Pairwise Learning to Rank Approach using Light Gradient Boosting Machine (LightGBM)

Tripheni Simanjuntak
Fakultas Informatika dan Teknik Elektro
Institut Teknologi Del
Balige, Indonesia
triphenisimanjuntak2017@gmail.com

Mika Lestari Valentina Manurung
Fakultas Informatika dan Teknik Elektro
Institut Teknologi Del
Lumban Julu, Indonesia
lestarimika07@gmail.com

Jessycha Royanti Tampubolon
Fakultas Informatika dan Teknik Elektro
Institut Teknologi Del
Balige, Indonesia
jessycha.royanti@gmail.com

Nita Sophia Winandi Sirait
Fakultas Informatika dan Teknik Elektro
Institut Teknologi Del
Lubuk Pakam, Indonesia
nitasophia05@gmail.com

***Abstrak* — *Paper* ini berkaitan dengan *learning to rank*, yaitu membangun model atau fungsi untuk menentukan peringkat suatu objek. *Learning to rank* diterapkan pada banyak aplikasi seperti *document retrieval* dan *collaborative filtering*. Salah satu pendekatan yang ada pada *learning to rank* adalah *pairwise*. Pada *pairwise*, dokumen diproses secara berpasang-pasangan. Metode yang diajukan adalah *Pairwise Approach* menggunakan *Light Gradient boosting Machine* (LightGBM) yang merupakan salah satu algoritma *Gradient Boosting*. *Gradient Boosting Decision Tree* (GBDT) adalah algoritma *machine learning* yang populer, dan memiliki beberapa implementasi yang efektif seperti XGBoost dan pGBRT. Meskipun teknik pengoptimalan telah banyak diadopsi dalam implementasi ini, efisiensi dan skalabilitasnya masih belum memuaskan saat *feature dimension* tinggi dan ukuran datanya besar. Alasan utamanya adalah untuk setiap fitur, mereka perlu memindai semua *sample data* untuk memperkirakan perolehan informasi dari semua kemungkinan titik pemisahan, yang sangat memakan waktu. Untuk mengatasi masalah ini, kami mengusulkan teknik baru yaitu LightGBM. Eksperimen kami pada beberapa kumpulan data publik menunjukkan bahwa LightGBM dapat memberikan hasil yang maksimal pada *learning to rank* dengan pendekatan *pairwise*.**

Kata kunci* — *Learning to rank, Pairwise, LightGBM

I. PENDAHULUAN

Learning to Rank (LTR) untuk *Information Retrieval* (IR) adalah sebuah tugas dalam pembangunan model pemeringkatan menggunakan data pelatihan, sehingga model tersebut dapat mengurutkan objek baru sesuai dengan relevansi, preferensi, atau kepentingannya [1]. Pemeringkatan adalah masalah utama di IR seperti *document retrieval, collaborative filtering, key term extraction, definition finding, important email routing, sentiment analysis, product rating,* dan *anti web spam.* Teknik LTR digunakan untuk meningkatkan teknologi IR. Secara khusus, terdapat beberapa pendekatan yang dapat digunakan untuk membangun model LTR sesuai dengan *loss functions* yang digunakan yaitu *pointwise approach* yang diterapkan ke setiap dokumen secara individual dan menganggapnya terpisah dari data latih lainnya, *pairwise approach* yang menghitung *priority* setiap kemungkinan pasangan kueri dan dokumen lalu mengurutkannya, dan *listwise approach* yang membandingkan skor setiap pasangan kueri dan dokumen pada list rank yang telah diperoleh.

Pairwise approach dikembangkan dan berhasil diterapkan untuk *document retrieval*. Pendekatan ini mengambil pasangan dokumen sebagai contoh untuk pelatihan, dan menyusunnya ke LTR sebagai klasifikasi. Terdapat keuntungan pendekatan *pairwise* yaitu metodologi yang ada pada klasifikasi dapat diterapkan secara langsung dan contoh pelatihan pasangan dokumen dapat dengan mudah diperoleh dalam skenario tertentu. *Pairwise* juga masih dapat bersaing dengan pendekatan *listwise* yang lebih baru dan jauh lebih kompleks. Pada pendekatan *pairwise*, apabila n adalah dokumen, maka n = 2 dimana model yang dibangun akan mengambil dua dokumen dan menentukan yang lebih relevan [2]

Kemudian kami menggunakan LightGBM yang merupakan singkatan dari *Light Gradient Boosting Machine* adalah *framework* peningkatan gradien yang didasarkan pada algoritma *decision tree* dan digunakan untuk peringkat, klasifikasi, dan tugas pembelajaran mesin lainnya. Fokus pengembangannya adalah pada kinerja dan skalabilitas. LightGBM memiliki banyak keunggulan seperti pengoptimalan *sparse, parallel training, multiple loss functions*, regularisasi, *bagging*, dan penghentian awal.

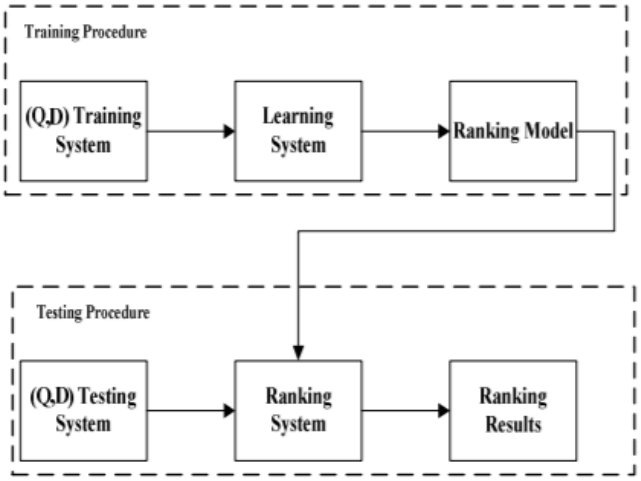
Kontribusi utama yang dijelaskan pada *paper* ini adalah bagaimana kami menyajikan performa model LTR yang dibangun menggunakan pendekatan *pairwise* dan menggunakan LightGBM dan menunjukkan bahwa model kami dan hasil eksperimen yang dilakukan pada set data LETOR MQ2008 cukup baik. *Paper* ini terdiri dari Bagian 1 Pendahuluan, Bagian 2 Metodologi, Bagian 3 Eksperimen, Bagian 4 Hasil dan Ucapan Terima Kasih.

II. METODOLOGI

A. *Learning to Rank*

Fokus penelitian *learning to rank* adalah penggunaan metode pembelajaran mesin seperti metode klasifikasi dan regresi untuk memproses tugas pemeringkatan. *Learning* dan *ranking* adalah dua prosedur untuk membangun model *learning to rank.* Dalam prosedur pembelajaran atau *learning*, diberikan sekelompok kueri. *Ranking documents* diambil sebagai *training data*, dan kemudian model peringkat dibangun menggunakan *training data* untuk mendapatkan parameter model yang optimal. Setelah itu, model pemeringkatan dapat menetapkan skor yang mewakili relevansi antara dokumen dan kueri untuk menguji dokumen dan merangkingnya dalam urutan berdasarkan skor untuk kueri dalam prosedur pemeringkatan.

Learning to rank telah menjadi tugas penting dalam banyak aspek terkait melayani kebutuhan informasi pengguna seperti memberi peringkat rekomendasi produk berdasarkan riwayat pembelian sebelumnya, memberi peringkat iklan *online* sehubungan dengan konten di halaman web, dan peringkat artikel berita dalam urutan minat pengguna. Keseluruhan prosedur membangun model *learning to rank* dapat diilustrasikan pada Gambar 1 [2].



Gambar 1 Prosedur Membangun Model *Learning to Rank*

B. *Pairwise Approach*

Beberapa contoh model pemeringkatan yang menggunakan *pairwise* adalah RankNet, RankBoost, LamdaRank dan LambdaMart. Pada *pairwise*, dokumen diproses secara berpasang-pasangan. *Pairwise* memberi peringkat pada dokumen berdasarkan perbedaan skor relatif dan bukan karena dekat dengan label. *Loss* didasarkan pada pasangan dokumen dengan perbedaan relevansi. Pendekatan *pairwise*, melihat sepasang dokumen *loss function.* Pendekatan *pairwise* bekerja lebih baik dalam praktiknya daripada pendekatan pointwise karena pendekatan ini memprediksi urutan relatif lebih dekat dengan sifat peringkat daripada memprediksi label kelas atau skor relevansi. *Pairwise* dan *listwise* biasanya lebih baik daripada *pointwise* karena masalah utama pemeringkatan dalam pencarian adalah untuk menentukan urutan dokumen tetapi tidak untuk menilai relevansi dokumen, yang mana menjadi tujuan dari kedua pendekatan tersebut [3].

Kriteria umum pendekatan *pairwise* adalah fungsi peringkat f memberi peringkat pada pasangan item (yaitu untuk $\{\vec{x}_i, \vec{x}_j\}$, apakah y_i lebih besar dari y_j ?) dimana fungsi *learning* menggunakan *paired input vectors* $f(\vec{x}_i - \vec{x}_j)$. Diberikan peringkat (1) sebagai informasi pelatihan, pendekatan *pairwise* mengekstrak semua preferensi *pairwise* dan menganggap preferensi ini sebagai contoh untuk tugas klasifikasi biner [4].

C. LightGBM

LightGBM adalah *gradient boosting framework* yang menggunakan algoritma pembelajaran berbasis *tree*. LightGBM memiliki beberapa keuntungan, yaitu *training* yang lebih cepat dan efisien, penggunaan memori yang lebih rendah, akurasi yang lebih baik dan mampu menangani data skala besar. Parameter LightGBM dapat disetel baik dalam *command line* dan *config file.* LightGBM memungkinkan untuk menggunakan beberapa metrik evaluasi. Secara *default,* LightGBM akan memetakan file data ke memori dan memuat fitur dari memori sehingga pemuatan data akan lebih cepat. LightGBM akan menyimpan *dataset* ke file biner. LightGBM akan memuat file skor awal secara otomatis.LightGBM menggunakan file tambahan untuk menyimpan data kueri dan akan memuat file kueri secara otomatis.

LightGBM menyediakan implementasi *Gradient Boosted Trees* yang berkinerja baik. LightGBM menggunakan dua teknik yang disebut *Gradient-based OneSide Sampling* (GOSS) dan *Exclusive Feature Bundling* (EFB). GOSS adalah algoritma *tree splitting* yang mempertimbangkan semua contoh data dengan gradien besar dan contoh data sampel secara acak dengan gradien kecil, mencapai perolehan informasi yang serupa tetapi dengan ukuran data yang lebih kecil. EFB bekerja dengan mengurangi jumlah fitur dengan mengelompokkan fitur yang saling eksklusif. Berikut adalah algoritma GOSS pada LightGBM [5].

```
Input: I: training data, d: iterations
Input: a: sampling ratio of large
gradient data
Input: b: sampling ratio of small
gradient data
Input: loss: loss function, L: weak
learner
models ? {}, fact ? (1-a)/b
topN ? a × len(I), randN ? b × len(I)
for i = 1 to d do
    preds ? models.predict(I) g ?
loss(I, preds), w ? {1, 1, ...}
    sorted ? GetSortedIndices(abs(g))
    topSet ? sorted[1:topN]
```

```
    randSet ?
RandomPick(sorted[topN:len(I)],
    randN)
    usedSet ? topSet + randSet
    w[randSet] × = fact . Assign
weight f act to the
    small gradient data.
    newModel ? L(I[usedSet],
g[usedSet],
    w[usedSet])
    models.append(newModel)
```

Gambar 2 Algoritma GOSS pada LightGBM

Untuk set pelatihan dengan n *instance* $\{x_1, \cdots, x_n\}$, di mana setiap x_i adalah vektor dengan dimensi s di ruang X^s. Dalam setiap iterasi peningkatan gradien, gradien negatif dari fungsi kerugian sehubungan dengan keluaran model dilambangkan sebagai $\{g_1, \cdots, g_n\}$. . Dalam metode GOSS ini, *instance* pelatihan diberi peringkat menurut nilai absolut dari gradiennya dalam urutan menurun. Kemudian, *instance* a × 100% teratas dengan gradien yang lebih besar disimpan dan mendapatkan subset *instance* A. Kemudian, untuk set A^c yang tersisa yaitu (1-a) × 100% *instance* dengan gradien yang lebih kecil, secara acak akan diambil sampel subset B dengan ukuran b × $|A^c|$.

$$V_{j|o}(d) = \frac{1}{n_o}\frac{\left(\sum_{\{x_i \in o: x_{ij} \le d\}} g_i\right)^2}{n^j_{l|o}(d)} + \frac{\left(\sum_{\{x_i \in o: x_{ij} \le d\}} g_i\right)^2}{n^j_{r|o}(d)}$$

Dimana $A_l = \{x_i ? A : x_{ij} ? d\}$, $A_r = \{x_i ? A : x_{ij} > d\}$, $B_l = \{x_i ? B : x_{ij} ? d\}$, $B_r = \{x_i ? B : x_{ij} > d\}$, dan koefisien *(1-a)/b* digunakan untuk menormalkan jumlah gradien di atas B kembali ke ukuran A^c. Berikut adalah algoritma EFB pada LightGBM.

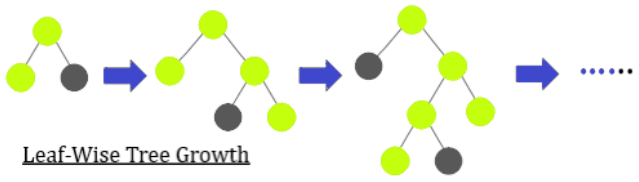
```
Input: numData: number of data
Input: F: One bundle of exclusive
features
binRanges ? {0}, totalBin ? 0
for f in F do
    totalBin += f.numBin
    binRanges.append(totalBin)
newBin ? new Bin(numData)
for i = 1 to numData do
    newBin[i] ? 0
    for j = 1 to len(F) do

        if F[j].bin[i] != 0 then
            newBin[i] ? F[j].bin[i] +
binRanges[j]
Output: newBin, binRanges
```

Gambar 3 Algoritma EFB pada LightGBM

EFB mengurangi dimensi untuk meningkatkan efisiensi sambil mempertahankan tingkat akurasi yang tinggi. Dengan EFB, maka dilakukan penggabungan fitur yang saling eksklusif, yaitu fitur yang jarang mengambil nilai bukan nol secara bersamaan, untuk mengurangi jumlah fitur. EFB menggunakan *greedy algorithm* untuk menggabungkan fitur yang saling eksklusif menjadi satu fitur dan dengan demikian mengurangi dimensionalitas. Algoritma EFB dapat menggabungkan banyak fitur eksklusif ke dense features yang jauh lebih sedikit, yang secara efektif dapat menghindari komputasi yang tidak perlu untuk nilai fitur nol. EFB dapat membantu mencapai percepatan yang signifikan pada kumpulan data skala besar [6]

High-dimensional data (data berdimensi tinggi) biasanya bersifat *sparse* sehingga memberi kemungkinan untuk merancang pendekatan tanpa kerugian untuk mengurangi jumlah fitur. Secara khusus, dalam ruang fitur yang *sparse*, banyak fitur yang saling eksklusif. Fitur eksklusif dapat digabungkan dengan aman ke dalam satu fitur (*Exclusive Feature Bundle*). Sehingga kompleksitas bangunan histogram berubah dari O (#data × #feature) menjadi O (#data × #bundle), sedangkan #bundle << # feature. Oleh karena itu, kecepatan *framework* pelatihan ditingkatkan tanpa mengurangi akurasi. Arsitektur LightGBM ditunjukkan pada gambar berikut.



Gambar 4 Arsitektur LightGBM

LightGBM membagi *tree leaf-wise* berlawanan dari algoritma *boosting* lainnya yang menggunakan *tree level-wise*. LightGBM memilih *leaf* dengan *delta loss* maksimum. Karena *leaf* yang dimiliki tetap, algoritma *leaf-wise* memiliki kerugian (*loss*) yang lebih rendah dibandingkan dengan algoritma *level-wise*. *Leaf-wise tree* dapat meningkatkan kompleksitas model dan dapat menyebabkan *overfitting* pada kumpulan data kecil.

III. EKSPERIMEN

Pada bagian ini, kami menyajikan hasil dari percobaan pada algoritma yang kami usulkan, yaitu

A. *Dataset*

Pada percobaan ini, *dataset* yang digunakan adalah *dataset* Benchmark LETOR. LETOR merupakan sebuah koleksi data yang dapat digunakan untuk percobaan *Learning to Rank* yang bersifat *open source* atau terbuka untuk umum dan diorganisasikan oleh microsoft. Lebih spesifik lagi, *dataset* LETOR yang digunakan pada percobaan ini adalah

LETOR 4.0 yaitu MQ2008. Desain eksperimen yang digunakan pada percobaan ini adalah dengan menggunakan *five-fold cross validation. Dataset* MQ2008 dibagi menjadi lima bagian dengan jumlah *query* yang sama untuk setiap bagian, dan setiap bagian dituliskan sebagai S1, S2, S3, S4, S5, dan kemudian menjadi satu *fold*. Untuk masing-masing *fold*, tiga bagian digunakan untuk melatih model *ranking (train set)*, satu bagian digunakan untuk memvalidasi model *ranking (valid set)*, dan yang lainnya digunakan untuk mengevaluasi model *ranking (test set)*, yang kemudian hasil rata-rata akurasi yang dicapai oleh lima *fold* tersebut akan digunakan sebagai akurasi *Learning to Rank* [7].

B. Evaluasi

DCG (*Normalized Discounted Cumulative Gain*) adalah ukuran kualitas peringkat. Dalam pencarian informasi, sering digunakan untuk mengukur efektivitas algoritma mesin pencari web atau aplikasi terkait. Terdapat dua asumsi dalam menggunakan DCG. Pertama, dokumen yang sangat relevan muncul lebih awal dan memiliki peringkat lebih tinggi. Kedua, dokumen yang sangat relevan lebih berguna daripada dokumen yang sedikit relevan. NDCG (*Normalized Discounted Cumulative Gain*) menghitung *Cumulative Gain* dari serangkaian hasil dengan menjumlahkan total relevansi setiap item dalam set hasil. Kemudian, posisi setiap item adalah *discounted*, artinya semakin rendah item yang relevan dalam daftar, semakin tinggi penalti atau *discount* yang diberikan item tersebut terhadap skor total. Jika dilakukan penghitungan DCG untuk kueri yang berbeda, akan ditemukan bahwa beberapa kueri lebih sulit daripada yang lain dan akan menghasilkan skor DCG yang lebih rendah daripada kueri yang lebih mudah. Maka, normalisasi dengan NDCG memecahkan masalah ini dengan menskalakan hasil berdasarkan hasil terbaik yang terlihat. Berikut merupakan penghitungan *Normalized Discounted Cumulative Gain* [8].

$$NDCG_n = \frac{DCG_n}{IDCG_n}$$

C. Rancangan Eksperimen

Dalam implementasinya, percobaan ini dilakukan dengan menggunakan bahasa pemrograman Python. *Library* yang dibutuhkan adalah lightgbm. Percobaan dilakukan dengan menggunakan metrik evaluasi NDCG@1, NDCG@3, NDCG@5, NDCG@10 menggunakan LETOR 4.0 MQ2008 yang terdiri atas 5 *Fold dataset* yang dipartisi. Pada percobaan ini, mq2008.train adalah data pelatihan, mq2008.vali adalah data validasi yang digunakan untuk memantau proses pelatihan dan mq2008.test adalah data uji. Data didapat dengan menjalankan ./wgetdata.sh.

Parameter LGBMRanker yang digunakan dalam *paper* ini ditampilkan pada Tabel 1. Pertama kami menggunakan nilai parameter *default* dari LGMBRanker yaitu *number of trees* 300, *learning rate* 0.05, *number of leaves* 31 untuk mengontrol kompleksitas model, *feature fraction* 0.9, dan *bagging fraction* was 0.9. Selanjutnya kami menyesuaikan untuk meningkatkan nilai NDCG nya.

Tabel 1 Parameter Setting

Parameter	Value
number of leaves	255
minimal data in leaf	1
minimal sum hessian in leaf	100
objective	*regression*
learning rate	0.1
number of threads	2
verbose	10
early stopping rounds	50

D. Hasil

Skor NDCG yang dihasilkan dari kelima *Fold* ditampilkan pada Tabel 2. Metrik untuk mengevaluasi algoritma peringkat yaitu NDCG@n, dimana @n menunjukkan bahwa metrik dievaluasi hanya pada n dokumen teratas. Misalnya NDCG@10 menunjukkan bahwa metrik dievaluasi hanya pada 10 dokumen teratas.

Tabel 2 Skor NDCG

Fold	NDGG@1	NDGG@3	NDGG@5	NDGG@10
1	0.664544	0.702462	0.742164	0.782247
2	0.679487	0.738718	0.770269	0.803714
3	0.675159	0.700883	0.736809	0.778698
4	0.660297	0.659529	0.710893	0.758665
5	0.630573	0.681832	0.702157	0.766917
Rata-rata	0.662012	0.6966848	0.7324584	0.7780482

Pada eksperimen yang dilakukan terhadap posisi peringkatan dapat dilihat bahwa NDCG@10 memiliki skor rata – rata NDCG lebih tinggi yaitu 0.7780482. Rata-rata dari NDCG@1, NDCG@3, NDCG@5 dan NDCG@10 kemudian dijumlahkan dan dicari lagi rata-ratanya sehingga diperoleh skor NDCG keseluruhan dari kelima *fold* yaitu sebesar 0.717301.

IV. KESIMPULAN

Percobaan ini dilakukan dengan mengusulkan pendekatan *pairwise* dengan algoritma LightGBM. Percobaan dilakukan dengan menggunakan *Benchmark Datasets* LETOR

MQ2008. Hasil percobaan yang dilakukan dengan menggunakan kelima *Fold*, maka didapatkan rata-rata skor NDCG dimulai dari NDCG@1, NDCG@3, NDCG@5 dan NDCG @10 adalah 0.717301. Rata-rata NDCG kelima *fold* yang sudah dihasilkan ini membuktikan bahwa penerapan algoritma LightGBM dan pendekatan *pairwise* sebagai pendekatan untuk membangun model *Learning to Rank*, menghasilkan skor NDCG yang dapat dikatakan baik untuk setiap *Fold* partisi pada dataset LETOR MQ2008 dan dapat disimpulkan bahwa model perangkingan yang dihasilkan juga sudah dapat dikategorikan baik.

UCAPAN TERIMA KASIH

Terima kasih kepada Tuhan Yang Maha Esa berkat kasih karunia Nya sehingga *paper* dengan judul *Pairwise Learning to Rank Approach Using Light Gradient Boosting Machine* (LightGBM) dapat selesai. Percobaan ini dilakukan untuk menjadi salah satu syarat lulus mata kuliah sistem temu balik sistem informasi. Setiap opini, temuan, dan kesimpulan yang diungkapkan dalam *paper* ini adalah milik penulis yang diperoleh berdasarkan hasil pengamatan terhadap percobaan dan studi literatur dari berbagai dokumen sejenis.

REFERENSI

[1] T.-Y. Liu, Learning to Rank for Information Retrieval, Beijing: Publishers Inc, 2011.

[2] M. Koppel, . A. Segne, M. Wagene, . L. Pense, A. Karwath dan S. Kramer, "Pairwise Learning to Rank by Neural Networks Revisited: Reconstruction, Theoretical Analysis and Practical Performance," *University of Birmingham*, pp. 1-16, 2019.

[3] X. Dong, X. Chen, Y. Guan, Z. Xu dan S. Li, "An Overview of Learning to Rank for Information Retrieval," *World Congress on Computer Science and Information Engineering*, pp. 600-606, 2009.

[4] Z. Hu, Y. Wang, Q. Peng dan H. Li, "Unbiased LambdaMART: An Unbiased Pairwise Learning-to-Rank Algorithm," *Association for Computing Machinery*, pp. 1-11, 2019.

[5] V. Melnikov, P. Gupta, . B. Frick, D. Kaimann dan E. Hullermeier, "Pairwise versus Pointwise Ranking: A Case Study," *Schedae Informaticae*, vol. 26, p. :73–83, 2016.

[6] S. Yunus, "Learning to Rank Place," pp. 1-62, 2019.

[7] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye dan T.-Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," *31st Conference on Neural Information Processing Systems*, pp. 1-9, 2017.

[8] A. A. Abdillah, H. Murfi dan D. Y. Satria, "Uji Kinerja Leraning To Rank Dengan Metode Support Vector Regression," *IndoMS Journal on Industrial and Applied Mathematic*, vol. 2, no. 1, pp. 15-25, 2015.

[9] H. Valizadegan, R. Jin, R. Zhang dan J. Mao, "Learning to Rank by Optimizing NDCG Measure," pp. 1-9, 2009.