-1} \sum_{j=1}^{N} g(j)D(r_j)$$

Now we replace the deterministic discount $D(r)$ with the expected discount. Thus we define softNDCG $\mathcal{G}$ as

$$\mathcal{G} = \quad G_{max}^{-1} \sum_{j=1}^{N} g(j)E[D(r_j)] \tag{2.5}$$

Using the fact that

$$E[D(r_j)] = \sum_{r=0}^{N-1} D(r)p_j(r) \tag{2.6}$$

the softNDCG can be written as :

$$\mathcal{G} = \quad G_{max}^{-1} \sum_{j=1}^{N} g(j) \sum_{r=0}^{N-1} D(r)p_j(r) \tag{2.7}$$

2.3 Gradient of softNDCG

Now we have derived an expression for a SoftNDCG, we now derive its gradient with respect to the weight vector. The derivative with respect to the weight vector with K element is:

$$\frac{\partial \mathcal{G}}{\partial w} = \begin{bmatrix} \frac{\partial s_1}{\partial w_1} & \dots & \frac{\partial s_1}{\partial w_1} \\ \dots & \dots & \dots \\ \frac{\partial s_1}{\partial w_K} & \dots & \frac{\partial s_N}{\partial w_K} \end{bmatrix} \begin{bmatrix} \frac{\partial \mathcal{G}}{\partial \bar{s_1}} \\ \dots \\ \frac{\partial \mathcal{G}}{\partial \bar{s_N}} \end{bmatrix} \tag{2.8}$$

It somehow looks like the chain rule..anyway..
We compute each derivatives :

$$\frac{\partial \mathcal{G}}{\partial \bar{s}_m} = G_{\max}^{-1} \sum_{j=1}^{N} g(j) \sum_{r=0}^{N-1} D(r) \frac{\partial p_j(r)}{\partial \bar{s}_m}. \tag{2.9}$$

Hence we need a parallel recursive computation to obtain the required derivative of $p_j(r)$. The authors write things in pretty way by denoting $\psi_{m,j}^{(i)}(r) = \frac{\partial p_j^{(i)}(r)}{\partial \bar{s}_m}$. But we do it in less elegant but clear way. Using the equation 2.4, we have

$$\begin{aligned} \frac{\partial p_j^{(i)}(r)}{\partial \bar{s}_m} &= \frac{\partial p_j^{(i-1)}(r-1)}{\partial \bar{s}_m} \pi_{ij} + p_j^{(i-1)}(r-1) \frac{\partial \pi_{ij}}{\partial \bar{s}_m} + \frac{\partial p_j^{(i-1)}(r)}{\partial \bar{s}_m}(1-\pi_{ij}) - p_j^{(i-1)}(r)\frac{\partial \pi_{ij}}{\partial \bar{s}_m} \\ &= \frac{\partial p_j^{(i-1)}(r-1)}{\partial \bar{s}_m} \pi_{ij} + \frac{\partial p_j^{(i-1)}(r)}{\partial \bar{s}_m}(1-\pi_{ij}) + (p_j^{(i-1)}(r-1) - p_j^{(i-1)}(r))\frac{\partial \pi_{ij}}{\partial \bar{s}_m}. \end{aligned} \tag{2.10}$$

Using the fact that

$$\frac{\partial}{\partial \mu} \int_0^\infty \mathcal{N}(x|\mu, \sigma^2) dx = \mathcal{N}(0|\mu, \sigma^2) \tag{2.11}$$

we obtain

$$\frac{\partial \pi_{ij}}{\partial \bar{s}_m} = \begin{cases} \mathcal{N}(0|\bar{s}_m - \bar{s}_j, 2\sigma^2) & m = i, m \neq j \\ \\ -\mathcal{N}(0|\bar{s}_i - \bar{s}_m, 2\sigma^2) & m \neq i, m = j \\ \\ 0 & m \neq i, m = j \end{cases} \tag{2.12}$$

We define the result of this computation as the N-vector over ranks:

$$\frac{\partial p_j(r)}{\partial \bar{s}_m} \equiv \Psi_{m,j} = [\psi^N_{m,j}(0), \ldots, \psi^N_{m,j}(N-1)] \tag{2.13}$$

Using the matrix notation we substitue the result in 2.9:

$$\frac{\partial \mathcal{G}}{\partial \bar{s}_m} = \frac{1}{G_{\max}} [g_1, \ldots, _N] \begin{bmatrix} \Psi_{m,0} \\ \ldots \\ \Psi_{m,N-1} \end{bmatrix} \begin{bmatrix} d_0 \\ \ldots \\ d_{N-1} \end{bmatrix} \tag{2.14}$$

We now define the gain vector $\mathbf{g}$, the discount vector $\mathbf{d}$ and the $N \times N$ square matrix $\boldsymbol{\Psi}_m$ whose rows are the rank distribution derivatives implied above:

$$\frac{\partial \mathcal{G}}{\partial \bar{s}_m} = \frac{1}{G_{\max}} \mathbf{g}^T \Psi_m \mathbf{d}. \tag{2.15}$$

Finally we have the gradient of the SoftNDCG!

$$\nabla \mathcal{G} = \begin{bmatrix} \frac{\partial \mathcal{G}}{\partial \bar{s}_1} & \ldots & \frac{\partial \mathcal{G}}{\partial \bar{s}_N} \end{bmatrix} \tag{2.16}$$

References

[1] Mike Taylor, John Guiver, Stephen Robertson, and Tom Minka. Softrank: Optimising non-smooth rank metrics. In *WSDM 2008*, February 2008.