Learning to Rank with Deep Autoencoder Features

Alberto Albuquerque
CS Dept., UFMG
Brazil
Email: alberto@dcc.ufmg.br

Tiago Amador
CS Dept., UFMG & UEFS
Brazil
Email: tiago@uefs.br

Renato Ferreira, Adriano Veloso
CS Dept., UFMG
Brazil
Email: {renato,adrianov}@dcc.ufmg.br

Nivio Ziviani
CS Dept., UFMG & Kunumi
Brazil
Email: nivio@kunumi.com

***Abstract*—Learning to rank in Information Retrieval is the problem of learning the full order of a set of documents from their partially observed order. Datasets used by learning to rank algorithms are growing enormously in terms of number of features, but it remains costly and laborious to reliably label large datasets. This paper is about learning feature transformations using inexpensive unlabeled data and available labeled data, that is, building alternate features so that it becomes easier for existing learning to rank algorithms to find better ranking models from labeled datasets that are limited in size and quality. Deep autoencoders have proven powerful as nonlinear feature extractors, and thus we exploit deep autoencoder features for semi-supervised learning to rank. Typical approaches for learning autoencoder features are based on updating model parameters using either unlabeled data only, or unlabeled data first and then labeled data. We propose a novel approach which updates model parameters using unlabeled and labeled data simultaneously, enabling label propagation from labeled to unlabeled data. We present a comprehensive study on how deep autoencoder features improve the ranking performance of representative learning to rank algorithms, revealing the importance of building an effective feature set to describe the input data.**

***Index Terms*—Deep Autoencoders, Learning to Rank**

I. Introduction

Information retrieval relies on methods that are able to distinguish relevant from irrelevant documents. Human ingenuity has produced many predictors of document relevance [1]. For example, a predictor can be the document PageRank score, the query length, a count of how many times the query words appeared in the document or how many documents in the data collection each query word appeared. Learning to rank (L2R) algorithms [2] use these predictors as features, and they achieve superior ranking performance by discovering a predictive relationship between these features through the minimization of some loss function using a labeled dataset [3]. Producing labeled datasets, however, is usually very costly as it requires human annotators (a.k.a., Google raters) to assess the relevance or to order the documents for some queries. Further, factors such as the ambiguity of query intent, the lack of domain knowledge and the vague definition of relevance levels make it difficult for human judges to give reliable labels to some documents [4]. Therefore, labeled datasets are limited not only in size, but also in quality [5], [6].

In this paper we propose an alternative way for semi-supervised L2R. First, a new feature space is found by solving a minimization problem which may involve both labeled and unlabeled data [7] which are inexpensive and available in large quantities. Then, the input data is projected into the new feature space, so that supervised L2R algorithms can be simply applied using the new features. Instead of learning representations from raw data, our approaches assume as input the large variety of existing relevance predictors. More specifically, our objective is to find a new feature space while taking into account the considerable domain expertise and human insights about document relevance. Our main hypothesis is that identifying features that are independent, robust to noise, and more discriminating than existing relevance predictors facilitates L2R algorithms to find more effective ranking models.

In summary, the research question we address in this paper is how to improve the ranking performance of existing L2R algorithms by describing the input data using more effective features. Deep autoencoders [8] have shown to produce effective representations of the input data, improving the supervised stage on a variety of applications [9]. We present an alternative representation for L2R algorithms in which the interactions between relevance predictors are considered on multiple levels of abstraction of an autoencoder. We employ different loss functions and the process of minimizing these functions results in diverse feature representations for L2R algorithms.

Contributions and Findings The main contributions of this paper are new semi-supervised L2R approaches. In practice, we claim the following benefits and contributions:

- We propose semi-supervised L2R approaches that use deep autoencoders for learning feature spaces so that it becomes easier for existing L2R algorithms to distinguish relevant from irrelevant documents. Our approaches differ on how unlabeled and labeled data are exploited. Specifically, the first approach employs only unlabeled data and learns new features by simply minimizing the reconstruction error. The second approach minimizes reconstruction error first and then minimizes prediction error, thus unlabeled and labeled data are used in two separate steps. The third approach employs unlabeled and labeled data simultaneously, propagating labels to unlabeled data while minimizing a hybrid loss function (i.e., reconstruction and prediction error simultaneously).
- Autoencoders have shown success in learning effective features for a variety of applications, but the use of deep autoencoders in L2R is new. We evaluate several autoencoder configurations including under- and over-complete,

dropout, and denoising autoencoders with varying depths. This provides multiple dimensions to assess the ranking performance of existing L2R algorithms using the corresponding autoencoder features. We show that a relevant fraction of plausible autoencoder configurations lead to significant improvements in the ranking performance of existing L2R algorithms. Further, gains are also reported when our approaches are compared with other semi-supervised L2R approaches.

II. Related Work

Our work relates to two lines of research in learning to rank: learning from unlabeled data and feature space transformation.

Unlabeled Data in L2R Large quantities of unlabeled documents and queries can always be collected at a low cost (from query logs, for instance). It would be very helpful if one can leverage such unlabeled data in the L2R process, possibly reducing the volume of required labeled data [10]. The typical approach used for learning from unlabeled data is to propagate the labels of the labeled documents to unlabeled documents, thus increasing the number of labels available for training. It is commonly assumed that unlabeled and labeled documents that are similar in the feature space should be associated with the same label (i.e., the manifold assumption). In [11], [12], the authors followed the label propagation approach in order to actively sample a training-set, which presents a significant reduction on the volume of labeled data required to learn ranking models. In [13], the authors introduced a regularizer favoring that similar documents are similar in preference to each other. A semi-supervised algorithm called SS$-\lambda$Rank was proposed, and improvements over λRank were reported. In [14], the authors proposed a semi-supervised ranking algorithm that learns query-dependent weights when combining multiple rankers in document retrieval. The proposed algorithm has to learn a different set of weights online for each query, which makes the algorithm infeasible for large-scale applications.

Feature Space Transformation in L2R While L2R algorithms have been intensively studied [2], [15], this is not the case for the features used by these algorithms, despite strong evidence showing that the way in which data are represented can make a huge difference in the success of learning algorithms [16]. In [17], the authors defined the importance of a feature in terms of the ranking performance on a loss function, and the correlation between the ranking results of two features as the similarity between them. Based on these definitions, they formulate feature selection as an optimization problem, for which it is to find the features with maximum total importance scores and minimum total similarity scores. In [18], the authors proposed a joint convex optimization formulation which minimizes ranking errors while simultaneously conducting feature selection.

In contrast to feature selection where the objective is to identify a subset of features that yield a comparable or even better effectiveness than using all the features, the key idea behind feature transformation is to derive better features to represent the input data. In [19], the authors proposed a transductive approach based on kernel PCA [20] to discover salient patterns using the unlabeled test set (i.e., principal directions). The labeled data are then projected onto the directions of these patterns, resulting in another feature space. Similarly, in [21], the authors proposed a transductive approach to learn ranking functions. The main intuition is to obtain paired preference data from the unlabeled test set using association rules.

Present Work Our work joins these two lines of research by developing semi-supervised L2R approaches that transform the feature space so that documents are more effectively represented in the new feature space. In contrast to [19], which assume a transductive scenario in which the test set is used to learn feature transformations, we assume a semi-supervised scenario in which a large amount of unlabeled data can be explored off-line. Further, from an algorithmic perspective, the use of deep autoencoders can model relatively complex relationships and non-linearities, while providing a series of additional options to explore, such as noise removal. Finally, we propose a novel approach which learns autoencoder features while enabling label propagation during the learning process. As a result, unlabeled data are gradually included into the training set and this information becomes available in further epochs of training.

III. Semi-Supervised L2R

The task of learning to rank in information retrieval is defined as follows. We have as input the *training set* (referred to as $\mathcal{D}$), which consists of a set of records of the form $< q, d, r >$, where q is a query, d is a document, and r is the *relevance* of d to q. The relevance draws its values from a discrete set of possibilities, ranging from completely irrelevant to near perfect match. The training set is used to learn a model which relates features of the query-document pairs to their corresponding relevance. The *test set* (referred to as $\mathcal{T}$) consists of records $< q, d, ? >$ for which only the query q and the document d are known, while the relevance of d to q is unknown. The model learned from the training set $\mathcal{D}$, denoted as $f(q, d)$, is used to produce an estimate of the relevance of documents to the corresponding queries, which can be used to produce a final ranking. That is, we would like to choose a model $f(q, d)$ such that $f(q, d^+) > f(q, d^-)$, expressing that a document d^+ should be ranked higher than another document d^-, given that d^+ is more relevant to q than d^-. We also assume the existence of an *unlabeled set* (referred to as $\mathcal{U}$) which consists of records of the form $< q, d >$.

We consider a learning scenario where labels are expensive to obtain and potentially noisy, and thus the training set $\mathcal{D}$ is limited both in size and quality. Unlabeled records, on the other hand, are inexpensive and available in large quantities, being safe to assume that $|\mathcal{U}| \gg |\mathcal{D}|$. A query-document pair $< q, d >$ is represented as a list of features $x = \{x_1, x_2, \ldots, x_n\}$, where each x_i is a relevance predictor. Datasets used by L2R algorithms are growing enormously in terms of number of features. These features can be grouped as follows:

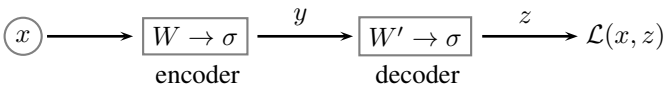


Fig. 1. Learning alternative feature representation using unlabeled data.

- Query-independent (or static features) − features that depend only on the document, but not on the query (e.g., PageRank or the length of the document).
- Query-dependent (or dynamic features) − features that depend both on the contents of the document and the query (e.g., TF-IDF score).
- Query level features − features that depend only on the query (e.g., the number of words in a query).

Typically, the dimensionality of datasets used by L2R algorithms are given in order of hundreds. Our proposed semi-supervised L2R approaches are based on learning feature transformations, and can be summarized in two separate steps:

1) The unlabeled set $\mathcal{U}$ is used to find an alternative feature space in which $< q, d >$ pairs become more separable and robust to noise. The training set $\mathcal{D}$ can also be exploited during this step.
2) The training set $\mathcal{D}$ is projected onto the new feature space, and then used by existing L2R algorithms to find the model $f(q, d)$.

The second step is straightforward, and thus we focus on the first step. We propose three approaches for learning feature transformations for semi-supervised L2R, which we describe next.

A. Learning from Unlabeled Data

We assume that feature representations of x that are useful to capture the input distribution $P(x)$ are also in part useful to capture $P(l = i|x)$, where i is a relevance level. That is, the input distribution $P(x)$ is structurally related to $P(l = i|x)$. Given this assumption, we apply autoencoders [8] in order to learn alternative representations for the inputs. An autoencoder is a feed-forward network composed of two components: an encoder and a decoder. The encoder takes the input $x = \{x_1, x_2, \ldots, x_n\}$, and maps it to a hidden representation $y = \{y_1, y_2, \ldots, y_m\}$, which is given by:

$$y = \sigma(Wx + b)$$

where b is the bias value, W is the weight matrix, and σ simply represents an element-wise sigmoid.

The latent or hidden representation y is mapped back to the original input x, using a decoder which is given by:

$$z = \sigma(W'y + b')$$

where b' is the bias value and W' is the weight matrix. In our context, W' is not necessarily the transpose of W.

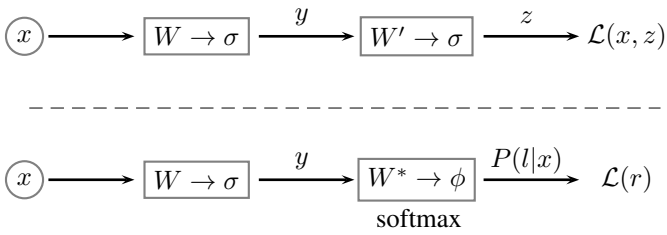


Fig. 2. Unlabeled data is exploited first, then labeled data is also exploited.

Unsupervised autoencoders use backpropagation to minimize the reconstruction loss given by:

$$\mathcal{L}(x, z) = ||x - z||^2$$

where inputs with low reconstruction error are associated with higher probabilities $P(x)$. Thus, autoencoder features are produced while approximating $P(x)$.

Figure 1 shows a conceptual illustration of an unsupervised autoencoder. Many autoencoders can be stacked and trained one after another − once the bottom encoders are trained, we can train the next autoencoder by computing the representation from the autoencoder below. The key idea is that each level produces a representation of the input that is more abstract than the one in the previous level, because it is obtained by composing more operations. Further, one typically regularizes the autoencoder by adding noise to the input data [8]. Specifically, input x is corrupted with noise, but the autoencoder is trained to recover the original input. By undoing the corruption process the learned representation y become robust to small irrelevant changes in input.

B. Learning from Unlabeled+Labeled Data

So far, the process of finding autoencoder features was completely unsupervised, but labeled data can also be used. In this case, the autoencoder is first trained on unlabeled data, and then its representation is fed to a classifier. The classifier uses a softmax function, which takes input x and produces a vector with real values in the range $(0, 1)$ that add up to 1. These values are interpreted as probabilities associated with each label, that is $P(l = i|x)$. The softmax classifier uses the log-likelihood loss function given by:

$$\mathcal{L}(r) = -\log(P(l = r|x))$$

where r is the ground-truth label. The use of log-likelihood loss provides extreme punishments for being both confident and wrong. Figure 2 shows a conceptual illustration of both unsupervised and supervised steps. We expect that using labeled data makes the training set $\mathcal{D}$ in the new feature space to become more separable.

C. Learning with Label Propagation

The following approach exploits labeled and unlabeled data simultaneously. As show in Figure 3, the input x is used by two independent networks. One network calculates the

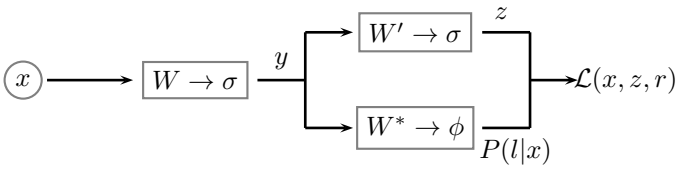


Fig. 3. Unlabeled and labeled data are exploited simultaneously. Labels are gradually propagated to unlabeled data.

reconstruction loss $\mathcal{L}(x, z)$, while the other network calculates the log-likelihood loss $\mathcal{L}(r)$. The network minimizes an hybrid loss function given by:

$$\mathcal{L}(x, z, r) = (1 - \alpha) \times \mathcal{L}(x, z) + \alpha \times \mathcal{L}(r)$$

where α $(0.0 < \alpha \leq 1.0)$ is used to give preference to either the reconstruction loss or the log-likelihood loss.

The proper choice of α is very important for balancing $\mathcal{L}(x, z)$ and $\mathcal{L}(r)$, that is, α essentially trade-off the supervised and unsupervised parts of the objective. If α is excessively low, the optimization process does not consider labeled data. On the other hand, if α is set too high, the optimization process does not benefit from unlabeled data.

Minimizing this hybrid loss function with respect to the model parameters W, W', and W^* is a non convex optimization problem that we solve with backpropagation. Next, we discuss an effective way that allows us to update the model parameters W, W' and W^* simultaneously on labeled and unlabeled data in the same batch of observations.

Label Propagation We assume that query-document pairs within the same manifold (or cluster) are likely to have the same relevance label. Given this assumption, our aim is to use unlabeled data to uncover these clusters, so that labels can be propagated through clusters with high density. Thus, we define the most probable relevance label for an arbitrary input x as:

$$r^* = \arg\max_i P(l = i|x)$$

and we say that r^* is a reliable label for x if $P(l = r^*|x) \geq \omega_{min}$, where ω_{min} is a user-specified threshold $(0.0 < \omega_{min} \leq 1.0)$. The idea is to use ω_{min} as a threshold indicating the minimum reliability necessary to regard the label as being the correct one. Further, reliable labels can be used to include new records into the training set $\mathcal{D}$, thus propagating labels to inputs in the unlabeled set $\mathcal{U}$. Finally, reliable labels may also be used to compute the log-likelihood loss. Specifically, if x is an unlabeled input and $P(l = r^*|x) \geq \omega_{min}$, then the reliable label r^* is assumed to be the correct label r, so that the log-likelihood loss can be simply computed.

Naturally, some labels are not reliable enough, given certain values of ω_{min}. In this case, W^* is not updated and the loss function simply reduces to the reconstruction loss. However, as new labeled records are included into the training set $\mathcal{D}$, new evidence is exploited while minimizing the loss function, hopefully increasing the reliability of labels that were unreliable in previous epochs of the optimization. The proper choice of ω_{min} is very important for label propagation. If ω_{min} is set too low, most of the labels included into $\mathcal{D}$ may be wrong. On the other hand, if ω_{min} is set too high, only few labels will be regarded as reliable ones.

IV. Experimental Setup

In this section we present datasets and baselines used for evaluating our semi-supervised L2R approaches, and then we discuss our evaluation procedure.

A. Datasets

We used pre-folded datasets to evaluate our algorithms:

- MQ2007-semi: this dataset contains approximately 70,000 labeled query-document pairs from 1,700 unique queries. There are 46 original features and three levels of relevance. There are also 701,102 unlabeled query-document pairs on this dataset.
- MQ2008-semi: this dataset contains approximately 15,000 labeled query-document pairs associated with 780 unique queries. There are 46 original features and three levels of relevance. There are also 531,050 unlabeled query-document pairs on this dataset.
- MSLR-WEB10K: this dataset contains approximately labeled 1.2 million query-document pairs from 10,000 unique queries. There are 136 original features and five levels of relevance. We removed 95% of labels in order to simulate unlabeled query-document pairs.

B. Baselines

We considered the following algorithms in order to provide strong baseline comparison:

- Supervised L2R algorithms: we consider $\lambda-$Mart [22], RankSVM [23], RankBoost [24], AdaRank [25], and ListNet [2] as state-of-the-art representatives. Our prime objective is to investigate how the alternate feature representations affect the ranking performance of these algorithms. In particular, we will investigate the design choices that lead to autoencoder features for which the ranking performance of L2R algorithms increases, decreases, or are not significantly affected.
- Semi-supervised algorithms: we consider T-KPCA [19] and SS$-\lambda$Rank [13] as state-of-the-art representatives.

C. Evaluation Procedure

To evaluate the ranking performance of the L2R algorithms, we have used the standard NDCG@10 (Normalized Discounted Cumulative Gain) measure [26]. We conducted five-fold cross validation − unlabeled data is held fixed while labeled data is arranged into five folds with about the same number of queries. At each run, three folds are used as training set, one fold is used for tuning the hyper parameters of the L2R algorithm,[1] and the remaining fold is used as test set. The results reported are the average of the five runs, and

[1]Parameters used are those that lead to the best results for each L2R algorithm given an autoencoder configuration.

are used to measure the overall ranking performance of L2R algorithms. We used the LETOR 4.0 evaluation tools.[2] In order to ensure the relevance of the results, we assess the statistical significance of our measurements by means of a pairwise t-test [27] with $p-$value ≤ 0.05.

Design Choices For each dataset and each semi-supervised L2R approach, we considered 600 autoencoder configurations. Each configuration employs a different combination of the following criteria:

- Depth: it is the number of autoencoder levels. We considered the values $\{1, 2, 3, 4, 5\}$.
- Dimension: it is the size of the final autoencoder representation. Its value is relative to the dimension of the original representation. We considered the values $\{0.2, 0.5, 0.9, 1.1,\ 1.5, 2.0\}$. The rate in which the autoencoder representations expand or contract depends on this value. A value of 0.5, for instance, means that the final representation has half the dimension of the original feature representation.
- Dropout: it is the fraction of units that are dropped. We considered the values $\{0.2, 0.3, 0.4, 0.5\}$.
- Noise: it is the level of corruption injected into the original features (Additive isotropic Gaussian noise [8]). We considered the values $\{0.0, 0.1, 0.2,\ 0.3, 0.4\}$.

For the "Label Propagation" approach we also varied α and ω_{min}. A larger focus on the supervised objective is important but a weight of α for the reconstruction error prevents overfitting, and thus we fixed $\alpha = 0.75$. After some inspection we fixed $\omega_{min} = 0.75$, since low ω_{min} values incur the inclusion of many wrong labels in $\mathcal{D}$. We ran each L2R algorithm on each autoencoder configuration for each dataset. Therefore, we reported results based on a total of $5 \times 3 \times 600 + 5 = 9,005$ models for each dataset (five L2R algorithms, three semi-supervised L2R approaches, and 600 autoencoder configurations plus the original feature space).

V. Experimental Evaluation

We aim to answer the following research questions related to the effectiveness of our semi-supervised L2R approaches:

- RQ1 Do existing L2R algorithms benefit from autoencoder features? Which algorithms are more likely to benefit from these features? Which semi-supervised L2R approach is more effective?
- RQ2 Which autoencoder configurations are likely to lead to better ranking models?
- RQ3 Are the transformed feature spaces more structured than the original feature space?
- RQ4 How our semi-supervised L2R approaches perform when compared with existing semi-supervised L2R approaches?

A. Effectiveness of Autoencoder Features

Our first experiment shows the fraction of autoencoder configurations that lead to (i) significant improvement (↑), (ii) similar (−), and (iii) significant decrease (↓) in the ranking performance of L2R algorithms. Table I shows improvement numbers for the MQ2007-semi dataset. The "Label Propagation" semi-supervised L2R approach leads to the largest number of performance improvements. Specifically, 47 to 63 out of 600 autoencoder configurations lead to significant improvements, depending on the L2R algorithm being considered. The "Unlabeled Only" semi-supervised L2R approach, which learns feature transformations using unlabeled data only, leads to the smaller number of improvements. More specifically, only 16 to 24 configurations lead to significant improvements, depending on the L2R algorithm being considered. The "Unlabeled+Labeled" semi-supervised L2R approach leads to improvements on 24 to 28 configurations, depending on the L2R algorithm being considered. The same trend is observed for the fraction of autoencoder configurations for which the L2R algorithms achieve a ranking performance which is similar to the one obtained using the original features. Finally, as expected, the opposite trend is observed for the fraction of autoencoder configurations that lead to significant decrease in the ranking performance of L2R algorithms.

Table II shows improvement numbers for the MQ2008-semi dataset. The overall results are very similar to the results obtained with the MQ2007-semi dataset. There are two main differences, however. First, it is observed a slightly increase in the fraction of configurations that lead to improvement in ranking performance. Second, the fraction of autoencoder configurations for which the ranking performance of the algorithms does not change significantly has decreased greatly for

TABLE I
VARIATION IN RANKING PERFORMANCE ON MQ2007-SEMI DATASET.

Algorithm		Unlabeled only			Unlabeled+Labeled			Label Propagation		
	N@10	↑	−	↓	↑	−	↓	↑	−	↓
RankBoost	.446	.033	.133	.833	.040	.253	.707	.078	.313	.608
RankSVM	.443	.027	.140	.833	.047	.247	.707	.085	.320	.595
ListNet	.444	.040	.133	.827	.047	.253	.700	.098	.327	.575
AdaRank	.437	.040	.140	.820	.040	.260	.700	.105	.333	.562
λMart	.447	.033	.140	.827	.047	.253	.700	.085	.340	.575

TABLE II
VARIATION IN RANKING PERFORMANCE ON MQ2008-SEMI DATASET.

Algorithm		Unlabeled only			Unlabeled+Labeled			Label Propagation		
	N@10	↑	−	↓	↑	−	↓	↑	−	↓
RankBoost	.225	.040	.100	.860	.060	.147	.793	.098	.213	.688
RankSVM	.228	.033	.113	.853	.053	.153	.793	.125	.207	.668
ListNet	.231	.040	.107	.853	.060	.153	.787	.105	.213	.682
AdaRank	.231	.033	.113	.853	.053	.140	.807	.098	.213	.688
$\lambda-$Mart	.231	.033	.113	.853	.060	.153	.787	.098	.220	.682

TABLE III
VARIATION IN RANKING PERFORMANCE ON MSLR-WEB10K DATASET.

Algorithm		Unlabeled only			Unlabeled+Labeled			Label Propagation		
	N@10	↑	−	↓	↑	−	↓	↑	−	↓
RankBoost	.385	.033	.187	.780	.060	.247	.693	.105	.380	.515
RankSVM	.383	.033	.180	.787	.060	.240	.700	.105	.373	.522
ListNet	.392	.033	.180	.787	.067	.247	.687	.112	.373	.522
AdaRank	.389	.033	.187	.780	.060	.2533	.687	.105	.380	.515
λMart	.392	.040	.193	.767	.060	.253	.687	.112	.387	.502

[2] http://research.microsoft.com/en-us/um/beijing/projects/letor

all three semi-supervised L2R approaches. Again, the semi-supervised L2R approach that leads to the largest number of significant improvements in ranking performance was "Label Propagation" – 10% to 12% of the configurations lead to significant improvements in the ranking performance of the L2R algorithms considered.

Table III shows improvement numbers for the MSLR-Web-10K dataset. For this dataset, improvement numbers for different L2R algorithms are much closer. For instance, considering the "Unlabeled Only" approach, all L2R algorithms achieve the same improvement numbers, with exception of λRank. Similarly, considering the "Unlabeled+Labeled" approach, all L2R algorithms achieve the same improvement numbers, with exception of ListNet. The same trend is observed for the fraction of autoencoder configurations that lead to results that are similar to the obtained with the original features. The decrease in variation of ranking performance between the considered algorithms may indicate that, specially for this dataset, the feature representations force the algorithms to find very similar ranking models.

B. Concentration of Best Configurations

Figures 4 and 5 show the range of configurations for which it is more likely that the ranking performance of algorithms are significantly improved. Due to lack of space we only consider feature transformations based on "Label Propagation" and RankBoost and ListNet as the L2R algorithms.

The autoencoder configurations that lead to the best results in MQ2007-semi and MQ2008-semi datasets are very similar. Specifically, best results for these datasets were usually obtained with two autoencoder levels (Figure 4(a)). These autoencoders are usually under-complete (Figure 4(b)), and dropout and denoising are usually important (Figures 4(c) and 4(d)). For MSLR-Web10K, the best configurations are usually associated with deeper networks, with three to four autoencoder levels (Figure 4(a)). These autoencoders may be either under- or over-complete (Figure 4(b)), and denoising is of paramount importance (Figure 4(d)). Particularly, in the over-complete cases, the autoencoder is doing a transformation from one feature space to another wherein the input data in the new feature space disentangles factors of variation, as will be discussed in the next section.

C. Feature Spaces

Figures 6 and 7 show t-SNE embeddings obtained from the original and transformed feature spaces for MQ2007-semi and MQ2008-semi datasets. The feature spaces shown in the figures correspond to the feature representations associated with the best autoencoder configurations for each semi-supervised L2R approach, and considering RankBoost as the L2R algorithm. For MQ2007-semi, the original feature space seems very well separable. Still, the transformed spaces clearly show more evident clusters, with the purest space being the one obtained with "Label Propagation." For MQ2008-semi, the original feature space seems much more scattered than the space observed for MQ2007-semi. Again, our approaches clearly produced feature spaces that are much more structured.

Figure 8 shows t-SNE embeddings obtained from the original and transformed feature spaces for MSLR-Web10K. The original feature space seems almost chaotic, and relevant documents are hardly seen in the figure. The alternate features placed relevant documents in the borders, while unlabeled data and irrelevant documents are placed in the middle of the feature space. This trend is particularly observed with the "Label Propagation" approach.

TABLE IV
NDCG@10 NUMBERS FOR SEMI-SUPERVISED L2R APPROACHES ON MQ2007-SEMI DATASET.

Dataset	Unlabeled only						
	RBoost	RSVM	LNet	ARank	λMart	T-KPCA	SS$-\lambda$Rank
MQ2007-semi	.453	.452	.453	.451	.456	.451	.455
MQ2008-semi	.233	.234	.236	.236	.238	.232	.236
MSLR-WEB10K	.394	.393	.399	.398	.400	.396	.403

TABLE V
NDCG@10 NUMBERS FOR SEMI-SUPERVISED L2R APPROACHES ON MQ2008-SEMI DATASET. UNDERLINED RESULTS ARE STATISTICALLY SUPERIOR THAN THE RESULTS OBTAINED BY T-KPCA AND SS$-\lambda$RANK.

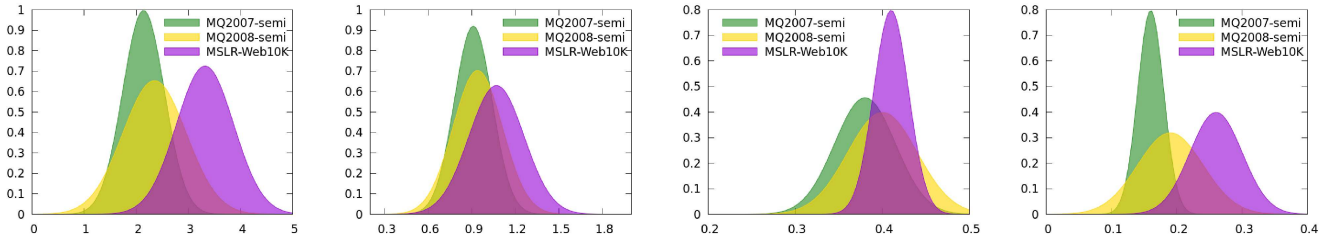
Dataset	Unlabeled+Labeled						
	RBoost	RSVM	LNet	ARank	λMart	T-KPCA	SS$-\lambda$Rank
MQ2007-semi	.457	.456	.458	.456	.459	.451	.455
MQ2008-semi	.240	.241	.242	.242	.245	.232	.236
MSLR-WEB10K	.404	.402	.408	.407	.409	.396	.403

TABLE VI
NDCG@10 NUMBERS FOR SEMI-SUPERVISED L2R APPROACHES ON MSLR-WEB10K DATASET. UNDERLINED RESULTS ARE STATISTICALLY SUPERIOR THAN THE RESULTS OBTAINED BY T-KPCA AND SS$-\lambda$RANK.

Dataset	Label Propagation						
	RBoost	RSVM	LNet	ARank	λMart	T-KPCA	SS$-\lambda$Rank
MQ2007-semi	.471	.469	.472	.468	.471	.451	.455
MQ2008-semi	.258	.259	.261	.260	.261	.232	.236
MSLR-WEB10K	.416	.416	.419	.418	.419	.396	.403

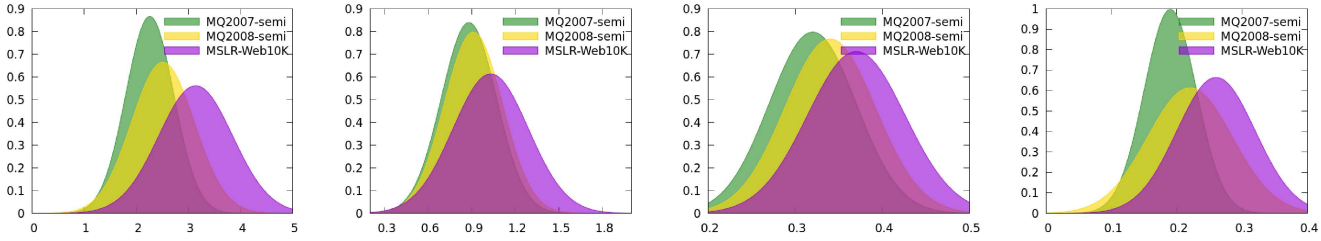
D. T-KPCA and SS$-\lambda$Rank

In the last set of experiments we present the comparison of our semi-supervised L2R approaches with T-KPCA and SS$-\lambda$Rank in terms of ranking performance. The results reported correspond to the average of the NDCG@10 numbers associated with the cases for which significant improvement was observed for each L2R algorithm. Table IV shows NDCG@10 numbers for "Labeled Only". In this case, no L2R algorithm was able to significantly surpass the ranking performance of T-KPCA and SS$-\lambda$Rank. Still, the ranking performance of the L2R algorithms were not significantly smaller when compared with T-KPCA and SS$-\lambda$Rank. Table V shows NDCG@10 numbers for "Unlabeled +Labeled." In this case, $\lambda-$Mart was able to achieve a ranking performance which is significantly superior than the performance of T-KPCA and SS$-\lambda$Rank. Table VI shows NDCG@10 numbers for "Label Propagation". In this case, the ranking performance of all L2R algorithms were significantly superior than the ranking performance of T-KPCA and SS$-\lambda$Rank. For MQ2007-semi,



(a) Depth. (b) Dimension. (c) Dropout. (d) Noise level.

Fig. 4. (Color online) Autoencoder configurations for which higher NDCG@10 numbers are more likely to be found. We consider feature transformations with "Label Propagation" and RankBoost as the L2R algorithm.



(a) Depth.. (b) Dimension. (c) Dropout. (d) Noise level.

Fig. 5. (Color online) Autoencoder configurations for which higher NDCG@10 numbers are more likely to be found. We consider feature transformations with "Label Propagation" and ListNet as the L2R algorithm.

gains range from 2.8% to 4.6%. For MQ2008-semi, gains range from 9.3% to 12.6%. Finally, for MSLR-Web10K, gains range from 3.2% to 5.7%. Gains over T-KPCA are more impressive, and are due to label propagation, since additional labels are included into the training set.

VI. Conclusions

Many researchers have found that the way in which data are represented can make a huge difference in the success of a learning algorithm. In this paper we focused on L2R algorithms, and we showed how deep autoencoder features can boost the ranking performance of these algorithms. Specifically, we considered a semi-supervised L2R scenario in which deep autoencoder features are found, and then the original input data are projected into a new feature space where relevance information is more evident. In addition to evaluating typical approaches for learning deep autoencoder features, we also proposed a more sophisticate approach which exploits labeled and unlabeled data simultaneously in the same batch of observations. This enables label propagation, gradually augmenting the training set with additional labels. Our experiments considered a large variety of autoencoder configurations, and we showed that using autoencoder features while performing label propagation leads to improved ranking models in $+10\%$ of the cases for different L2R algorithms. Lastly, we showed that using autoencoder features while performing label propagation leads to superior ranking performance when compared with semi-supervised approaches.

Acknowledgements

This work was partially funded by projects InWeb (grant MCT/CNPq 573871/2008-6) and MASWeb (grant FAPEMIG/PRONEX APQ-01400-14), and by the authors individual grants from CNPq, FAPEMIG and Kunumi.

References

[1] R. A. Baeza-Yates and B. A. Ribeiro-Neto, *Modern Inf. Retrieval - the concepts and technology behind search, Second edition*, 2011.

[2] Z. Cao, T. Qin, T. Liu, M. Tsai, and H. Li, "Learning to rank: from pairwise approach to listwise approach," in *Proc. of the 24th Intl. Conf. on Machine Learning*, 2007, pp. 129–136.

[3] C. J. C. Burges, R. Ragno, and Q. V. Le, "Learning to rank with nonsmooth cost functions," in *Proc. of the 20th Conf. on Neural Inf. Processing Systems*, 2006, pp. 193–200.

[4] S. Niu, Y. Lan, J. Guo, X. Cheng, and X. Geng, "What makes data robust: a data analysis in learning to rank," in *Proc. of 37th ACM SIGIR Conf. on Research and Development in Inf. Ret.*, 2014, pp. 1191–1194.

[5] A. Kumar and M. Lease, "Learning to rank from a noisy crowd," in *Proc. of the 34th ACM SIGIR Conf. on Research and Development in Inf. Retrieval*, 2011, pp. 1221–1222.

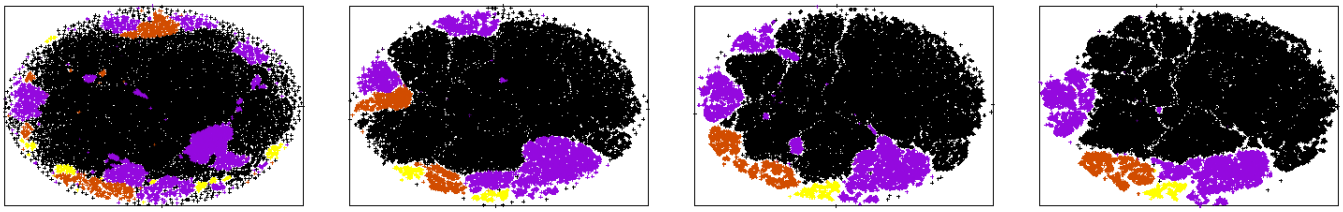
[6] J. Xu, C. Chen, G. Xu, H. Li, and E. R. T. Abib, "Improving quality of training data for learning to rank using click-through data," in *Proc. of the 3rd Conf. on Web Search and Data Mining*, 2010, pp. 171–180.

[7] Y. Bengio, A. C. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013.

[8] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[9] J. Xie, R. B. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. of the 33rd Intl. Conf. on Machine Learning*, 2016, pp. 478–487.

[10] T.-Y. Liu, "Semi-supervised ranking," in *Learning to Rank for Inf. Retrieval*, 2011, pp. 123–126.



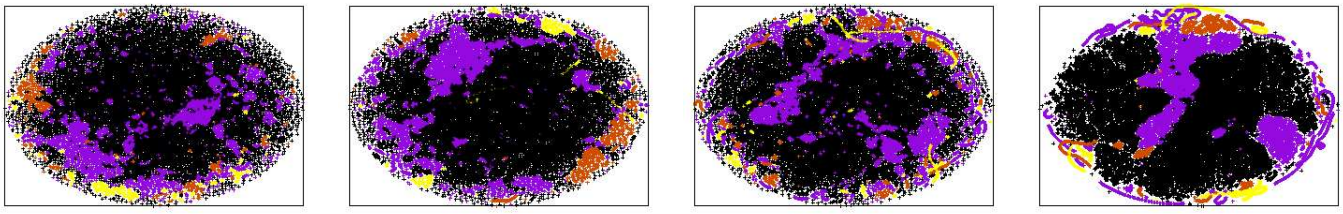
(a) Points represent query-document pairs.

(b) Representation obtained with unlabeled data only.

(c) Representation obtained with labeled and unlabeled data.

(d) Representation obtained with label propagation.

Fig. 6. (Color online) Input representations for MQ2007-semi. Black points — unlabeled documents. Purple — irrelevant documents. Orange — relevant documents. Yellow — very relevant documents.



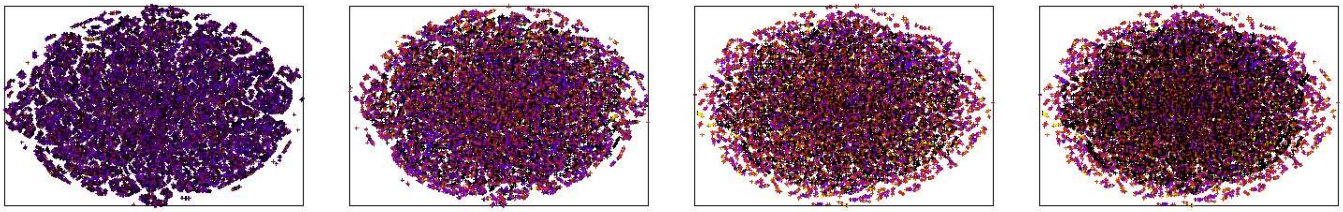
(a) Points represent query-document pairs.

(b) Representation obtained with unlabeled data only.

(c) Representation obtained with labeled and unlabeled data.

(d) Representation obtained with label propagation.

Fig. 7. (Color online) Input representations for MQ2008-semi. Black points — unlabeled documents. Purple — irrelevant documents. Orange — relevant documents. Yellow — very relevant documents.



(a) Points represent query-document pairs.

(b) Representation obtained with unlabeled data only.

(c) Representation obtained with labeled and unlabeled data.

(d) Representation obtained with label propagation.

Fig. 8. (Color online) Representations for MSLR-Web10K dataset. Black points — unlabeled documents. Relevance levels: purple, blue, red, orange, yellow.

[11] R. M. Silva, M. A. Gonçalves, and A. Veloso, "Rule-based active sampling for learning to rank," in *Proc. of the 2011 European Conf. on Machine Learning and Knowledge Discovery in Databases, Part III*, 2011, pp. 240–255.

[12] R. L. de Oliveira Jr., A. Veloso, A. Pereira, W. M. Jr., R. Ferreira, and S. Parthasarathy, "Economically-efficient sentiment stream analysis," in *Proc. of the 37th ACM SIGIR Conf. on Research and Development in Inf. Retrieval*, 2014, pp. 637–646.

[13] M. Szummer and E. Yilmaz, "Semi-supervised learning to rank with preference regularization," in *Proc. of the 20th ACM Conf. on Inf. and Knowledge Management*, 2011, pp. 269–278.

[14] S. C. H. Hoi and R. Jin, "Semi-supervised ensemble ranking," in *Proc. of the 23rd AAAI Conf. on Artificial Intelligence*, 2008, pp. 634–639.

[15] A. Veloso, H. de Almeida, M. Gonçalves, and W. M. Jr., "Learning to rank at query-time using association rules," in *Proc. of 31st ACM SIGIR Conf. on Research and Development in Inf. Ret.*, 2008, pp. 267–274.

[16] D. Erhan, A. C. Courville, Y. Bengio, and P. Vincent, "Why does unsupervised pre-training help deep learning?" in *Proc. of the 13th Intl. Conf. on Artificial Intelligence and Statistics*, 2010, pp. 201–208.

[17] X. Geng, T. Liu, T. Qin, and H. Li, "Feature selection for ranking," in *Proc. of the 30th ACM SIGIR Conf. on Research and Development in Inf. Retrieval*, 2007, pp. 407–414.

[18] H. Lai, Y. Pan, Y. Tang, and R. Yu, "FSMRank: Feature selection algorithm for learning to rank," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 24, no. 6, pp. 940–952, 2013.

[19] K. Duh and K. Kirchhoff, "Learning to rank with partially-labeled data," in *Proc. of the 31st ACM SIGIR Conf. on Research and Development in Inf. Retrieval*, 2008, pp. 251–258.

[20] B. Schölkopf, A. J. Smola, and K. Müller, "Kernel principal component analysis," in *Proc. of the 7th Intl. Conf. on Artificial Neural Networks*, 1997, pp. 583–588.

[21] Y. Pan, H. Luo, H. Qi, and Y. Tang, "Transductive learning to rank using association rules," *Expert Syst. Appl.*, vol. 38, pp. 12 839–12 844, 2011.

[22] Q. Wu, C. J. C. Burges, K. M. Svore, and J. Gao, "Adapting boosting for information retrieval measures," *Inf. Retr.*, vol. 13, pp. 254–270, 2010.

[23] T. Joachims, "Optimizing search engines using clickthrough data," in *Proc. of the 8th ACM SIGKDD International Conf. on Knowledge Discovery and Data Mining*, 2002, pp. 133–142.

[24] Y. Freund, R. D. Iyer, R. E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," *Journal of Machine Learning Research*, vol. 4, pp. 933–969, 2003.

[25] J. Xu and H. Li, "Adarank: a boosting algorithm for information retrieval," in *Proc. of the 30th ACM SIGIR Conf. on Research and Development in Inf. Retrieval*, 2007, pp. 391–398.

[26] Y. Wang, L. Wang, Y. Li, D. He, and T. Liu, "A theoretical analysis of NDCG type ranking measures," in *Proc. of the 26th Annual Conf. on Learning Theory*, 2013, pp. 25–54.

[27] T. Sakai, "Statistical reform in information retrieval?" *SIGIR Forum*, vol. 48, no. 1, pp. 3–12, 2014.