Pairwise Learning to Rank Approach using LambdaRank

Angelia Regina Ginting
Fakultas Informatika dan Teknik Elektro
Institut Teknologi Del
Medan, Indonesia
gintingangelia@gmail.com

Rezky Prayitno Simanjuntak
Fakultas Informatika dan Teknik Elektro
Institut Teknologi Del
Tarutung, Indonesia
rezkys1999@gmail.com

Mega Christy Silalahi
Fakultas Informatika dan Teknik Elektro
Institut Teknologi Del
Pematang Siantar, Indonesia
christysilalahi05@gmail.com

Siti Berliana Manurung
Fakultas Informatika dan Teknik Elektro
Institut Teknologi Del
Lumban Julu, Indonesia
sitiberlianamanurung@gmail.com

Abstrak — *Learning to Rank* merupakan salah satu masalah *machine learning* yang tujuannya adalah untuk membangun suatu model perangkingan dari data pembelajaran sehingga diperoleh urutan yang memiliki relevansi dan preferensi yang optimal. Pada *paper* ini akan menjelaskan tentang penyelesaian masalah *learning to rank* dengan menggunakan *dataset LETOR* 4.0 *MQ2008* dengan menggunakan pendekatan *pairwise* dan metode *Lambda Rank*. *Lambda Rank* pada IR (*Information Retrieval*) mampu memecahkan masalah dengan menentukan gradien dari biaya yang diberikan di tempat tujuan. *Lambda Rank* memperoleh hasil yang lebih baik saat memperkuat gradien dengan perubahan NDCG yang muncul akibat dari pertukaran dokumen.Hasil yang diperoleh pada paper ini adalah NDGC pada *fold* 1 sebesar 0.621774, untuk *fold* 2 sebesar 0.654353, untuk *fold* 3 sebesar 0.639651, untuk *fold* 4 sebesar 0.615033 dan *fold* 5 sebesar 0.59328. Hasil rata-rata NDGC dengan 5 *fold* pada paper ini adalah sebesar 0.624818

Kata Kunci—*Learning to rank, Lambda Rank, LETOR, NDGC*

I. PENDAHULUAN

Information retrieval adalah salah satu metode yang digunakan untuk mengambil data terstruktur yang tersimpan dalam koleksi dokumen, lalu menyediakan informasi yang diperlukan. *Information Retrieval* atau sistem temu balik informasi bertujuan untuk mengambil dan menampilkan dokumen yang relevan dengan pengguna *(query)* [1].

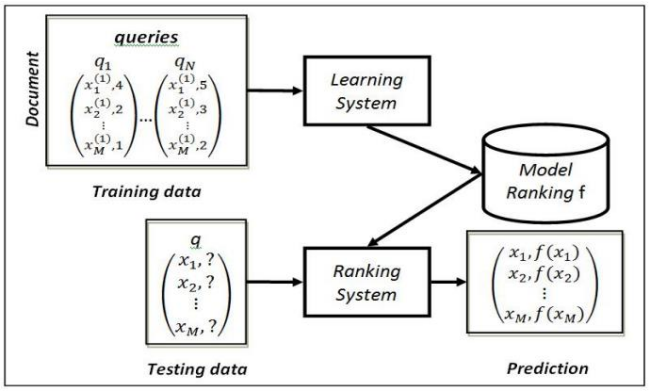
Informasi yang tersimpan dalam suatu gudang informasi dengan jumlah yang sangat banyak tentu akan memerlukan mekanisme agar informasi dapat dimunculkan kembali dan dapat bermanfaat bagi pengguna dan seiring berjalannya waktu informasi tersebut akan semakin banyak sehingga munculnya banjir informasi. Namun informasi dengan jumlah yang sangat banyak dapat diatasi dengan menggunakan sebuah mekanisme agar mendapatkan kembali informasi yang sesuai dengan kebutuhan pengguna dan salah satu caranya adalah dengan melakukan *Ranking*. *Ranking* adalah bagian penting dari masalah pencarian informasi seperti pengambilan dokumen, penyaringan informasi, penempatan iklan *online*, dan lain sebagainya. Contoh aplikasi Ranking yang tidak asing lagi bagi kita adalah terdapat pada *search engine* yaitu aplikasi *Google* dan *Yahoo*.

Hingga saat ini sudah banyak metode yang dikembangkan untuk melakukan ranking namun masih banyak hasil yang diperoleh belum optimal karena dokumen-dokumen yang ditampilkan belum tentu merupakan dokumen yang dibutuhkan oleh pengguna. Namun untuk menyelesaikan permasalahan *ranking* tersebut dapat dilakukan dengan *machine learning*. *Machine Learning* merupakan salah satu metode atau dapat disebut sebagai model yang dapat belajar dari data historis sehingga menjadi cerdas atau memiliki kemampuan untuk menggeneralisasi data baru yang belum dipelajari sebelumnya misalnya untuk memprediksi, mengklasifikasi, ranking dan lain lain. Hadirnya *machine learning* membuat orang mulai mencoba untuk mempelajari pola pengguna dalam menentukan relevansi suatu *web* terhadap suatu *query* yang diberikan atau dapat disebut *"Learning to Rank"*. *Learning to Rank* merupakan salah satu masalah machine learning yang tujuannya adalah untuk membangun suatu model perangkingan dari data pembelajaran sehingga diperoleh urutan yang memiliki relevansi dan preferensi yang optimal [2]. Pada paper ini akan menjelaskan tentang penyelesaian masalah *learning to rank* dengan menggunakan *dataset LETOR MQ2008* dan peneliti mengusulkan menggunakan pendekatan *Pairwise* dengan *metode LambdaRank.*

II. METODOLOGI

A. Learning to Rank

Learning to Rank merupakan sebuah metode yang menggunakan teknologi *machine learning*, yang bertujuan untuk membangun model ranking dari data pembelajaran, sehingga menghasilkan urutan yang memiliki relevansi dan preferensi yang optimal. *Learning to Rank* terdiri dari beberapa tahapan proses yaitu sebagai berikut:



Gambar 1 Skema *learning to rank*

(Sumber: Uji Kinerja *Learning to Rank dengan Metode Support Vector Regression*, 2015)

Dari gambar tersebut diketahui bahwa data training dan data testing pada *learning to rank* yaitu terdiri dari pasangan *query* q = {qi}i=1,2,...,N serta data yang terdiri dari vektor *feature* x = {xj}j=1,2,...M dan label relevansi. Vektor *feature* adalah sebuah kombinasi dari beberapa metode ranking konvensional untuk menjadi elemen-elemen pada setiap vektor. Pada umumnya tahapan pada *learning to rank* terdiri dari dua bagian utama yaitu *learning system* dan *ranking system. Learning system* dilakukan untuk melakukan pelatihan dengan menggunakan suatu metode *ranking* terhadap data training, yang bertujuan untuk mendapatkan model *ranking* yang paling optimal, yang berarti mendapatkan hasil akurasi ranking paling tinggi. Kemudian untuk tahap selanjutnya yaitu model *ranking* akan dilakukan pengujian pada data testing dengan *query* baru, yang bertujuan untuk memperoleh hasil urutan sebagai *output* dari pemilihan *query* yang baru tersebut. *Learning to Rank* dapat dilakukan dengan menggunakan beberapa metode ranking konvensional yaitu *Vector Space Model* (VSM), *Term Frequency* (TF), *Pagerank*, *Language Model for Information Retrieval* (LMIR), *Boolean Model* (BM25), dan lain sebagainya. *Learning to Rank* dapat diklasifikasikan menjadi beberapa model yaitu *pointwise models*, *pairwise models*, dan *listwise models*. Pada model-model tersebut *training* data yang digunakan yaitu dokumen individual (*individual documents*), pasangan dokumen (*document pairs*), dan daftar dokumen (*document lists*). Pada model tersebut menggunakan teknologi klasifikasi dan regresi yang digunakan untuk menyesuaikan parameter serta meningkatkan presisi peringkat [2].

B. Pairwise

Pairwise dilatih oleh setiap pasangan dokumen. Pada setiap pasangan dokumen, dua dokumen diberikan dua skor relevansi oleh setia orang. Setiap dokumen pada *Pairwise* akan diberi peringkat dengan mempertimbangkan prioritas utama antara pasangan dari kueri dan dokumen. Model *Pairwise* memiliki keuntungan yaitu dapat digunakan secara langsung mendukung penerapan teori dan algoritma yang ada pada klasifikasi dan juga pelatihan pasangan dokumen dapat dengan mudah diperoleh dalam skenario tertentu. Sedangkan untuk kerugian dalam menggunakan model *pairwise* yaitu *loss function* dinilai dari pasangan satu dokumen ke dokumen lain dan bukan secara keseluruhan, sehingga sulit untuk menentukan daftar rankingnya. Pendekatan Pairwise berfokus untuk memaksimalkan jumlah pasangan dokumen yang benar. Untuk dapat memiliki model ranking yang dapat mengklasifikasikan semua pasangan dengan benar berarti model tersebut sudah dapat melakukan ranking untuk semua dokumen yang benar [3].

C. LambdaRank

Lambda MART merupakan *boosted tree* (versi pohon yang ditingkatkan) dari LambdaRank yang mengambil dasarnya pada RankNet. RankNet, LambdaRank, dan LambdaMART telah terbukti menjadi algoritma yang berhasil memecahkan masalah *real world ranking* [4].

Pada sebagian besar tugas pembelajaran mesin (*machine learning tasks*), *target cost* digunakan untuk menilai keakuratan sistem pada waktu pengujian dan biaya pengoptimalan, dimana biasanya perkiraan untuk *target cost* digunakan untuk melatih (*train*) sistem. Jenis-jenis dari *target cost* yaitu berupa MAP, MRR, Mean NDCG, dan lain-lain. Lambda Rank pada IR (*Information Retrieval*) mampu memecahkan masalah dengan menentukan gradien dari biaya yang diberikan di tempat tujuan. Pada awalnya, gradien LambdaRank didefinisikan untuk *Mean* NDCG, dimana gradien ini didefinisikan dengan menetapkan aturan tentang bagaimana menukar dua dokumen setelah mengurutkannya berdasarkan skor untuk kueri tertentu, dalam hal ini biaya berubah [5].

Selama prosedur pelatihan dari *original* RankNet, ditemukan bahwa penghitungan biaya tidak diperlukan, sebaliknya gradien dari biaya cukup untuk menentukan peringkat yang diprediksi untuk pasangan item. Hal ini dapat dilihat melalui gambar dibawah dengan visualisasi panah yang menunjukkan arah pergerakan item tertentu.

Gambar 2 Visualisasi gradien biaya dengan panah yang menunjukkan pergerakan

(Sumber: From RankNet to LambdaRank to LambdaMART: An Overview)

Dalam hal ini, Lambda Rank memperoleh hasil yang lebih baik saat memperkuat gradien dengan perubahan NDCG yang muncul akibat dari pertukaran dokumen. Lambda Rank terbukti lebih cepat dalam pelatihan (*train*) sistem, ditambah dengan peningkatan dalam akurasi [6].

Salah satu *key observation* atau kunci pengamatan dari Lambda Rank bahwa untuk melatih model atau *train* tidak perlu melakukan biaya sendiri, cukup hanya memerlukan gradien atau biaya *(cost)* yang terkait dengan model score. pada Lambda Rank simbol atau panah (λ's) disebut sebagai gradien. λ's akan memberikan URL tertentu atau U1 untuk mendapatkan kontribusi dari semua URL lain untuk kueri yang sama dan memiliki label yang berbeda. λ's dapat juga diartikan sebagai gaya *(forces)* dimana gradien berfungsi potensial jika gaya konservatif. Jika U2 lebih relevan dari U1 maka U1 akan mendapat dorongan berdasarkan ukuran atau |λ| dan U2 akan terdorong ke atas. Jika U2 kurang relevan dari U1 maka U1 akan mendapat dorongan ke atas berdasarkan ukuran atau |λ| dan sebaliknya U2 akan berlawanan dan akan terdorong ke bawah. Persamaan pada Lambda Rank adalah sebagai berikut [4]:

$$\lambda_{ij} = \frac{\partial C(s_i - s_j)}{\partial s_i} = \frac{-\sigma}{|1 + e^{\sigma(s_i - s_j)}|} |\Delta_{NDCG}|$$

Dimana pada persamaan diatas dilakukan dengan mengalikan ukuran hasil perubahan dalam NDCG atau (|ΔNDCG| dengan menukar posisi peringkat U1 dan U2 atau dengan membiarkan posisi peringkat dari semua URL lainnya tidak berubah [4].

III. EKSPERIMEN

A. The LETOR Benchmark Datasets

Dataset yang digunakan pada percobaan ini yaitu dataset LETOR 4.0 MQ2008. LETOR adalah sebuah kumpulan data untuk *learning to rank* yang bersifat umum atau *open source* dan diorganisasi oleh *Microsoft.* Million Query 2008 (MQ 2008) adalah set data benchmark dari dataset LETOR 4.0. MQ 2008 dibuat oleh Microsoft Research Asia dan datanya berasal dari Million Query track TREC 2008. Dataset terdiri dari 784 queries dari Gov2 web page. Gov2 merupakan koleksi yang terdiri dari 25 juta halaman web yang didistribusikan oleh *University of Glasgow*. Untuk setiap pasangan kueri atau dokumen diwakili oleh feature vector 46 dimensi. Terdapat 15211 pasangan kueri atau dokumen pada dataset MQ2008. Desain eksperimen yang digunakan yaitu *five-fold cross validation*, yaitu *dataset* MQ2008 akan dibagi menjadi lima bagian untuk setiap bagian dengan jumlah *query* yang sama. lima bagian tersebut yang telah dibagi akan dituliskan sebagai S1, S2 S3, S4, S5, kemudian bagian tersebut akan disusun sebagai satu *fold*. Pada setiap *fold* tiga bagian akan digunakan untuk melatih model *ranking*, kemudian satu bagian akan digunakan untuk melakukan validasi model *ranking* dan untuk sisanya akan digunakan untuk mengevaluasi model *ranking*. Dari lima *fold* yang digunakan akan menghasilkan hasil rata-rata akurasi yang akan digunakan sebagai akurasi *learning to rank* [2].

B. Evaluation Measures

Evaluasi yang digunakan pada percobaan ini yaitu dengan menggunakan NDCG. *Normalized Discounted Cumulative Gain at Position* (NDCG) adalah sebuah ukuran evaluasi pada n dokumen teratas yang berasal dari hasil perangkingan dengan menggunakan lebih dari dua label relevansi [2].Berikut merupakan formula yang digunakan pada NDCG yaitu sebagai berikut:

$$N(n) = Z_n \sum_{j=1}^{n} \frac{2^{r(j)} - 1}{\log_2(1 + j)}$$

C. Experimental Settings

Untuk mengimplementasikan percobaan ini maka digunakan bahasa pemrograman python dan untuk menguji hasil percobaan dilakukan dengan menggunakan metrik NDGC. Metode yang digunakan adalah dengan menggunakan Lambda Rank. Dalam percobaan ini dilakukan percobaan dengan dataset LETOR 4.0 MQ2008. Setiap proses dilakukan *5-fold cross-validation*. Parameter Lambda Rank yang digunakan untuk mengimplementasikan percobaan ini dapat dilihat pada tabel berikut.

Tabel 1 Parameter LambdaRank yang digunakan untuk implementasi

Parameter	Value
Input size	Integer
Hidden layers size	100
Activation	relu
solver	adam

D. Results

Metrik yang digunakan untuk mengevaluasi perangkingan ini yaitu menggunakan metrik NDCG dengan menggunakan *Fold* 1 hingga *Fold* 5, dimana dapat dilihat pada tabel dibawah ini.

Tabel 2 Hasil Evaluasi Metrik NDCG

Fold	Hasil NDCG
1	0.621774
2	0.654353
3	0.639651
4	0.615033
5	0.59328
NDGC Average	**0.624818**

Berdasarkan hasil yang diperoleh dari tabel diatas, dapat dilihat bahwa NDCG dengan *Fold* 2 memiliki skor NDCG yang lebih tinggi yaitu sebesar 0.654353 dan NDCG dengan *Fold* 5 memiliki skor NDCG yang lebih rendah yaitu sebesar 0.59328 dan berdasarkan hasil NDGC dari kelima *fold* tersebut maka diperoleh nilai rata-rata *(Average)* NDGC adalah sebesar **0.624818.**

IV. KESIMPULAN

Kesimpulan yang didapatkan berdasarkan percobaan ini yaitu pada percobaan ini dilakukan perankingan terhadap dataset LETOR 4.0 MQ2008 yang terdiri dari *Fold* 1 hingga *Fold* 5 dengan menggunakan metrik evaluasi NDCG pada kelima *Fold* tersebut. Perangkingan yang dilakukan pada percobaan ini yaitu menggunakan pendekatan *pairwise* dengan metode LambdaRank. Dari hasil yang sudah diperoleh, dapat dilihat bahwa nilai rata-rata (*average*) NDCG yang diperoleh dari kelima *Fold* tersebut sebesar 0.624818.

REFERENCES

[1] H. R. F. R. Wina Witanti, "PEMBANGUNAN SISTEM TEMU BALIK INFORMASI (INFORMATION RETRIEVAL) DALAM PEMILIHAN PEMAIN SEPAK BOLA BERKUALITASDI INDONESIA BERBASIS ANALISIS SENTIMEN," *Seminar Nasional Teknologi Informasi dan Komunikasi*, pp. 1-2, 2016.

[2] H. M. ABDUL AZIS ABDILLAH, "UJI KINERJA LEARNING TO RANK DENGAN METODE SUPPORT VECTOR REGRESSION," *IndoMS Journal on Industrial and Applied Mathematics*, vol. 2, no. 1, p. 15, 2015.

[3] X. Dong, "An Overview of Learning to Rank for Information Retrieval," *2009 World Congress on Computer Science and Information Engineering*, pp. 602-603, 2009.

[4] C. J.C, "From RankNet to LambdaRank toLambdaMART," *BurgesMicrosoft Research Technical Report*, pp. 8-10, 2010.

[5] P. Donmez, K. M.Svore and C. J. C. Burges, "On the Local Optimality of LambdaRank," pp. 460-467, 2009.

[6] D. Goenka, "Learning to Rank for Information Retrieval: A Deep Dive into RankNet," towards data science, 26 September 2020. [Online]. Available: https://towardsdatascience.com/learning-to-rank-for-information-retrieval-a-deep-dive-into-ranknet-200e799b52f4. [Accessed 19 Mei 2021].