CS410 Technology Review – Learning to Rank

Greg Harrison (gdh3)

Learning to rank is a topic related to producing a function which will compare features and rank them by some criteria. Ranking algorithms are used in many applications within the field of information retrieval, natural language processing, and data mining [1]. Examples include document retrieval, collaborative filtering, and sentiment analysis. In these applications, learning to rank is used to create a list of documents ranked by how relevant they are to a query. Modern algorithms typically use supervised machine learning techniques to perform the task of ranking.

In learning to rank, feature vectors act as the input data to a model. A feature vector is created for each query and document pair. There are different methods for determining the values of these feature vectors. Additionally, to train a supervised machine learning model, the feature vectors for training are labeled based on the relevance of the document to the query in the document-query pairing for that feature. These labels act as the ground truth and there are different methods for obtaining them, for example, labels may be obtained by having humans explicitly score how relevant documents are to a given query; alternatively, labels may be obtained implicitly by tracking the number of times humans click on document [2]. When these scores are sorted, they create the ground truth ranking of the document query pairs, i.e., feature vectors [1]. A loss function is used to quantifies the difference between the ranking that the model outputs and the ground truth ranking. In learning to rank, the goal in training a model is to minimize this loss function.

Ranking algorithms have been categorized into pointwise, pairwise, and listwise approaches. The difference in these approaches is primarily based on the loss function used and how the feature vectors are treated within the loss function.

Pointwise Approach:

The pointwise approach looks at each feature individually, without consideration for the other features. The goal of a model using this approach is to produce a score for each feature. A ranked list is then created by sorting the features based on the score computed by the model. The loss function in the pointwise approach tries to minimize the difference between the score output from the model and the ground truth score from the label. Many standard regression models can be used to do this.

Pairwise Approach:

In the pairwise approach, the data is transformed by taking pairs of feature vectors and combining them to create a new vector for each pair of feature vectors. A new label for each of these new vectors is creature by computing some value for the relative relevance of that pair.

For example, the new vector could be the difference vector of the two pairs, and the new label could be the difference of the two labels associated with those pairs. These new vectors and labels are then used to train a model, where the goal is to create a model that will minimize the number of pairs that are not in the correct order when compared to the ground truth ranking [2]. This type of approach is used in models such as Ranking SVM, IR SVM, RankBoost, and RankNet [1].

For information retrieval, the pairwise approach typically produces better results than the pointwise approach. The pairwise approach is, however, more computationally expensive, because the total number of pairs to be evaluated can be significant. Additionally, models that use the pairwise approach have been found to be bias towards queries that have more document pairs. This is because the number of document pairs associated with each query tend to be significantly different from query to query [2].

Listwise Approach:

In the listwise approach, permutation of the feature vectors in entire ranked lists are used to train a model. Since the listwise approach uses complete lists, standard ranking evaluation metrics like mean average precision (MAP) and normalized discounted cumulative gain (NDCG) can be used in the loss function. For each permutation of a ranked list, a score can be produced using one of these evaluation metrics. The loss function will take the difference between this score and the score for the perfect permutation, which is usually 1. As list permutations become closer to the perfect permutation, i.e., the ideal ranking, the evaluation metric is maximized, and the loss function is minimized [2].

The main issue with this approach is that a function which scores every computation is O(n!) where n is the number of features in a permutation. Therefore, more efficient listwise implementations are typically used. The top one probability method is one of such approaches which is used in the ListNet algorithm [2].

Algorithms that use the listwise approach include ListNet, ListMLE, AdaRank, SVM MAP, and SoftRank [1]. These algorithms tend to outperform both the pairwise and pointwise approaches; although, when there are only two documents in a list, the pairwise and listwise approaches produce the same results. The ListNet algorithm has time complexity $O(m * n_{max})$ where m is the number of queries used in training and n_{max} is the number of documents for the query that has the largest number of associated documents. This is more efficient and has been found to produce better results than the pairwise RankNet algorithm, which has complexity $O(m * n_{max}^2)$ [2].

Conclusion:

The pointwise, pairwise, and listwise approaches to learning to rank are distinguished from each other based on the loss functions used in their computations. Pointwise approach uses a

single feature in its loss function, the pairwise approach uses pairs of features, and listwise approach uses an entire list. In general, the listwise approach produces the best results of the three.

References:

[1] Hang Li. A Short Introduction to Learning to Rank, IEICE Trans. Inf. & Syst. E94-D, 10 (Oct. 2011): n.p.

[2] Cao, Zhe & Qin, Tao & Liu, Tie-Yan & Tsai, Ming-Feng & Li, Hang. (2007). Learning to Rank: From Pairwise Approach to Listwise Approach. Proceedings of the 24th International Conference on Machine Learning. 227. 129-136. 10.1145/1273496.1273513.