Diversity in recommender systems

Andy Valjakka, Wilson Poon, Heng Gui, Toni Mikkola
Source material


Santos et al. 2015: Influence sampling

\[ D \subset R \subset S \]

Diversity set is a subset of Result set which is a subset of the Set of every item

- diversity is introduced to a result set in two phases:
  - reduction of search space
  - trade-off between similarity and diversity

- again two phases:
  - candidate filtering
  - diversity computation
Trade-off briefly

diversity problem: “how to retrieve elements similar to the query center, but also diverse enough to generate a more heterogeneous and useful result set”

● first, we have query center $dq$
● trade-off is a parameter $\lambda \in [0, 1]$
  ○ aka diversity preference
  ○ $\lambda = 0 \rightarrow$ preferably no diversity
  ○ $\lambda = 1 \rightarrow$ preferably only diversity
    ■ “users tend to prefer approx. $\lambda = 0.4$”
● this is bi-criteria optimization
  ○ similarity and diversity are competing
Reduction of search space

- aims at excluding irrelevant elements asap
- previous papers
  - “obtain sets of explanations”
  - “structure results according to item distance”
    - then get items with appropriate distances
- all prior research focuses on diversity computation
  - focus on candidate filtering instead
- proposal: result diversification based on influence
Result Diversification based on Influence - RDI

- goal: exclusion of elements with low contribution
- *separation distance* principle
  - if the distance between two items is less than minimum, they have equivalent information
- minimum distance comes from *influence intensity*
  - if item $d_1$ is closer to $d_2$ than query center $d_q$, then $d_2$ has more influence on $d_1$ than $d_q$
    - this means that $d_1$ and $d_2$ are, in essence, the same item and only one of them is needed for the result set
- basis of *BRID (Better Result with Influence Diversification)*
Algorithm in filtering candidate

- **k nearest neighbor (kNN)**
- **Random selection method (Rnd)**
  - randomly select elements with enough relevance.
- **Clustering-based method (CLT)**
  - pick representative element from each cluster
    - kNN filtering!
- **Influence selection (BRID)**
  - select items that each pairs of items will not be similar. So that each elements can carry enough information.(short-distance items can only provide same information)
Brief introduction of diversity computation.

- **Incremental strategy**: greedy, aim at best item each iteration
  - MMR: maximal marginal relevance
  - GMC: greedy marginal contribution
  - MSD: max-sum dispersion

- **Exchanging**: replace current items if iterated ones are better
  - SWAP

- **Meta-heuristic**: greedily build a list, then swap iteratively
  - GMC: greedy marginal contribution
Evaluation of processing time

only diversity computation phase

candidate filtering + diversity computation
Evaluation of quality
Conclusion

● All the algorithms contribute to the diversity computation phase.
● BRID method can directly get a good result. (the processing time of diversity computation is almost zero)
● The quality of results is almost the same (difference less than 2%) no matter what methods are used.
● Based on processing time, the BRID and Rnd are the best in candidate filtering, and MMR and SWAP is the best in the diversity computation.
● Actually, totally random is a pretty good method.
Aspects are expressed in user interests and item descriptions

- e.g. Genres in context of movie recommendations
- Explicit - directly available from input data
- Latent - i.e. implicit, learned from user-item interaction

Constrained PLSA model use explicit aspects, but learns the aspect probabilities to directly optimise their predictive performance -> Latent aspects

Baseline recommender system generate recommender list

xQuAD do reranking with recommender list by using probabilities of aspects
**xQuAD re-ranker** (explicit Query Aspect Diversification framework)

- intent aware diversification comes from IR field
- re-ranker reorder recommendation list by probabilities of aspects
- inputs: predefined set of k> 0 aspects $A=\{a_1, a_2, ..., a_k\}$ and list of recommendations

1. Compute aspects probability distribution such as $\sum_a p(a|u)=1$
2. Starting with $S=\emptyset$ (re-ranked list), and given score $s(u, i)$ by the baseline recommender, uses $\lambda$-value (I guess founded by empirical tests)
   - every iteration: greedily selecting the item $i$ that satisfies
     \[
     i^* = \arg \max_{i \in R_u \setminus S} (1 - \lambda)s(u, i) + \lambda \sum_{a \in A} p(a|u)p(i|u, a) \prod_{j \in S} (1 - p(j|u, a)),
     \]
   - updating $S \leftarrow S \cup \{i^*\}$
Probabilities

item | user probability:

\[ p(i|u) = \sum_a p(a|u)p(i|u, a) \]

Estimates of probability (aspect | user) and probability(item | user, aspect)

\[ p(a|u) \sim \frac{|\{i \in I_u : a \in A_i\}|}{\sum_{a' \in A} |\{i \in I_u : a' \in A_i\}|}, \quad p(i|u, a) \sim \frac{1_{A_i}(a)s(u, i)}{\sum_{j \in \mathcal{R}} 1_{A_i}(a)s(u, j)} \]
Wasilewski et al 2016 (cont)

- **ExAs-Co0 - explicit aspects co-occurrence estimation method**
  - does not learn from user behaviour. \( p(a \mid u) \) equal weight for all

- **pLSA model (probabilistic latent semantic analysis)**
  - Uses latent aspects as learning technique for machines. Aims to represent some sort of relationship between items in terms of their proximity in the semantic space
  - Assumes \( p(i \mid u, a) \) is independent of \( u \) and learns \( p(a \mid u) \) and \( p(i \mid a) \) by optimisation on the training data

- **C-pLSA - constrained pLSA.**
  - Uses explicit aspects
  - constraint of \( p(i \mid a) = 0 \) when ‘a’ ‘not in \( A_i \)
  - user aspect probabilities learned more accurately and weights can be derived
Wasilewski et al 2016 - Experiment Background

- Precision & Recall
  - Precision is the proportion of top recommendations that are relevant
  - Recall is the proportion of all relevant results included in top recommendations

- NDCG - Normalized Discounted Cumulative Gain
  - Measures the performance of a recommendation system based on the graded relevance of the recommended entities. It varies from 0.0 to 1.0, with 1.0 ideal ranking

- ERR-IA - Expected Reciprocal Rank (Intent Aware), cascade model
  - Expected reciprocal length of time that it takes the user to find a relevant item
  - Top-down search method, user stops at position $p$. Once the user is satisfied, search is terminated and items below this result are not examined regardless of their position

- MF - Matrix Factorization
  - Characterizes items and users by vectors of factors inferred from item rating patterns
  - High correspondence between both lead to a recommendation
Wasilewski et al 2016 - Results

Movielens: 800,000 ratings used as training set; 200,000 ratings as test

4 algorithms (MF, PLSA, UB, IB) used to generate recommendation candidate lists of 100 items. Re-ranked using the following methods below:

xQuAD_e = ExAs-Co0
xQuAD_c = c-pLSA
xQuAD_p = pLSA

Diversity - ExAs-Co0 gives equal weights thus more items for the re-ranker to work with while ExAs-Co0 focus only on those items that contribute to relevance

Despite the lower capacity in diversity, belief that C-pLSA represents user’s true intents
Wasilewski et al 2016 - Conclusion

- For this paper, results were not very significant between methods
- Using probability as latent aspect modeling for user behaviour
  - important as explicit information is very limited
  - Represent all of user’s interests and avoiding filter bubbles
- There is a growing number of options for studying and tweaking recommender systems

Thank you, Any Questions?