TFDHRによる移動距離を考慮したPOI推薦の提案と性能評価

M2016SC015 尾崎 俊介
指導教員：河野 浩之

1 はじめに

近年，Foursquare (https://ja.foursquare.com/) や Weibo (https://www.weibo.com/login.php) などの Locallion Base Social Networking service (LBSNs) と呼ばれる SNS が全世界で普及している．これらのサービスは観光地や店舗をユーザーが訪れた際にチェックインを行い，その場所の評価を行う．ユーザが興味のある場所は Point Of Interest (POI) と呼ばれ，観光地や店舗を推薦する手法の研究に使用されている．しかし，一人のユーザがチェックインする POI は全ての POI と比べるとごくわずかであり，評価行列で表す際に非常に疎なデータとなる．また，POI 推薦にはユーザの訪問する POI への移動距離や多く訪問される時間帯などいくつかの情報を分析する必要がある．

本研究ではテンソル因子分解と移動距離から訪問確率を予測する手法を統合した推薦手法を提案し，移動距離ができるだけ短くてかつ訪問確率の高い POI を推薦することを目的とする．

推薦には協調フィルタリングと呼ばれる手法があり，未評価の値を補完する効果がある．協調フィルタリングには KNN やアソシエーション分析等，複数の手法があるが，本研究では複数の条件から同時に相関を調べることのできるテンソル因子分解を用いる．それに加えて，POI への訪問確率は移動距離の大きさによって変化する [2]．そこで，現在位置と推薦する POI の移動距離をスコア計算に導入し，そのスコアに基づいて推薦を行うことで現在位置から近くて訪問確率の高い推薦を行う手法である Tensor Factorization and Distance Hybrid Recommendation (TFDHR) を提案する．

2 章では POI 推薦に関連する研究の紹介とその問題点を提示する．3 章では 2 章で提示した問題点を解決する TFDHR を提案する．4 章では提案手法の性能を評価するための実験方法を示す．5 章では TFDHR の性能評価，6 章でむすびとする．

2 POI推薦に関する先行研究

本章では POI 推薦に関連する研究の紹介と，それらの問題点についてまとめる．Yong Liu らはユーザが訪れたことのある POI 付近の POI が今後訪問する確率があるとし，“ユーザ-POI” の訪問回数行列の行列因子分解のスコアと訪問したことのある POI の付近の POI の訪問確率のスコアと統合する手法を提案している [1]．

Quan Yuan らはユーザ POI を訪れる際の時間的情報と POI の距離などの地理的情報が POI 推薦の精度に大きく影響するとし，それらの情報を考慮した協調フィルタリングの推薦手法を提案している [2]．データのスパース性に対処するためにチェックインされている時間枠の周辺の枠にも値が入るようスムージング手法を用いている．さらに，あるチェックインを行った後，次に他のチェックインする可能性が POI 間の距離が遠くなるほど低下することを示し，訪問履歴から訪問確率を予測する確率分布を作成し，推薦に用いている．

2 つの先行研究の長所と問題点を表 1 に示す．Yong Liu らの研究は POI を訪問する時間帯が考慮されていない．また，Quan Yuan らの研究は POI を訪れる時間情報と地理情報を考慮しているが，訪問履歴の POI 間の距離を分析した推薦であり現在位置については考えられていない．

表 1 先行研究の長所と問題点

	長所	問題点
[1]	移動距離を考慮した行列因子分解手法	POI の訪問時間を考慮していない
[2]	距離と時間帯を考慮した協調フィルタリング	現在位置を考慮していない

そこで本研究では，ユーザの特徴を多次元で分析することができるテンソル因子分解を用いて “ユーザ”，“POI”，“時間” の 3 つの特徴を同時に分析する．さらに，このスコアに併せて現在位置からの移動距離を考慮する推薦を提案する．また，本研究のようなテンソル因子分解と移動距離の訪問確率のスコアを統合する手法は存在せず，新規性がある．

3 距離と時間を考慮したテンソル因子分解推薦手法 (TFDHR) の提案

本章では TFDHR について説明する．まず，3.1 節で TFDHR の流れを説明する．次に本研究で使用するテンソル因子分解と移動距離から POI の訪問確率を推定するための方法を 3.2 節と 3.3 節で説明する．最後に 3.4 テンソル因子分解と移動距離の訪問確率のスコアの統合方法について説明する．

3.1 TFDHR の流れ

本節では本研究で提案する推薦手法の全体の流れを説明する．図 1 は TFDHR ののアーキテクチャである．

(1) では Foursquare ユーザのチェックイン情報からを取得し，これを用いて 3 階テンソル X を作成する．作成に使用するチェックイン情報はチェックインしたユーザ ID，チェックイン時刻，チェックイン POI を示す．(2) ではテンソル因子分解を用いて，作成したテンソルを因子分解し，復元したテンソル X_{rec} を作成する．また，本研究ではテンソル分解手法の一つである Non-negative Tensor Factorization (NTF) を用いる．(3) では復元テンソルからユーザの各時間枠の POI とその予測値を抽出し各時間

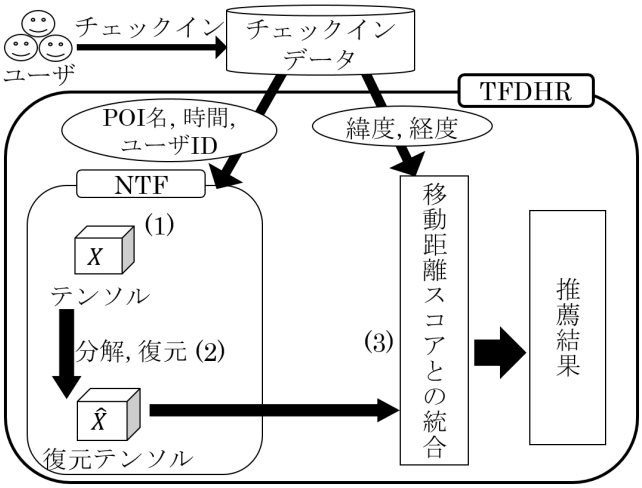

図 1　TFDHR の流れ

帯の推薦候補とする．次にテンソルに含まれるすべての POI の緯度と経度を用いて現在地の POI と推薦候補の POI の移動距離を計算し，事前に作成した訪問確率の分布からスコアを計算する．最後に NTF のスコアと移動距離の訪問確率のスコアを統合し，統合スコアを用いて現在地の周辺の予測値の高い POI のみを推薦結果として出力する．

3.2　Non-negative Tensor Factorization(NTF)

テンソル因子分解とは，多次元配列であるテンソルを複数の行列やテンソルの積に近似する手法である．さらに近似した行列の積から復元したテンソルは復元前のテンソルで欠損値だった値を補完する効果があり，この値を用いてユーザの未訪問 POI についても訪問確率を予測することが可能である．POI 推薦ではテンソルの軸を “ユーザ”，“POI”，“時間” で表し，各要素を「あるユーザが POI である時間帯にチェックインした回数」としてテンソルで表現する．各要素は訪問回数であるため非負値である．

そこで本研究では非負制約下のコンテンツ分析に有効であるテンソル因子分解手法の一つである Non-negative Tensor Factorization(NTF)[3] を用いて，チェックインデータから作成したテンソルを因子分解する．本研究で用いる NTF は CP 分解を用いてテンソルを分解する．CP 分解とはテンソルを次元数と同じ数の行列の内積に分解する手法である．本研究では “ユーザ”，“POI”，“時間” を属性とした 3 階テンソル $X = [x_{ijk}] \ni \mathbb{R}^{I\times J\times K}$ を分解する．ここで，$U = [u_1, u_2, \ldots, u_i, \ldots, u_I]$ はユーザを表す．$P = [p_1, p_2, \ldots, p_j, \ldots, p_J]$ は POI を示す．$T = [t_1, t_2, \ldots, t_k, \ldots, t_K]$ は時間枠を示す．時間枠は 24 時間を 1 時間ごとで区切る．例えば am5:00-am6:00 を一つの時間枠とする．$x_{i,j,k}$ はユーザ u_i が地点 p_j で時間枠 t_k にチェックインした回数を示し，多ければ多いほどその条件下での興味が高いと仮定する．

テンソル因子分解後はテンソル X の各属性の特徴行列であるユーザ特徴行列 $A = [a_{ir}] \ni \mathbb{R}^{I\times R}$，POI 特徴行列 $B = [b_{jr}] \ni \mathbb{R}^{J\times R}$，時間特徴行列 $C = [c_{kr}] \ni \mathbb{R}^{K\times R}$ に分解する．R は分解後の行列の因子数であり任意の自然数を設定する．また，式 (1) のように行列 A, B, C の内積を求めることで復元テンソル $\hat{X} = [\hat{x}_{ijk}] \ni \mathbb{R}^{I\times J\times K}$ を得る．

$$\hat{X} = A \otimes B \otimes C \tag{1}$$

復元テンソル $\hat{X}$ はコスト関数 D を最小化することで X と近似する．コスト関数は KL ダイバージェンスを用いて 2 つのテンソルの距離を計算する．コスト関数 D を式 (2) に示す．

$$D(X, \hat{X}) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}(x_{ijk}\log\frac{x_{ijk}}{\hat{x}_{ijk}} - x_{ijk} + \hat{x}_{ijk}) \tag{2}$$

最終的にこれを最小化するための更新式を作成し，これを複数回実行して a_{ir}, b_{jr}, c_{kr} を更新していくことで X に近似した $\hat{X}$ が得られる．$\hat{X}$ には X でゼロ要素 (未評価) だった値に非ゼロ要素 (予測値) が補完される．補完された値を用いることでユーザが未訪問の POI でも訪問確率を予測することができる．

3.3　距離を考慮した推薦スコアの計算

本節では $\hat{X}$ を考慮しつつ，現在訪問している POI の周辺に位置している POI を優先的に推薦する手法を提案する．Yuan Quan らの距離を考慮した POI 推薦手法 [2] では移動距離が増加するほど移動先の POI への訪問確率がべき乗則にしたがって低下していることを考慮して，移動距離から訪問確率が得られるべき乗関数をチェックインデータから推定し推薦に使用している．

本研究の性能評価実験で扱うデータセットでも同様の傾向があるか，事前実験として確認する．図 2 は後述する 4.1 節のデータセットの連続チェックインされた POI 間の距離とその頻度の関係を表した散布図である．ここからわかるように Yuan Quan らの研究 [2] と同様の傾向が表れており，連続チェックインされる POI 間の移動距離が長くなるにつれてその距離での訪問回数が低下していることがわかる．

そこで本研究では Yuan Quan らの研究の移動距離から訪問確率を推定する手法を用いて，現在地から推薦候補の POI への訪問確率を推定する．式 (3) に訪問確率を予

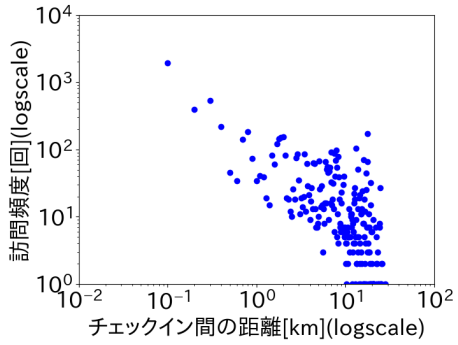

図 2　POI 間の移動距離とその訪問回数

測するべき乗関数を示す.

$$wi(dis(\tilde{p}, p_j)) = \beta_1 (dis(\tilde{p}, p_j))^{\beta_2} \tag{3}$$

関数 $dis()$ は 2 つの POI の緯度と経度を用いて POI 間の直線距離を計算する関数である．$\tilde{p}$ は現在位置の POI，p_j はテンソルに含まれる $\tilde{p}$ を除く全ての POI である．β_1, β_2 は $wi()$ の分布を決定するためのパラメータである．次に式 (3) の両辺を対数でとると，一次関数で表現できる．よって図 2 のデータ群から最小 2 乗法で一次関数を推定し，パラメータ β_1, β_2 を推定できる．

3.4 NTF と距離推薦スコアの統合

本節では 3.2 節と 3.3 節で計算した各スコアの統合方法について説明する．まず，NTF のスコアを一人のユーザの各時間帯の訪問確率とするために以下の式 (4) で $\dot{S} = [\dot{s}_{ijk}]$ を計算する．

$$\dot{s}_{ijk} = \frac{\hat{x}_{ijk}}{\sum_{d=1}^{J} \hat{x}_{idk}} \tag{4}$$

次に式 (5) で $\ddot{S} = [\ddot{s}_{ijk}]$ を計算する．

$$\ddot{s}_{ijk} = \frac{wi(dis(\tilde{p}, p_j))}{\sum_{d=1}^{J} wi(dis(\tilde{p}, p_d))} \cdot \dot{s}_{ijk} \tag{5}$$

$\dot{S}$ は訪問確率であるためこれと統一するために $wi(dis(\tilde{p}, p_j))$ についても全 POI の $wi()$ 総和から各 p_j の訪問確率を計算する．しかし，訪問確率のスコアのまま $\dot{S}$ と統合した場合，過去に誰も訪問していない時間帯の POI であっても現在位置から近いだけで上位に推薦されてしまう．そこでテンソル因子分解のスコアを掛けることで訪問確率の低い POI のスコアを低下させる．

最後に以下の式 (6) で $\dot{S}$ と $\ddot{S}$ を線形結合で統合し $\hat{S} = [\hat{s}_{ijk}]$ を得る．

$$\hat{s}_{ijk} = \alpha \dot{s}_{ijk} + (\alpha - 1)\ddot{s}_{ijk} \tag{6}$$

ここで α は $[0, 1]$ のパラメータで任意の値を設定する．最後にユーザ u_i に時間帯 t_k でスコア $\hat{s}_{ijk}$ の高い POIp_j 上位 N 件を推薦結果 $L(u_i, t_k) = \{l_1, l_2, \ldots l_n\}$ として出力する．

4 TFDHR の性能評価実験

本章では 3 章の TFDHR の有効性を確かめるための性能評価実験を行う．まず，4.1 節では使用するデータセットについて説明し，4.2 節で実験環境と実験方法について説明する．

4.1 Foursquare データセット

実験では Dingqi Yang らが収集した Foursquare Dataset の Tokyo Check-in Dataset[4] を用いた．データセットの概要を表 2 に示す．

チェックイン数が極めて少ないユーザや POI がある場合，適切に分析できないことが考えられるためこのデータセットから小さなデータセットを抽出する．総チェックイン数が 500 を超えるユーザのチェックインのみを選択，総訪問数が 50 未満の POI のチェックインを削除する．

表 2 データセットの概要

収集日	2012/4/12-2013/2/16
収集した都市	東京
チェックイン数	573,703 件
ユーザ数	1,939
カテゴリ数	251
POI 数	61,858
その他の属性	ユーザ ID,POIID, カテゴリ, 緯度, 経度, チェックイン日時

次に，本研究の推薦目的である「現在位置を訪れた後に訪れる POI」のチェックインを持つユーザとそのチェックインを抽出する．推薦目的のチェックインとなる正解データは「現在位置の POI から半径 1km 以内にある POI」または「現在位置をチェックインした後にチェックインした POI」のチェックインとする．また，POI には駅や空港などの公共交通機関の POI や学校や会社などの関係者以外は訪れることのない POI を除く．最終的に 23 人の 18,940 件のチェックインを用いて性能評価を行う．

4.2 実験環境と実験方法

実験環境を表 3 に示す．Anaconda とは Numpy や Scipy などの多次元配列の計算に必要なライブラリと Python 本体を備えたパッケージである．NTF には nmf-and-ntf ライブラリ (https://github.com/drumichiro/nmf-and-ntf) を使用してテンソル因子分解を行った．

表 3 実験環境

OS	Ubuntu14.04
CPU	AMD FX-8350 Eight- Core
メモリ	16GB
使用言語	Python2.7.13 Anaconda4.4.0(64-bit)

次に，TFDHR の性能評価実験の方法について述べる．現在位置は東京駅，新宿駅，渋谷駅，池袋駅にそれぞれ設定して推薦結果を出力した．推薦結果の出力は一時間ごとに行い，性能評価の指標は Mean Reciprocal Rank (MRR) を用いた．式 (7) に各時間帯の MRR の計算方法を示す．

$$MRR = \frac{1}{T} \sum_{k=1}^{K} \frac{1}{N} \sum_{i=1}^{U} \frac{1}{n_{ik}} \tag{7}$$

MRR は推薦結果の順位を 1 位からみたときに最初に適合した POI の順位 n_{ik} の逆数をスコアとして用いる指標である．総合的なスコアは各時間帯で全ユーザの平均をとったあと，全時間帯で平均をとって計算する．チェックインデータは各ユーザの 20 %のチェックインを正解データ，残りを教師データとする．またユーザが POI を訪問する時間帯の誤差を考えるために，推薦時間枠の ± 1 時間で推薦した POI をユーザがチェックインした場合も推薦成功とみなす．

5 TFDHR の性能評価実験結果

本章では TFDHR の性能評価実験の結果と考察を示す．ここで，NTF のパラメータは因子数 R を 128，更新アルゴリズムの反復回数を 100 回に設定した．式 (6) のパラメータ α は MRR が最も高い値を採用した．

まず，現在位置を東京駅に設定したときの推薦結果を表 4 に示す．これはユーザ ID347 に 19 時の推薦スコアが高い POI を降順に並べた上位 5 件である．また，正解データと一致した POI は「正」列に○，一致しなかった POI は×を示す．NTF の推薦は訪問回数のみが推薦スコアに影響するため現在位置から遠い POI が 1 位や 2 位に推薦されてしまっている．しかし，TFDHR を適用すると NTF で 4 位に位置する「ビックカメラ有楽町店」が TFDHR では 1 位に推薦された．また，新宿駅に設定したときの推薦結果の表 5 でも TFDHR を適用することで同様の変化が見られる．これは現在位置からの移動距離が推薦スコアに考慮されたため，NTF のスコアが高くかつ移動距離の短い POI が上位に推薦されるからである．

表 4 東京駅の推薦結果上位 5 件

NTF			
順位	POI	距離	正
1	東海道本線六郷川橋梁	16.8km	×
2	多摩川	17.1km	×
3	セブンイレブン新木場 1 丁目店	6.6km	×
4	ビックカメラ有楽町店	0.7km	○
5	鶴見川	20.1km	×

TFDHR			
順位	POI	距離	正
1	ビックカメラ有楽町店	0.7km	○
2	東海道本線六郷川橋梁	16.8km	×
3	多摩川	17.1km	×
4	セブンイレブン新木場 1 丁目店	6.6km	×
5	京葉線連絡通路動く歩道	0.29km	×

表 5 新宿駅の推薦結果上位 5 件

NTF			
順位	POI	距離	正
1	ヨドバシカメラ Akiba	6.8km	×
2	高田馬場駅前ロータリー	2.6km	×
3	AKIHABARA ゲーマーズ	6.5km	×
4	アニメイト新宿店	0.4km	×
5	西武池袋本店	4.4km	×

TFDHR			
順位	POI	距離	正
1	ヨドバシカメラ Akiba	6.8km	×
2	高田馬場駅前ロータリー	2.6km	×
3	アニメイト新宿	0.4km	×
4	AKIHABARA ゲーマーズ	6.5km	×
5	ビックカメラ新宿西口店	0.8km	○

次に，表 6 に NTF と TFDHR の推薦結果上位 20 件で出力したときの MRR を示す．各地点の MRR を比較すると，東京駅は約 29.2%，新宿駅は約 18.1%，池袋駅は約 31.1%，渋谷駅は約 9.0%だけ TFDHR が NTF を上回っていた．また，4 地点の MRR の平均を求めた時，TFDHR が NTF を約 22.7%上回っていた．この結果から表 4 と表 5 のような変化がどの地点を設定しても表れることがわかり，TFDHR を用いることで現在位置から移動距離が近くてユーザが好む POI を上位で推薦できている．

表 6 MRR の性能評価

現在位置	NTF	TFDHR
東京駅	0.352	0.455
新宿駅	0.336	0.397
池袋駅	0.322	0.422
渋谷駅	0.255	0.278
平均	0.316	0.388

6 むすび

本研究ではテンソル因子分解と POI への移動距離を考慮した手法を統合した POI 推薦手法 (TFDHR) を提案し，NTF を用いて TFDHR の実装を行った．次に Foursquare データセットを用いて TFDHR の性能評価実験を行った．評価方法として NTF と TFDHR の MRR を計算し，性能指標の比較を行った．その結果，上位 5 件の POI 推薦で TFDHR は NTF よりも約 22.7%だけ MRR が向上した．

参考文献

[1] Yong Liu, Wei Wei, Aixin Sun, Chunyan Miao, *"Exploiting Geographical Neighborhood Characteristics for Location Recommendation"*, Proceedings of the 23rd ACM International Conference on Information and Knowledge Management pp.739-748 (2014).

[2] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, Nadia Magnenat-Thalmann, *"Time-aware Point-of-Interest Recommendation"*, Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval pp.363-372 (2013).

[3] Koh Takeuchi, Ryota Tomioka, Katsuhiko Ishiguro, Akisato Kimura, Hiroshi Sawada, *"Non-negative Multiple Tensor Factorization"*, Proceedings of IEEE 13th International Conference on Data Mining, pp.1199-1204 (2013).

[4] Dingqi Yang, Daqing Zhang, Vincent W. Zheng, Zhiyong Yu, *"Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs"*, IEEE Trans. on Systems, Man, and Cybernetics: Systems, vol.45, no.1, pp.129-142 (2015).