

#### Background: response prediction

In computational advertising, a content publisher (e.g. CNN, AOL, et cetera) is approached by several advertisers who wish for their ad(s) to be displayed. The advertisers bid for a display on the publisher's page by offering some amount of money if the publisher displays their ad, and some action is performed. We'll assume here that the action is the ad being clicked by a user. The publisher decides which ads to show by conducting an auction based on expected revenue. This requires finding the **clickthrough rate** (**CTR**) of each ad, which is the probability of the ad being clicked.



**Response prediction** is the problem of estimating this clickthrough rate for an ad when shown on a publisher page. The straightforward approach to doing so is using the maximum likelihood estimate, viz the empirical probability of an ad being clicked based on historical data. This is generally very noisy, since most ads are displayed only a few (or zero) times on a given page. An alternative is to use classical supervised learning techniques, such as logistic regression, on explicit features for publishers and ads.

### A collaborative filtering approach

This work asks the question: can we use ideas from **collaborative filtering**, a technique of recommending items (e.g. movies, books) to users, to aid in response prediction? The connection between the two problems is evident: pages are "users", ads are "items", and a page's "rating" for an ad is the ad's clickthrough rate:



We will specifically look to import a popular approach to collaborative filtering based on learning latent features for users and movies. The basic idea is that users and movies live in some latent space, and that ratings measure affinity in this space. If the set of observed ratings is  ${\cal O}$ , and  $X_{i\,i}$  is the rating user i gives movie j, then we learn latent features lpha, etavia the regularized objective

$$\sum_{(i,j)\in\mathcal{O}}\ell(X_{ij},\alpha_i^T\beta_j) + \lambda\Omega(\alpha,\beta)$$

for some loss function  $\ell$ , such as square-loss, and regularizer  $\Omega$ , such as  $rac{1}{2}(||lpha||_F^2+||eta||_F^2)$ . Intuitively, we can think of the elements of  $\Omega_i$  as being some latent characteristics of the user *i*, e.g. whether she likes indie-style movies, whether she likes movies with rich orchestration. The corresponding elements of  $\beta_{2}$  measure how strongly these features are represented in the movie *j*.

For response prediction, the latent feature approach is very different to both maximum-likelihood estimation and prediction based on explicit features. A distinct advantage over the maximum-likelihood approach is that we can make sensible predictions even for cells that have limited historical data. The reason is that the latent vectors are estimated based on behaviour across all pages and ads. An advantage over logistic regression, say, is that we attempt to let the data "speak for itself" in terms of determining what characteristics of pages and/or ads influence clickthrough rates.

The reduction to collaborative filtering is not unconditional. First, the "ratings" here have a notion of confidence: we are more confident about the CTR of an ad that has been clicked 50 times from 100 displays than one clicked 1 time from 2 displays. To handle this, we will think of each cell as comprising a number of positive and negative ratings, corresponding to the clicks and views-but-not-clicks respectively. We will now pay more attention to cells with a large amount of historical data.



Second, we are interested in meaningful probabilities of ratings, and not just a good ranking. So, we need to directly model the probability of ad *j* being clicked when displayed on page *i*. Following from the above, we have a table where each cell comprises multiple positive and negative examples. We will then model

$$\hat{P}_{ij} = \sigma(\alpha_i^T \beta_j)$$

where  $\sigma(\cdot)$  denotes the standard sigmoid function, which can be thought of as a matrix analogue to logistic regression. The vectors  $\alpha_i$ ,  $\beta_i$  represent the latent features for page i and ad j respectively. We can now minimize negative loglikelihood on our data under this model. Let  $C_{i,i}$  denote the number of clicks that ad j receives when shown on page i, and let  $V_{i\,i}$  denote the number of views. We can learn latent features lpha, eta by minimizing:

 $-\sum_{(i,j)\in\mathcal{O}} C_{ij} \log \sigma(\alpha_i^T \beta_j) + (V_{ij} - C_{ij}) \log(1 - \sigma(\alpha_i^T \beta_j)) + \frac{\lambda}{2} (||\alpha||_F^2 + ||\beta||_F^2)$ 

With this basic model in place, we now study two important extensions to the factorization model in turn: how to incorporate side-information, and how to incorporate hierarchies. The resulting model will be shown to have superior performance to both LMMH and the feature-based methods.

# Response prediction using collaborative filtering with hierarchies and side-information

Aditya Krishna Menon\*, Krishna-Prasad Chitrapura\*\*, Sachin Garg\*\*, Deepak Agarwal\*\*\* and Nagaraj Kota\*\*

\* UC San Diego, \*\* Yahoo! Labs Bangalore, \*\*\* Yahoo! Research Santa Clara

## Incorporating side-information

Pages and ads possess explicit features other than just their unique identifiers, such as the content of the ad, its placement on the page, et cetera. In collaborative filtering, such features are known as side-information. To incorporate this information into the factorization model, we use a linear combination of the latent features and explicit features:

where  $x_{ij}$  consists of the explicit features for the cell (*i*, *j*). It is clear that our predictions will now be influenced by the side-information, with w measuring the relative importance of the explicit features over the latent features. To train this augmented model, note that it may be rewritten as:

$$\hat{P}_{ij} = \sigma([1; u$$

which is a logistic regression model where the factorization estimates  $\alpha_i^T \beta_i$  are treated as additional input features. This suggests a simple learning strategy: first, train the standard factorization model. Then, feed the features  $x'_{ij} = |\alpha_i^T \beta_j; x_{ij}|$  to a standard logistic regression model. The resulting solution predicts the click probability using both latent explicit information.

Given the above model, a simple iterative procedure can be applied to further improve performance. Let us rewrite the confidence weighted factorization model as

$$\min_{\alpha,\beta} \frac{1}{|\mathcal{O}|} \sum_{(i,j)\in\mathcal{O}} -V_{ij} \left( P_{ij}^{\text{MLE}} \log \hat{P}_{ij} - (1 - P_{ij}^{\text{MLE}}) \log(1 - \hat{P}_{ij}) \right) + \Omega(\alpha,\beta).$$

If a cell has a small number of views, then  $P_{ii}^{MLE}$  will be noisy. This means that our confidence weighting is *itself* noisy. Ideally, we would like to weight each entry by the true click probability; but of course, if we knew this, our learning would be complete. Yet this motivates the following EM-style procedure: we take the predictions from our factorization model, and use these in place of  $P_{ii}^{MLE}$  in the above equation. We now re-learn our factorization model using these new confidences. We can iterate by feeding the results of the newly learned factorization into a logistic regression model, and use the resulting estimates as a fresh set of confidence weights.

### Exploiting hierarchical information

A key challenge in response prediction is the extreme sparsity of data: most ads are not shown on a particular publisher page, and even if they are, they are very rarely clicked. To get reliable estimates of clickthrough rates, we need to exploit special structure in the problem. It turns out that we often have available categorical information about pages and ads that construct a hierarchy over them. For example, we can think of ads as being clustered by their advertising campaign, campaigns as being clustered by their advertiser, and so on. We assume that this hierarchy induces correlations amongst the clickthrough rates, and wish to use this to learn a better model.

The question then is how to use this information to improve our model. We use three simple procedures:

Hierarchical regularization: Our first idea is to let every node in the hierarchy possess its own latent vector, and use this to construct priors that constrain the latent vectors. We specify the prior for each latent vector so that it behaves like its parent node's in expectation. The natural choice is to modify the mean of the Gaussian prior used in  $\ell_2$  regularization from zero to that of the parent vector:



We learn MAP estimates for all latent vectors in the hierarchy, which corresponds to modifying the regularization term so that every latent vector is encouraged to be close to that of its parent node:

$$\Omega(\alpha) = \frac{\lambda_{\alpha}}{2} \sum_{i \in \mathcal{H}^P} ||\alpha_i|$$

To see why the regularizer helps, suppose there are two siblings *u,v* with a common parent, and that node *u* has only a few views while node v has many views. For v, the dominating term in the objective will be the loss function, so its parameters will be optimized to be predictive for the CTR. For *u*, the regularizer will dominate and push its latent vector to be similar to the parent node. In turn, the parent is encouraged to be close to its children, and so *u* will ``borrow strength" from *v*.

Agglomerative fitting: Above, the latent vectors for non-leaf nodes only appear in the regularizer, and hence are only indirectly affected by the click and view data. To do this, we use the hierarchy to agglomerate the click/view data across many pages and ads, and try to predict this data using the appropriate latent vectors. For example, for a (page, campaign) pair (*i*, *c*), we agglomerate the clicks/views for all children of c when shown on page *i*. We model the resulting data using the vectors  $\alpha_i$  and  $\beta_c$ . This will learn a sensible prior for the children's latent vectors.





 $\hat{P}_{ij} = \sigma(\alpha_i^T \beta_j + w^T x_{ij}),$ 

 $^{I} \ \left[\alpha_{i}^{T}\beta_{j};x_{ij}\right]),$ 

 $-\alpha_{\operatorname{Par}(i)}||_{2}^{2} + \frac{\lambda_{R}}{2}||\alpha_{\operatorname{Root}}||_{2}^{2}$ 

**Residual fitting**: We modify this prediction itself based on the hierarchy. Specifically, for the pair (*i*, *j*), we use the prediction  $\sigma( ilde{lpha}_i^T ilde{eta}_i)$ , where  $ilde{lpha}, ilde{eta}$  modify the original vectors lpha, eta based on the hierarchy. A simple choice is the additive model

and similarly for  $\beta$ . Here, the fine-grained latent features for each page are modelled as corrections over coarser latent features of the ancestor nodes, which may be thought of as bias terms.

# Experimental results

We ran experiments on three real-world Yahoo! ad datasets: **PVC** (post-view click), **PCC** (post-click conversion), and **Click** (ad-click). We compared to logistic regression using cross-products of explicit features, and the state-of-the-art LMMH method. There are ~(90B, 3B) (train, test) records for Click, ~(7B, 250M) for PVC, and ~(500M, 20M) for PCC. The three datasets also include *interaction features* for the user involved in each interaction (e.g.) the age and gender of the user that clicks on an ad, how recently the ad was shown to the user, et cetera).

We trained our models using stochastic gradient descent (SGD), and used the MapReduce code from the Apache Mahout project to scale to the challenging sizes of the datasets. We parallelized the optimization by fixing the page latent features  $lpha_i$  and then optimizing for the advertisement latent features  $eta_i$  using SGD. This optimization of each individual  $eta_i$  can be done in parallel.

We find that the basic latent feature method underperforms due to the extreme sparsity of the datasets. Adding sideinformation manages to improve performance, but LMMH still manages to outperform. However, finally combining with hierarchical information yields the best results on all datasets. Further, studying the lifts on the Click dataset after each application of both the factorization and logistic regression models, we note that we almost always improve the loglikelihood at each iteration.



We study log-log plots of the ratio of predictions of our final model and the logistic regression model to the test set CTR, ordered by increasing number of views on the training set. There are two striking characteristics in the plots: first, our model has significantly less variance than the logistic regression model, which shows that its factorization component captures most of the structure in the data through latent features. Second, our model converges much quicker to the true CTR than logistic regression. This shows that our model can successfully smoothen at a much greater degree of sparsity in the training data, corresponding to dyads with a few number of views.



#### Conclusions

Response prediction can be attacked using ideas from collaborative filtering. However, the extreme sparsity of data requires domain-specific adapation. By exploiting hierarchical information about publishers and advertisements, and incorporating explicit information about the same, we show how latent features can give state-of-the-art results for response prediction on real world ad datasets.



$$\tilde{\alpha}_{ik} = \alpha_{ik} + \sum_{u \in \text{Path}(i) - \{i\}} \alpha_{uk}$$