Text Technologies for Data Science

INFR11145

Learning to Rank

Instructor
Björn Ross
Pre-Lecture

• Only one lecture today
• Last lecture in the course
• No lab
• After the lecture: Info on group project (coursework 3)
Lecture Objectives

• **Learn** about:
  • IR as a classification task
  • Learning to Rank approaches
Classical Models vs. ML in IR

• Classical Models:
  • Features (factors): only a few, e.g., TF, IDF, |D|, P(t|corpus) etc.
  • Structure: optimized for the a few particular features
  • Parameter & training
    • Often 1-2; not every factor has a parameter controlling its influence
    • Hand-tuning or data-based; can tune exhaustively since just 1-2 parameters
  • tfidf or BM25 or LIMIR? PRF? What $n_d$, $n_t$?

• ML in IR
  • Features: can include up to hundreds, thousands, or even more
  • Define the basic structure of a model
  • Quite generic: such as a weighted linear combination of all features
  • Parameters & training
    • Many; control the influence of each feature and their combinations
    • Impossible to tune by hand; Must be data-driven
  • Let the ML decide what is better!
Text Classification in IR

- Text Classification:
  - Classify a document into one of two or more classes
  - Different features could be used, e.g. BOW

- Can we model IR as classification?
  - Classify document to C1: R or C2: NR
  - Challenges?
    - Training data?
    - Features? BOW?

- BOW features cannot work
  - Spam? Viagra, @ed.ac.uk
  - Sentiment? happy, sad
  - Relevant? Trump, hurricane
  - Relevance depends on the query!
From Classification to IR

• Transforming features
  • Text classification: Input (D) $\rightarrow$ output (yes/no)
  • Information Filtering: Input (D|Q) $\rightarrow$ output (yes/no)

• Feature set:
  • Independent of absolute words
  • More on relation between doc and query
  • Mostly numbers (formulas, frequencies, …)
  • As consistent as possible among different Q,D pairs
  • e.g.:
    • TFIDF, BM25
    • Query in page title? Heading?
    • Query in anchor text linking pages
    • PageRank of doc
    • Number of times page clicked for the same query
# Popular Features

<table>
<thead>
<tr>
<th>Column in Output</th>
<th>Description</th>
<th>Column in Output</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TF(Term frequency) of body</td>
<td>24</td>
<td>LMIR.JM of body</td>
</tr>
<tr>
<td>2</td>
<td>TF of anchor</td>
<td>25</td>
<td>BM25 of anchor</td>
</tr>
<tr>
<td>3</td>
<td>TF of title</td>
<td>26</td>
<td>LMIR.ABS of anchor</td>
</tr>
<tr>
<td>4</td>
<td>TF of URL</td>
<td>27</td>
<td>LMIR.DIR of anchor</td>
</tr>
<tr>
<td>5</td>
<td>TF of whole document</td>
<td>28</td>
<td>LMIR.JM of anchor</td>
</tr>
<tr>
<td>6</td>
<td>IDF(Inverse document frequency) of body</td>
<td>29</td>
<td>BM25 of title</td>
</tr>
<tr>
<td>7</td>
<td>IDF of anchor</td>
<td>30</td>
<td>LMIR.ABS of title</td>
</tr>
<tr>
<td>8</td>
<td>IDF of title</td>
<td>31</td>
<td>LMIR.DIR of title</td>
</tr>
<tr>
<td>9</td>
<td>IDF of URL</td>
<td>32</td>
<td>LMIR.JM of title</td>
</tr>
<tr>
<td>10</td>
<td>IDF of whole document</td>
<td>33</td>
<td>BM25 of URL</td>
</tr>
<tr>
<td>11</td>
<td>TF*IDF of body</td>
<td>34</td>
<td>LMIR.ABS of URL</td>
</tr>
<tr>
<td>12</td>
<td>TF*IDF of anchor</td>
<td>35</td>
<td>LMIR.DIR of URL</td>
</tr>
<tr>
<td>13</td>
<td>TF*IDF of title</td>
<td>36</td>
<td>LMIR.JM of URL</td>
</tr>
<tr>
<td>14</td>
<td>TF*IDF of URL</td>
<td>37</td>
<td>BM25 of whole document</td>
</tr>
<tr>
<td>15</td>
<td>TF*IDF of whole document</td>
<td>38</td>
<td>LMIR.ABS of whole document</td>
</tr>
<tr>
<td>16</td>
<td>DL(Document length) of body</td>
<td>39</td>
<td>LMIR.DIR of whole document</td>
</tr>
<tr>
<td>17</td>
<td>DL of anchor</td>
<td>40</td>
<td>LMIR.JM of whole document</td>
</tr>
<tr>
<td>18</td>
<td>DL of title</td>
<td>41</td>
<td>PageRank</td>
</tr>
<tr>
<td>19</td>
<td>DL of URL</td>
<td>42</td>
<td>Inlink number</td>
</tr>
<tr>
<td>20</td>
<td>DL of whole document</td>
<td>43</td>
<td>Outlink number</td>
</tr>
<tr>
<td>21</td>
<td>BM25 of body</td>
<td>44</td>
<td>Number of slash in URL</td>
</tr>
<tr>
<td>22</td>
<td>LMIR.ABS of body</td>
<td>45</td>
<td>Length of URL</td>
</tr>
<tr>
<td>23</td>
<td>LMIR.DIR of body</td>
<td>46</td>
<td>Number of child page</td>
</tr>
</tbody>
</table>
Training Data

- Training data: \( \{R,X\} \)
  - \( X \): feature representation of (D,Q) pairs
  - \( R = \{-1,+1\} \) … is D relevant to Q or no

- Samples:
  - Large set of (D,Q) pairs
  - Wide range of Q’s (long/short, frequent/rare, …)
  - Wide range of D’s for each Q (top/deep ranked, recent/old pages, …)

- Labels:
  - Manually labelled: assessors judge relevance of docs to queries (similar to standard IR)
  - Automatically labelled: click-through data
Classification or Ranking?

• Click-through data
  • User clicks can give indication of relevance
  • What about non-relevance?
  • A list of ranked results: \( D_1 \rightarrow D_2 \rightarrow D_3 \)
    user clicked on \( D_3 \) and neglected \( D_1 \) & \( D_2 \)
    what does it mean?
    • \( D_3 \) is relevant and \( D_1 \) & \( D_2 \) are not relevant?
    • Relevance: \( D_3 > D_1 \) & \( D_2 \)?

• It might be better to model the problem as ranking
  • Label \rightarrow \text{Ranking preference (e.g. gain=[4,3,2,1,0])}
  • Learning \rightarrow \text{to optimize } Doc_X > Doc_Y
    not to classify them to R/NR
  • Input: features for set of docs for a given query
    Objective: rank them (sort by relevance)
ML & IR: History

- Considerable interaction between these fields
  - Rocchio algorithm (60s) is a simple learning approach
  - 80s, 90s: learning ranking algorithms based on user feedback
  - 2000s: text categorization

- Limited by amount of training data

- Web query logs have generated new wave of research
  - L2R (LTR): “Learning to Rank”
What is Learning-to-Rank?

• Purpose
  • Learn a function automatically to rank results effectively

• Point-wise approach
  • Classify document to R / NR

• List-wise
  • The function is based on a ranked list of items
  • given two ranked list of the same items, which is better

• Pair-wise
  • The function is based on a pair of item
  • e.g., given two documents, predict partial ranking
Point-wise Approaches

• The function is based on features of a single object
  • e.g., regress the rel. score, classify docs into R and NR

• Very similar to classification
  • Examples of (D,Q) pairs with labels 1 or 0

• Classic retrieval models are also point-wise:
  • Calculate score(Q, D)
  • *If* score(Q,D) > \( \theta \) \( \rightarrow \) relevant
    else, irrelevant

• Referred to as *information filtering*
  • Standing query + new documents coming
  • Decide whether a new document is R or NR
List-based Approaches

• Given: ranked list A and ranked list B
  Task: decide which is better

• Need a loss function on a list of documents

• Challenge is scale
  • Huge number of potential lists

• Can develop tricks
  • Consider only possible re-rankings of top N retrieved by some fixed method

• Still expensive
  • No clear benefits over pairwise ones (so far)
Pair-wise Approaches

• Trying to classify
  • Which document of two should be ranked at a higher position?

• Optimize based on:
  • Margin between decision hyperplane and instances
  • Errors
  • Weighted based on some hyper-parameter C
  • Evaluation metric

• Example: SVM-rank
  • A generalization of SVM that supports ranking
    [Herbrich et al. 1999, 2000; Joachims et al. 2002]
# SVM-rank Example

<table>
<thead>
<tr>
<th>Q1</th>
<th>qid:1</th>
<th>1:1</th>
<th>2:1</th>
<th>3:0</th>
<th>4:0.2</th>
<th>5:0</th>
<th># 1A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>qid:1</td>
<td>1:0</td>
<td>2:0</td>
<td>3:1</td>
<td>4:0.1</td>
<td>5:1</td>
<td># 1B</td>
</tr>
<tr>
<td></td>
<td>qid:1</td>
<td>1:0</td>
<td>2:1</td>
<td>3:0</td>
<td>4:0.4</td>
<td>5:0</td>
<td># 1C</td>
</tr>
<tr>
<td></td>
<td>qid:1</td>
<td>1:0</td>
<td>2:0</td>
<td>3:1</td>
<td>4:0.3</td>
<td>5:0</td>
<td># 1D</td>
</tr>
<tr>
<td></td>
<td>qid:2</td>
<td>1:0</td>
<td>2:0</td>
<td>3:1</td>
<td>4:0.2</td>
<td>5:0</td>
<td># 2A</td>
</tr>
<tr>
<td></td>
<td>qid:2</td>
<td>1:1</td>
<td>2:0</td>
<td>3:1</td>
<td>4:0.4</td>
<td>5:0</td>
<td># 2B</td>
</tr>
<tr>
<td></td>
<td>qid:2</td>
<td>1:0</td>
<td>2:0</td>
<td>3:1</td>
<td>4:0.1</td>
<td>5:0</td>
<td># 2C</td>
</tr>
<tr>
<td></td>
<td>qid:2</td>
<td>1:0</td>
<td>2:0</td>
<td>3:1</td>
<td>4:0.2</td>
<td>5:0</td>
<td># 2D</td>
</tr>
<tr>
<td></td>
<td>qid:3</td>
<td>1:0</td>
<td>2:0</td>
<td>3:1</td>
<td>4:0.1</td>
<td>5:1</td>
<td># 3A</td>
</tr>
<tr>
<td></td>
<td>qid:3</td>
<td>1:1</td>
<td>2:1</td>
<td>3:0</td>
<td>4:0.3</td>
<td>5:0</td>
<td># 3B</td>
</tr>
<tr>
<td></td>
<td>qid:3</td>
<td>1:1</td>
<td>2:0</td>
<td>3:0</td>
<td>4:0.4</td>
<td>5:1</td>
<td># 3C</td>
</tr>
<tr>
<td></td>
<td>qid:3</td>
<td>1:0</td>
<td>2:1</td>
<td>3:1</td>
<td>4:0.5</td>
<td>5:0</td>
<td># 3D</td>
</tr>
</tbody>
</table>

- **Q3:** 3C>3A, 3C>3B, 3C>3D, 3B>3A, 3B>3D, 3A>3D
Pair-wise Approaches

- The most popular approach
- Learning methods: SVM-rank, RankBoost, GBRank, Ranknet, LambdaRank, LambdaMART
- Pairwise ranking error often has better correlations with evaluation metrics than the loss/objective functions in point-wise approaches
  - Why: evaluation measures only care about rankings!
    - e.g., ground-truth: rel(D1) = 3, rel(D2) = 2
      - Regression model 1: pred.rel(D1) = 2, pred.rel(D2) = 3
      - Regression model 2: pred.rel(D1) = 1, pred.rel(D2) = 0
      - Model 1 is better than model 2 by criterion of evaluation regression (the prediction error), but model 2 yields a correct ranking of docs
- Still, issues with ranking SVM e.g. it does not directly optimize an evaluation metric
Pair-wise Approaches

• LambdaMART:
  • Misordered pairs are not equally important
  • Depends on how much they contribute to the changes in the target evaluation measure
**Pair-wise Approaches**

- Optimizing for an evaluation metric
  - The general idea is to weight loss/objective function or gradient with pairwise changes in evaluation measure.
  - e.g., in LambdaMART: lambda gradient

- Can we optimize all measures?
  - Not necessarily
  - For some measures, pairwise changes do not only relate to the two documents themselves, but also others …
    - Position-based measures do not have the issues (pairwise change only depends on the two documents)
    - Cascade measures may have issues
Pair-wise Approaches: Example

- Experiments
  - 1.2k queries, 45.5K documents with 1890 features
  - 800 queries for training, 400 queries for testing

<table>
<thead>
<tr>
<th>Method</th>
<th>MAP</th>
<th>P@1</th>
<th>ERR</th>
<th>MRR</th>
<th>NDCG@5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ListNET</td>
<td>0.2863</td>
<td>0.2074</td>
<td>0.1661</td>
<td>0.3714</td>
<td>0.2949</td>
</tr>
<tr>
<td>LambdaMART</td>
<td>0.4644</td>
<td>0.4630</td>
<td>0.2654</td>
<td>0.6105</td>
<td>0.5236</td>
</tr>
<tr>
<td>RankNET</td>
<td>0.3005</td>
<td>0.2222</td>
<td>0.1873</td>
<td>0.3816</td>
<td>0.3386</td>
</tr>
<tr>
<td>RankBoost</td>
<td>0.4548</td>
<td>0.4370</td>
<td>0.2463</td>
<td>0.5829</td>
<td>0.4866</td>
</tr>
<tr>
<td>RankingSVM</td>
<td>0.3507</td>
<td>0.2370</td>
<td>0.1895</td>
<td>0.4154</td>
<td>0.3585</td>
</tr>
<tr>
<td>AdaRank</td>
<td>0.4321</td>
<td>0.4111</td>
<td>0.2307</td>
<td>0.5482</td>
<td>0.4421</td>
</tr>
<tr>
<td>pLogistic</td>
<td>0.4519</td>
<td>0.3926</td>
<td>0.2489</td>
<td>0.5535</td>
<td>0.4945</td>
</tr>
<tr>
<td>Logistic</td>
<td>0.4348</td>
<td>0.3778</td>
<td>0.2410</td>
<td>0.5526</td>
<td>0.4762</td>
</tr>
</tbody>
</table>
L2R in Practice

Capannini, G., et al.
Quality versus efficiency in document scoring with learning-to-rank models.
IP&M 2016.
Current work in L2R

• Deep learning models are mainly used
• No manual feature extraction is applied
• Using word-embeddings to represent queries and docs, then learn the features automatically
• Content-independent models: try to learn the pattern of relations between terms in Q and D
• Content dependent: dependent on the terms
Types of Deep LTR Models

- **Early Interaction-based**: Learn on the signals from a query-document interaction.
- **Late Interaction (Representation) based**: Learn independent representations of queries and documents and then consider the interaction between them.
- Early interaction based approaches, e.g. DRMM, are relatively independent of the content (terms themselves) – tend to generalize well.
- Late interaction based approaches, e.g. ColBERT, are usually data hungry approaches – hence likely not to generalize well on standard ad-hoc IR collections.

By: Debasis Ganguly
• DRMM (left) uses histograms of word pair similarities (between doc and query) terms as inputs to a feed-forward network.

• The model seeks to utilize inherent patterns in these histograms to distinguish relevance from non-relevance.

• KNRM (right) does not need to rely on histograms. Instead it applies 1D convolution.

By: Debasis Ganguly
Summary

- IR as a classification task
- Learning to rank (L2R) approaches
  - Point-wise
    - Information Filtering
  - List-wise
  - Pair-wise
    - Ranking SVM
    - LambdaMART
- Current work in L2R depends on deep learning models and word-embedding representations
Resources


• SVM^RANK: [http://svmlight.joachims.org/](http://svmlight.joachims.org/)

• L2R test sets: