Using Non-Task Data To Solve Cold Start Problems in Heterogeneous Peer Prediction

Brian Hentschel Anna S. Hilgard Casey Meehan

Abstract

- This document describes the expected style, structure, and rough proportions for your final project write-up.
- While you are free to break from this structure, consider it a strong prior for our expectations of the final report.
- Length is a hard constraint. You are only allowed max **8 pages** in this format. While you can include supplementary material, it will not be factored into the grading process. It is your responsibility to convey the main contributions of the work in the length given.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1. Introduction

In peer prediction settings with heterogeneous users, the goal is to make truthful reporting a low-regret strategy for any agent. To fully minimize expected regret while allowing for heterogeneity, this would require the computation and use of individualized user-user signal correlations for every pair of users. This is in general computationally infeasible, so methodologies typically focus instead on clustering similar users while trying to minimally distort their individual signal distributions.

Additionally, because this method explicitly requires an existing signal distribution, there is a cold start problem: if we do not already have a significant amount of task data for a new user, we cannot know how properly to reward them in comparison to their peers. In our project, we focus on two specific goals:

1. Clustering is usually done by comparing each user's reports with other users. We aim to speed up this clustering by using latent variable matrix factorization and then clustering on much smaller latent variables.

2. Bootstrapping the clustering process by using non-task data to form an initial clustering before agent reporting begins.

2. Background

In Shnayder et al. (2016), Shnayder et al. prove that the Correlated Agreement multi-task peer prediction mechanism maximally incentivizes truthful reporting without a ground truth for homogeneous users. The CA mechanism can be generalized to non-homogeneous users but requires a prohibitive number of user reports; thus, Agarwal et al. (2017) bounds expected regret by clustering users with similar signal-report distributions. This regret bound is proportional to the L1 difference between user-user 'Δ Matrices' and those of their respective clusters, where a Δ Matrix of users q, r is defined as:

$$D_{q,r}(i,j) = p(\text{q reports i, r reports j}) - p(\text{q reports i})p(\text{r reports j})$$

The primary dataset we use is that of the *Good Judgment Global Forecasting Competition*, available at `https://dataverse.harvard.edu/dataverse.xhtml?alias=gjp`. There are many challenges inherent to this dataset. First, the idea of a task is not well-defined. Users enter predictions in continuous time, and the true probability of an event will change over time, such that each minute could be considered a different task (Note that if we expect users only update their predictions

in response to new information, these time-question tasks will be independent conditional on the signal). We choose to bucket these responses by week to mediate between the desire to allow for variation in true probability over time while maintaining sufficient task response density. This results in 1,499,294 predictions by 2,052 users on 1,548 tasks. Secondly, the signal itself can be expected to have a high degree of noise, as the probability of any of these global events is not well-defined even in the absence of individual interpretation effects.

To study the effects of the clustering methodologies in a slightly less noisy realm, we additionally use a second dataset of user judgments on whether or not websites contained adult content. This dataset, termed *Adult*, was pulled from the Square Project at the University of Texas. The data is available at `https://github.com/ipeirotis/Get-Another-Label/tree/master/data`. We limit the dataset to users who gave at least one report of each signal, resulting in 65,144 reports by 389 users on 10,381 tasks.

Both datasets required significant preprocessing to account for sparsity of signal distributions and to compute the Δ matrices. Additionally, because Good Judgment reports were continuous rather than discrete, we bucketed the responses into five uniform bins, which we use to compute the matrices.

3. Related Work

(Agarwal et al., 2017) is the only paper we know of specifically considering clustering methods for heterogeneous peer prediction. The paper differs from our work in that it assumes all of the relevant data exists and seeks to prove a theoretical upper bound on regret rather than to study empirical average regret in settings with sparse data. While the authors also use the *Adult* dataset among others, they restrict it to a small subset of only those users for whom there is a full joint distribution between any two users, resulting in only 269 users.

Further, while the authors prove a theoretical bound in the case of empirically estimated ground truth, the tests they run on real data rely on the existence of a known ground truth label. This allows them to calculate individual signal confusion matrices using methods developed in Dawid & Skene (1979) and leverage tensor decomposition techniques from Anandkumar et al. (2014) and Zhang et al. (2016) to compute average clusters directly. We seek to show that in the more general case of incomplete data with no ground truth labels, clustering instead on user-specific latent variables learned through matrix factorization can lead to similar computational savings as tensor decomposition while eliminating the need for labels. Further, we show that these computationally efficient clusterings are similarly effective with respect to incentivizing truthfulness as clustering on Δ matrices directly.

The use of the Correlated Agreement mechanism (Shnayder et al., 2016) requires that the Δ matrices are already known or can be empirically estimated from observed data. Nothing in the peer prediction literature addresses this cold start problem yet. However, it is well studied in the field of matrix factorization. The most similar method to ours, in that it incorporates non-task data, is (Kula, 2015).

4. Model

Given a sparse matrix containing tasks, users, and users‘ reports on each task, the goal is to produce clusters which accurately describe users‘ reporting behavior. Because of the sparse matrix data, we focus on matrix factorization models to generate latent factor representations of users. That is, we assume that there is some latent variable representation of a user‘s forecasting behavior, u_i that interacts with related task-specific factors, t_j. These could be thought of, for example as the task‘s topic areas and the user‘s expertise in those specific topic areas. After transforming the predictions to range from -1 to 1 instead of 0 to 1, we use the following model, shown in Figure 1:

$$p_{i,j} \sim \mathcal{N}(\tanh\left(\mu_{i_M}\left[u_i^T t_j + g + \mu_j\right] + \mu_{i_A}\right), \sigma^2)$$

Users tend to exhibit two forms of individualized biases in this space. The multiplicative bias, μ_{i_A} captures either risk aversion or overconfidence, such that they tend to predict values either closer to or farther from 0 (.5) than are warranted by their knowledge. The additive bias, μ_{i_M} captures both availability bias or normalcy bias which would result in user having a tendency to predict a specific level. The task bias μ_j captures the general prediction likelihood of the task absent user-specific factors, and the global bias g captures the overall mean of the user-task predictions. The tanh restricts the space of the predicted mean to the domain of the target value.

Then, we must infer $k + 2$ parameters for each user, $k + 1$ parameters for each task, and one global bias, where k is the length of the latent vector representation we choose for users and tasks. Only the predictions $p_{i,j}$ are known in this model.

For *Adult*, which does not have this probablistic representation and has reports ranging from 1 to 4, we use only additive biases and remove the tanh.

XX TODO PREDICTING LATENT USER VECTORS FROM PSYCH DATA XX

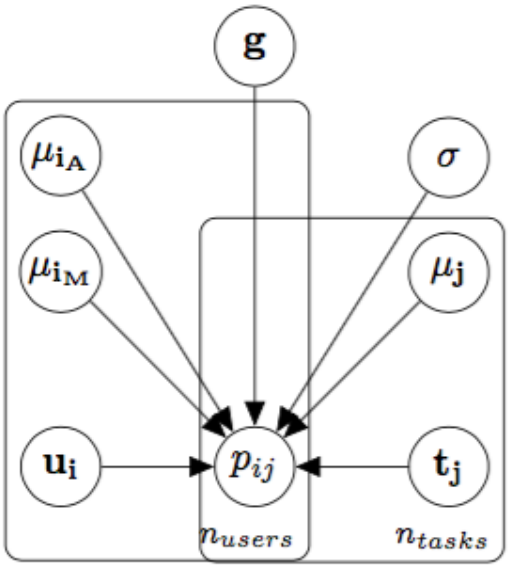

Figure 1. Model of prediction generation.

5. Inference (or Training)

For learning the latent user parameters, we use mini-batch stochastic gradient descent to find the maximum likelihood estimate. We choose not to use a prior for regularization as we find that the training and validation performance do not generally diverge. This is likely because there is already enough noise in the training data for each parameter that it is difficult to overfit the model. We do not explicitly code the initialization of the model, so of course it is possible it finds a local rather than global minimum. However, the performance of the model is consistent across random starts.

Algorithm 1 Mini-Batch SGD Parameter Fitting

learning rate = α
for epoch in epochs **do**
  for batch in batches **do**
    Calculate the gradient of the MSE Loss for each parameter for all users and tasks in the batch
    Update the parameters:
    param $\leftarrow$ param - α * $\frac{d}{dparam}$(MSE Loss)
  end for
end for

TODO COLD START MODEL

6. Methods

For the *Good Judgment* Dataset, we use the file `survey_fcasts.yr4.tab`, consisting of all user forecasts for the fourth year of the tournament. The file has 1,685,784 lines, each consisting of details about a specific user id and forecast. We restrict to only binary valued questions, for which a single probability of 'yes' is given as the answer (as opposed to, for example, "Who will be inaugurated as President of Russia in 2012?", which had 3 possible answers). We restrict to only users who responded to at least 30 Individual Forecasting Problems (IFPs) over the course of the year. Given that users can (and ideally do) update their predictions for IFPs over time, any given time period could be thought of as a new IFP. Following the scoring procedure of the Good Judgment, we assume that if a user enters a prediction and then does not update it, this implies that they would like their existing prediction to be carried forward until they do update. We allow each week an IFP is active to be a different prediction task to balance temporal variability in task signals with our desire for higher task report density.

In computing the user latent vectors, we transform the predictions from -1 to 1 as described above to allow for the multiplicative user bias but otherwise allow them to remain continuous. We train with stochastic gradient descent for 50 epochs with a learning rate of .9.

As a metric of the clustering success, we use the average element-wise L1 distance between clustered Δ matrices and the original user-user Δ matrices for each pair of users. This requires binning the continuous predictions into discrete buckets. We choose to use 5 uniformly spaced bins from 0 to 1. We then compute the empirical joint distribution for any two users based on these discrete signals, as well as the empirical signal distributions of each individual. As mentioned above, the Δ matrices can be calculated from these as

$$D_{q,r}(i,j) = p(\text{q reports i, r reports j}) - p(\text{q reports i})p(\text{r reports j})$$

To cluster, we use the scikit-learn implementation of K-means on all cores. The baseline we compare to is that of clustering on the Δ matrices directly, which is the method described in Agarwal et al. (2017) and should in general be better at guaranteeing similarity of Δ matrices than clustering on other factors. To compute a user's features for clustering, we append all of the Δ matrices between that user and all other users, leaving NaNs where not enough data exists to compute a distribution, and flatten this into a vector. When clustering, we iteratively impute feature means for NaNs such that they are not taken into consideration in the clustering.

The *Adult* Dataset has 92,720 lines, each of which has a report by a user id on a website. Ratings are discrete and ordinal, and we use them as given for both learning latent user variables and for computing Δ matrices.

TODO COLD START MODEL

Table 1. Distribution of L1 Distances Across Clusterings

	Δ (100)	K=2 (100)	Δ (500)	K=2 (500)
mean	.0168	.0215	.0145	.0192
std	.0243	.0290	.0218	.0258
min	0	0	0	0
25%	.0021	.0043	.0015	.0036
50%	.0084	.0108	.0071	.0099
75%	.0216	.0271	.0186	.0244
max	.7118	.6864	.6536	.6191

Table 2. Runtime (s) of KMeans algorithm on User Representation

Clusters	Δ	K=2	K=3	K=4
100	423	.57	.55	.65
150	625	.66	.66	.66
200	800	.66	.77	.87
250	959	.87	.88	.97
300	1187	.96	1.1	1.1
350	1198	1.1	1.2	1.2

7. Results

- What were the results comparing previous work / baseline systems / your systems on the main task?
- What were the secondary results comparing the variants of your system?
- This section should be fact based and relatively dry. What happened, what was significant?

While we achieved the lowest training and validation loss using 10 latent user factors (See Figure 2), the best L1 loss against the original Δ matrices was achieved by clustering on only 2 latent user factors (See Figure 3). Distribution results are given in Table 3

However, clustering on the latent factors for all values of K was significantly faster than clustering directly on the Δ matrices, as the flattened concatenated matrices result in 51,300 features for each user.

8. Discussion

- What conclusions can you draw from the results section?
- Is there further analysis you can do into the results of the system? Here is a good place to include visualizations, graphs, qualitative analysis of your results.
- What questions remain open? What did you think might work, but did not?

One possible explanation is that this setup forces more of the user variability into the multiplicative and additive biases, which are more specifically relevant to the task of predicting a distribution of reports given signals.

9. Conclusion

- What happened?
- What next?

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

References

Agarwal, Arpit, Mandal, Debmalya, Parkes, David C, and Shah, Nisarg. Peer prediction with heterogeneous users. 2017.

Anandkumar, Animashree, Ge, Rong, Hsu, Daniel, Kakade, Sham M, and Telgarsky, Matus. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15:2773–2832, 2014.

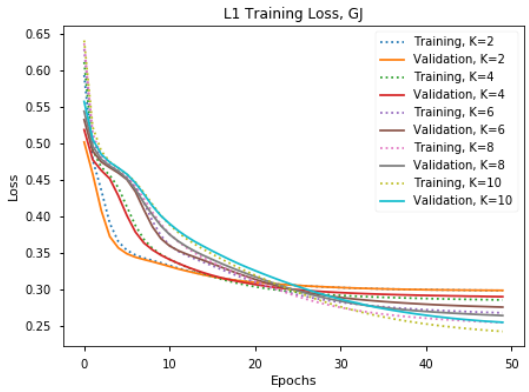

Figure 2. Training Results, L1 Loss.

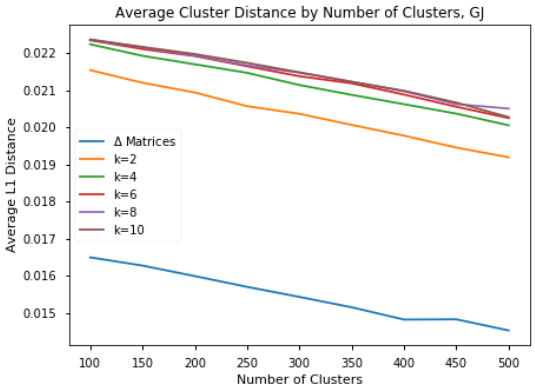

Figure 3. Average L1 Distance.

Figure 4. Visualizations of the internals of the system.

Dawid, Alexander Philip and Skene, Allan M. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied statistics*, pp. 20–28, 1979.

Kula, Maciej. Metadata embeddings for user and item cold-start recommendations. *arXiv preprint arXiv:1507.08439*, 2015.

Shnayder, Victor, Agarwal, Arpit, Frongillo, Rafael, and Parkes, David C. Informed truthfulness in multi-task peer prediction. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pp. 179–196. ACM, 2016.

Zhang, Yuchen, Chen, Xi, Zhou, Dengyong, and Jordan, Michael I. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. *The Journal of Machine Learning Research*, 17(1):3537–3580, 2016.