Pólya-Gamma分布を使ったBayesian Factorization Machines

北原　洋一†

† 株式会社リブセンス　〒105–7510 東京都港区海岸 1-7-1 東京ポートシティ竹芝 10 階
E-mail: †yoichi.kitahara@livesense.co.jp

あらまし　Factorization Machines は Matrix Factorization を一般化したモデルで、コンテキスト情報を利用可能な協調フィルタリングに使うことができる。評価情報が 2 値の場合ロジスティック回帰タイプのモデルが使われることが多いが、ロジスティック回帰タイプの Factorization Machines に関するギブスサンプリング法は知られていなかった。本研究では Pólya-Gamma 分布を使ったデータ拡大法を利用して、ロジスティック回帰タイプの Bayesian Factorization Machines のパラメータをギブスサンプリングする方法を提案する。

キーワード　協調フィルタリング，行列分解

1 導　　入

レコメンデーションは、大量の情報の中からユーザーの興味や関心があるものを効率的に提示するのに有用で、さまざまな領域で広く使われている。例えば、オンラインショッピングでは顧客に適切な商品を提案するのに使われているし [1]、人材等のマッチングサービスなどでも利用されている。よりよいレコメンデーションを実現するため、レコメンデーション技術のさらなる発展が望まれている。

協調フィルタリングはレコメンデーションにおいて有効な手法の一つである。協調フィルタリングでは、行動パターンが類似した他のユーザーの評価情報に基づいてアイテムを推薦するため、パーソナライズされた推薦ができるという特徴がある。モデルベース協調フィルタリングを使う場合、評価情報がスコアのような連続値であれば回帰、クリックや購入のような離散値であれば分類として扱う。商品やエンターテイメントコンテンツのレビューなどでは回帰として扱われることが多いが、サービスの KPI に関わる機能では分類問題として扱われることが多い。

協調フィルタリングでは行動データがないあるいは少ないユーザーの扱いが困難であるというコールドスタート問題がある。レビューを利用したエンターテイメントコンテンツのレコメンデーションなどでは継続的なサービス利用が多いためコールドスタート問題の影響は比較的小さい。しかし、人材等のマッチングサービスではサービスの質が向上するほどユーザーの利用期間が短くなるためコールドスタート問題の影響を受けやすい。コールドスタート問題に対処する方法としては、コンテキスト情報を利用可能な協調フィルタリング手法が挙げられる。

Factorization Machines(FM) [2] [3] は Matrix Factorization(MF) [4] を一般化したモデルで、　コンテキスト情報を利用可能な協調フィルタリングに使うことができる。多くの MF 派生モデルを表現可能な表現能力の高いモデルであるにもかかわらず、高速な学習が可能であることが知られている [5]。しかしながら、学習データにオーバーフィットしやすいため、モデルパラメータの推定にはオーバーフィットが生じにくいベイズ推定が有効である。

FM のベイズ推定手法はいくつか提案されている。回帰については、ギブスサンプリング [6] や変分ベイズ [7] [8] が提案されている。分類については、プロビット回帰タイプのギブスサンプリング [3] やロジスティック回帰タイプの変分ベイズ [8] が提案されている。しかし、ロジスティック回帰タイプのギブスサンプリングは知られていない。

本研究では、協調フィルタリングでの利用を想定して、ロジスティック回帰タイプの Bayesian FM のパラメータをギブスサンプリングする方法を提案する。ロジットモデルのパラメータの条件付き分布を導出するために、Pólya-Gamma 分布を使ったデータ拡大法 [9] を利用する。また、[3], [5], [6] と同様に、ギブスサンプリングにおいてはモデルパラメータが一つずつサンプリングされるという特徴を活かした計算量の削減を図る。

2 関 連 研 究

FM は、スパースデータの交互作用を表現するのに優れたモデルである。協調フィルタリングでよく用いられる MF だけでなく、その派生モデルである Pairwise Interaction Tensor factorization [10] や SVD++ [11]、FPMC [12]、BPTF [13]、TimeSVD++ [14] といった幅広い行列分解モデルを表現可能である [3]。さらに、学習データにコンテキスト情報を含めることで、コンテキスト情報を利用可能な協調フィルタリングに使うこともできる。

FM のモデルパラメータを Altrnating Least Square(ALS) もしくはギブスサンプリングによって推定する場合は、事前計算を利用することで高速な学習が可能であることが知られている [5] [3]。FM がモデルパラメータについて線形であることと、ALS のパラメータ更新やギブスサンプリングのサンプリングではパラメータを独立させて扱うことを利用して、更新あるいはサンプリング対象となったパラメータに関わる値の差分のみを計算することで大幅な計算量が削減できる。近年では自動

微分を利用した機械学習フレームワークが充実しているため、様々な機械学習モデルや統計モデルのパラメータ推定が容易になったものの、このようなモデル特有の計算高速化手法を適用するのは難しい。なお、[5] で提案された計算量削減方法は、Stochastic Gradient Descent(SGD) のように一度に複数のモデルパラメータを更新する手法では使えない。さらに、内部に同一のパターンを有しているデータについては、ブロック構造をまとめて計算することで高速に計算する方法も提案されている [15]。

FM をベイズ推定する方法もいくつか提案されている。[6] では、モデルパラメータを表す分布に共役事前分布を仮定することで FM のモデルパラメータをギブスサンプリングする Bayesian Factorization Machines(BFM) が提案されている。モデルパラメータを表す分布のパラメータであるハイパーパラメータも推定することで、オーバーフィットを抑制するためのパラメータ調整を容易にしている。また、[5] で提案された事前計算を利用した高速な計算も利用できるという利点もある。[3] では、分類問題に対応した切断正規分布を利用したプロビット回帰タイプの FM のパラメータをギブスサンプリングする方法が紹介されている。[7] では、FM の変分ベイズ推定がバッチ学習とオンライン学習の二つについて提案されている。[8] では、機械学習フレームワークを使った変分ベイズ推定が提案されており、回帰だけでなく分類問題に対応したロジスティック回帰も扱っている。

FM の派生モデルも提案されている。[16] では特徴量の種類を Field として考慮した拡張がなされている。[17] では、より高次の交互作用を含むモデルの効率的な計算アルゴリズムが提案されている。また、Deep learning への応用もなされている [18] [19] [20]。

3 Bayesian Logistic Fctorization Machines

3.1 Factorization Machines

FM は、特徴量数を M、因子数を K とし、目的変数 $y \in \mathbb{R}$、説明変数 $\mathbf{x} \in \mathbb{R}^M$、モデルパラメータ $w_0 \in \mathbb{R}$、$\mathbf{w} \in \mathbb{R}^M$、$\mathbf{V} \in \mathbb{R}^{M \times K}$ を使って

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{M} w_i x_i + \sum_{i=1}^{M} \sum_{j=i+1}^{M} \hat{w}_{i,j} x_i x_j \tag{1}$$

$$\hat{w}_{i,j} = \sum_{f=1}^{K} v_{i,f} v_{j,f} \tag{2}$$

と表される。式 (1) の第 3 項はそのまま計算すると $O(KM^2)$ の計算量となるが、変形することで $O(KM)$ で計算可能であることが知られている [2]。因子数 K は調整パラメータで、大きくするほど表現能力は高まるが、学習データにオーバーフィットしやすくなる。そのため、データに応じて適切な因子数を選択する。

2 値の分類問題に対応するため、ロジスティック回帰タイプの FM を考える。式 (1) の左辺を改めて $\hat{\psi}(\mathbf{x})$ とおき、目的変数を $y \in [0, 1]$ とし

$$\hat{\psi}(\mathbf{x}) = w_0 + \sum_{i=1}^{M} w_i x_i + \sum_{i=1}^{M} \sum_{j=i+1}^{M} \hat{w}_{i,j} x_i x_j \tag{3}$$

$$\tilde{y}(\mathbf{x}) = \frac{\exp(\hat{\psi}(\mathbf{x}))}{1 + \exp(\hat{\psi}(\mathbf{x}))} \tag{4}$$

としたモデルを本稿では Logistic Factorization Machines(LFM) と呼ぶ。

FM と同様に LFM の式 (3) は、いずれのモデルパラメータについても線形であるため、ある一つのモデルパラメータ $\theta \in \Theta = \{w_0, w_1, ..., w_M, v_{1,1}, ..., v_{M,K}\}$ のみに着目すると

$$\hat{\psi}(\mathbf{x}) = g_\theta(\mathbf{x}) + \theta h_\theta(\mathbf{x}) \tag{5}$$

と表記できる。$h_\theta(\mathbf{x})$ は

$$h_\theta(\mathbf{x}) = \begin{cases} 1 & (\theta = w_0) \\ x_l & (\theta = w_l) \\ x_l \sum_{j=1}^{M} v_{jf} x_j - v_{lf} x_l^2 & (\theta = v_{l,f}) \end{cases} \tag{6}$$

となる。ギブスサンプリングでは一つずつモデルパラメータをサンプリングするので、式 (5) を使うことでモデルパラメータを見通しよく統一的に扱うことができる。そこで、本稿でも式 (5) の表記を利用する。

3.2 Pólya-Gamma 分布を使ったデータ拡大法

ロジットモデルは解析的な扱いが難しいが、[9] にて Pólya-Gamma 分布を使ったデータ拡大法によって効率的なベイズ推論が可能であることが示されている。[9] では、次の関係が証明されている。

$$\frac{\left(e^{\psi}\right)^a}{\left(1 + e^{\psi}\right)^b} = 2^{-b} e^{\kappa\psi} \int_0^\infty e^{-\omega\psi^2/2} p(\omega) d\omega \tag{7}$$

ここで、$\kappa = a - b/2$ であり、$p(\omega) = PG(\omega|b, 0)$ は Pólya-Gamma 分布である。式 (7) の左辺はロジットモデルに類似した形になっており、右辺は ω を与えたとき ψ について正規分布のカーネルになっていることを利用する。a を 2 値の目的変数 $y \in [0, 1]$ とし、$b = 1$ として、1 サンプルの条件付き分布は

$$P(y|\psi) = \frac{\left(e^{\psi}\right)^y}{1 + e^{\psi}} \tag{8}$$

$$\propto \exp\left(\kappa\psi - \frac{\omega\psi^2}{2}\right) \tag{9}$$

となる。このとき、潜在変数 ω の分布は

$$P(\omega|\psi) = PG(1, \psi) \tag{10}$$

となる。Pólya-Gamma 分布はガンマ分布の無限和となっているが、 $PG(1, \psi)$ から効率的なサンプリングが可能であることが知られている [9]。本稿でも、式 (9)、(10) の関係を利用してギブスサンプリングに必要な条件付き分布を導出する。

ロジットモデルであれば式 (9) の関係を広く利用できるため、多くの応用事例がある。例えば、MF を含む因子モデル [21] や Tensor Factorization [22] にも利用されている。

3.3 Bayesian Logistic Factorization Machines

本節では、Bayesian Logistic Factorization Machines(BLFM) のモデルと、ギブスサンプリングに必要な条件付き分布を示す。モデルパラメータの事前分布は [3] とほぼ同じものを用い、モデルパラメータを表す分布のパラメータであるハイパーパラメータも推定されるようにする。モデルパラメータの数が多いケースを想定し、モデルパラメータ間の相関を考慮しない事前分布となっている。

BLFM の各パラメータの分布は次のように設定する。

$$P(y_i|\mathbf{x}_i,\Theta,\omega) \propto \kappa_i\psi(\mathbf{x}_i,\Theta) - \frac{1}{2}\omega_i\psi(\mathbf{x}_i,\Theta)^2 \quad (11)$$

$$P(\omega_i|\mathbf{x}_i,\Theta) = PG(1,\psi(\mathbf{x}_i,\Theta)) \quad (12)$$

$$P(\theta|\Theta_H) = N(\mu_\theta,\lambda_\theta) \quad (13)$$

$$P(\mu_\theta|\Theta_H,\Theta_0) = N(\mu_\mu,\gamma_\mu\lambda_\theta) \quad (14)$$

$$P(\lambda_\theta|\Theta_0) = Gam(\alpha_\lambda,\beta_\lambda) \quad (15)$$

データサンプル数は N とし、i 番目のデータサンプルを $(y_i,\mathbf{x}_i)$ と表記した。ハイパーパラメータはまとめて $\Theta_H = \{\mu^w,\lambda^w,\mu_1^v,\cdots,\mu_K^v,\lambda_1^v,\cdots,\lambda_K^v\}$ と表した。また、モデルパラメータ θ に関するハイパーパラメータを μ_θ、λ_θ と表した。例えば、$\theta = w_1$ のハイパーパラメータは $\mu_\theta = \mu^w$、$\lambda_\theta = \lambda^w$ であり、$\theta = v_{2,3}$ であれば $\mu_\theta = \mu_3^v$、$\lambda_\theta = \lambda_3^v$ である。ハイパーパラメータの分布のパラメータは定数とし、まとめて $\mathbf{\Theta}_0 = \{\mu_\mu,\lambda_\mu,\alpha_\lambda,\beta_\lambda,\mu_0,\lambda_0\}$ と表記した。$N(\mu,\sigma^2)$ は平均 μ、分散 σ^2 の正規分布、$Gam(\alpha,\beta) \propto x^{\alpha-1}\exp(-\beta x)$ はガンマ分布である。

図 1 に BLFM のプレート表現を示す。

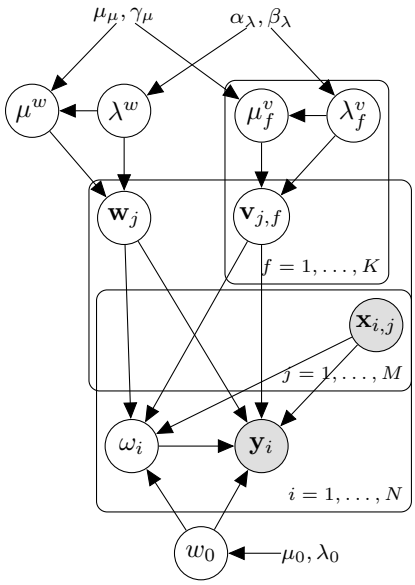

図 1　Bayesian Logistic Factorization Machines のプレート表現

次に、ギブスサンプリングに使う条件付き分布を示す。

式 (5) の表記を利用すると、モデルパラメータの条件付き分布は次のようになる。

$$P(\theta|\mathbf{y},\mathbf{X},\mathbf{\Theta}\setminus\{\theta\},\mathbf{\Theta}_H,\omega) \propto N(\tilde{\mu_\theta},\tilde{\sigma_\theta^2}) \quad (16)$$

$$\tilde{\mu_\theta} = \tilde{\sigma_\theta^2}\big(\mu_\theta\lambda_\theta + \sum_{i=1}^{N}\{\kappa_i - \omega_i g_\theta(\mathbf{x}_i)\}h_\theta(\mathbf{x}_i)\big) \quad (17)$$

$$\tilde{\sigma_\theta^2} = \big(\lambda_\theta + \sum_{i=1}^{N}\omega_i h_\theta(\mathbf{x}_i)^2\big)^{-1} \quad (18)$$

そのまま計算すると $h_\theta(\mathbf{x}_i)$ と $g_\theta(\mathbf{x}_i)$ の計算に時間がかかる。そこで、[5] と同様に 4.2 節で説明する方法で計算量を削減する。

パラメータ θ を表す分布のパラメータであるハイパーパラメータ μ_θ と λ_θ の条件付き分布は

$$P(\mu_\theta|\mathbf{\Theta},\mathbf{\Theta}_H\setminus\{\mu_\theta\},\mathbf{\Theta}_0) \propto N(\tilde{\mu_{\mu_\theta}},\tilde{\sigma_{\mu_\theta}^2}) \quad (19)$$

$$\tilde{\mu_{\mu_\theta}} = \tilde{\sigma_{\mu_\theta}^2}\lambda_\theta(\mu_\mu\gamma_\mu + \sum_{j=1}^{M}\theta_j) \quad (20)$$

$$\tilde{\sigma_{\mu_\theta}^2} = \big(\lambda_\theta(\gamma_\mu + M)\big)^{-1} \quad (21)$$

$$P(\lambda_\theta|\mathbf{\Theta},\mathbf{\Theta}_H\setminus\{\lambda_\theta\},\mathbf{\Theta}_0) \propto Gam(\tilde{\alpha_{\lambda_\theta}},\tilde{\beta_{\lambda_\theta}}) \quad (22)$$

$$\tilde{\alpha_{\lambda_\theta}} = \frac{1}{2}(2\alpha_\lambda + M + 1) \quad (23)$$

$$\tilde{\beta_{\lambda_\theta}} = \frac{1}{2}\Big\{\sum_{j=1}^{M}(\theta_j - \mu_\theta)^2 + \gamma_\mu(\mu_\theta - \mu_\mu)^2\Big\} + \beta_\lambda \quad (24)$$

となる。

また、潜在変数 ω_i の条件付き分布は

$$P(\omega_i|\mathbf{y},\mathbf{X},\mathbf{\Theta}) = PG(1,\hat{\psi}(\mathbf{x}_i,\mathbf{\Theta})) \quad (25)$$

となる。

4 協調フィルタリングにおける実装

4.1 学習データの構造

協調フィルタリングに FM を利用する場合、学習データはユーザー、アイテム、状況といった 3 種類のデータが連結されたものとなる。

ユーザーデータは、ユーザーが決まると一意に定まるデータで、ユーザー ID とユーザー属性から構成される。ユーザー ID は、ユーザー ID に対応するカラムの値が 1、それ以外が 0 となるスパースなデータとして表現される。そのため、ユーザー ID を表すカラムはユーザー数と同数必要になる。ユーザー属性は、ユーザーの特徴を表すデータである。例えば、ユーザーの年齢を表す数値、性別や職業を表すダミー変数などはユーザー属性である。また、過去に閲覧したアイテムなどもユーザー属性である。

アイテムデータは、アイテムが決まると一意に定まるデータで、アイテム ID とアイテム属性から構成される。ユーザー ID と同様にアイテム ID も、アイテム ID に対応するカラムの値が 1 それ以外が 0 となるスパースなデータとして表現される。アイテム属性は、アイテムの特徴を表すデータである。例えば、アイテムのリリース年やジャンルを表すダミー変数などはアイテム属性である。

状況データは、ユーザーやアイテムが決まっても定まらないデータである。例えば、アクセス時間やアクセス媒体などは状

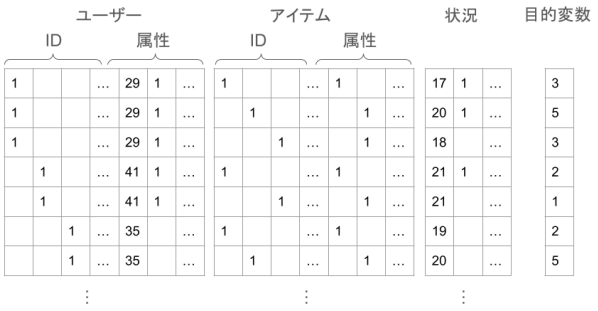

図 2　協調フィルタリングで使う FM のデータ

況データである。

協調フィルタリングで使うデータを図に示すと図 2 となる。例えば、図 2 において、ID データはカラム順に ID が連番で対応しているとし、一番左のユーザー属性は年齢、一番左のアイテム属性は何らかのジャンル A、一番左の状況データは見た時間とすると、図 2 のデータにおいて、29 歳のユーザー ID=1 のユーザーは 17 時にジャンル A に属するアイテム ID=2 のコンテンツを見たときに、評価などの目的変数の値が 3 であったことを示す。図 2 からもわかるように、ID データはカラム数が多い非常にスパースなデータになるため、実装においてはスパースデータ構造を用いるなどの対応が必要になる。

協調フィルタリングにおいては、あるユーザーに対してアイテムと状況のデータが与えられた時のスコアやコンバージョン率などの予測値を計算し、予測値が高いアイテムを予測値が高くなる状況においてレコメンドする。

4.2　計算量の削減

モデルパラメータのサンプリング時と予測値計算時の双方において計算量の削減を図る。

モデルパラメータのサンプリング時の計算量削減は、[5] とほぼ同様の方法を使う。共通計算部分を事前に計算しておくことで繰り返し計算を回避することと、差分のみを計算することで計算量の多い $\psi(\mathbf{x}_i, \Theta)$ や $g_\theta(\mathbf{x}_i)$ の直接計算を回避することで計算量の削減を図る。

まず、$\psi(\mathbf{x}_i, \Theta)$ の計算について考える。更新後のパラメータを θ^* で表すと、パラメータ更新後の $\psi_i(\mathbf{x}_i, \Theta^*)$

$$\psi_i(\mathbf{x}_i, \Theta^*) = \psi_i(\mathbf{x}_i, \Theta) + (\theta^* - \theta) h_\theta(\mathbf{x}_i) \tag{26}$$

となる。そのため、事前に $\psi_i(\mathbf{x}_i, \Theta)$ を計算し記憶しておけば、モデルパラメータ更新時に $O(1)$ で更新できる。

次に、$h_\theta(\mathbf{x}_i)$ の計算について考える。$h_\theta(\mathbf{x}_i)$ は θ が w_0 あるいは w_l の場合は容易であるため、$\theta = v_{l,f}$ のケースについてのみ考える。$\theta = v_{l,f}$ のときの $h_\theta(\mathbf{x}_i)$ は

$$\bar{q}_{i,f} = \sum_{j=1}^{M} v_{j,f} x_{i,j} \tag{27}$$

を事前計算しておくことで、次のように $O(1)$ で計算できる。

$$\begin{aligned} h_\theta(\mathbf{x}_i) &= x_{i,l} \sum_{j=1}^{M} v_{j,f} x_{i,j} - v_{l,f} x_{i,l}^2 \\ &= x_{i,l} \bar{q}_{i,f} - v_{l,f} x_{i,l}^2 \end{aligned} \tag{28}$$

$v_{l,f}$ が更新された時 $\bar{q}_{i,f}$ は次のように $O(1)$ で更新することができる。

$$\bar{q}^*_{i,f} = \bar{q}_{i,f} + (v^*_{l,f} - v_{l,f}) x_{i,j} \tag{29}$$

なお、$g_\theta(\mathbf{x}_i)$ は $\psi(\mathbf{x}_i, \Theta) - \theta h_\theta(\mathbf{x}_i)$ から計算する。$\psi(\mathbf{x}_i, \Theta)$ と $h_\theta(\mathbf{x}_i)$ が計算されていれば、$g_\theta(\mathbf{x}_i)$ を直接計算する必要はない。

以上のようにすることで、1 パラメータの更新を $O(N)$ で行うことができるので、1 エポックあたり $O(NMK)$ で計算が可能になる。これは、回帰タイプの FM と同じ計算量である。

次に、予測値の計算方法について説明する。

[3] では、モデルパラメータのサンプリング時にテストデータに関する予測値を計算するアルゴリズムを提案しているが、評価実験などでは有用でも実運用では利用しにくいという問題がある。評価実験等では予測値計算対象となるデータ量は学習データのデータ量と同等もしくはそれより少ないことが多い。一方、実際に協調フィルタリングで利用する場合は、最大でユーザー数とアイテム数と状況数を掛け合わせた数のデータに対して予測値を計算する。状況データの特徴量数は多くないことが多いので以下では状況データを無視して考える。ユーザーとアイテムのみを考えユーザー数を N^u、アイテム数を N^i、サンプリング数を S としたとき、1 データの予測値を計算するのに $O(SMK)$ の計算量が必要なので、予測計算に $O(SMKN^uN^i)$ の計算が必要になる。学習時と予測時のデータ量の比はデータに強く依存するものの現実のデータでは $N \ll N^uN^i$ であることが多く、レビューデータなど 1 人あたりのサンプルデータが多いデータではおよそ 10 倍以上、人材サービスデータなどの 1 人あたりのサンプルデータが少ないデータではおよそ 100 倍以上、予測時のデータ量が多くなる。　そのため、実運用においては [3] にて提案されている方法を使うのは現実的ではないことがある。

本稿では、事前計算を利用して予測計算全体の計算量を特徴量数に依存しない $O(SKN^uN^i)$ にする方法を考える。大量の計算が必要になるのはユーザーとアイテムのデータを組み合わせた計算のときに生じるため、可能な限りユーザーのみあるいはアイテムのみで計算可能な部分を事前に計算しておくことで計算量の削減を図る。

予測値算出式について、ユーザーの特徴量数を M^u、アイテムの特徴量数を M_i、サンプル数を S とし、サンプル番号を下付き添字 $_s$ で示し、ユーザーデータとアイテムデータを表すのに上付き添字 ^{u} と ^{i} を使って、式 (3) を書き直すと

$$\hat{\psi}(\mathbf{x}) = \frac{1}{S}\sum_{s=1}^{S}\Big[w_{s,0} + \sum_{j=1}^{M^u} w_{s,i}^u x_j^u + \sum_{j=1}^{M^i} w_{s,j}^i x_j^i + \sum_{l=1}^{M^u}\sum_{j=l+1}^{M^u} \hat{w}_{s,l,j}^u x_l^u x_j^u + \sum_{l=1}^{M^i}\sum_{j=l+1}^{M^i} \hat{w}_{s,l,j}^i x_l^i x_j^i + \sum_{l=1}^{M^u}\sum_{j=1}^{M^i} \hat{w}_{s,l,j}^{u,i} x_l^u x_j^i\Big] \quad (30)$$

$$\hat{w}_{s,l,j}^u = \sum_{f=1}^{K} v_{s,l,f}^u v_{s,j,f}^u \quad (31)$$

$$\hat{w}_{s,l,j}^i = \sum_{f=1}^{K} v_{s,l,f}^i v_{s,j,f}^i \quad (32)$$

$$\hat{w}_{s,l,j}^{u,i} = \sum_{f=1}^{K} v_{s,l,f}^u v_{s,j,f}^i \quad (33)$$

となる。

ユーザーのみ、アイテムのみで計算可能な式 (30) の第 2、3、4、5 項は、ユーザーのみあるいはアイテムのみのデータで

$$\bar{r}^u = \frac{1}{S}\sum_{s=1}^{S}\Big[\sum_{j=1}^{M^u} w_{s,j}^u x_j^u + \sum_{l=1}^{M^u}\sum_{j=l+1}^{M^u} \hat{w}_{s,l,j}^u x_l^u x_j^u\Big] \quad (34)$$

$$\bar{r}^i = \frac{1}{S}\sum_{s=1}^{S}\Big[\sum_{j=1}^{M^i} w_{s,j}^i x_j^u + \sum_{l=1}^{M^i}\sum_{j=l+1}^{M^i} \hat{w}_{s,l,j}^i x_l^u x_j^u\Big] \quad (35)$$

として、学習時に計算できる。ユーザーとアイテムそれぞれ $O(SKM^uN^u)$ と $O(SKM^iN^i)$ で計算できるため計算量が問題になることは少ない。

計算量が多くなるのはユーザーとアイテムの組み合わせの項である式 (30) の第 6 項である。ここで、

$$\bar{q}_{s,f}^u = \sum_{j=1}^{M^u} v_{s,j,f}^u x_j^u \quad (36)$$

$$\bar{q}_{s,f}^i = \sum_{j=1}^{M^i} v_{s,j,f}^i x_j^i \quad (37)$$

をサンプリング時に事前計算しておくと、

$$\begin{aligned}
&\frac{1}{S}\sum_{s=1}^{S}\sum_{l=1}^{M^u}\sum_{j=1}^{M^i} \hat{w}_{s,l,j}^{u,i} x_l^u x_j^i \\
&= \frac{1}{S}\sum_{s=1}^{S}\sum_{l=1}^{M^u}\sum_{j=1}^{M^i}\sum_{f=1}^{K} v_{s,l,f}^u v_{s,j,f}^i x_l^u x_j^i \\
&= \frac{1}{S}\sum_{s=1}^{S}\sum_{f=1}^{K}\left(\sum_{l=1}^{M^u} v_{s,l,f}^u x_l^u\right)\left(\sum_{j=1}^{M^i} v_{s,j,f}^i x_j^i\right) \\
&= \frac{1}{S}\sum_{s=1}^{S}\sum_{f=1}^{K} \bar{q}_{s,f}^u \bar{q}_{s,f}^i
\end{aligned} \quad (38)$$

と $O(SK)$ で計算が可能になる。つまり、学習時に $\bar{r}^u$ と $\bar{r}^i$、$\bar{q}_{s,f}^u$、$\bar{q}_{s,f}^i$ を計算しておくことで、式 (30) は $O(SK)$ で計算できるため、予測計算全体の計算量を $O(SKN^uN^i)$ とすることができる。

4.3 アルゴリズム

式 (16) から式 (25) の θ に具体的なモデルパラメータ を当てはめ、4.2 節の計算量削減を行うと、BLFM のギブスサンプリングは Algorithm1 となる。なお、$\mathbf{h} = (h_1, \cdots, h_N)^\top$ など、サンプルデータを一括して扱っているところではベクトル表記を用いている。Algorithm1 ではモデルパラメータの初期値をすべて 0 としているが乱数を使用しても構わない。モデルパラメータの初期値を全て 0 とすると、ψ や $\mathbf{q}$ も 0 になるため初期計算を省略できる利点がある。Algorithm1 では明示していないが、初期のサンプルはバーンインとして予測には使わないようにすることが望ましい。また、協調フィルタリングではデータがスパースなため、実装にはスパースデータ構造を利用するなどスパースデータを効率的に扱うための工夫が必要である。

5 検証実験

本節では、プロビットタイプの FM と比較した検証実験の結果を示す。評価指標には ROC 曲線の AUC(Area Under Curve) と Accuracy を用いた。比較対象モデルとして、libfm[1]のプロビットタイプの FM を用いた。以下、提案手法は lfm、比較対象モデルは libfm と表記する。検証はロジスティックタイプとプロビットタイプの FM で以下の点に違いが生じるかに着目して行われた。

- データ量
- 2 値のバランス
- 属性の有無
- 1 人あたりの評価データ数
- 因子数

データは Movielens[2]の 100K データセットと 1M データセットを利用した。100K データセットはあらかじめ 5 分割されたテストデータが用意されているため、それらをそのまま利用した。1M データセットについてはランダムに 10 分割したデータを用いた交差検証を行った。モデルパラメータの初期値については、libfm は 0 を平均とし 0.1 を標準偏差とした乱数とし、lfm は全て 0 としている。また、定数として与える、ハイパーパラメータを表す分布のパラメータは、$\mu_\mu = \mu_0 = 0$、$\lambda_\mu = \lambda_0 = \alpha_\lambda = \beta_\lambda = 1$ としている。なお、バーンインは 30 としている。

2 値のバランスの影響を調べるため、目的変数については、評価 4 点以上を 1 それ以外を 0 としたデータセットと、評価 5 点以上を 1 それ以外を 0 としたデータセットの 2 種類を用いた。100K データセットでは評価 4 点以上の割合は 55.4%、5 点以上の割合は 21.2%であり、1M データセットでは評価 4 点以上の割合は 57.5%、5 点以上の割合は 22.6%である。2 値のバランスが、評価 4 点以上を 1 としたときは比較的均衡しており、5 点以上を 1 とした場合は不均衡になっているため、前者を均衡データ、後者を不均衡データと呼ぶ。コンバージョンなどを

1：http://www.libfm.org
2：https://grouplens.org/datasets/movielens/

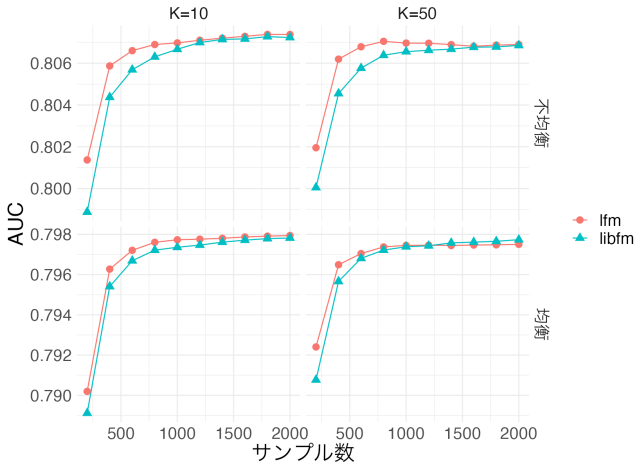

図 3　サンプル数と AUC の関係 (100K データセット)

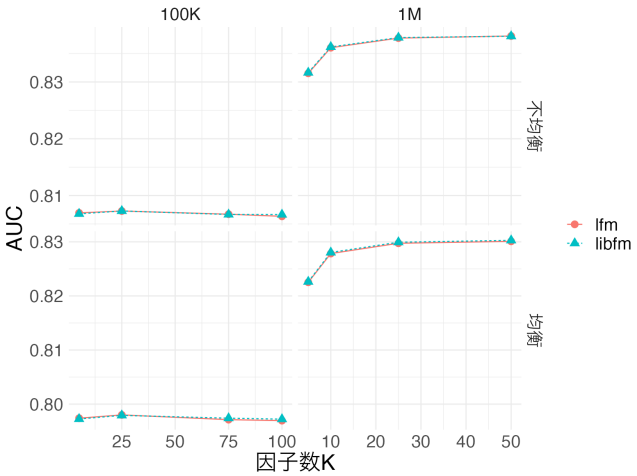

図 4　属性なしでのデータサイズ (100K, 1M) による AUC の違い (左段:100K, 右段:1M、上段: 不均衡データ、下段: 均衡データ)

扱う場合は不均衡データとなることが多いため、不均衡データにおいて高精度なモデルが望ましい。

属性の影響を調べるため、特徴量にユーザーの性別と映画ジャンルを加えたケースについて検証した。いずれの特徴量も 1 もしくは 0 をとるダミー変数とした。

1 人あたりの評価データ数が少なくなったときの影響を調べるため、1M データセットの学習データにおいてユーザーごとに 5%、10%、20%の評価しか学習に利用しないケースについて検証した。なお、人材サービスのレコメンデーションでは、1 人あたりの評価データ数が少くなる傾向にあるため、1 人あたりの評価データ数が少なくなったときでも精度が維持されるモデルが望ましい。

属性情報なしの 100K データセットを使って、均衡、不均衡データについて、因子数 $K = 10, K = 50$ における ROC 曲線の AUC を計算し、サンプル数との関係をプロットしたものを図 3 に示す。概ねサンプル数が増加するほど AUC は上昇する傾向が見られるが、1,000〜1,500 あたりから上昇幅が急激に小さくなる。そこで、本研究では、サンプル数を 1,500 とした。ただし、属性情報なしの 100K データセットではサンプル数 2,000 の結果を示す。なお、提案手法と libfm とでサンプル数が少ない時の AUC に違いが生じている原因としては、モデルの違いに加えて初期値の扱いの違いが考えられる。

属性なしの 100K データセットと 1M データセットそれぞれの不均衡データと均衡データを使って ROC 曲線の AUC を因子数別にプロットしたものを図 4 に示す。提案手法と libfm とでは、データサイズや 2 値のバランスが異なっても AUC にほとんど差が生じないことがわかる。なお、いずれのモデルでも因子数 K の影響を受けることも確認できる。

100K データセットで属性ありのデータとなしのデータを使って ROC 曲線の AUC と Accuracy を計算し因子数別にプロットしたものを図 5 に示す。提案手法と libfm とでは、Accuracy にほとんど差がないことがわかる。属性なしの AUC でも提案手法と libfm にはほとんど差がないものの、属性ありの AUC ではやや違いが見られる。均衡データ、不均衡データいずれにおいても、libfm と比較して提案手法は因子数 K が小さい時に

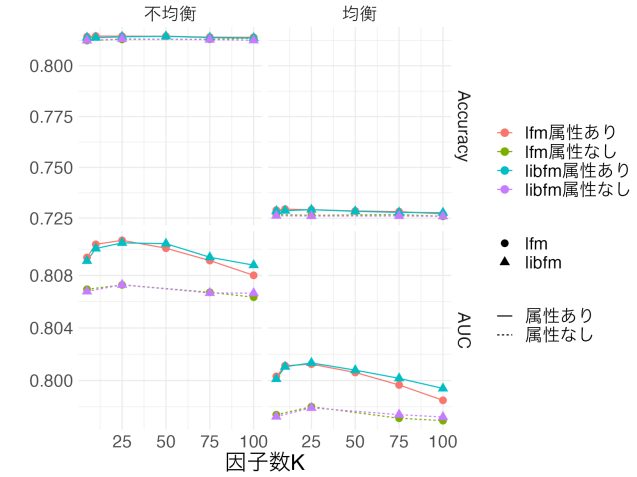

図 5　100K データセットにおける属性情報の有無による AUC と Accuracy の違い (左段: 不均衡データ, 右段: 均衡データ、上段:Accuracy、下段:AUC)

AUC が比較的高く、K が大きくなると AUC が比較的小さくなる傾向が見られる。ただし、AUC が高くなる因子数 K が 5〜25 の範囲ではモデル間で大きな差は生じておらず、モデルの違いより適切な因子数 K の選択のほうが AUC の向上に影響すると考えられる。また、いずれのモデルでも因子数 K の影響を受けることや属性を使うことで AUC が向上することが確認できる。

1M データセットにおいてユーザーごとに 5%、10%、20%の評価のみを学習データとして使って均衡データと不均衡データについて AUC を計算し因子数別にプロットした結果を図 6 に示す。いずれのケースにおいても libfm と比較して提案手法の AUC がわずかに高くなっているものの、大きな差はないことがわかる。評価データ数を 20%、10%、5%に削減してもモデル間の AUC の差はほとんど変わらないため、評価データ数による影響も見られない。

プロビット回帰とロジスティック回帰の予測結果は類似していることが知られている。サンプル数が十分にあり適切な因子

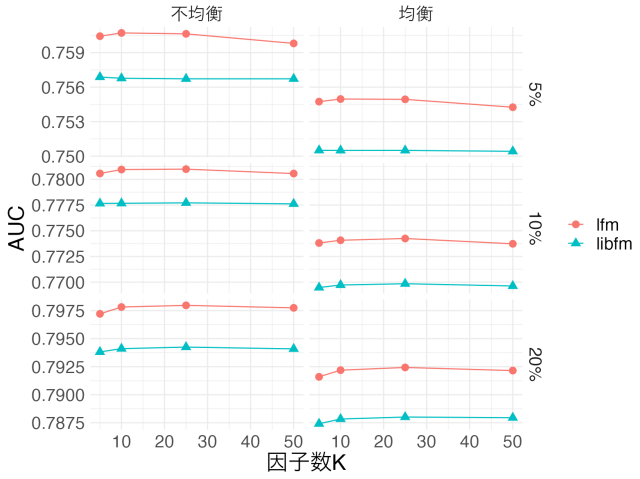

図 6 (1M データセットにおいて一部のデータのみを学習したときの違い (左段: 不均衡データ, 右段: 均衡データ、上段:5%データで学習、中段:10%データで学習、下段:20%データで学習))

数を選択した場合、プロビットタイプの FM とロジスティック回帰タイプの FM の予測精度の差はほとんどないことから、FM の場合でも類似した予測結果になりやすいと考えられる。

6 ま と め

本研究では Pólya-Gamma 分布を使ったデータ拡大法を利用することで、ロジスティック回帰タイプの Bayesian Factorization Machines のパラメータを、近似なしの条件付き分布を使ってギブスサンプリングする方法を提案した。また、学習時および予測時における計算量の削減方法も示した。

検証実験では、データサイズ、2 値のバランス、属性の有無、1 人あたりの評価データ数、因子数によって、ロジスティックタイプの FM とプロビットタイプの FM の精度に差が生じるかについて検証した。その結果、サンプル数が十分にあり適切な因子数が選択された場合、ロジスティックタイプの FM とプロビットタイプの FM はほとんどのケースにおいてほぼ同程度の精度となることが確認された。

文 献

[1] Linden, G., Smith, B. and York, J.: Amazon.Com Recommendations: Item-to-Item Collaborative Filtering, *IEEE Internet Computing*, Vol. 7, No. 1, pp. 76–80 (2003).
[2] Rendle, S.: Factorization Machines, in *2010 IEEE International Conference on Data Mining*, pp. 995–1000 (2010).
[3] Rendle, S.: Factorization Machines with libFM, *ACM Transactions on Intelligent Systems and Technology*, Vol. 3, No. 3, pp. 1–22 (2012).
[4] Srebro, N. and Jaakkola, T.: Weighted low-rank approximations, in *Proceedings of the 20th international conference on machine learning (ICML-03)*, pp. 720–727 (2003).
[5] Rendle, S., Gantner, Z., Freudenthaler, C. and Schmidt-Thieme, L.: Fast Context-Aware Recommendations with Factorization Machines, in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 635–644 (2011).
[6] Freudenthaler, C., Schmidt-Thieme, L. and Rendle, S.: Bayesian Factorization Machines, in *Proceedings of the NIPS Workshop on Sparse Representation and Low-rank Approximation* (2011).
[7] Saha, A., Misra, R., Acharya, A. and Ravindran, B.: Scalable Variational Bayesian Factorization Machine (2017).
[8] Vie, J.-J., Rigaux, T. and Kashima, H.: Variational Factorization Machines for Preference Elicitation in Large-Scale Recommender Systems, in *2022 IEEE International Conference on Big Data (Big Data)*, pp. 5607–5614 (2022).
[9] Polson, N. G., Scott, J. G. and Windle, J.: Bayesian Inference for Logistic Models Using Pólya – Gamma Latent Variables, *Journal of the American Statistical Association*, Vol. 108, No. 504, pp. 1339–1349 (2013).
[10] Rendle, S. and Schmidt-Thieme, L.: Pairwise Interaction Tensor Factorization for Personalized Tag Recommendation, in *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, pp. 81–90 (2010).
[11] Koren, Y.: Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model, in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 426–434 (2008).
[12] Rendle, S., Freudenthaler, C. and Schmidt-Thieme, L.: Factorizing Personalized Markov Chains for Next-Basket Recommendation, in *Proceedings of the 19th International Conference on World Wide Web*, pp. 811–820 (2010).
[13] Xiong, L., Chen, X., Huang, T.-K., Schneider, J. and Carbonell, J. G.: Temporal Collaborative Filtering with Bayesian Probabilistic Tensor Factorization, in *Proceedings of the 2010 SIAM International Conference on Data Mining*, pp. 211–222 (2010).
[14] Koren, Y.: Collaborative Filtering with Temporal Dynamics, in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 447–456 (2009).
[15] Rendle, S.: Scaling Factorization Machines to Relational Data, *Proceedings of the VLDB Endowment*, Vol. 6, No. 5, pp. 337–348 (2013).
[16] Juan, Y., Zhuang, Y., Chin, W.-S. and Lin, C.-J.: Field-Aware Factorization Machines for CTR Prediction, in *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 43–50 (2016).
[17] Blondel, M., Fujino, A., Ueda, N. and Ishihata, M.: Higher-Order Factorization Machines, in *Advances in Neural Information Processing Systems*, Vol. 29 (2016).
[18] Wen, P., Yuan, W., Qin, Q., Sang, S. and Zhang, Z.: Neural Attention Model for Recommendation Based on Factorization Machines, Vol. 51, No. 4, pp. 1829–1844 (2021).
[19] He, X. and Chua, T.-S.: Neural factorization machines for sparse predictive analytics, in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 355–364 (2017).
[20] Li, Z., Wu, S., Cui, Z. and Zhang, X.: GraphFM: Graph Factorization Machines for Feature Interaction Modeling (2022).
[21] Klami, A.: Polya-Gamma Augmentations for Factor Models, in *Proceedings of the Sixth Asian Conference on Machine Learning*, Vol. 39, pp. 112–128 (2015).
[22] Rai, P., Hu, C., Harding, M. and Carin, L.: Scalable Probabilistic Tensor Factorization for Binary and Count Data, in *Proceedings of the 24th International Conference on Artificial Intelligence*, pp. 3770–3776 (2015).

Algorithm 1 BLFM のギブスサンプリング

function BLFM_GS($\mathbf{y}$, $\mathbf{x}, K, S$)
  $w_0, \mathbf{w}, \mathbf{v}, \psi, \mathbf{q}, \bar{w_0}, \mathbf{r}^u, \mathbf{r}^i \leftarrow 0,$
  for $s \in \{1, \ldots, S\}$ **do**
    $\mu_{\tilde{\mu^w}} \leftarrow \sigma^2_{\tilde{\mu^w}} \lambda^w \left(\mu_\mu \gamma_\mu + \sum_{j=1}^M w_j\right)$
    $\sigma^2_{\tilde{\mu^w}} \leftarrow \left(\lambda^w(\gamma_\mu + M)\right)^{-1}$
    sample μ^w from $N(\mu_{\tilde{\mu^w}}, \sigma^2_{\tilde{\mu^w}})$
    $\alpha_{\tilde{\lambda^w}} \leftarrow \frac{1}{2}(2\alpha_\lambda + M + 1)$
    $\beta_{\tilde{\lambda^w}} \leftarrow \frac{1}{2}\left\{\sum_{j=1}^M (w_j - \mu^w)^2 + \gamma_\mu(\mu^w - \mu_\mu)^2\right\} + \beta_\lambda$
    sample λ^w from $Gam(\alpha_{\tilde{\lambda^w}}, \beta_{\tilde{\lambda^w}})$
    for $f \in \{1, \ldots, K\}$ **do**
      $\mu_{\tilde{\mu^v_f}} \leftarrow \sigma^2_{\tilde{\mu^v_f}} \lambda^v_f (\mu_\mu \gamma_\mu + \sum_{j=1}^M v_{j,f})$
      $\sigma^2_{\tilde{\mu^v_f}} \leftarrow \left(\lambda^v_f(\gamma_\mu + M)\right)^{-1}$
      sample μ^v_f from $N(\mu_{\tilde{\mu^v_f}}, \sigma^2_{\tilde{\mu^v_f}})$
      $\alpha_{\tilde{\lambda^v_f}} \leftarrow \frac{1}{2}(2\alpha_\lambda + M + 1)$
      $\beta_{\tilde{\lambda^v_f}} \leftarrow \frac{1}{2}\left\{\sum_{j=1}^M (v_{j,f} - \mu^v_f)^2 + \gamma_\mu(\mu^v_f - \mu_\mu)^2\right\} + \beta_\lambda$
      sample λ^v_f from $Gam(\alpha_{\tilde{\lambda^v_f}}, \beta_{\tilde{\lambda^v_f}})$
    end for
    sample ω from $PG(1, \hat{\psi}(\mathbf{x}_i, \mathbf{\Theta}))$
    $\mu_{\tilde{w_0}} \leftarrow \sigma^2_{\tilde{w_0}}\left(\mu_0\lambda_0 + \sum_{i=1}^N \{\kappa_i - \omega_i(\psi(\mathbf{x}_i) - w_0)\}\right)$
    $\sigma^2_{\tilde{w_0}} \leftarrow \left(\lambda_0 + \sum_{i=1}^N \omega_i\right)^{-1}$
    sample w_0^* from $N(\mu_{\tilde{w_0}}, \sigma^2_{\tilde{w_0}})$
    $\psi \leftarrow \psi + (w_0^* - w_0)$
    $w_0 \leftarrow w_0^*$
    for $j \in \{1, \ldots, M\}$ **do**
      $\mu_{\tilde{w_j}} \leftarrow \sigma^2_{\tilde{w_j}}\left(\mu^w\lambda^w + \sum_{i=1}^N \{\kappa_i - \omega_i(\psi(\mathbf{x}_i) - w_j x_{i,j})\} x_{i,j}\right)$
      $\sigma^2_{\tilde{w_j}} \leftarrow \left(\lambda^w + \sum_{i=1}^N \omega_i x_{i,j}^2\right)^{-1}$
      sample w_j^* from $N(\mu_{\tilde{w_j}}, \sigma^2_{\tilde{w_j}})$
      $\psi \leftarrow \psi + (w_j^* - w_j)\mathbf{x}_j$
      $w_j \leftarrow w_j^*$
    end for
    for $f \in \{1, \ldots, K\}$ **do**
      for $j \in \{1, \ldots, M\}$ **do**
        for $i \in \{1, \ldots, N\}$ **do**
          $h_i \leftarrow x_{i,j}\bar{q}_{i,f} - v_{j,f}x_{i,j}^2$
        end for
        $\mu_{\tilde{v_{j,f}}} \leftarrow \sigma^2_{\tilde{v_{j,f}}}\left(\mu^v_f\lambda^v_f + \sum_{i=1}^N \{\kappa_i - \omega_i(\psi(\mathbf{x}_i) - v_{j,f}h_i)\}h_i\right)$
        $\sigma^2_{\tilde{v_{j,f}}} \leftarrow \left(\lambda^\theta_f + \sum_{i=1}^N \omega_i h_i^2\right)^{-1}$
        sample $v^*_{j,f}$ from $N(\mu_{\tilde{v_{j,f}}}, \sigma^2_{\tilde{v_{j,f}}})$
        $\psi \leftarrow \psi + (v^*_{j,f} - v_{j,f})\mathbf{h}$
        $\mathbf{q}_f \leftarrow \mathbf{q}_f + (v^*_{j,f} - v_{j,f})\mathbf{x}_j$
        $v_{j,f} \leftarrow v^*_{j,f}$
      end for
    end for
    $\bar{w_0} \leftarrow \bar{w_0} + w_0$
    for $i \in \{1, \ldots, N^u\}$ **do**
      $r^u_i \leftarrow r^u_i + \sum_{j=1}^{M^u} w^u_i x^u_{i,j} + \sum_{l=1}^{M^u}\sum_{j=l+1}^{M^u} \hat{w}^u_{l,j} x^u_{i,l} x^u_{i,j}$
    end for
    for $i \in \{1, \ldots, N^i\}$ **do**
      $r^i_i \leftarrow r^i_i + \sum_{j=1}^{M^i} w^i_i x^i_{i,j} + \sum_{l=1}^{M^i}\sum_{j=l+1}^{M^i} \hat{w}^i_{l,j} x^i_{i,l} x^i_{i,j}$
    end for
    for $f \in \{1, \ldots, K\}$ **do**
      $\bar{q}^u_{s,f} \leftarrow \sum_{j=1}^{M^u} v^u_{s,j,f} x^u_j, \quad \bar{q}^i_{s,f} \leftarrow \sum_{j=1}^{M^i} v^i_{s,j,f} x^i_j$
    end for
  end for
  $\bar{w_0} \leftarrow \bar{w_0}/S$, $\bar{\mathbf{r}}^u \leftarrow \mathbf{r}^u/S$, $\bar{\mathbf{r}}^i \leftarrow \mathbf{r}^i/S$
  return $\bar{w_0}, \bar{\mathbf{q}}^u, \bar{\mathbf{q}}^i, \bar{\mathbf{r}}^u, \bar{\mathbf{r}}^i$
end function