Half Transductive Ranking

Bing Bai$^{(1)}$ **Jason Weston**$^{(2)}$ **David Grangier**$^{(1)}$
Ronan Collobert$^{(1)}$ **Corinna Cortes**$^{(2)}$ **Mehryar Mohri**$^{(2)(3)}$

$^{(1)}$NEC Labs America, Princeton, NJ
{bbai, dgrangier, collober}@nec-labs.com
$^{(2)}$ Google Research, New York, NY
{jweston, corinna, mohri}@google.com
$^{(3)}$ NYU Courant Institute, New York, NY
mohri@cs.nyu.edu

Abstract

We study the standard retrieval task of ranking a fixed set of documents given a previously unseen query and pose it as the *half-transductive* ranking problem. The task is partly *transductive* as the document set is fixed. Existing transductive approaches are natural non-linear methods for this set, but have no direct out-of-sample extension. *Functional* approaches, on the other hand, can be applied to the unseen queries, but fail to exploit the availability of the document set in its full extent. This work introduces a *half-transductive* approach to benefit from the advantages of both *transductive* and *functional* approaches and show its empirical advantage in supervised ranking setups.

1 Introduction

The task of ranking a set of documents given a query is the core task of Information Retrieval (IR). In most setups, the set of documents to rank is fixed (or slowly growing) while the set of submitted queries is unknown. This environment gives rise to an interesting learning problem, *half-transductive ranking*, in which all the documents to rank are available at training time, while the test queries are only revealed once the model is trained.

Although frequent in Information Retrieval, previous literature mainly ignores this specific aspect and considers a setup in which the ranking model is learned to generalize to both new documents and new queries. Most ranking models learn a scoring function which assigns a real valued score to a feature vector describing a query/document pair. Given a query, the function is applied to each document and documents are ranked by decreasing scores. Models relying on this paradigm include the ranking perceptron [1], ranking SVM [2] or rankNet [3] among others. Hence, these approaches can be referred to as *functional*.

In contrast, *non-functional* approaches have been proposed in the literature for fully transductive setups, where the learnt similarity measure can only be assessed between objects available at training time. These approaches, such as as Isomap [4], Locally Linear Embedding [5] or Laplacian Eigenmaps [6], perform nonlinear dimensionality by learning a parameter vector for each training object. Compared to functional approaches, non-functional strategies provide highly non-linear embeddings, while relying on simple optimization. Moreover, they do not tie the object embedding to a specific feature representation but rather focus on the inter-object relationships.

Despite these advantages, non-functional approaches are hard to apply in our setup, which is only half-transductive: all the documents are available for training, but the test queries are not. Of course, one could advocate for using out-of-sample extensions that allow embedding a new object into a learned space [7]. Such a strategy is however not desirable in a retrieval context since: (i) it requires solving a (potentially expensive) optimization problem after the submission of a new query, which is the most time critical period for a retrieval system; and (ii) it neglects that available training queries could be used to identify a good strategy for embedding test queries.

This work proposes a direct solution to the *half transductive ranking* problem by combining a functional model for embedding queries and a non-functional model for embedding objects. This approach hence benefits from the advantages of the non-functional approaches mentioned above, while retaining the efficiency and generalization ability of functional approaches for coping with new queries. In the following, we first briefly describe existing ranking models before introducing the proposed approach. Then, we empirically compare this solution to its alternatives. Finally, we draw some conclusions and delineate future research directions.

2 Functional and Non-Functional Models

Our ranking problem is classical. We are first given a fixed set of m objects, $\{y_i\}_{i=1}^m$, and we consider the task of ranking these objects given a new query x, unknown at training time. In the following, we classify models addressing this problem as either *functional* or *non-functional*.

Functional Models represent query x and object y using a joint feature representation $\Phi(x, y)$ and scores the pair (x, y) using a function $x, y \rightarrow f_w(\Phi(x, y))$ parametrized by w. Instances of such functions include: functions linear in the feature space [1, 2], neural network functions [3] or Gaussian Processes [8]. These models can be learned with training objectives linked to the pairwise ranking loss, Normalized Discounted Cumulative Gain (NDCG) or Mean Averaged Precision (MAP).

Functional models also include Latent Semantic Indexing (LSI) [9], which corresponds to a linear function with a specific choice of features and parameter regularization, i.e. one can notice that LSI scoring, $f_W(x, y) = \phi(x)^{\mathrm{T}} W^{\mathrm{T}} W \phi(y), \quad \text{where} \quad \phi(x), \phi(y) \in \mathbb{R}^d, \quad W \in \mathbb{R}^{d \times n}, \quad n < d$, is equivalent to a linear model, when one defines $w = W^{\mathrm{T}} W$ and $\Phi(x, y) = \phi(x)\phi(y)^{\mathrm{T}}$. In this case, the parameters are learned with a mean-squared reconstruction objective, by applying Singular Value Decomposition to the document-term matrix [9]. The same parametrization is also used by Supervised Semantic Indexing (SSI [10]), which learns the parameters from a supervised signal, to minimize the pairwise ranking loss.

Non-Functional Models are *transductive* approaches which assign a parameter vector $v_i \in \mathbb{R}^d$ to each object y_i available at training time. Contrary to functional approaches, these strategies do not tie the object representation v_i to a feature representation $\Phi(\cdot)$ through a function. Instead, they consider a function f comparing examples in the embedding space $\mathbb{R}^d$, e.g. the Euclidean distance $f(y_i, y_j) = ||v_i - v_j||_2$ or the dot-product $f(y_i, y_j) = v_i \cdot v_j$, and learn $\{v_i\}_{i=1}^m$ by considering desired relationships between objects.

Several transductive embedding approaches have been proposed in the recent years. Most of them are rooted either in factor analysis (e.g. Principal Component Analysis) or Multi-Dimensional Scaling, including kernel PCA [11], Isomap [4], Locally Linear Embedding [5] and Laplacian Eigenmaps [6]. For example the latter embeds points by minimizing $\sum_{ij} L(v_i, v_j, A_{ij}) = \sum_{ij} A_{ij} ||v_i - v_j||_2^2$ where A is a similarity ("affinity") matrix, under constraints that ensure a trivial solution is not reached. An overall review of this family of methods can be found in [12].

Non-functional, transductive models hence offer a flexible framework to learn highly non-linear models and benefit from available information on relationships between objects. However, extensions to new objects – such as queries in our framework – are computationally costly and depends on the availability of information on the relationship between the new object and the already available objects [7].

3 Half-Transductive Ranking

In this work, we propose *Half-Transductive Ranking* to benefit from the advantages of both functional and non-functional approaches. Like transductive approaches, we assign a vector v_i to each object y_i in the fixed set to be ranked. However, we then consider a query x can be any object, not necessarily available during training. In that case, a function is applied to map x into the embedding space. Specifically, our scoring function is

$$f(x, y_i) = \langle W\phi(x), v_i \rangle = \phi(x)^{\mathrm{T}} W^{\mathrm{T}} v_i, \quad \text{where} \quad W \in \mathbb{R}^{d \times n}, \quad \forall i = 1 \dots m,\ v_i \in \mathbb{R}^d,$$

i.e. the dot product is used to compare embedded examples and a linear projection is used for mapping queries (nonlinear embeddings are implemented via $\phi(\cdot)$ being a fixed nonlinear map).

The parameters of our model, i.e. W and $\{v_i\}_{i=1}^m$ are learned to optimize the pairwise ranking loss. From a set of preference relations $\mathcal{R}$, where $\forall (x, y_+, y_-) \in \mathcal{R}$ expresses that document y_+ is preferred to document y_- for query x, we would like to learn f such that $f(x, y_+) > f(x, y_-)$. Like the margin ranking perceptron [1] or the ranking SVM [2], we minimize the hinge loss over such pairwise constraints,

$$L_{\mathrm{HTR}}(\mathcal{R}) = \sum_{(x,y_+,y_-) \in \mathcal{R}} \max(0, 1 - \phi(x)^{\mathrm{T}} W^{\mathrm{T}} v_+ + \phi(x)^{\mathrm{T}} W^{\mathrm{T}} v_-).$$

In our experiments, we noticed that a better linear mapping W could be obtained by learning to embed documents *functionally* as well and we introduce an hyperparameter γ to weight this second learning objective, i.e.

$$L(\mathcal{R}; \gamma) = L_{\mathrm{HTR}}(\mathcal{R}) + \gamma \sum_{(x,y_+,y_-) \in \mathcal{R}} \max(0, 1 - \phi(x)^{\mathrm{T}} W^{\mathrm{T}} W \phi(y_+) + \phi(x)^{\mathrm{T}} W^{\mathrm{T}} W \phi(y_-)).$$

We propose to train this model using stochastic gradient descent as this optimization strategy allows to achieve great scalability while avoiding poor local optima. The learning algorithm is hence simple: initially, the entries of W and $\{v_i\}_{i=1}^m$ are drawn randomly using from a normal distribution $\mathcal{N}(0, 1)$. Then, one iteratively picks a random triplet $(x, y_+, y_-) \in \mathcal{R}$ and updates the parameters according to the gradient of $(W, v_+, v_-) \rightarrow L(\{(x, y_+, y_-)\}; \gamma)$. One can note that this approach is computationally interesting since each iteration only updates W and two of the vectors $\{v_i\}_{i=1}^m$. Moreover, one can further exploit the potential sparsity of $\phi(x)$ for further gains. In our experiments, regularization is achieved through early stopping, i.e. one stops training when the performance over held-out validation data stops improving.

4 Experiments and Results

Our ranking experiments are carried out on Wikipedia and use the link structure of this dataset to build a large scale ranking task. Wikipedia provides a large document set with a meaningful, closed link structure. Compared to benchmark datasets, we can work with bigger query sets compared to TREC[1] and we can extract document-level features as opposed to LETOR[2].

The dataset we consider consists of 1,828,645 English Wikipedia documents and 24,667,286 links[3] which is randomly split into two portions, 70% for training (and validation) and 30% for testing. The following task is then considered: given a query document x, rank the other documents such that if x links to y then y should be highly ranked. Documents are represented with bag-of-word vectors, with a normalized TFIDF weighting [10].

Our results are compared to four alternative models: simple cosine similarity (TFIDF), LSI [9], margin ranking perceptron [1] and SSI [10]. We report results in two settings, (i) using only the top 30,000 most frequent words; (ii) using all 2.5 million words in Wikipedia. In the first setting, we rely on a margin percetpron with $\Phi(x, y) = \phi(x)\phi(y)^{\mathrm{T}}$. In the second setting, we rely on perceptron using Hash Kernels [13] as the previous feature choice does not allow the model to fit in memory.

Table 1 reports the performance using the pairwise ranking error, Mean Averaged Precision, MAP, and precision at top 10, P@10. It shows that supervised models (Perceptron, SSI, Half-Transductive) outperform unsupervised ones (TFIDF, LSI). It also shows that embedding the data in a lower dimensional space is an effective capacity control mechanism (SSI versus Perceptron). More importantly, it shows that the additional non-linearity added by introducing the non-functional parameter vectors

[1]TREC datasets offer ~ 50 to ~ 500 queries, `http://trec.nist.gov`

[2]LETOR provides features for query/doc. pairs only, `http://learningtorank.spaces.live.com/`

[3]We removed links to calendar years as they provide little information while being very frequent.

Table 1: Document-document ranking on Wikipedia

Algorithm	30k words			2.5M words		
	Rank-Err (%)	MAP	P@10	Rank-Err (%)	MAP	P@10
TFIDF	1.62	0.33	0.16	0.84	0.43	0.19
LSI	1.28	0.35	0.17	0.72	0.43	0.19
Perceptron	0.41	0.48	0.21	1.37	0.33	0.16
SSI	0.30	0.52	0.23	0.16	0.55	0.24
Half-Transductive	**0.20**	**0.56**	**0.24**	**0.11**	**0.61**	**0.26**

$\{v_i\}_{i=1}^m$ greatly enhances performance (SSI versus Half-Transductive). In fact, the half-transductive model outperforms all alternatives. A last remark on γ, this hyper-parameter is important: it pushes W toward the solution provided by SSI which results in significant gain ($\gamma = 0$ yields a pairwise error of 0.38% over the 30k-word test data compared to 0.20%).

5 Conclusions

This work studies the task of ranking a fixed set of documents given a previously unseen query. This classical IR setting yields us to introduce *Half Transductive Ranking*. Like transductive, non-functional embeddings, we learn a parameter vector to represent each of the known documents. At the same time, we learn a functional mapping to project test queries onto the document embedding space. This approach is different from test-time out-of-sample extensions for nonlinear dimensionality reduction techniques since we learn this function from training queries at training time. This approach is also different from classical functional ranking approaches which focus on models generalizing to both new documents and new queries while our strategy explicitly considers the availability of the documents at both train and test time. Our experiments over Wikipedia data demonstrate the advantages of our approach over functional ranking alternatives.

Acknowledgments Part of this work has been conducted while J. Weston was with NEC Labs America.

References

[1] M. Collins and N. Duffy. New ranking algorithms for parsing and tagging: kernels over discrete structures, and the voted perceptron. In *ACL*, 2001.

[2] T. Joachims. Optimizing search engines using clickthrough data. In *KDD*, 2002.

[3] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *ICML*, 2005.

[4] M. Balasubramanian, E.L. Schwartz, J.B. Tenenbaum, V. de Silva, and J.C. Langford. The Isomap algorithm and topological stability. *Science*, 295(5552):7–7, 2002.

[5] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.

[6] M. Belkin and P. Niyogi. Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. *Neural Computation*, 15(6):1373–1396, 2003.

[7] Y. Bengio, J.F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-sample extensions for LLE, isomap, MDS, eigenmaps, and spectral clustering. In *NIPS*, 2004.

[8] J. Guiver and E. Snelson. Learning to rank with softrank and gaussian processes. In *SIGIR*, 2008.

[9] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[10] B. Bai, J. Weston, R. Collobert, and D. Grangier. Supervised semantic indexing. In *ECIR*, 2009.

[11] B. Schlkopf, A. Smola, and K.R. Müller. Kernel principal component analysis. *Advances in kernel methods: support vector learning*, pages 327–352, 1999.

[12] J. A. Lee and M Verleysen. *Nonlinear Dimensionality Reduction*. Springer, New York, 2007.

[13] Q. Shi, J. Petterson, G. Dror, J. Langford, A. Smola, A. Strehl, and V. Vishwanathan. Hash kernels. In *AISTATS*, 2009.