Using Gradient Descent to Optimize Language Modeling Smoothing Parameters

Donald Metzler
metzler@cs.umass.edu
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003

Categories and Subject Descriptors: H.3.3 Information Storage and Retrieval: Information Search and Retrieval

General Terms: Theory, Experimentation

Keywords: Language modeling, parameter estimation, gradient descent.

1. INTRODUCTION

Most information retrieval models have some set of tunable parameters. It is often the case that the parameters of such models are either set to default values or tuned in a supervised or unsupervised manner. Historically, many retrieval models have been tuned to maximize some underlying retrieval metric, such as mean average precision. However, as the number of parameters increases, the direct maximization techniques become computationally expensive. For this reason, there has been a growing interest in both the information retrieval and machine learning communities to develop parameter estimation techniques that scale well.

Since most information retrieval metrics are non-smooth with respect to model parameters, the machine learning techniques have focused on maximizing or minimizing some surrogate function that attempts to mimic or be highly correlated with retrieval metrics. Standard optimization techniques can then easily be applied to the surrogate function in order to estimate approximate parameters.

There have been, however, few studies from an information retrieval perspective into how well such surrogate functions compare to the direct search approach. Recent work has investigated how the effectiveness of using BM25F parameters estimated by minimizing the RankNet cost function correlates with the effectiveness of an approach that directly maximizes NDCG [2]. The results showed that RankNet acts as a good surrogate and produces reasonable effectiveness when compared to a more computationally expensive direct search technique.

Following up on this work, we wish to explore how to use the RankNet cost function to optimize language modeling smoothing parameters. In addition, we explore how well the parameters learned using RankNet compare to those found by a direct search technique for various metrics that rely on *binary* relevance judgments, such as mean average precision, binary preference, and precision at 10.

2. PARAMETER ESTIMATION

In this section we describe, in general terms, how to estimate parameters using direct search and the RankNet cost function.

2.1 Direct Optimization

Given a model with one or more parameters, a set of relevance judgments, and a retrieval metric, it is straightforward to implement various algorithms that directly attempt to find the parameter setting that maximizes or minimizes the metric. One of the most naïve approaches is to search for the global optimum via a brute force search over the entire parameter space. Depending on whether the parameter space is bounded and the number of parameters, this may be computationally infeasible.

Techniques such as coordinate ascent (and its variants) may be more efficient, but are not guaranteed to find a global optimum. These techniques are forced to estimate derivatives via finite differences or perform line searches, since analytical derivates cannot be computed in general.

2.2 RankNet Optimization

Rather than directly search within the original, non-smooth retrieval metric space, we may instead optimize a surrogate function. In our work, we choose the RankNet cost function as our surrogate [1, 2]. The RankNet cost function is defined over pairwise preferences. That is, for a given query Q, we define the set $\mathcal{R}_Q$, such that $(d_1, d_2) \in \mathcal{R}_Q$ implies that document d_1 should be ranked higher than document d_2. Given a set of binary relevance judgments, it is easy to construct $\mathcal{R}_Q$ by taking the cross product of relevant and non-relevant documents. There are other ways to construct $\mathcal{R}_Q$, as well, although we found that we achieved the best results by using all of the pairwise preferences we had available to us.

Once we have defined our pairwise preferences, the RankNet cost is computed according to:

$$C(\mathcal{Q}, \mathcal{R}) = \sum_{Q \in \mathcal{Q}} \sum_{(d_1,d_2) \in \mathcal{R}_Q} \log(1 + \exp(Y))$$

where $\mathcal{Q}$ is the set of queries, $Y = g(Q; d_2) - g(Q; d_1)$, and $g(Q; d)$ is the score of document d with respect to query Q using the current parameter setting.

In order to minimize C, we perform coordinate descent, which requires us to compute gradients with respect to our model parameters. Given some model parameter α, we apply the chain rule in order to compute the gradient of C with respect to α as:

$$\frac{\delta C}{\delta \alpha} = \sum_{Q \in \mathcal{Q}} \sum_{(d_1,d_2) \in \mathcal{R}_Q} \frac{\delta C}{\delta Y} \frac{\delta Y}{\delta \alpha}$$

Using the RankNet cost function, it is easy to see that $\frac{\delta C}{\delta Y}$ is computed as:

$$\frac{\delta C}{\delta Y} = \frac{\exp\left[g(Q;d_2) - g(Q;d_1)\right]}{1 + \exp\left[g(Q;d_2) - g(Q;d_1)\right]}$$

Therefore, the final piece that needs to be computed depends on the underlying scoring function. In order to analytically compute the partial, we must be able to differentiate our scoring function with respect to each parameter. It is then straightforward to compute $\frac{\delta Y}{\delta \alpha}$ according to:

$$\frac{\delta Y}{\delta \alpha} = \frac{\delta g(Q;d_2)}{\delta \alpha} - \frac{\delta g(Q;d_1)}{\delta \alpha}$$

Note that the RankNet cost function does not depend on any specific retrieval metric. Therefore, RankNet will always learn the same parameters for a given set of pairwise preferences, regardless of the metric we wish to optimize for.

3. LANGUAGE MODELING RANKING

We wish to estimate the parameters for the language modeling query likelihood ranking function using two-stage smoothing [3]. The two-stage smoothing estimate, which has two parameters, generalizes both Dirichlet and Jelinek-Mercer smoothing, and therefore makes for a good general purpose language modeling ranking function. A document D is scored against query Q under this model as follows:

$$g(Q;D) = \sum_{w \in Q} \log\left((1-\lambda) \frac{tf_{w,D} + \mu P(w|C)}{\mu + |D|} + \lambda P(w|C) \right)$$

where λ and μ are the smoothing parameters that we wish to estimate. Note that we rank according to the log query likelihood in order to simplify the mathematical derivations.

The partial derivates of the scoring function, with respect to λ and μ, are computed as follows:

$$\frac{\delta g(Q;D)}{\delta \lambda} = \sum_{w \in Q} \frac{P(w|C) - \frac{tf_{w,D} + \mu P(w|C)}{\mu + |D|}}{(1-\lambda)\frac{tf_{w,D} + \mu P(w|C)}{\mu + |D|} + \lambda P(w|C)}$$

$$\frac{\delta g(Q;D)}{\delta \mu} = \sum_{w \in Q} \frac{(1-\lambda)\frac{|D|}{(\mu + |D|)^2}\left(P(w|C) - \frac{tf_{w,D}}{|D|}\right)}{(1-\lambda)\frac{tf_{w,D} + \mu P(w|C)}{\mu + |D|} + \lambda P(w|C)}$$

4. EVALUATION

In this section we evaluate how the effectivness of language modeling parameters learned by minimizing the RankNet cost function compare to the effectiveness of parameters learned by using direct search. Experiments are carried out on four standard TREC *ad hoc* retrieval test collections, including three newswire data sets (AP '88'-'90, WSJ '87-'92, and the 2004 Robust Track data set), and a large web data set (wt10g). For training purposes, each set of topics is split into a training and test set.

The results are given in Table 1. As the table shows, the outcomes for MAP and BPREF are the same, with direct search being significantly better than RankNet on the WSJ and ROBUST data sets. In addition, RankNet is significantly worse than optimal for every data set except AP, whereas direct search only differs significantly from optimal for WSJ and ROBUST. The results for P@10 show that RankNet is only significantly worse than direct search for WSJ. Indeed, RankNet appears to be more stable for P@10 than MAP and BPREF.

Metric	Estimate	ap	wsj	robust	wt10g
MAP	Direct	.2072	**.3255**	**.2920**	.1930
	RankNet	.2081	.2987	.2756	.1922
	Optimal	.2088	***.3290***	***.2931***	**.2003**
BPREF	Direct	.3490	**.3392**	**.2821**	.1834
	RankNet	.3437	.3294	.2661	.1805
	Optimal	.3504	***.3557***	***.2840***	**.1907**
P@10	Direct	.3360	**.4820**	.4323	.3204
	RankNet	.3400	.4240	.4384	.3143
	Optimal	*.3520*	**.4960**	.4434	**.3388**

Table 1: Test set effectiveness for parameters estimated using direct search and RankNet. Optimal effectivness values are also provided as an upper bound. Effectiveness is measured in terms of mean average precision, binary preference, and precision at 10. Italicized values indicate statistically significant improvements over direct search. Bold values indicate significant improvements over RankNet.

These results indicate that RankNet is never significantly better than direct search for estimating the two-stage language modeling parameters. While RankNet often does produce reasonable effectiveness, it is considerably less consistent than direct search. This is likely the result of the cost function minimizing a surrogate function that only tends to be correlated with actual retrieval metrics. As pointed out by Taylor et al., it may be possible to improve the consistency of RankNet by measuring the retrieval metric on some validation set and using that as a stopping criteria [2].

Therefore, when computationally feasible, direct search still seems to be the most appropriate method for estimating parameters. However, RankNet acts as a good surrogate that scales better and often produces reasonable results.

Acknowledgments

This work was supported in part by the CIIR, in part by NSF grant #CNS-0454018, in part by ARDA and NSF grant #CCF-0205575, and in part by Microsoft Live Labs. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the sponsor.

5. REFERENCES

[1] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 89–96, 2005.

[2] M. Taylor, H. Zaragoza, N. Craswell, S. Robertson, and C. Burges. Optimisation methods for ranking functions with multiple parameters. In *Proc. 15th CIKM*, pages 585–593, 2006.

[3] C. Zhai and J. Lafferty. Two-stage language models for information retrieval. In *Proc. 25th SIGIR*, pages 49–56. 2002.