Learning TO Rank (LETOR)

Muhammad Ali Norozi
mnorozi@idi.ntnu.no

Department of Computer and Information Science
Norwegian University of Science and Technology

04th April 2014
Trondheim, Norway
Learning to Rank
Learning to Rank
Learning to Rank
Learning to Ranking
Outline

1 Learning
2 Ranking
3 Learning to Rank
4 Active Learning to Rank
5 Online Learning to Rank
6 Conclusions
Outline

1 Learning

2 Ranking

3 Learning to Rank

4 Active Learning to Rank

5 Online Learning to Rank

6 Conclusions
Learning to Rank
Active Learning to Rank
Online Learning to Rank
Conclusions
Learning to Rank
Machine Learning – Introduction

We are using it dozen of times a day without even knowing it.
Machine Learning – Day to Day
Machine Learning – Day to Day

Recommendation systems – Collaborative filtering
Machine Learning – Day to Day
Machine Learning – Definition

Arthur Samuel (1959):

Machine Learning is a Science of getting the computers to “learn”, without being explicitly programmed!
Machine Learning – Definition

*Tom Mitchell (1998):*

A computer is said to *learn* from experience $E$ with respect to some task $T$ and some performance measure $P$, if its performance on $T$, as measured by $P$, improves with experience $E$. 
Machine Learning – Algorithms

1. Unsupervised
2. Supervised
3. Semi-supervised
Machine Learning – Algorithms

1 Unsupervised – let the machine learn itself
   • No class information
   • No training data
   • Clustering (K-means clustering)
Machine Learning – Algorithms

1. Unsupervised – let the machine learn itself
   - No class information
   - No training data
   - Clustering (K-means clustering)

2. Supervised – teach the machine how to learn
   - Learning from past experience (training)
   - Classification
   - Inductive learning – learning by example
   - Decision Trees
   - Ranking
Machine Learning – Algorithms

1 **Unsupervised** – let the machine learn itself
   - No class information
   - No training data
   - Clustering (K-means clustering)

2 **Supervised** – teach the machine how to learn
   - Learning from past experience (training)
   - Classification
   - Inductive learning – learning by example
   - Decision Trees
   - **Ranking**

3 **Semi-supervised** – teach and let the machine learn itself
   - Supervised learning which also make use of unlabelled data
   - A small amount of labelled and a large amount of unlabelled training data.
   - $\approx$ Ranking
Outline

1 Learning

2 Ranking

3 Learning to Rank

4 Active Learning to Rank

5 Online Learning to Rank

6 Conclusions
Ranking – IR
Traditional Ranking – Documents
Traditional Ranking – Queries
Traditional Ranking – Relevance Estimation
Traditional Ranking – functions

- Estimation of relevance of documents to given query
- Manually design the ranking function, for example:
  - Boolean ranking
  - Vector space models
  - Probabilistic models ($tf \times idf$, BM25)
  - Language Models
  - Linked Analysis Ranking models
  - etc...
Traditional Ranking – functions

- Estimation of relevance of documents to given query
- *Manually* design the ranking function, for example:
  - Boolean ranking
  - Vector space models
  - Probabilistic models ($tf \times idf$, BM25)
  - Language Models
  - Linked Analysis Ranking models
  - etc...
Traditional Ranking – functions

- Estimation of relevance of documents to given query
- *Manually* design the ranking function, for example:
  - Boolean ranking
  - Vector space models
  - Probabilistic models \((tf \times idf, BM25)\)
  - Language Models
  - Linked Analysis Ranking models
  - etc...
Traditional Ranking – functions

- Estimation of relevance of documents to given query
- Manually design the ranking function, for example:
  - Boolean ranking
  - Vector space models
  - Probabilistic models \( (tf \times idf, \text{BM25}) \)
  - Language Models
  - Linked Analysis Ranking models
  - etc...
Traditional Ranking – functions

- Estimation of relevance of documents to given query

- *Manually* design the ranking function, for example:
  - Boolean ranking
  - Vector space models
  - Probabilistic models \((tf \times idf, \text{BM25})\)
  - Language Models
  - Linked Analysis Ranking models
  - etc...
Traditional Ranking – functions

- Estimation of relevance of documents to given query
- Manually design the ranking function, for example:
  - Boolean ranking
  - Vector space models
  - Probabilistic models ($tf \times idf$, BM25)
  - Language Models
  - Linked Analysis Ranking models
  - etc...
Outline

1. Learning
2. Ranking
3. Learning to Rank
4. Active Learning to Rank
5. Online Learning to Rank
6. Conclusions
Learning to Rank
Learning to Rank:
Using Machine “Learning” technologies to solve the problem of “Ranking”
Learning to Rank:
Using Machine "Learning"^AI technologies to solve the problem of "Ranking"^IR
<table>
<thead>
<tr>
<th>Learning</th>
<th>Ranking</th>
<th>Learning to Rank</th>
<th>Active Learning to Rank</th>
<th>Online Learning to Rank</th>
<th>Conclusions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Learning TO Rank**
ML for Ranking or Learning to Rank
Phases – Training
Phases – Learning

Training Phase
Phases – Testing
Not just Learn to Rank but also Learn to –

• Crawl
• Index
• Mine the Data
• Frontend

Most of them are supervised, which means judgements can possibly be expensive.
Categorization – Approaches to LETOR

- Pointwise
- Pairwise
- Listwise
Categorization – Approaches to LETOR

- **Pointwise**
  - Existing ML methods
  - Exact relevance degree of each document
  - Transforming ranking to regression, classification, or ordinal regression
  - SVM-based method, in case of ranking it is binary classification (relevant or irrelevant)
  - input: single document
  - output: ground truth labels \( y = f(x) \), relevance score
Categorization – Approaches to LETOR

• Pairwise
  • Pairwise classification
  • Order correctly pairs of documents
  • Closer to ranking than pointwise
  • Classification on document pairs
  • Minimize the number of misclassified document pairs.
  • input: pair of documents
  • output: binary labels $y \in \{-1, +1\}$ which indicate if the documents are in correct order
  • complexity: quadratic number of documents
Categorization – Approaches to LETOR

- Listwise
  - Entire set of documents associated with query
  - Straightforwardly represents learning to rank problem
  - Input: n-dimensional feature vectors of all m candidate docs for given query
  - Output: scores of all candidate docs (permutation of feature vectors \( \{ f(x_i) \}_{i=1}^{n} \)).
Categorization – Approaches to LETOR

- Pointwise
- Pairwise
- Listwise

Pairwise and Listwise approaches are more suitable for learning to rank problem (e.g., ranking in search)
Issues in Learning to Rank

- Labelling of the data (usually manual task)
- Feature Extraction (based on scenario)
- Learning Method (model, loss function, algorithm)
- Evaluation Measures (to materialize the gain / loss)
Issues in Learning to Rank

- Labelling of the data (usually manual task)
- Feature Extraction (based on scenario)
- Learning Method (model, loss function, algorithm)
- Evaluation Measures (to materialize the gain / loss)
Labelling the Training data

Manual relevance judgement
Labelling the Training data

Manual relevance judgement

- In the supervised ML for ranking, we need:
  - Large enough labelled data for training
Labelling the Training data

Manual relevance judgement

- In the supervised ML for ranking, we need:
  - Large enough labelled data for training
  - Because the quality of the ranking function is highly correlated with the amount and quality of the training data.
Labelling the Training data

Manual relevance judgement

- In the supervised ML for ranking, we need:
  - Large enough labelled data for training
  - Because the quality of the ranking function is highly correlated with the amount and quality of the training data.
  - It is easy to collect unlabelled data.
Labelling the Training data

Manual relevance judgement

- In the supervised ML for ranking, we need:
  - Large enough labelled data for training
  - Because the quality of the ranking function is highly correlated with the amount and quality of the training data.
  - It is easy to collect unlabelled data.
  - It is expensive and painstakingly hard to label and update the training data.
Labelling the Training data

Manual relevance judgement

- In the supervised ML for ranking, we need:
  - Large enough labelled data for training
  - Because the quality of the ranking function is highly correlated with the amount and quality of the training data.
  - It is easy to collect unlabelled data.
  - It is expensive and painstakingly hard to label and update the training data.

- Offline process.
Labelling the Training data

Manual relevance judgement

• In the supervised ML for ranking, we need:
  • Large enough labelled data for training
  • Because the quality of the ranking function is highly correlated with the amount and quality of the training data.
  • It is easy to collect unlabelled data.
  • It is expensive and painstakingly hard to label and update the training data.

• Offline process.
• Why do we need to update the training data?
Labelling the Training data

Manual relevance judgement

- In the supervised ML for ranking, we need:
  - Large enough labelled data for training
  - Because the quality of the ranking function is highly correlated with the amount and quality of the training data.
  - It is easy to collect unlabelled data.
  - It is expensive and painstakingly hard to label and update the training data.
- Offline process.
- Why do we need to update the training data?
  - Users interests change!
Labelling the Training data

Manual relevance judgement

- In the supervised ML for ranking, we need:
  - Large enough labelled data for training
  - Because the quality of the ranking function is highly correlated with the amount and quality of the training data.
  - It is easy to collect unlabelled data.
  - It is expensive and painstakingly hard to label and update the training data.
- Offline process.
- Why do we need to update the training data?
  - Users interests change!
  - Companies interest change
Labelling the Training data

Manual relevance judgement

- In the supervised ML for ranking, we need:
  - Large enough labelled data for training
  - Because the quality of the ranking function is highly correlated with the amount and quality of the training data.
  - It is easy to collect unlabelled data.
  - It is expensive and painstakingly hard to label and update the training data.

- Offline process.
- Why do we need to update the training data?
  - Users interests change!
  - Companies interest change
  - Users need dynamic results
Labelling the Training data

Manual relevance judgement

How to solve this dilemma?
The answer is:

Active “Learning” to “Rank”
Outline

1 Learning
2 Ranking
3 Learning to Rank
4 Active Learning to Rank
5 Online Learning to Rank
6 Conclusions
For any *supervised* learning system to perform well, it must often be trained on hundreds (even thousands) of labeled instances.

Sometimes *part of* these labels comes at a little or no-cost:

- Spam flag on email
- Ratings
Active Learning to Rank

Active learning to rank systems attempt to overcome the labelling bottleneck.

The learning algorithm is allowed to choose the data from which it learns.

It will perform better with less training.
Active Learning to Rank – Cycle

learn a model

labeled training set $\mathcal{L}$

oracle (e.g., human annotator)

machine learning model

unlabeled pool $\mathcal{U}$

select queries
Co-active Learning to Rank

Co-active learning where both the system and user actively explore possible solution to speedup learning.

Interactions are modeled such that the system presents an initial ranked list, which is then improved by the user.

It was shown that feedback provided in this way can lead to effective learning.
Problems with Active Learning

The focus of active learning is to reduce manual labelling effort. However:

- They are not designed to learn from natural user interactions, happening live!
- Because they are offline!
- There is a need for online learning!
# Problems with Active Learning

The focus of active learning is to reduce manual labelling effort. However:

- They are not designed to learn from natural user interactions, happening live!
- Because they are offline!
- There is a need for online learning!
Problems with Active Learning

The focus of active learning is to reduce manual labelling effort. However:

- They are not designed to learn from natural user interactions, happening live!
- Because they are offline!
- There is a need for online learning!
Problems with Active Learning

The focus of active learning is to reduce manual labelling effort. However:

- They are not designed to learn from natural user interactions, happening live!
- Because they are offline!
- There is a need for online learning!
Outline

1 Learning
2 Ranking
3 Learning to Rank
4 Active Learning to Rank
5 Online Learning to Rank
6 Conclusions
The problem with supervised or semi-supervised learning to rank approach is:
The problem with supervised or semi-supervised learning to rank approach is:

- Once learned they usually do not continue to learn.
The problem with supervised or semi-supervised learning to rank approach is:

- Once learned they usually do not continue to learn.
- There is natural need for a self correcting learning and prediction process.
That is:

*Online “Learning” to “Rank”*
Online Learning to Rank

Learning to rank problem is settled in an online settings.

- System directly learns from live user interactions.
- Labelled data is not provided!
- But need to be collected through interaction with users
- It has to be inferred from online user interactions.
- The system transparently adapts to the users “true” preferences

Well! what else do we want from learning to rank?

But wait! online LETOR also have to face some challenges!
Online Learning to Rank

Learning to rank problem is settled in an online settings.

- System directly learns from live user interactions.
- Labelled data is not provided!
  - But need to be collected through interaction with users
  - It has to be inferred from online user interactions.
  - The system transparently adapts to the users “true” preferences

Well! what else do we want from learning to rank?

But wait! online LETOR also have to face some challenges!
Online Learning to Rank

Learning to rank problem is settled in an online settings.

- System directly learns from live user interactions.
- Labelled data is not provided!
- But need to be collected through interaction with users
  - It has to be inferred from online user interactions.
  - The system transparently adapts to the users “true” preferences

Well! what else do we want from learning to rank?

But wait! online LETOR also have to face some challenges!
Online Learning to Rank

Learning to rank problem is settled in an online settings.

- System directly learns from live user interactions.
- Labelled data is not provided!
- But need to be collected through interaction with users
- It has to be inferred from online user interactions.
- The system transparently adapts to the users “true” preferences

Well! what else do we want from learning to rank?

But wait! online LETOR also have to face some challenges!
Online Learning to Rank

Learning to rank problem is settled in an online settings.

- System directly learns from live user interactions.
- Labelled data is not provided!
- But need to be collected through interaction with users
- It has to be inferred from online user interactions.
- The system transparently adapts to the users “true” preferences

Well! what else do we want from learning to rank?

But wait! online LETOR also have to face some challenges!
Online Learning to Rank

Learning to rank problem is settled in an online settings.

- System directly learns from live user interactions.
- Labelled data is not provided!
- But need to be collected through interaction with users
- It has to be inferred from online user interactions.
- The system transparently adapts to the users “true” preferences

Well! what else do we want from learning to rank?

But wait! online LETOR also have to face some challenges!
Online Learning to Rank

Learning to rank problem is settled in an online settings.

- System directly learns from live user interactions.
- Labelled data is not provided!
- But need to be collected through interaction with users
- It has to be inferred from online user interactions.
- The system transparently adapts to the users “true” preferences

Well! what else do we want from learning to rank?

But wait! online LETOR also have to face some challenges!
Challenges

The main challenges are:

1. Quality of the available feedbacks, e.g., click through data.

2. The need to learn quickly and reliably, while maintaining high result quality.
Challenges

The main challenges are:

1. **Quality of the available feedbacks, e.g., click through data.**

2. The need to learn quickly and reliably, while maintaining high result quality.
Quality of user feedback

How to interpret users interactions and hence behaviour for LETOR?

- Explicit relevance feedback
  - Users indicate if the items are relevant or not-relevant
  - Expensive
  - Requires users time and efforts

- Implicit relevance feedback
  - Log user interaction information and use it to infer users’ satisfaction
  - All aspects of users interactions, click, mouse movements, dwell time, gaze, already installed items, you name it!
  - Much cheaper, in comparison to explicit feedback, by product of user interaction
  - But typically much noisier than explicit
  - Therefore interpretation and use is much more difficult

Lets take an example of the implicit feedback.
Quality of user feedback

How to interpret users interactions and hence behaviour for LETOR?

- **Explicit relevance feedback**
  - Users indicate if the items are relevant or not-relevant
  - Expensive
  - Requires users time and efforts

- **Implicit relevance feedback**
  - Log user interaction information and use it to infer users’ satisfaction
  - All aspects of users interactions, click, mouse movements, dwell time, gaze, already installed items, you name it!
  - Much cheaper, in comparison to explicit feedback, by product of user interaction
  - But typically much noisier than explicit
  - Therefore interpretation and use is much more difficult

Lets take an example of the implicit feedback.
Quality of user feedback

How to interpret users interactions and hence behaviour for LETOR?

- **Explicit relevance feedback**
  - Users indicate if the items are relevant or not-relevant
  - Expensive
  - Requires users time and efforts

- **Implicit relevance feedback**
  - Log user interaction information and use it to infer users’ satisfaction
  - All aspects of users interactions, click, mouse movements, dwell time, gaze, already installed items, you name it!
  - Much cheaper, in comparison to explicit feedback, by product of user interaction
  - But typically much noisier than explicit
  - Therefore interpretation and use is much more difficult

Lets take an example of the implicit feedback.
Quality of user feedback

How to interpret users interactions and hence behaviour for LETOR?

- Explicit relevance feedback
  - Users indicate if the items are relevant or not-relevant
  - Expensive
  - Requires users time and efforts

- Implicit relevance feedback
  - Log user interaction information and use it to infer users' satisfaction
  - All aspects of users interactions, click, mouse movements, dwell time, gaze, already installed items, you name it!
  - Much cheaper, in comparison to explicit feedback, by product of user interaction
  - But typically much noisier than explicit
  - Therefore interpretation and use is much more difficult

Lets take an example of the implicit feedback.
Click through data – highly valuable source of relevance information

Largely used implicit feedback, in most of the learning to rank for recommendation and IR.

To some extent depict users behaviour.
Click through data – highly valuable source of relevance information

Largely used implicit feedback, in most of the learning to rank for recommendation and IR.

To some extent depict users behaviour.

Can be collected in large quantity at reasonably low cost. But:
Click through data – highly valuable source of relevance information

Largely used implicit feedback, in most of the learning to rank for recommendation and IR.

To some extent depict users' behaviour.

Can be collected in large quantity at reasonably low cost. But:

- How to make sense out of it?
Click through data – highly valuable source of relevance information

Largely used implicit feedback, in most of the learning to rank for recommendation and IR.

To some extent depict users behaviour.

Can be collected in large quantity at reasonably low cost. But:

- How to make sense out of it?
- How to accurately interpret it?
Click through data – highly valuable source of relevance information

Largely used implicit feedback, in most of the learning to rank for recommendation and IR.

To some extent depict users behaviour.

Can be collected in large quantity at reasonably low cost. But:

- How to make sense out of it?
- How to accurately interpret it?
- Higher ranked results usually get more clicks, independent of how relevant they are!
Click through data – highly valuable source of relevance information

Largely used implicit feedback, in most of the learning to rank for recommendation and IR.

To some extent depict users behaviour.

Can be collected in large quantity at reasonably low cost. But:

- How to make sense out of it?
- How to accurately interpret it?
- Higher ranked results usually get more clicks, independent of how relevant they are!
- It is too noisy
Click through data – highly valuable source of relevance information

Largely used implicit feedback, in most of the learning to rank for recommendation and IR.

To some extent depict users' behavior.

Can be collected in large quantity at reasonably low cost. But:

- How to make sense out of it?
- How to accurately interpret it?
- Higher ranked results usually get more clicks, independent of how relevant they are!
- It is too noisy
- But still found to be useful in both research and practice.
Click through data – highly valuable source of relevance information

Clicks to be used **NOT** as an absolute feedback but relative to its context (whether a clicked item is more or less relevant to the non-clicked item in the proximity)
Classical click models

- Position model
- Cascade model
Classical click models

- **Position model** – click depends on both relevance and examination.
  - Each document has certain probability of being clicked (examined).
  - Which decays by, and only depends on rank positions.
  - A click on document indicate the document is examined and relevant.
  - *In the hind side, this model treats documents individually*
Classical click models

- **Position model** – click depends on both relevance and examination.
  - Each document has certain probability of being clicked (examined).
  - Which decays by, and only depends on rank positions.
  - A click on document indicates the document is examined and relevant.
  - *In the hind side, this model treats documents individually*

- **Cascade model** – user examines results sequentially and stops as soon as relevant doc is clicked.
  - The probability of examination is indirectly determined by two factors:
    1. Rank of the doc
    2. Relevance of all previous docs
  - *It makes strong assumption that only one doc is clicked per search and hence does not explain abandoned search and multiple clicks.*
Classical click models

There is difference between the *perceived* relevance and *actual* relevance.
Classical click models

There is difference between the *perceived* relevance and *actual* relevance.

There are other models as well which solves the above mentioned issues, but out of scope of this presentation.
Scalability

The need to learn quickly and reliably!
Scalability

The need to learn quickly and reliably!

Out of scope of this presentation
Outline

1 Learning

2 Ranking

3 Learning to Rank

4 Active Learning to Rank

5 Online Learning to Rank

6 Conclusions
Conclusions

- Huge theoretical and practical potential.
- Large amount of existing work and focus from both IR and AI communities.
- Over 100 publications in the top IR venues SIGIR, CIKM, KDD and others.
- Benchmark datasets – publicly available from both Yahoo! and Microsoft (from Learning to Rank challenge).
- Wide range of application scenarios.
- High prospects in near and possibly far future.
References and Further reading

Chapelle, Olivier and Chang, Yi
Yahoo! Learning to Rank Challenge Overview.

Fabrizio Silvestri
Mining Query Logs: Turning Search Usage Data into Knowledge,

Settles, Burr.
Active learning literature survey.
University of Wisconsin, Madison, pages 55–66, 2010

Liu, Bing
Web data mining,
References and Further reading

- **Joachims, Thorsten.**
  Optimizing search engines using clickthrough data,

- **Schuth, Anne and Hofmann, Katja and Whiteson, Shimon and de Rijke, Maarten.**
  LEROT: An online learning to rank framework,

- **Microsoft Research**
  LETOR: Learning to Rank for Information Retrieval,

- **Liu, Tie-Yan**
  Learning to Rank for Information Retrieval,
  Book – Springer, 2011
References and Further reading

Andrew Ng
Stanford Machine Learning course,
https://www.coursera.org/course/ml, 2014

Long, Bo and Chapelle, Olivier and Zhang, Ya and Chang, Yi and Zheng, Zhaohui and Tseng, Belle
Active learning for ranking through expected loss optimization,

Shivaswamy, Pannaga and Joachims, Thorsten.
Online structured prediction via coactive learning, 2012
Thank you for your attention!