Targeted Training for Multi-organization Recommendation

KIRAN TOMLINSON, Cornell University, USA
MENGTING WAN, CAO LU, BRENT HECHT, JAIME TEEVAN, and LONGQI YANG, Microsoft, USA

Making recommendations for users in diverse organizations (*orgs*) is a challenging task for workplace social platforms such as Microsoft Teams and Slack. The current industry-standard model training approaches either use data from all organizations to maximize information or train organization-specific models to minimize noise. Our real-world experiments show that both approaches are poorly suited for the multi-org recommendation setting where different organizations' interaction patterns vary in their generalizability. We introduce *targeted training*, which improves on standard practices by automatically selecting a subset of orgs for model development whose data are cleanest and best represent global trends. We demonstrate how and when targeted training improves over global training through theoretical analysis and simulation. Our experiments on large-scale datasets from Microsoft Teams, SharePoint, Stack Exchange, DBLP, and Reddit show that in many cases targeted training can improve mean average precision (MAP) across orgs by 10–15% over global training, is more robust to orgs with lower data quality, and generalizes better to unseen orgs. Our training framework is applicable to a wide range of inductive recommendation models, from simple regression models to graph neural networks (GNNs).

CCS Concepts: • **Information systems** → **Recommender systems**; **Enterprise applications;**

Additional Key Words and Phrases: Recommendation system, multi-organization, graph learning

ACM Reference format:
Kiran Tomlinson, Mengting Wan, Cao Lu, Brent Hecht, Jaime Teevan, and Longqi Yang. 2023. Targeted Training for Multi-organization Recommendation. *ACM Trans. Recomm. Syst.* 1, 3, Article 12 (July 2023), 18 pages.
https://doi.org/10.1145/3603508

1 INTRODUCTION

Workplace communication and social platforms (e.g., Microsoft Teams and Slack) have become essential productivity and collaboration tools as organizations across the industry underwent accelerated digital transformation in recent years [34]. Recommendation systems are critical components of these platforms to address the problem of workplace information overload [9]. Different from recommending content in traditional consumer-facing applications, a crucial challenge that commercial software face is the need to serve a diverse range of **organizations** (***orgs***). For example, Microsoft Teams makes post recommendations for users from firms of varied sizes—from small businesses to large corporations (similar recommendation settings arise in

Authors' addresses: K. Tomlinson, Cornell University, 107 Hoy Rd, Ithaca, NY 14853; email: kt@cs.cornell.edu; M. Wan, C. Lu, B. Hecht, J. Teevan, and L. Yang, Microsoft, One Microsoft Way, Redmond, WA 98052; emails: mengting.wan@microsoft.com, cao.lu@microsoft.com, brent.hecht@microsoft.com, teevan@microsoft.com, longqi.yang@microsoft.com.

2770-6699/2023/07-ART12 $15.00
https://doi.org/10.1145/3603508

Slack, Salesforce Chatter, and Google Workspace, to name just a few). To address this, platforms across industry adopted two classes of strategies: (1) training a global model using data from all organizations [10]—"the more data the better," and (2) training a dedicated model for each org [32] or first clustering organizations and then training a model for each cluster—providing more customization to an org's particular user patterns.

In this article, we show that neither data maximization nor per-org customization are ideal solutions to the multi-organization recommendation problem. Instead, our real-world experiments reveal the pervasiveness of "super organizations" where one can train models that significantly outperform both global models trained on all data and local models trained on organization-specific datasets. For example, in a Microsoft Teams post recommendation task, 30% of orgs in our sample produced per-org models that outperform the global model. Moreover, models trained on these super orgs generalize better than global models to new orgs not seen at all before test time. In fact, in the Teams post recommendation task, we can make better recommendations on out-of-sample test orgs by targeted training on a subset of distinct orgs than by adding the test orgs to the training set and using global training. We argue that this is evidence of shared patterns of user preference across orgs, but that those patterns are much easier to learn in some orgs' data than others. For instance, this could be because some orgs exhibit more distribution shifts over time around the global mean. We show theoretically that even simple temporal noise patterns can result in super orgs.

To leverage this unique pattern, we propose a lightweight, practical, and theoretically-informed framework, *targeted training*, that outperforms existing solutions against a wide variety of recommendation algorithms, from logistic regression to **graph neural networks** (**GNNs**). In targeted training, we train per-org models and use cross-org validation to identify super orgs whose models perform well across the board. We then train a final model on combined data from a set of these orgs. Our approach improves recommendation quality across orgs and results in highly robust models, producing effective recommendations even in out-of-sample orgs. We provide theoretical justification for this approach based on disparate noise levels across orgs and validate this reasoning in simulated data. We also show the benefits of targeted training on a wide range of datasets: two anonymized telemetry datasets collected on Microsoft Teams and SharePoint, one with a user-user recommendation task (855 orgs) and one with a user-post recommendation task (426 orgs)—and three public datasets—coauthorships in 522 different computer science venues extracted from DBLP [37], question answers in 346 Stack Exchange communities [35], and comments in 192 popular Reddit communities [4]. On the two large-scale Microsoft Teams datasets, targeted training on only 5 orgs outperforms per-org training by 6.3%–28.7%, global training by 0.6%–12.6%, and clustered training by 0.2%–8.9% (in average MAP[1] across all orgs and all model types). On the public Stack Exchange data, we find that excluding orgs from training can dramatically improve robustness over global training: while it doesn't benefit the mean MAP for a simple logistic regression baseline, targeted training improves mean MAP by 84% for a more noise-sensitive GNN. In the Reddit data, we find a particularly skewed distribution of org noise, allowing top-1 targeted training to outperform global training by up to 19% in mean MAP. Finally, we discuss conditions under which different multi-org recommendation frameworks perform best. For instance, in the DBLP co-authorship data, we see high consistency across venues, resulting in equivalent performance from global, clustered, and targeted training.

To summarize, we make the following three contributions:

— We show that existing multi-org recommendation frameworks are often suboptimal and that maximizing training data is not always advisable.

[1]mean average precision, a widely used ranking metric [46].

— We propose a multi-org recommendation approach, *targeted training*, that can improve recommendation across orgs.
— We provide extensive empirical and theoretical evidence for the efficacy of targeted training.

2 MULTI-ORG RECOMMENDATION

We begin by formalizing the multi-organization recommendation problem and describing existing approaches. A *multi-organization recommendation* setting consists of a collection of n organizations labeled $1, \ldots, n$, each of which has a set of users $\mathcal{U}_i$ and a set of items $\mathcal{I}_i$ over which we wish to make recommendations. Note that in the case of user-user recommendations (e.g., "people you may know"), $\mathcal{I}_i = \mathcal{U}_i$. Crucially, the organizations are disjoint, with no overlap in users or items between orgs: for any $i \neq j$, $\mathcal{U}_i \cap \mathcal{U}_j = \mathcal{I}_i \cap \mathcal{I}_j = \emptyset$. In each organization i, we want to identify items in $\mathcal{I}_i$ to recommend to each user in $\mathcal{U}_i$ (for instance, this may take the form of a personalized ranking). The disjoint nature of orgs necessitates *inductive* recommendation models that learn patterns applicable to unseen users and items, in contrast to *transductive* models that learn user- or item-specific parameters (as in collaborative filtering).

2.1 Existing Multi-org Training Frameworks

Per-org training. One simple way to make multi-org recommendations is to maintain a separate model for each organization, with each model trained on data from its org. This is beneficial if there is significant heterogeneity in user behavior among orgs. However, this requires each organization to have sufficient high-quality data for training. Additionally, per-org training also requires maintaining many sets of separate parameters, which is not feasible in cases like Microsoft Teams, where there are a huge number of organizations. As such, we focus our attention on other approaches, but we use per-org training as one of our baselines for completeness.

Global training. To avoid the maintenance costs of training and deploying many separate models, a natural solution is to train on combined data from all orgs (or a random subsample, if resource constraints demand it). This is the standard training framework in the industry [10]. Global training has many advantages: it uses the maximum amount of data, is very simple, and produces a single model. However, it ignores heterogeneity in user behavior. More subtly, it also overlooks variation in data quality. In our experiments, we find this second point to be more important. Including data from all organizations can significantly degrade recommendation quality.

Clustered training. A middle ground between per-org and global training is clustering, where a single model is used per cluster of organizations. This potentially shares the advantages of both frameworks, combining data from multiple sources while also parsimoniously accounting for heterogeneity. However, clustering also dilutes these advantages, increasing complexity and maintenance over global training while allowing for less customization than per-org training. Clustering can be performed based on org meta-information, if it is available, or based on per-org model transfer performance (using the idea that orgs in the same cluster should have models that transfer well to each other).

3 REAL-WORLD EVIDENCE FOR SUPER ORGANIZATIONS

Which multi-org recommendation frameworks are best suited to an application depends on the collection of organizations and the recommendation task. For instance, if users behave very consistently across orgs, then clustering and per-org training are rendered less useful. In our recommendation settings, we found strong evidence for *super organizations* whose data produce models that generalize extremely well across all organizations, enabling our new approach to multi-org recommendation.

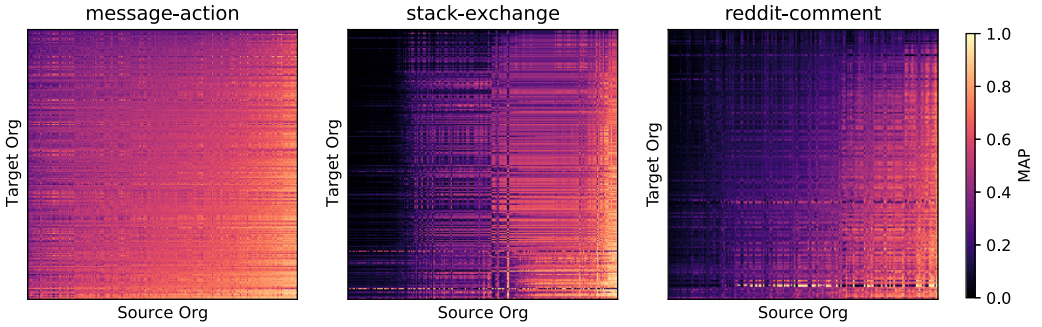


Fig. 1. Performance of every per-org logistic regression model on every target organization for MESSAGE-ACTION, STACK-EXCHANGE, and REDDIT-COMMENT. Source orgs are sorted left to right by mean MAP over targets, while target orgs are sorted top to bottom by mean MAP over sources. The light vertical regions on the right sides indicate the presence of "super orgs" whose models generalize well across all targets.

We trained three types of inductive recommendation models (described in more detail in Section 6.4) on a sample of organizations that use Microsoft Teams in order to recommend posts users are likely to engage with (we call this dataset MESSAGE-ACTION). We also trained the same models to recommend questions on Stack Exchange sites that a user might wish to answer (here, each Stack Exchange site serves as an org) and to recommend posts on Reddit (where subreddits are treated as orgs). Our datasets, including two others we discuss later, are described in more detail in Section 6. Due to the need for privacy between orgs in the Microsoft Teams setting, we only use non-content data for these recommendation models, such as recency, activity, and structural network features (described in detail in Section 6). We trained models per-org and evaluated each org's model on each other org's data. In Figure 1, we show heatmaps for the cross-org performance of the logistic regression models, measured by **mean average precision** (**MAP**).

We found that some orgs produced models that performed well across the board, while other orgs produced models that performed poorly across the board (notice the light vertical strips on the right side of the heatmaps, representing "super orgs" whose models perform well on all targets). We found no evidence of subclusters with distinct user preferences, although there are clusters of orgs producing worse models, especially in STACK-EXCHANGE and REDDIT-COMMENT. We note that the high apparent similarity in user preferences across orgs is likely driven by the activity-based item featurizations that are standard in inductive recommendation [10]: while preferences for specific content may vary widely, users everywhere are more likely to engage with items that are recent and popular. Two key findings (described in more detail in Section 7) are that (1) super orgs' models perform even better than a global model trained on combined data from all orgs and (2) super orgs' models also perform much better than a target org's own local model. In light of these results, our goal is to best aggregate data across super orgs.

4 TARGETED TRAINING

Based on our previous observation, we develop a new approach to multi-org recommendation: *targeted training*. In targeted training, we identify a small subset of k orgs on which high-quality models can be trained. We then only use those k orgs for training a single final model and apply it globally. To perform the initial org selection, we train models per-org and measure their cross-org performance on validation sets (we use MAP, but any desired metric could be used). We then select the source orgs with the highest mean performance over all targets and use those to train a final model (see Algorithm 1 for pseudocode). Since targeted training results in one final model, it shares

ALGORITHM 1: Targeted training for multi-org recommendation.

Input: training and validation data for n orgs (actions by users in $\mathcal{U}_i$ on items in $\mathcal{I}_i$ for each org i), k, p
$T \leftarrow$ a p-fraction subsample of the orgs
for org i in T **do**
    train model $\mathcal{M}_i$ on i
    for org j in T **do**
        evaluate $\mathcal{M}_i$ on j's validation data
    $\overline{f}(i) \leftarrow$ mean performance of $\mathcal{M}_i$ over all j
$T^* \leftarrow k$ orgs i with highest $\overline{f}(i)$
return model trained on combined data from T^*

a low maintenance cost with global training. However, there is an added computational cost to the initial org selection phase. This can be reduced if the fraction of super orgs is sufficiently high that a small subsample contains good training candidates. We find that a relatively small sample often suffices.

Since the approach of discarding training data from some orgs is counter-intuitive, we analyze a simplified scenario where training on global data is suboptimal. We show that using data from only a few orgs can produce better model estimates and that model selection on a validation set can identify desirable orgs for training. This setting relies on different orgs exhibiting different amounts of noise in their data, a defining feature of our real-world data. Noise in recommendation data can manifest in many forms—in the theoretical analysis below, we show that even simple zero-mean noise from temporal fixed effects (i.e., effects that are shared by all orgs at a given timestep but vary over time) is sufficient to induce bad performance from global training. Such noise across orgs could arise in real-world message recommendation scenarios due to current events (e.g., people discussing recent sporting events or elections), disruptions due to weather or natural disasters (e.g., global pandemics), or seasonal effects (e.g., tax seasons). We emphasize that many other forms of noise could also contribute to the poor performance of global training; our goal with the analysis below is to demonstrate one way in which targeted training can outperform global training in a plausible setting. Our simulation results in Section 5 and our real-world results in Section 7 back up this claim.

4.1 Theoretical Model

Suppose we have organizations $1, \ldots, n$. We will model users in an org i at time period t as having stochastic preferences $\boldsymbol{\theta_{it}} \in \mathbb{R}^d$ (for the sake of simplicity, we assume all users in an org have the same preferences). When a user encounters an item j described by features $\boldsymbol{x_j}$, the user interacts with j with some probability $p(\boldsymbol{\theta_{it}}, \boldsymbol{x_j})$. We model global temporal effects on user preferences, such as those caused by current events and seasonality, using zero-mean time fixed effects around a global preference vector $\boldsymbol{\theta^*}$, where the time fixed effect has different magnitudes in different orgs. That is, we consider the model $\boldsymbol{\theta_{it}} = \boldsymbol{\theta^*} + \sigma_i \boldsymbol{\delta_t}$, where $\mathrm{E}[\boldsymbol{\delta_t}] = 0$ and σ_i controls how much the time fixed effect $\boldsymbol{\delta_t}$ influences org i. Intuitively, this model describes a scenario where users across all orgs share some basic patterns in their preferences (such as a preference for recent posts), but some users in some orgs are more strongly impacted by temporal events. For instance, a tax attorney's office is likely more strongly affected by tax seasons than the average org. For the sake of tractability, we assume the fixed effects $\boldsymbol{\delta_t}$ are independent across t. Note that $\boldsymbol{\theta^*}$ is a fixed vector, while $\boldsymbol{\delta_t}$ and, by extension, $\boldsymbol{\theta_{it}}$ are random vectors. Our goal is to learn $\boldsymbol{\theta^*}$ with observations from every org, but from few time periods. We consider an idealized model training

procedure that, given observations from users with preferences $\boldsymbol{\theta_1}, \ldots, \boldsymbol{\theta_m}$, produces the estimate $\frac{1}{m}\sum_{j=1}^{m}\boldsymbol{\theta_j}$.

Suppose we observe m interactions from each org in two time periods 1 and 2. Training a global model on these observations is equivalent to training a model on $2mn$ observations of an org with the mixture of preferences $\tilde{\boldsymbol{\theta}} = \frac{1}{2n}\sum_{i=1}^{n}(\boldsymbol{\theta_{i1}} + \boldsymbol{\theta_{i2}}) = \frac{1}{2n}\sum_{i=1}^{n}[2\boldsymbol{\theta^*} + \sigma_i(\boldsymbol{\delta_1} + \boldsymbol{\delta_2})]$. Notice that $\tilde{\boldsymbol{\theta}}$ has mean $\boldsymbol{\theta^*}$, but we can make the variance of $\tilde{\boldsymbol{\theta}}$ arbitrarily large by increasing some of the σ_i. By our simplified model training procedure, we can thus make the variance of the global model's estimate of $\boldsymbol{\theta^*}$ arbitrarily high. In particular, we can make it higher than the variance of a model trained only on data from the org(s) with smallest σ_i. In these settings, we would want to only train a model on orgs with small σ_i.

How do we identify such orgs with data from two time periods? Intuitively, orgs with low noise will produce period 1 estimates closer to those in period 2, since the time fixed effects are independent and zero mean. More formally, consider comparing each org's model estimate from period 1 to the estimates of all orgs in period 2. An org i's model estimate from time period 1 is $\boldsymbol{\theta^*} + \sigma_i\boldsymbol{\delta_1}$. Moreover, the expected estimate of any model from period 2 is $\boldsymbol{\theta^*}$, since $\boldsymbol{\delta_t}$ is zero-mean. Thus, given a fixed period 1 estimate, the expected difference between org i's estimate in period 1 and period 2 is $|(\boldsymbol{\theta^*} + \sigma_i\boldsymbol{\delta_1}) - \boldsymbol{\theta^*}| = \sigma_i\boldsymbol{\delta_1}$. Thus the org with the smallest σ_i has the smallest expected difference between its period 1 and period 2 estimates. We can therefore select the orgs whose models transfer best between periods 1 and 2 and expect them to have small σ_i.

This is exactly the idea behind targeted training, where the validation set acts as period 2 data to select orgs whose data lead to generalizable models. Notice that this analysis predicts that targeted training outperforms global the most when a sufficient number of orgs are noisy enough to disrupt the global model's estimate, but we also need enough low-noise orgs to have clean data for targeting and to be able to identify them through cross-validation. While this argument uses a simplified view of model training, we demonstrate through simulation in Section 5 that an actual model training procedure exhibits this phenomenon. Our simulation also demonstrates the surprising phenomenon that using a second time period for model selection through targeted training can perform better than using that time period as training data.

5 SIMULATION EXPERIMENT

Our theoretical analysis of targeted training indicates that its performance depends on the distribution of data noise across organizations. To investigate this more thoroughly, we simulate collections of organizations with different data distributions. Specifically, we simulate a binary user-item interaction prediction problem, where each item j is described by an observed vector of features $\boldsymbol{x_j} \in \mathbb{R}^d$ and users have a global preference vector $\boldsymbol{\theta^*} \in \mathbb{R}^d$. However, we assume that a fraction of the orgs (the *noisy* orgs) experiences time fixed effects over preferences, so that their users' preference vector at time t is $\boldsymbol{\theta^*} + \boldsymbol{\theta_t}$. We model the probability of interaction between a user in org i and item j at time t as $S(\boldsymbol{\theta_{it}}^T\boldsymbol{x})$, where S is the sigmoid function, $\boldsymbol{\theta_{it}} = \boldsymbol{\theta^*}$ if i is a clean org, and $\boldsymbol{\theta_{it}} = \boldsymbol{\theta^*} + \boldsymbol{\theta_t}$ if i is a noisy org. We assume that training data is from a single time period for all orgs and validation data is from another time period. The goal is to recover $\boldsymbol{\theta^*}$ by observing labeled user-item interactions across all the simulated organizations in the training and validation sets, without knowing which orgs are noisy. In the simulation, as in our theoretical argument above, there are no systematic patterns in the time fixed effects, so learning $\boldsymbol{\theta^*}$ is the best we can hope to do for prediction in future time periods.

5.1 Simulation Details

We begin by drawing the global user preference vector $\boldsymbol{\theta^*} \sim \mathcal{N}(0, I)$. We use dimension $d = 32$ for the user and item vectors. We simulate 100 orgs, of which a p-fraction are noisy. For each

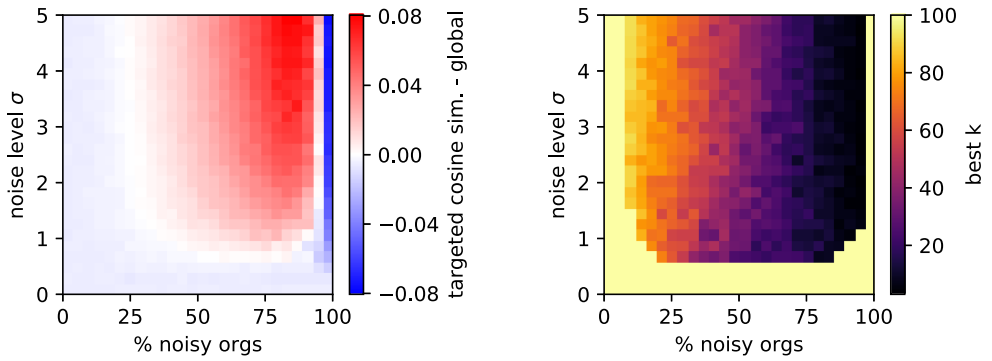


Fig. 2. Performance of targeted training in simulated data as the fraction of noisy orgs and the noise level varies. Left: difference in estimation quality between top-10 targeted and global training, measured by cosine similarity between estimated and true user preferences (red means top-10 targeting beats global). Right: best values of k for top-k targeted training. With a larger fraction of noisy orgs, targeted training on fewer orgs performs better.

dataset (training and validation), we draw a time fixed-effect $\boldsymbol{\theta_t} \sim \mathcal{N}(0, \sigma^2 I)$, where σ controls the magnitude of the noisiness. We then simulate 100 user-item interactions per org. For each interaction in org i, we draw an item vector $\boldsymbol{x_j} \sim N(0, I)$ and label it 1 with probability $\sigma(\boldsymbol{\theta_{it}}^T \boldsymbol{x})$. We ensure every org has at least one positive and one negative sample, redrawing items if necessary. We then perform top-k targeted training for every $k = 1, \ldots, 99$, using mean absolute error on validation to select the targeted orgs and training a final model on combined training data from the targeted orgs. Note that $k = 100$ corresponds to global training. For the predictive model, we use `scikit-learn`'s logistic regression with fully default parameters. To make global training as competitive as possible, we train the global model on combined training and validation data, while targeted training uses the validation set just for org selection. For each combination of the noise level σ and the fraction of noisy orgs, we perform 16 independent trials.

5.2 Simulation Results

To measure the success of different training approaches, we measure the cosine similarity between the learned parameter $\hat{\boldsymbol{\theta}}$ and the underlying global preference vector $\boldsymbol{\theta^*}$. In Figure 2 (left), we show the gap in estimation accuracy for global training and top-10 targeted training as the org distribution parameters vary. When 30–90% of the orgs are noisy, top-10 targeted training provides a more accurate estimate of $\boldsymbol{\theta^*}$, despite training on much less data than the global model: top-10 targeted training uses only 5% as many samples as global training. The performance gap increases with the noise level σ. We also show the best-performing value of k for top-k targeted training (where $k = 100$ is global) on the right of Figure 2. With a larger fraction of noisy orgs, $k \leq 10$ is optimal, but the optimal k gradually increases as more orgs are clean. Note that the amount of noise has little effect on the optimal number of orgs for targeting: regardless of how much noise there is, we want to pick the top 10% of orgs for targeting if 90% of orgs are noisy. If the noise level is too low, if there are too few noisy orgs, or if there are too few clean orgs, global training performs better than targeted training.

6 REAL-WORLD EXPERIMENTS AND RECOMMENDATION MODELS

Having seen that targeted training can perform well in simulated data, we perform experiments on five real-world multi-organization datasets, two collected from Microsoft Teams and SharePoint

(CHAT-COACCESS and MESSAGE-ACTION) and three public datasets (DBLP-COAUTHOR [37], STACK-EXCHANGE [35], and REDDIT-COMMENT [4]). For the two Microsoft datasets, we sampled 1432 anonymized organizations, stratified by active user count to ensure a diverse sample.

6.1 User-user Data: CHAT-COACCESS

We constructed user-user collaboration networks for each org by joining Microsoft Teams chats with file co-accesses on SharePoint for the six-week period from March 1 to April 11, 2021. In this dataset, the task is to recommend likely collaborators to a user from within their org. We form an edge between two users if they either both sent a message to the same thread or if they accessed the same file. We add the number of shared threads and shared files, storing this as the weight of an edge. We use the first two weeks for training, the middle two weeks for validation, and the final two weeks for testing. Within each two-week period, we use the first week (Week 1) to construct features for each user and use these to predict edges in the second week (Week 2). In each two-week period, Week 1 can be thought of as providing the model with the user's recent activity patterns (which we would have access to in a deployed setting), while Week 2 provides the actual prediction instances. For each of the training, validation, and test sets, we remove users who are not active in both Weeks and who do not belong to the largest connected component in the Week 1 graph. Since the Week 1 graph represents recent activity patterns that our models should draw on, we use the Week 1 graph for GNN convolutions. We remove orgs with fewer than 10 active users in any of the six weeks, leaving 855 orgs in CHAT-COACCESS. Finally, we subsample Week 2 edges down to 100 k if necessary.

Next, we sample corresponding negative instances for each positive sample. The corresponding negative sample has the same source user as the positive sample, but a different target user (i.e., item). For computational efficiency, we sample negatives from the full set of users rather than just the targets each source user did not interact with (since that would require sampling over a different support per user, which is expensive). This means that some "negative" samples are just additional positives—but this is rare in data with sparse interactions, and we label them correctly depending on whether they are in fact negatives or real interactions.

We compute two different types of features: *instance features* associated with a particular candidate edge in Week 2 (positive or negative sample) and *node features* associated with nodes in the Week 1 graph. These features are based on standard link-prediction techniques [26, 48]. As instance features of the candidate edge (s, t), we compute (1) the log degree of t, (2) whether the edge (s, t) existed in Week 1, (3) the log weight of the edge (s, t) in Week 1 (or 0), (4) the log weighted degree of t, (5) the PageRank [11] score of t, (6) whether s and t shared neighbors in Week 1, (7) the Jaccard index of s and t in Week 1, and (8) the Adamic/Adar [1] index of s and t in Week 1. Note that while the user-user networks are undirected, the asymmetry in features arises because we are trying to decide which users to recommend to s. For node features, we use PageRank score, log degree, and local clustering coefficient [41].

To evaluate the models using ranking metrics, we also sample (up to) 500 source nodes in the test and validation periods and compute instance features for all edges incident on each sampled source (these are the positive instances for our ranking). Additionally, we sample (up to) 500 negative samples to rank alongside the positives. We store the true label for each instance (edge or no edge in Week 2). We construct ranking datasets for both the validation and test periods (but not training, which uses only the binary classification data).

6.2 User-item Data: MESSAGE-ACTION

We also gathered Microsoft Teams chat react and reply data for each organization from the same period as CHAT-COACCESS. Here, the task is to rank messages posted within a user's teams that they

are likely to be interested in (note that a single org may have many teams in Microsoft Teams, such as HR, Engineering, Sales, etc). We store a timestamped edge from a user node to a message node each time a user reacts or replies to a message. As with CHAT-COACCESS, we split each subset into Week 1 and Week 2, using the graph from Week 1 for the GNNs and taking positive samples from Week 2. Again, we discard orgs with fewer than 10 active users in any week, leaving 426 orgs in MESSAGE-ACTION. For each of the training, validation, and test sets, we also remove data from any team that had actions on fewer than 10 unique messages in Week 2. Finally, we subsample edges in Week 2 down to 100 k if necessary, ignoring actions on the earliest message posted in each team in Week 2 (this will ensure a valid negative sample exists).

As with CHAT-COACCESS, we match up each positive sample (edge in Week 2) with a negative sample. We ensure that a negative sample is a message from the same team as the positive sample and that it existed at the time of the positive sample's action. This is done to ensure the user had the ability to react or reply to the negative sample message. For ranking evaluation, we use the same setup as with CHAT-COACCESS, with one additional constraint: we allow negative samples for source nodes (i.e., users) only from teams in which they were active so that we do not recommend inaccessible messages. As in the binary classification data, we only take samples from Week 2 and ignore the earliest message posted in each team in Week 2. However, we need to assign timestamps to negative sampled actions (which have no timestamps, since they did not occur) in order to compute the temporal instance features described below. To do this, we use a form of hot deck imputation [2], assigning each negative sample to the time of the soonest positive action by the user after the negative was posted (or the end of Week 2, if no subsequent positive exists). As before, we construct ranking data for both the validation and test periods.

Since messages are transient, temporal effects are crucial to recommendations in this dataset. As a temporal distance measure, compute the *recency* of actions (reacts and replies) on a message, defined as $\log^{-1}(2+\text{ seconds since action})$ or 0 if there is no previous action. We use the following instance features for a timestamped interaction between a user u and message m (drawing from timestamped link-prediction methods [30]): (1) whether m had previous actions, (2) the number of previous actions on m, (3) the recency of an action on m, (4) the number of distinct users who have acted on m, (5) whether m had previous actions by u, (6) the number of previous actions on m by u, (7) the recency of any action on m by u, and (8) the recency of m's posting. We use the same node features as in CHAT-COACCESS for simplicity, noting that the clustering coefficient is always 0 in bipartite (e.g., user-message) graphs.

6.3 Public Data: DBLP-COAUTHOR, STACK-EXCHANGE, and REDDIT-COMMENT

To examine our frameworks on public data, we compiled three collections of networks, one with user-user interactions (DBLP-COAUTHOR) and two with user-item interactions (STACK-EXCHANGE and REDDIT-COMMENT). In DBLP-COAUTHOR, the task is to provide a ranking of likely coauthors to someone aiming to publish at a particular computer science venue.[2] In STACK-EXCHANGE, the task is to recommend within-community questions to a user based on the likelihood the user will want to answer them. In REDDIT-COMMENT, the task is to rank posts within a particular subreddit that a user is likely to comment on. In all of these public datasets, there is some amount of user overlap between orgs; however, there is a lack of available public data arising from non-overlapping multi-org settings. These datasets were selected as the closest public analogues to the multi-org recommendation problem encountered in platforms like Microsoft Teams.

[2]This is a somewhat contrived task, but public multi-org user-user data is limited.

To construct DBLP-COAUTHOR, we extracted per-venue coauthorship networks from the ArnetMiner DBLP V13 data[3] [37]. We treated each venue as an organization, forming an edge between each pair of authors who collaborated on a article within a venue (timestamped by year). While there is overlap in authors between venues, we ignore these edges to make this a multi-org recommendation problem. We sort the coauthorships by year for each venue and use the first third for training, the second third for validation, and the final third for testing. We perform the same negative sampling and feature computation procedures as in CHAT-COACCESS (splitting each the train, validation, and test sets into first and second halves to mimic computing features in Week 1 and predicting edges in Week 2). We perform several preprocessing steps. We remove articles if they were missing an ID, title, authors, publication year, venue, references, or citation count. We also remove uncited articles. This filtering reduces the number of articles from 5,354,309 to 2,875,947. We standardize venue names by removing punctuation and ignoring case. We also discard two outlier "venues" in the DBLP data simply named "San Diego, CA" and "San Francisco, CA" (it is not clear why these cities, in particular, are listed as venues for some articles). Finally, to remove the large number of venues with insufficient training data, we filter out venues with fewer than 1,000 articles over their lifetime, resulting in a final count of 522 venues (including, e.g., ICML, KDD, and NeurIPS.). (We need a sufficiently large number of articles for there to be a large connected component in the coauthorship network that meaningfully evolves over time—coathorship networks in smaller venues tend to have tiny components.)

To create the STACK-EXCHANGE dataset, we downloaded the public Stack Exchange Data Dump[4] [35], which contains data from every public Stack Exchange question-answering forum. We consider all question answers to be an action taken by the answerer on the posted question. We split the answers temporally into thirds to form the training, validation, and test sets for each community. The data dump contains a total of 351 Stack Exchange sites, of which 346 remain after filtering out communities with fewer than 10 actions in any of the three sets. We perform the same feature computation steps as in MESSAGE-ACTION, again using the first half of each set as the graph and the second half for instances to predict. We note that each Stack Exchange site has an associated "meta" community where users discuss the forum itself, which we also include in our experiments. The particular data dump we accessed is missing data from `monero.stackexchange.com` due to faulty malware detection on the data hosting site.[5] We filter out actions from deleted users or with timestamps before the question post time (it is not clear how the data contains such instances) and then select the 1 million most recent actions (this only affects the two largest communities, Stack Overflow and Mathematics Stack Exchange). Any community with fewer than 10 actions in any set is discarded, leaving 346/351 communities in the final dataset (the excluded communities are {`tezos`,`eosio`,`cardano`,`stellar`,`iota`}`.meta.stackexchange.com`). As in the other datasets, we cap the number of positive samples in each org at 100k.

Finally, we constructed REDDIT-COMMENT from the Pushshift data dumps[6] [4]. We downloaded all posts and comments from the first six weeks of 2010 and selected the 192 subreddits which had at least 10 posts and 10 distinct contributors in each of the six weeks of data. Since we then have six weeks of posts and comments, we follow the same data processing and feature computation procedure as for MESSAGE-ACTION.

[3] https://www.aminer.org/citation.
[4] https://archive.org/details/stackexchange; accessed September 7, 2021.
[5] https://meta.stackexchange.com/a/369521.
[6] https://files.pushshift.io/reddit/.

6.4 Inductive Recommendation Models

To demonstrate the flexibility of targeted training, we test three different kinds of inductive recommendation models: logistic regression, a **graph convolutional network** (**GCN**) [21], and a state-of-the-art recommendation model, IGMC [49]/SEAL [48] (SEAL is designed for link prediction, while IGMC is designed for recommendation, but they are essentially the same). All three models are trained as binary classifiers for whether a user interacts with an item. The confidence scores of the models are used to rank items.

6.4.1 Logistic Regression. We train a binary logistic regression model for each dataset using the instance features described above and binary cross-entropy loss. We use the Rprop optimizer [31] with no minibatching, initial learning rate 0.05, and grid search over $L2$ regularization weights $0.001, 0.01, 0.1$, and 1, picking the value resulting in lowest validation loss.[7] We train for 500 epochs or until the squared gradient magnitude falls below 10^{-8}.

6.4.2 Graph Convolutional Network. We use an inductive GCN architecture to ensure that a model can be applied to a different organization than it was trained on. To ease training, we pre-compute three features per node: log degree, PageRank score [11], and local clustering coefficient [41]. We feed these node features into a two-layer GCN [21] with tanh activation and dropout. We concatenate the embeddings produced by the first and second layers. We then feed the GCN embeddings of candidate source-target pairs concatenated with the instance features of that pair into a two-layer MLP that produces final prediction scores. In our implementation, we use hidden dimension 8 and output dimension 8 for the GCN. The hidden dimension of the MLP is equal to its input dimension. To train the GCN, we also use batch Rprop with an initial learning rate of 0.05 and binary cross-entropy loss. We perform a grid search over $L2$ regularization weights $10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}$ and dropout probabilities $0.1, 0.25, 0.5$. When training on data from multiple orgs, we fix the GCN dropout rate at 0.5 since we found it not to have an effect on performance. As before, we train for 500 epochs or until the squared gradient magnitude dips below 10^{-8}.

6.4.3 IGMC. We apply a method based on **Inductive Graph-based Matrix Completion** (**IGMC**) [49], a state-of-the-art recommendation method. The key idea behind IGMC is to extract and label subgraphs relevant to a candidate link before passing it through a GCN. Crucially, IGMC is fully inductive, since node labels within each subgraph depend only on the network relationship between the node and the candidate link. We apply the same training procedure as for the GCN—the only difference between the GCN and IGMC is the labeled subgraph extraction.

6.4.4 Training on Data from Multiple Organizations. Because GCNs operate over a graph, we iterate through orgs, taking a small number of optimization steps on each one. We treat the number of optimization steps per org as a hyperparameter, selected from $1, 5, 10$. During a second pass over orgs, we store model parameters after the optimization steps on each org and average them to obtain final model parameters. This ensures that parameter settings are not dominated by the last few orgs.[8]

6.5 Baselines

For each of the inductive models described above, we compare targeted training to three baselines: per-org, global, and clustered training. Per-org and global training are straightforward: we either

[7] Note that we do not search over learning rates since Rprop adapts learning rates per-parameter—we found it to converge quickly regardless of the initial learning rate.

[8] In preliminary experiments, we found that this parameter-averaging approach also tended to outperform using the final parameters.

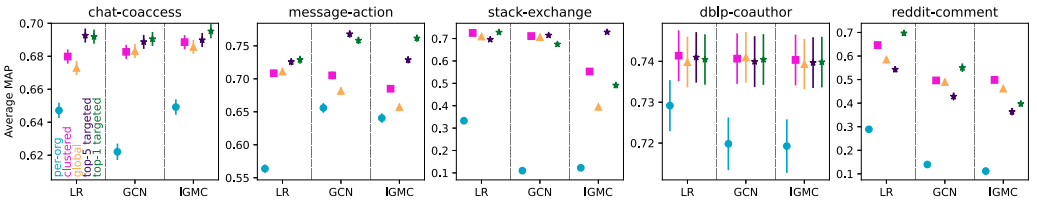


Fig. 3. Average MAPs of different multi-org training frameworks. Errorbars (in some cases tiny) show standard error. Targeted training, either top-1 or top-5, often outperforms global, clustered, and per-org training.

train one model for each org on its own data or train a single model on combined data from all orgs, respectively. For clustered training, we group organizations based on the transfer performance of their per-org models. We use spectral co-clustering [16] on the matrix of cross-org validation MAPs to identify clusters of source orgs whose models transfer with high MAP to clusters of target orgs. We then apply models trained on the source clusters to the target clusters. We test using 1-5 clusters (where cluster count 1 is global training).

7 RESULTS

7.1 Performance Comparison

We compare the performance of top-k targeted training (with $k = 1, 5, 10, 25$, and 50 training orgs) to our baselines in terms of MAP. We find that targeted training consistently matches or beats global and clustered training, even when only training on 1–5 target organizations (in contrast, global trains on combined data from hundreds of orgs) (see Figure 3). This is most evident in MESSAGE-ACTION, where top-5 targeted training improves GCN performance by 12.6% over global training. Without targeted training, the GNN methods are unable to outperform logistic regression on MESSAGE-ACTION. Additionally, in STACK-EXCHANGE, IGMC performs particularly poorly with per-org (MAP 0.12), clustered (0.55), and global (0.40) training, but top-5 targeted training (0.73) allows it to surpass logistic regression and GCN. All frameworks are similarly competitive on DBLP-COAUTHOR dataset (this dataset captures very different behavior than the other four, and over much longer time scales). On REDDIT-COMMENT, the GNN methods perform particularly poorly. However, for both logistic regression and GCN, top-1 targeted training performs better than any other framework, beating the global MAP by 19% and 12%, respectively. As we will see, the especially skewed distribution of org model performance in REDDIT-COMMENT results in top-1 targeting beating top-5.

We also investigated the effect of cluster count on clustering performance. Increasing the cluster count has no consistent effect, except a slight decrease in average MAP after 2 clusters in CHAT-COACCESS. This indicates that there is little cross-org behavioral heterogeneity in these datasets, resulting in no benefit from having different models for different organizations.

7.2 A Closer Look at Targeted Training

In Figure 4, we show the effect of k in top-k targeted training. For all three MESSAGE-ACTION models, STACK-EXCHANGE IGMC, and REDDIT-COMMENT LR, targeted training performance degrades as we train on more organizations (interestingly, these are the three user-post datasets). In the two user-user datasets, CHAT-COACCESS and DBLP-COAUTHOR, performance is roughly constant up to $k = 50$.

To better understand the generalizability of targeted training, we test how well it performs when we use only a subset of orgs for model selection and then test it on held-out orgs. Using the matrix of all cross-org MAPs, we select a random subset of training orgs and pick the best (and 5th best) performing org in mean MAP within that subset. We then measure the mean MAP of that model on the held-out orgs. We repeat this sampling procedure 64 times (results for MESSAGE-ACTION

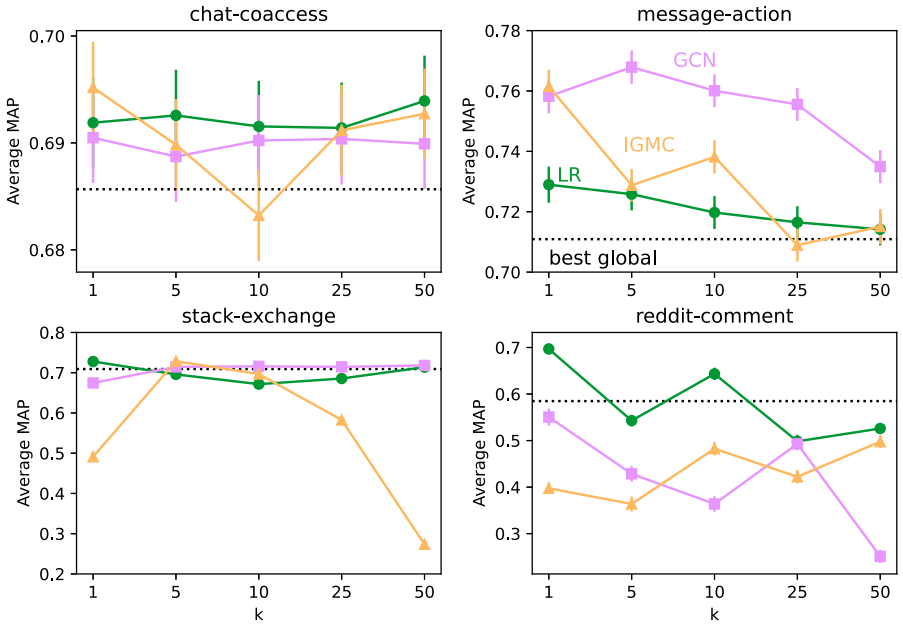


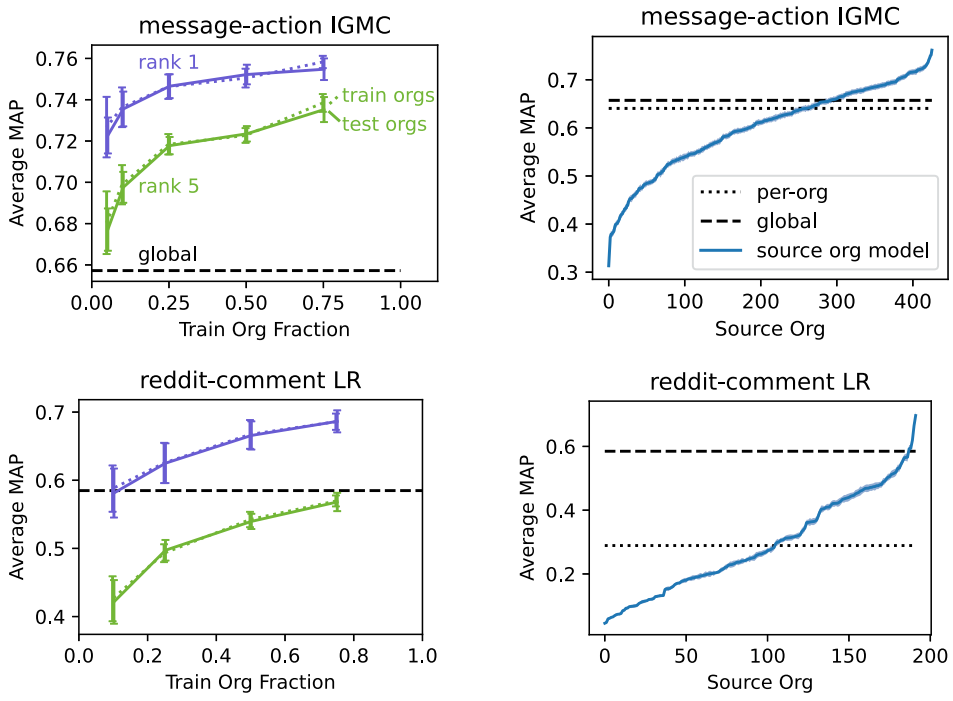
Fig. 4. Effect of the number of orgs k selected for targeting on top-k targeted training. The dotted line shows the best-performing globally trained model. We omit DBLP-COAUTHOR from this figure since it has constant performance across k for all models.

IGMC and REDDIT-COMMENT logistic regression in Figure 5(a)). Across the board, we found that held-out org performance was highly consistent with train org performance. Additionally, top-1 targeting on even a small subset of orgs (down to 5%) performed very well, beating global training in MESSAGE-ACTION for all three models. How small of a subsample is sufficient depends on the fraction of high-data-quality orgs. In REDDIT-COMMENT, note that only the top-ranked model beats global, and it needs more than 5% of orgs for training, indicating that there are fewer high-data-quality orgs in this dataset. We can visualize the distribution of org data quality by plotting the distribution of average cross-org MAPs of each org's model (Figure 5(b)). In MESSAGE-ACTION, we found about 30% of orgs had models beating global training, while in REDDIT-COMMENT, fewer than 5% of orgs had models beating global. This reveals why top-1 targeting was necessary in REDDIT-COMMENT, while top-5 worked well in MESSAGE-ACTION. The distributions in Figure 5(b) emphasize why we need to perform targeted targeting on a small number of orgs rather than simply discarding a small number of outliers: most orgs have very noisy data. Rather than finding a small number of outliers, we need to find a small number of super orgs.

7.3 Organizations Producing the Best Models

We observed that models trained on certain "super orgs" can perform very well on all targets—the natural followup questions: which orgs and why? To answer these questions, we perform linear regressions on the mean transfer performance of each organization's model. We consider the following network and data features (in the training set) for each organization: log node count, log edge count, log community count,[9] modularity [29], approximate diameter,[10] 25th- and

[9]We use the max-modularity Leiden algorithm [38] to cluster the networks and count communities. We also use the Leiden algorithm clustering to compute modularity.
[10]Found by computing eccentricity (max dist. to another node) for 1 k sampled nodes.



(a) Hard transfer experiment

(b) Transfer MAP distribution

Fig. 5. MESSAGE-ACTION IGMC (top row) and REDDIT-COMMENT logistic regression (bottom row) hard transfer experiments (left column) and transfer MAP distributions (right column). (a) In 64 trials, we select a random subset of train orgs. We plot the average MAP of the best and 5th best model from the train subset (dotted line: MAP on train orgs; solid line: MAP on test orgs). Error bars show standard error. MAP on test orgs is consistent with train orgs, and a small subsample suffices to beat global training. (b) Average MAP of each org's model over all targets (std. err. shaded). Source orgs are sorted by average MAP. For MESSAGE-ACTION IGMC, 31% of orgs have models better than global, while only 3% do for REDDIT-COMMENT logistic regression.

75th-percentile degree, max degree, degree assortativity [28], component count, and number of training samples. Additionally, we compute log node count, log community count, and modularity in the largest connected component. In CHAT-COACCESS and DBLP-COAUTHOR, we also use average, 25th percentile, and 75th percentile clustering coefficient [41] (which is always 0 in bipartite graphs like in MESSAGE-ACTION and STACK-EXCHANGE). In MESSAGE-ACTION and STACK-EXCHANGE, we include the log user count and log post count. Finally, we also include an organization's log team count for MESSAGE-ACTION. We standardize all features so that regression coefficients can be interpreted as the effect of a one-standard-deviation change on average transfer MAP.

We find several significant coefficients in CHAT-COACCESS. Organizations with fewer but more interconnected members tend to produce better models (small $|V|$, large $|E|$). This is also reflected in the positive coefficient for average clustering coefficient. In MESSAGE-ACTION, only log sample count is significant at $p < 0.001$ (for GCN and IGMC), with coefficients -0.14 (0.02) and -0.10 (0.02), respectively. However, team count ($p = 0.002$) and user count ($p = 0.001$) are on the margin for MESSAGE-ACTION logistic regression, with coefficients 0.02 (0.01) and -0.07 (0.02). Just as in CHAT-COACCESS, having more users surprisingly results in worse models—even more surprising

are the large negative coefficients for sample count. This may be due to larger organizations with more data having more internal variation in user behavior (e.g., there are likely more distinct roles in a larger company). In the public datasets, no coefficients are significant at $p < 0.001$, and r^2 values are all below 0.2.

Overall, there is some evidence that smaller, more tightly connected orgs result in better user-user recommendation models. However, the correlations are relatively weak. In some datasets, we could not predict transfer performance from our data and network features. This motivates the direct use of validation performance as the model selection criterion in targeted training.

8 RELATED WORK

The literature on recommender systems (see [8, 42, 52] for surveys) has largely focused on collaborative filtering [5, 6, 15, 18, 22, 33], content-based methods [27, 56, 57], and hybrid approaches combining the two [40, 47, 53]. As discussed in this article, neither collaborative filtering nor content-based methods are well-suited to multi-org recommendation due to the need for inductive models and the non-overlapping nature among users and items. There are some purely feature-based recommendation systems that have the potential to be applied to multi-org settings (e.g., [13]), but their applicability has not been comprehensively studied.

A related problem is cross-domain recommendation [17, 58], but orgs differ from domains in that they are disjoint but have the same recommendation tasks. Cross-domain methods can be divided into embedding-based, rating pattern-based, and content-based transfer learning [58]. Many cross-domain methods rely on user overlap between domains [20, 44, 54, 55], preventing their application to multi-org recommendation. Methods that do not rely on user overlap (e.g., [12, 51]) make other assumptions about known relationships between domains, such as social network information or common subpopulations. One recent article on cross-domain recommendation by Krishnan et al. [23] addresses a similar scenario to targeted training: non-overlapping one-to-many multi-domain transfer learning. However, the approach taken by Krishnan et al. is different, exploiting shared contextual information across domains (e.g., identifying items popular on weekends). Local recommendation [14, 24] also deals with a related setting where different models are applied to different subgroups of users, but these methods are transductive and rely on user or item overlap between groups [45].

Another related setting is federated learning [25, 43], where a collection of orgs collaborate to train models together. However, the key distinction is that orgs in federated learning each operate independently and handle their own data and training. The lack of trust between orgs forms the basis of all federated learning methods, such as differential privacy and secure multiparty computation [39]. In contrast, in multi-org recommendation, there is a central service provider with access to all orgs' data, which allows a single unified approach to model training without the same issues of trust.

While multi-org recommendation has not received attention in the literature, it is of significant importance in industry. A talk on Slack's approach to multi-org recommendation was given at RecSys 2018 [10], describing a global metadata-based pipeline. The authors emphasized the importance of privacy and the ensuing need to ignore content-based features. However, the talk did not address alternate frameworks, such as clustering and targeted training. Our use of validation sets for recommendation model selection is inspired by focused learning [7], where the goal is to improve recommendations for under-served items. There is a rich literature on cross-validation for model selection [3, 36]—targeted training is closely related to multi-fold cross-validation [50]. However, note the difference between sampling subsets of a dataset for model training (the standard cross-validation paradigm) and cross-org validation, since orgs actually have data drawn from different distributions.

9 DISCUSSION

Our initial steps in exploring multi-org recommendations were necessarily limited in scope. For instance, our models were all trained as binary classifiers and then used for ranking. Future work could apply targeted training to multinomial ranking or choice models. Additionally, we focused on macro-average model performance across organizations—there may be cases where micro-average performance (or some other weighting) is of interest. There are also a number of important practical considerations for the production use of targeted training that merit further investigation, such as the possible drift of optimal target orgs over time—it may be necessary to retrain (and pick new target orgs) periodically, although this would likely depend on the particular application. Likewise, the optimal number or training targets k may vary between application settings; luckily, our experiments suggest that targeting using a small sample of orgs (and picking the best k within this sample) generalizes well the to full collection of orgs.

In our experiments, we found consistently low behavioral heterogeneity, evidenced by single-org targeted training beating out per-org and clustered training. This may not be universal for all multi-org recommendation problems; investigating what types of recommendation tasks and what collections of orgs exhibit more behavioral heterogeneity would be a valuable follow-up. In such instances, we conjecture that a hybrid of clustering and targeted training ("targeted clustering") could perform well. Targeted clustering would consist of selecting a small number of orgs per cluster on which to train cluster-wide models, thus accounting for behavioral heterogeneity between orgs while maintaining the benefit of targeted training in handling variation in data quality. Future work could also investigate how much behavioral heterogeneity there is among users within the same org and how this impacts the quality of that org's data for model training.

As another extension to targeted training, it may be possible to perform training data selection at a level other than the organization. For instance, in MESSAGE-ACTION, selection could be performed for individual Microsoft Teams channels within an org. At an even finer-grained level, one could imagine discarding individual users or even samples from training data that degrade validation (and eventually test) performance. However, this poses a number of considerable challenges, including identifying which samples to discard, avoiding over-fitting, and ensuring fairness to users. Another extension to targeted training, drawing inspiration from inverse-variance weighting in statistical meta-analysis [19], would be to weight data from different orgs in training, placing lower weight on higher-variance orgs rather than discarding their data outright. Methods for determining these weights would require additional investigation. An entirely different strategy for multi-org recommendation is to make globally-trained models more robust to noisy training data—but as we have seen, both simple logistic regression and complex neural models are harmed by low-data-quality orgs.

The privacy of multi-org recommendation also merits further investigation. We sidestepped such issues by ignoring content, but perhaps a more sophisticated method could incorporate content with privacy guarantees (e.g., techniques from federated learning and differential privacy). Although our content-agnostic models possess much less sensitive information, it is important to formally assess the privacy risks of training a model on multiple orgs (as in top-5 targeted training) relative to training on a single org.

REFERENCES

[1] Lada A. Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Social Networks* 25, 3 (2003), 211–230.
[2] Rebecca R. Andridge and Roderick J. A. Little. 2010. A review of hot deck imputation for survey non-response. *International Statistical Review* 78, 1 (2010), 40–64.
[3] Sylvain Arlot and Alain Celisse. 2010. A survey of cross-validation procedures for model selection. *Statistics Surveys* 4 (2010), 40–79.

[4] Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The Pushshift Reddit dataset. In *Proceedings of the ICWSM*. 830–839.
[5] Robert M. Bell and Yehuda Koren. 2007. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Proceedings of the ICDM*. IEEE, 43–52.
[6] James Bennett and Stan Lanning. 2007. The netflix prize. In *Proceedings of the KDD Cup and Workshop*. Vol. 2007, New York, NY, 35.
[7] Alex Beutel, Ed H. Chi, Zhiyuan Cheng, Hubert Pham, and John Anderson. 2017. Beyond globally optimal: Focused learning for improved recommendations. In *Proceedings of the WWW*. 203–212.
[8] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. 2013. Recommender systems survey. *Knowledge-based Systems* 46 (2013), 109–132.
[9] Dirk Bollen, Bart P. Knijnenburg, Martijn C. Willemsen, and Mark Graus. 2010. Understanding choice overload in recommender systems. In *Proceedings of the RecSys*. 63–70.
[10] Renaud Bourassa. 2018. Building recommender systems with strict privacy boundaries. In *Proceedings of the RecSys*. 486–486.
[11] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems* 30, 1-7 (1998), 107–117.
[12] Wei Chen, Wynne Hsu, and Mong Li Lee. 2013. Making recommendations from multiple domains. In *Proceedings of the KDD*. 892–900.
[13] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide and deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. 7–10.
[14] Evangelia Christakopoulou and George Karypis. 2016. Local item-item models for top-n recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 67–74.
[15] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the RecSys*. 191–198.
[16] Inderjit S. Dhillon. 2001. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the KDD*. 269–274.
[17] Ignacio Fernández-Tobías, Iván Cantador, Marius Kaminskas, and Francesco Ricci. 2012. Cross-domain recommender systems: A survey of the state-of-the-art. In *Proceedings of the Spanish Conference on Information Retrieval*. 1–12.
[18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the WWW*. 173–182.
[19] Larry V. Hedges. 1982. Estimation of effect size from a series of independent experiments. *Psychological Bulletin* 92, 2 (1982), 490.
[20] Liang Hu, Jian Cao, Guandong Xu, Longbing Cao, Zhiping Gu, and Can Zhu. 2013. Personalized recommendation via cross-domain triadic factorization. In *Proceedings of the WWW*. 595–606.
[21] Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the ICLR*.
[22] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
[23] Adit Krishnan, Mahashweta Das, Mangesh Bendre, Hao Yang, and Hari Sundaram. 2020. Transfer learning via contextual invariants for one-to-many cross-domain recommendation. In *Proceedings of the SIGIR*. 1081–1090.
[24] Joonseok Lee, Samy Bengio, Seungyeon Kim, Guy Lebanon, and Yoram Singer. 2014. Local collaborative ranking. In *Proceedings of the 23rd International Conference on World Wide Web*. 85–96.
[25] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine* 37, 3 (2020), 50–60.
[26] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the Association for Information Science and Technology* 58, 7 (2007), 1019–1031.
[27] Pasquale Lops, Dietmar Jannach, Cataldo Musto, Toine Bogers, and Marijn Koolen. 2019. Trends in content-based recommendation. *User Modeling and User-Adapted Interaction* 29, 2 (2019), 239–249.
[28] Mark E. J. Newman. 2002. Assortative mixing in networks. *Physical Review Letters* 89, 20 (2002), 208701.
[29] Mark E. J. Newman and Michelle Girvan. 2004. Finding and evaluating community structure in networks. *Physical Review E* 69, 2 (2004), 026113.
[30] Jan Overgoor, George Pakapol Supaniratisai, and Johan Ugander. 2020. Scaling choice models of relational social data. In *Proceedings of the KDD*. 1990–1998.
[31] Martin Riedmiller and Heinrich Braun. 1992. Rprop-a fast adaptive learning algorithm. In *Proceedings of the ISCIS VII), Universitat*. Citeseer.

[32] Aditya Sakhuja. 2021. Building a Multi-tenant Content-based Recommender with Automated Training. Retrieved from https://pycon.blogspot.com/2021/05/building-multi-tenant-content-based.html, accessed 3/1/2023.
[33] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the WWW*. 111–112.
[34] Anu Sivunen and Kaisa Laitinen. 2019. Digital communication environments in the workplace. In *Workplace Communication*, Leena Mikkola and Maarit Valo (Eds.). Routledge, New York, NY, 41–53.
[35] Stack Exchange, Inc. 2021. Stack Exchange Data Dump. Retrieved from https://archive.org/details/stackexchange, accessed September 7, 2021.
[36] Mervyn Stone. 1974. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society: Series B* 36, 2 (1974), 111–133.
[37] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. ArnetMiner: Extraction and mining of academic social networks. In *Proceedings of the KDD'08*. 990–998.
[38] Vincent A. Traag, Ludo Waltman, and Nees Jan Van Eck. 2019. From Louvain to Leiden: Guaranteeing well-connected communities. *Scientific Reports* 9, 1 (2019), 1–12.
[39] Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. 2019. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the AISec*. 1–11.
[40] Xinxi Wang and Ye Wang. 2014. Improving content-based and hybrid music recommendation using deep learning. In *Proceedings of the MM*. 627–636.
[41] Duncan J. Watts and Steven H. Strogatz. 1998. Collective dynamics of 'small-world' networks. *Nature* 393, 6684 (1998), 440–442.
[42] Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. 2022. Graph neural networks in recommender systems: a survey. *Comput. Surveys* 55, 5 (2022), 1–37.
[43] Jie Xu, Benjamin S. Glicksberg, Chang Su, Peter Walker, Jiang Bian, and Fei Wang. 2021. Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research* 5, 1 (2021), 1–19.
[44] Huan Yan, Xiangning Chen, Chen Gao, Yong Li, and Depeng Jin. 2019. Deepapf: Deep attentive probabilistic factorization for multi-site video recommendation. *TC* 2, 130 (2019), 17–883.
[45] Longqi Yang, Tobias Schnabel, Paul N. Bennett, and Susan Dumais. 2021. Local factor models for large-scale inductive recommendation. In *Proceedings of the 15th ACM Conference on Recommender Systems*. 252–262.
[46] Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims. 2007. A support vector method for optimizing average precision. In *Proceedings of the SIGIR*. 271–278.
[47] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the KDD*. 353–362.
[48] Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*, Vol. 31.
[49] Muhan Zhang and Yixin Chen. 2020. Inductive matrix completion based on graph neural networks. In *Proceedings of the ICLR*.
[50] Ping Zhang. 1993. Model selection via multifold cross validation. *The Annals of Statistics* 21, 1 (1993), 299–313.
[51] Qian Zhang, Dianshuang Wu, Jie Lu, Feng Liu, and Guangquan Zhang. 2017. A cross-domain recommender system with consistent information transfer. *Decision Support Systems* 104 (2017), 49–63.
[52] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys* 52, 1 (2019), 1–38.
[53] Shuai Zhang, Lina Yao, and Xiwei Xu. 2017. AutoSVD++: An efficient hybrid collaborative filtering model via contractive auto-encoders. In *Proceedings of the SIGIR*. 957–960.
[54] Yu Zhang, Bin Cao, and Dit-Yan Yeung. 2010. Multi-domain collaborative filtering. In *Proceedings of the UAI*. 725–732.
[55] Cheng Zhao, Chenliang Li, and Cong Fu. 2019. Cross-domain recommendation via preference propagation graphnet. In *Proceedings of the CIKM*. 2165–2168.
[56] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A deep reinforcement learning framework for news recommendation. In *Proceedings of the WWW*. 167–176.
[57] Lei Zheng, Vahid Noroozi, and Philip S. Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the WSDM*. 425–434.
[58] Feng Zhu, Yan Wang, Chaochao Chen, Jun Zhou, Longfei Li, and Guanfeng Liu. 2021. Cross-domain recommendation: Challenges, progress, and prospects. In *Proceedings of the IJCAI*.

Received 28 November 2022; revised 2 March 2023; accepted 21 May 2023