A Reviewer Recommendation Framework

Group 48 - CS771 Project Report

Mentor - Dr. Purushottam Kar
Assistant Professor
Department of CSE
IIT Kanpur

Homanga Bharadhwaj
150294
Department of CSE
IIT Kanpur
homangab@iitk.ac.in

Anmol Porwal
150108
Department of CSE
IIT Kanpur
anmolp@iitk.ac.in

Varun Khare
150714
Department of CSE
IIT Kanpur
varun@iitk.ac.in

Amrit Singhal
150092
Department of CSE
IIT Kanpur
amrits@iitk.ac.in

Harsh Vardhan
150283
Department of EE
IIT Kanpur
harshv@iitk.ac.in

Abstract

The process of reviewer assignment to the papers of a conference is a very challenging and sensitive task. The choice of reviewers for a particular paper plays a crucial role in determining whether the paper is accepted into the conference. Automating the task of choosing reviewers is one of the recent challenges being actively researched by the machine learning community. The state of the art paper-matching system is the Toronto Paper Matching System which uses an LDA based topic modelling approach to identify topics in the entire paper and a simple dot product to assess similarity between reviewer topics and paper topics. Our approach builds on this line of work and develops an alternating optimization approach for completing the matrix of relevance scores between paper vectors and author vectors.

Contents

1 Problem Statement

The problem statement formulated for this project is as follows: Given a paper and a list of available reviewers, recommend the reviewers best suited for reviewing the given paper. Formally stating, given a set of N reviewers, we need to find the subset of best k reviewers for a given 'unseen' paper using the set of resources mentioned in the Resources section.

The work described is in a scenario where all the papers have a format close to the NIPS format and each of these papers have an explicit Experiment Section or a Related works section. Nonetheless, our approach is somewhat general in nature, which can be applied to any paper with slight modifications in the algorithm.

2 Motivation

The problem of expert recommendation and assignment is a typical challenge in both industry and academia. In this paper, we focus on the task of paper-reviewer recommendation for conferences and journals. A large number of conferences now routinely receive more than one thousand papers which have to be assigned to reviewers from a large group of over a hundred reviewers. The assignment of each paper to a set of suitable reviewers demands both the expertise of the reviewers and of the specific theme of the paper. Hence for a typical conference, this process of reviewer assignment in beyond the scope of a single person. Moreover, decentralized methods lead to natural questions of human bias being infused in the process. Hence, an automated method of paper-matching is of utmost necessity. The Toronto Paper matching system[2] and more recently, the paper 'A Robust Model for Paper Reviewer Assignment'[7] are novel attempts towards this objective of automation, but both have their limitations, which we have tried to address in our project.

The approach we present in this paper tackles two central drawbacks of the TPMS algorithm, namely for topic modelling an LDA based model is used and that this model does not weigh differently the topics generated from different sections of the paper. We believe that not all sections of the paper yield topics suitable for the process of reviewer assignment; for instance, the Related Works and Simulation Results sections often make references to existing papers with similar themes and hence should be given more weightage in topic modelling.

3 Related Works

In this section we highlight previous work on paper-reviewer assignment through information retrieval and machine learning techniques. The very popular Toronto Paper Matching System[2] addresses the assignment problem as a problem of minimizing the cost of network flow using some various similarity measures. This work also considered reviewers bids, which express their interest in specific papers, as available in the form of feedback. Hettich et al. [6] formulated TF-IDF to quantitatively evaluate a similarity index between the drafts and reviewers. Tang et al. [12] justified an assumption that every reviewer has an expertise level, which is already known. Then they defined some specific matching criteria to optimize the reviewer arrangement procedure. Charlin et al. in TPMS [2] essentially used an LDA[1] model, standard linear regression techniques and collaborative filtering to determine reviewer assignments. Rodriguez et al. [11] have designed a co-authorship graph with the references of a submitted paper as initial steps for suggesting reviewers. Conry et al.[3] address both the modeling of reviewer-paper preferences (which can be cast as a learning problem) and the optimization of reviewing assignments to satisfy global conference criteria. As described, a bulk of the existing papers have focused on improving the similarity between query and experts. Expertise has almost exclusively been considered the major factor in these approaches.[7]

4 Resources

We aimed to use the data-sets experimented on in the TPMS paper as this would allow us to compare our model with the SOTA systems currently available. But, due to unavailability of those datasets, we had to switch to using the ones of Liu et al [7]. We mailed the authors of the paper[7], and requested them to provide us with the datasets for our project, to which they agreed. So, we also produced results on TPMS like models, after training them on this dataset, to be able to perform a comparative study of the results produced by our models. Additionally we also also procured paper-reviewer pairs and, most importantly, feedback of the reviews from the NIPS website. The feedback and the reviewer rating was used for training the recommendation system. We reason that these metrics represent the most suited reviewer for the paper and hence will serve as a reliable source for validating out model. We also used the recommended reviewers section given by the authors of the paper to be reviewed along with the bidding done by the reviewers[8] in order to achieve a more realistic evaluation of the predicted reviewers.

4.1 Dataset Description

Three different datasets were used for the different learning tasks involved in the project:

1. **LDA training:** For obtaining relevant LDA topics for the papers, the LDA model was trained on a data-set of **6560** complete papers from NIPS 1987-2015 which was obtained from Kaggle. [5]
2. **Constructing Author Profiles:** We used a collection of 14913 paper abstracts authored by the reviewers. Each abstract was assigned a vector according using the trained LDA model and the reviewer vector was calculated as the average of his authored paper vectors.
3. **Recommendation System:** This dataset[9] was used for actual matrix computation and collaborative filtering.
   - Abstracts of **148** papers published in *NIPS 2006* conference.
   - A reviewer list of 364 possible reviewers.
   - 650 paper-reviewer relevance judgments obtained from prominent researchers from the NIPS community. This is assumed to be the ground truth for the problem. Scores are as follows
     - Very Relevant: Score = 3
     - Relevant: Score = 2
     - Slightly Relevant: Score = 1
     - Irrelevant: Score = 0

5 The Proposed Approaches

In this section, we first describe our attempt at replicating the core techniques used in TPMS followed by an analysis of its shortcomings. Then, we describe our novel contributions to the problem, where we propose various methods to overcome these shortcomings. We also present the experimental results of the implementations for these approaches.

5.1 Replicating TPMS Model:

The paper-reviewer matching problem does not have any established benchmarks to test performance. For replicating the performance of the TPMS model, we used LDA-based topic modelling, where we represented papers as vectors of topics and the reviewer as the average vector comprising of all his/her paper vectors. The topic model was trained on a scientific corpora to create suitably relevant relevant topics.

Having created all the author vectors and the paper vectors, we tried to fill the incomplete reviewer paper relevance matrix by using hybrid collaborative filtering[4].

5.1.1 Hybrid Collaborative Filtering

It is an item-based and user-based collaborative filtering. In both these methods we need a metric to compute similarity between any 2 vectors. The similarity metrics used by us are cosine, jaccard and pearson.

Cosine similarity score :

$$Cosine(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{d} \mathbf{x}_i \mathbf{y}_i}{\sqrt{\sum_{i=1}^{d} \mathbf{x}_i^2} \sqrt{\sum_{i=1}^{d} \mathbf{y}_i^2}}$$

Jaccard Similarity Score:

$$Jaccard(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n} min(\mathbf{x}_i, \mathbf{y}_i)}{\sum_{i=1}^{n} max(\mathbf{x}_i, \mathbf{y}_i)}$$

Pearson Similarity Score :

$$Pearson(\mathbf{x}, \mathbf{y}) = \frac{n \sum_{i=1}^{n} \mathbf{x}_i \mathbf{y}_i - \sum_{i=1}^{n} \mathbf{x}_i \sum_{i=1}^{n} \mathbf{y}_i}{\sqrt{n \cdot (\sum_{i=1}^{n} \mathbf{x}_i^2) - (\sum_{i=1}^{n} \mathbf{x}_i)^2} \sqrt{n \cdot (\sum_{i=1}^{n} \mathbf{y}_i^2) - (\sum_{i=1}^{n} \mathbf{y}_i)^2}}$$

In user-based collaborative filtering, for each user who has rated any item, each item is assigned a score based on the scores of similar users for that item weighted by the similarity between that user and the other users and the dot-product of that user with the given item. For example the (i, j) element of the reviewer paper relevance matrix is updated at the t^{th} iteration by the user based collaborative filtering and is given by

$$R^t(i,j)_{user} = \rho_{user} \left(R^{t-1}(i,j)\right) + (1 - \rho_{user}) \left(\frac{\sum_{k=1,k \neq i}^{n} R^t(k,j) SimR(i,k)}{\sum_{k=1,k \neq i}^{n} SimR(k,j)} \right)$$

Where, $SimR(k, j)$ is the similarity coefficient between k^{th}, j^{th} reviewers and $0 \leq \rho_{user} \leq 1$.

In item-based collaborative filtering, the functions performed on user and items are interchanged. In our method we calculate the user-based and item based relevance scores and take their average. For example the (i, j) element of the reviewer paper relevance matrix is updated at the t^{th} iteration by the item based collaborative filtering is given by

$$R^t(i,j)_{item} = \rho_{item} (R^{t-1}(i,j)) + (1 - \rho_{item}) \left(\frac{\sum_{k=1,k \neq j}^{d} R^t(i,k) SimP(j,k)}{\sum_{k=1,k \neq j}^{n} SimP(k,j)} \right)$$

Where $SimP(k, j)$ is the similarity coefficient between k^{th}, j^{th} papers and $0 \leq \rho_{item} \leq 1$.

$$R^t(i,j) = \frac{R^t(i,j)_{user} + R^t(i,j)_{item}}{2}$$

The relevance matrix obtained after this method makes predictions using dot product of reviewer and paper and reviewer paper similarities, thus it is used to replicate the min ideas used in TPMS.

5.1.2 Problems with TPMS

We found TPMS lacking in two main aspects:

1. Topic Modelling used by TPMS employs a BoW or LDA based topic modelling approach. Substituting it with a semantically richer topic modelling like Word2vec can improve results of the paper-reviewer assignment.
2. LDA or BoW Topic Modelling used by TPMS gives equal importance to all sections of the paper. Not all sections of the paper are equally important for deciding reviewers. The related works section and the Experiment sections of the paper are more important than the rest of the paper

5.2 Unsupervised Clustering of Paper Sections

We proposed that giving importance to certain sections like experiments, related works, etc.(i.e. the topics in these sections) is necessary as they have a higher stake in deciding a good reviewer as opposed to general topics from across the paper. A reviewer who has used the same experimental techniques/data-sets has a higher chance of better understanding the work. He/she may already be familiar with most of the intricate details of the data-set and so his/her review would be to the point and precise. Similarly, it is easy to understand that citations of related works are highly important for assigning a reviewer. In order to incorporate these details we aim to cluster all the sentences in the paper and assign them priority weights. These weight can be used in two ways*:

1. Use the weights in boosting the relevance score of the cited person if available in the paper-reviewer matrix.
2. Boost the topic frequency whose sentences lie on a cluster of higher priority

5.2.1 Sent2Vec using vector averaging

This method uses average of word vectors ($\mathbf{w}^i$) to evaluate the sentence vector ($\mathbf{S}$)[10] i.e.

$$\mathbf{S} = \frac{\sum_{i=1}^{n} \mathbf{w}^i}{n}$$

We applied agglomerative clustering on these vectors to get hierarchical clusters. But from our results, we obtained that the merger distance is significantly high for the merging to make sense. Also most of the sentences belonged to the same (central) cluster. We believe this is because averaging causes loss of information of all the words present. This can be seen in the following example:
Assume a sentence has 2 words with vectors $\mathbf{w}^1 = -\mathbf{1}$ and $\mathbf{w}^2 = \mathbf{1}$. Then we have,

$$\mathbf{S} = \frac{\sum_{i=1}^{n} \mathbf{w}^i}{n} = 0$$

This $\mathbf{S}$ neither represents **1** nor **-1** and hence the mess.

5.2.2 Bag of Vectors

This approach uses sum of cosines of all word vector pairs in the sentences to calculate the similarity score. Assume we have n and m words in sentence $\mathbf{S}_1$ and sentence $\mathbf{S}_2$ respectively.

$$\mathbf{S}_1 = \{\mathbf{w}^i : \mathbf{w}^i \text{ is in sentence } 1\}$$

$$\mathbf{S}_2 = \{\mathbf{w}^i : \mathbf{w}^i \text{ is in sentence } 2\}$$

Then after removing stop words, we have pairs $\langle \mathbf{u}_i, \mathbf{v}_j \rangle$ st $(\mathbf{u}_i, \mathbf{v}_j) \in \mathbf{S}_1 \times \mathbf{S}_2$ and the score is:

$$\text{score} = \sum_{\substack{\mathbf{u}_i \in \mathbf{S}_1 \\ \mathbf{v}_j \in \mathbf{S}_2}} \frac{\langle \mathbf{u}_i, \mathbf{v}_j \rangle}{\|\mathbf{u}_i\|_2 \|\mathbf{v}_j\|_2}$$

We will apply spectral clustering based on this affinity matrix.

5.3 Alternating Optimization approach

We hypothesized that the LDA topic vectors $\mathbf{x}$ obtained for each paper and for each reviewer are not optimal for the paper-reviewer assignment problem. Thus our problem now expands to finding the actual reviewer-paper relevance matrix and the optimal vectors for each paper and each reviewer. We try to solve this problem by using an alternating optimization approach:-

1. Initialize $\mathbf{u}_i, \mathbf{v}_j$ to LDA vectors
2. Optimizing the reviewer-paper relevance matrix from the given paper and reviewer vector

3. Using this matrix to obtain the optimal paper and reviewer vectors via matrix factorization.

Obtaining the reviewer-paper relevance matrix is done as described in the base case scenario.

Using Low Rank Matrix Factorization[13], we break the obtained reviewer-paper relevance matrix into the Paper matrix and Reviewer matrix. Low Rank Matrix factorization is itself an Alternating Optimization algorithm updating the paper vectors using the matrix and the reviewer vectors and then updating the reviewer vectors using the matrix and paper vectors. For low rank matrix factorization we use a stochastic gradient descent approach to find the paper and user vectors to solve the optimization problem

$$\arg\min_{\substack{U\in\mathbb{R}^{m\times k}\\ V\in\mathbb{R}^{n\times k}}} \sum_{i=1}^{m}\sum_{j=1}^{n}(X_{ij} - \langle \mathbf{u}^i, \mathbf{v}^j \rangle)^2 + \lambda \left(\sum_{i=1}^{m} \|\mathbf{u}^i\|_2^2 + \sum_{i=1}^{n} \|\mathbf{v}^i\|_2^2 \right)$$

The update steps for GD are as follows:

$$\mathbf{u}^{i^t} \leftarrow \mathbf{u}^{i^t} - 2\eta_t (\lambda \cdot \mathbf{u}^{i^t} - (X_{i^t j^t} \dot{-} \langle \mathbf{u}^{i^t}, \mathbf{v}^{i^t} \rangle) \cdot \mathbf{v}^{j^t})$$

$$\mathbf{v}^{j^t} \leftarrow \mathbf{v}^{j^t} - 2\eta_t (\lambda \cdot \mathbf{v}^{j^t} - (X_{i^t j^t} \dot{-} \langle \mathbf{u}^{i^t}, \mathbf{v}^{j^t} \rangle) \cdot \mathbf{u}^{i^t})$$

The initial matrix was the partially filled relevance score matrix and the initial reviewer and paper vectors were their LDA vectors.

But this seemed to work more like a regularizer which only smooths the relevance matrix. So, after the discussion, during the presentation, we have decided to change the optimization problem to

$$argmin_{\substack{U\in\mathbb{R}^{m\times k}\\ V\in\mathbb{R}^{n\times k}}} \sum_{i=1}^{m}\sum_{j=1}^{n}(X_{ij} - \langle \mathbf{u}^i, \mathbf{v}^j \rangle)^2 + \lambda \left(\sum_{i=1}^{m} \|\mathbf{u}^i - \mathbf{u}_*^i\|_1 + \sum_{i=1}^{n} \|\mathbf{v}^i - \mathbf{v}_*^i\|_1 \right)$$

Here $\mathbf{u}_*^i, \mathbf{v}_*^i$ refer to the vectors at the start of the function. This change will enforce the paper and reviewer vectors not to deviate too much from the LDA vectors ensuring document topics to play an influential role in relevance score.

5.4 Modelling Cited Authors

We claim that the authors of the papers cited in the *Experiments* or the *Related Works* section of any paper are the best reviewers for that paper. So, to model these sections explicitly, we first find the papers cited in these sections. So, if the authors of such cited papers are present in the reviewer list then we assign those reviewers high relevance scores for that paper. If those authors aren't present in the reviewer list then we compute their author profiles and add these authors as new rows in the reviewer-paper relevance matrix with a high score in the column corresponding to the paper in which they have been cited. We complete the matrix according to the method described in base case approach. The end result of this addition of filled rows would be to increase the relevance scores of reviewers who are similar to the authors cited in the experiments and related works sections of the paper.

6 Experiments

6.1 Preprocessing Steps

We employed standard pre-processing processes for Natural Language Processing and also some strategies specific to our project, namely-

- **Stemming** - We used the standard Porter stemming algorithm for all the papers in our data-set.

- **Stop-word removal** - We used a standard list of stop-words in order to eliminate them from the papers in our data-set.
- **Extracting Experiment Section** - Owing to the great diversity in the title of this section (e.g. Results, Experimentation, Simulation Studies etc.), we manually extracted this section from each paper in our data-set.
- **Extracting References** - We used the bibtex files for the papers in our data-set to extract the references present in them.
- **Creating Author profiles** - An author is the 'average' of the topics in all of his/her papers. We used a simple average of the paper vectors of each author.

6.2 Implementation Details

- For the LDA training, we used Uni-grams and 127 topics and 5 passes on the data-set with a minimum probability of 0.
- For collaborative filtering used in part 1, we set an initial value of 1.0 to each unfilled entry and used all three similarity metrics (cosine, Jaccard, Pearson) for training, with the best metric (according to cross-validation accuracy) being used for predictions.
- Root Mean Square Error (RMSE) was calculated by a 10-fold cross validation; Cross Validation (CV) dividing the actual relevance scores into 10 parts, with training done using only 9 parts and RMSE calculated between the predicted values and the actual values for the last part.
- We then implemented the Alternating Optimization approach for 5 epochs and calculated the changes in 10-fold CV RMSE.
- For the Low Rank Matrix Factorization, 5000 iterations were performed with step length 0.0002 and $\lambda = 0.02$
- For LMRC, we used the author and paper vectors obtained from previous iteration as the current initial vectors.

6.3 Results

Technique	**RMSE value from our model**	**RMSE value from TPMS implementation**
Cosine	1.029	1.033
Jaccard	1.054	1.065
Pearson	1.029	1.044

- The RMSE values obtained from our model is slighly better than those obtained from the TPMS equivalent.
- *Note:* Relevance Score is a number between 0 and 3.

7 Challenges Encountered

The lack of a 'proper' data-set was the primary hurdle in realizing this project satisfactorily. Specifically, data-set with reviewer-paper assignment was difficult to find owing to confidentiality clauses attached with such matching. Most of the existing work in this front had specifically requested for data from the Conference organizing committees. Since we had a time constraint on implementing our project, we could not pursue this method of data acquisition to a significant level. However, we had mailed a number of authors of previous papers and managed to acquire the data-set used in the paper 'A Robust Model for Paper-Reviewer Assignment'.[7]

Moreover, we spent a bulk of our time and resources implementing the TPMS paper and trying to reproduce the results. Only partial code snippets for this paper are available freely and even those modules are not very 'portable'.

The culture of open-sourcing codes of research papers has only recently been advocated for promoting a culture of reproducible research. Eventually, we were successfully able to implement the TPMS algorithm on our data-set and compare the performance of our approach and that of TPMS.

For comparison, we required web crawlers for extracting papers of the authors cited but most freely available web crawlers are not compatible with the latest Google Scholar design. Also, another challenge was that the cited name of the authors and their profile on Google Scholars often differed (e.g. cited V S Saxena but the Google Scholar profile reads Vikram S Saxena). This made it a tedious process to manually resolve out the conflicts.

Finally, due to the absence of a standard benchmark limited our tests of how effective the system we designed indeed is. The ideal test of a reviewer recommendation system is for it to be implemented in an actual conference and receive feedback of the authors after the conference. This was not feasible in our current setting.

8 Future Works

1. A lot of our proposed approaches have not been hypertuned to perfection due to paucity of time. So, getting them tuned and running, and comparing their performances with the base case of TPMS should give us more insights into the paper-reviewer matching problem.
2. The final reviewer vectors and paper vectors can be used to decide which topics are most essential in paper-reviewer assignment both for the reviewers and papers. This can possibly be used to further divide the papers into groups with each group having a weight vector. Now the topics of a new paper vector can be automatically boosted according to the weight vector of the group to which the papers belongs.
3. Also, constructing the reviewer profiles and paper profiles using only abstracts discards a lot of information and an immediate improvement would be to use the complete paper instead of just abstracts.
4. Identification of sections of the paper important for reviewer assignment is something which we have done using only the actual relevance scores. If we focus on a method which finds these sections independent of the relevance scores we might obtain significantly better results even with very few or even corrupted relevance scores
5. Extending this method to papers not having a clearly defined experiment and/or related works section is another important aspect. The experiment section is just an example. If we can compute the sections of the paper essential for the review process, then we can apply it to all papers.
6. A **citation graph-based approach** which measures similarity between paper and reviewer by convolving their 2 graphs offers a very different perspective to this problem. The encoding of information in such graphs appears to be much more suited to the reviewer-paper matching problem than in the topic models. Weighing different sections of the paper boils down to the assigning different weights to different citations in the graph.

9 Acknowledgments

We thank our mentor Dr. Purushottam Kar for his patient guidance during the first leg of our project. We also thank our classmates of CS771 for the insightful discussions both inside and outside the classroom. Finally, we express our heartfelt gratitude to Xiang Liu, Torsten Suel and Nasir Memon of New York University for kindly consenting to share with us their data-set.

References

[1] BLEI, D. M., NG, A. Y., AND JORDAN, M. I. Latent dirichlet allocation. *Journal of machine Learning research 3*, Jan (2003), 993–1022.

[2] Charlin, L., and Zemel, R. The toronto paper matching system: an automated paper-reviewer assignment system.

[3] Conry, D., Koren, Y., and Ramakrishnan, N. Recommender systems for the conference paper assignment problem. In *Proceedings of the third ACM conference on Recommender systems* (2009), ACM, pp. 357–360.

[4] Ekstrand, M. D., Riedl, J. T., and Konstan, J. A. Collaborative filtering recommender systems. *Foundations and Trends in HumanComputer Interaction 4*, 2 (2011), 81–173.

[5] Hammer, B. Nips papers. `https://www.kaggle.com/benhamner/nips-papers/data`.

[6] Hettich, S., and Pazzani, M. J. Mining for proposal reviewers: lessons learned at the national science foundation. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* (2006), ACM, pp. 862–871.

[7] Liu, X., Suel, T., and Memon, N. A robust model for paper reviewer assignment. In *Proceedings of the 8th ACM Conference on Recommender systems* (2014), ACM, pp. 25–32.

[8] Microsoft. Welcome to cmt: Microsofts academic conference management service. `https://cmt.research.microsoft.com/cmt/`.

[9] Mimno, D., and McCallum, A. Expertise modeling for matching papers with reviewers. In *KDD* (2007).

[10] Pagliardini, M., Gupta, P., and Jaggi, M. Unsupervised learning of sentence embeddings using compositional n-gram features. *arXiv preprint arXiv:1703.02507* (2017).

[11] Rodriguez, M. A., and Bollen, J. An algorithm to determine peer-reviewers. In *Proceedings of the 17th ACM conference on Information and knowledge management* (2008), ACM, pp. 319–328.

[12] Tang, W., Tang, J., and Tan, C. Expertise matching via constraint-based optimization. In *Web intelligence and intelligent agent technology (wi-iat), 2010 ieee/wic/acm international conference on* (2010), vol. 1, IEEE, pp. 34–41.

[13] Yeung, A. A. Matrix factorization: A simple tutorial and implementation in python. `http://www.quuxlabs.com/blog/2010/09/matrix-factorization-a-simple-tutorial-and-implementation-in-python/`.