DEIM2020 P2-18(day2 p91)

双曲空間におけるニュース推薦のためのオンライングラフ埋め込み

飯塚洸二郎† 関 喜史†

† 株式会社 Gunosy 〒 107-6016 東京都港区赤坂 1 丁目 12－32
E-mail: †{kojiro.iizuka,yoshifumi.seki}@gunosy.com

あらまし 近年、EC や音楽視聴をはじめとしたサービスにおいて、推薦システムはユーザーとの接地点と位置づけられ重要な役割を担っている。様々なサービスの中でも、ニュースサービスやソーシャルネットワークサービスなどは、推薦対象アイテムの更新が頻度が極めて高い、つまり、高 Velocity のサービスであるため、推薦に即時性が求められることが知られている。多くの推薦システムが埋め込み手法を用いることで高い精度を達成してきている中で、高い Velocity を持つサービスにおいてユーザーからのフィードバックを用いた埋め込み手法を用いることは計算効率の観点で難しかった。そこで、本研究では高 Velocity のサービスで活用できる双曲空間でのオンライングラフ埋め込み手法を提案し、学習の効率性を検証した。次に、双曲空間において埋め込みを行うことで、従来手法よりも推薦精度が高くなることをオフライン実験を通して確認した。最後に、実際のニュースサービスに本手法を組み込むことで、クリック数や滞在時間が有意に向上することを A/B テストを通して確認した。

キーワード 情報推薦，グラフ埋め込み，オンライン学習

1 はじめに

近年、EC や音楽視聴をはじめとしたサービスにおいて、推薦システムはユーザーとの接地点と位置づけられ重要な役割を担っている [5]。推薦システムには、サービスやユーザーの特性にあったコンテンツの配信が求められる。様々な配信サービスの中でも、ニュースサービスやソーシャルネットワークサービスなどは、推薦対象アイテムの更新が頻度が極めて高い、つまり、高 Velocity のサービスであるため、推薦に即時性が求められることが知られている。その制約の中で、推薦アルゴリズムの精度を上げるための研究が近年注目を集めている [7], [9], [10]。高い Velocity を持つサービスにおいて推薦精度を高めるための推薦アルゴリズムの構築にはいくつかの課題が存在する。そのうちの 1 つは推薦アイテムの埋め込み手法の効率化である。

近年の推薦アルゴリズムは埋め込み表現に大きく依存しており、埋め込み表現の精度を上げることが、推薦精度を向上させることに直結する [1], [2], [12]。しかし、高い Velocity の状況下においては、埋め込み表現の学習時間が短いことが期待され、既存の学習に時間の掛かるモデルは適用できない。また学習時間の短縮のために、新しく得られる学習データのみを入力し、学習を一から逐次やり直す方針を取ると、過去に得られたアイテムの埋め込み表現とその時点で学習した埋め込み表現の整合性がとれなくなってしまう問題がある。

そこで、本研究では高い Velocity を持つサービスにおいて、過去に学習したモデルと新しく得られる学習データを入力として、モデルの差分更新を行う効率的な埋め込み手法を提案する。本研究での貢献は以下の通りである。

- 推薦システムにおけるユーザーの行動ログが、双曲空間における学習に適した性質を持つことを初めて検証した
- 高い Velocity を想定した双曲空間におけるオンライングラフ埋め込み手法を提案した
- オフライン実験によって提案手法の学習の効率性と精度の良さを確認した
- 実際のニュースサービスに提案手法を組み込むことによって、クリック数や滞在時間が向上することを確認した

2 関連研究

深層学習を実際の推薦システムに適用することで、推薦精度は近年大きく向上した [1], [12]。特に、自然言語処理の研究から発展した深層学習の手法は、推薦システムへの応用に相性が良いことが知られている [2]。ユーザーの商品やウェブページの行動履歴を文章のコンテキストとみなして、直接アイテム間の関係性を学習させることで、アイテム同士の推薦のみならず商品のユーザーに対するパーソナライゼーションにも活用されてきた [5]。またアイテム間の関係性を低次元でより正確に表現する手法も現れ始めている。この手法は、従来の埋め込み空間をユークリッド空間ではなく双曲空間に拡張することで、階層構造を持つデータに対して埋め込み表現の精度を大幅に向上させた [8]。本研究では、このポアンカレ埋め込みを双曲空間への埋め込みとして扱う。

しかしながら、従来の深層学習の手法が適用が困難な推薦ドメインが存在する。例えば、ニュースサービスやソーシャルネットワークサービスでは推薦対象のアイテムの入れ替わりが高頻度で発生するため、アイテム間の関係性の学習に長い時間をかけることができず、精度を担保するのが難しかった。この問題に対しては、推薦アイテムのタイトルやカテゴリ情報などのコンテンツ情報を RNN の入力とする手法が提案されている [9]。一方で、コンテンツベースの埋め込み表現よりもユーザーのフィードバックを用いた協調フィルタリングベースの埋め込み表現のほうがより高いランキングメトリクスを達成することが

知られている [3]。そのため、高い Velocity を持つサービスにおいてもユーザーフィードバックを用いて直接的にアイテム間の関係性を学習することで、より精度の高い手法が実現できることが期待される。

3 分　　析

双曲空間への埋め込みは階層構造を持つグラフへの埋め込みが得意とされる。そのため、推薦システムにおけるグラフ構造が階層構造の性質をどれほど持っているかを clustering coefficient によって確認する。clustering coefficient は、グラフ内のノードがどれほど階層化する傾向にあるかを示す指標であり、以下のように定義される。

$$C(k_i) = \frac{1}{|V|}\sum_{i=1}^{|V|}\frac{|\{e_{jk} : v_j, v_k \in N_i, e_{jk} \in E\}|}{k_i\,(k_i - 1)} \qquad (1)$$

ここで、N_i は v_i に隣接するノードの集合であり、$k_i = |N_i|$ である。階層構造を持つグラフの場合、$C(k) \simeq k^{-\beta}$ となることが知られている [11]。

図 1 と図 2 が clustering coefficient を可視化した図である。グノシーのデータに関しては、$\beta = 0.6$、ニュースパスに関しては $\beta = 0.6$ として補助線を引いた。補助線に対して、cluster coefficent が当てはまっていることが確認できる。そのため、双曲空間の埋め込みは本実験で使用する推薦システムのデータに対しても適していると言える。

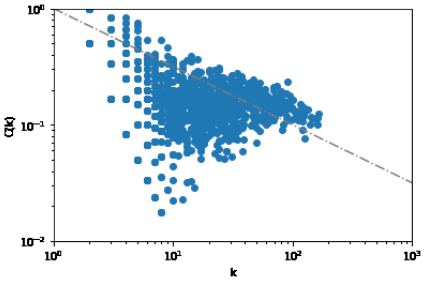

図 1　グノシーにおける cluster coefficent

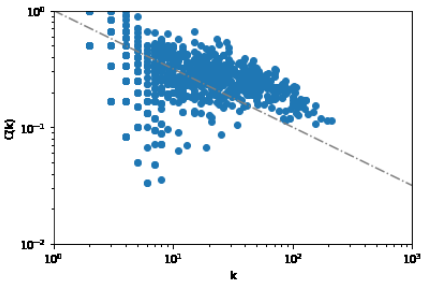

図 2　ニュースパスにおける cluster coefficent

4 提 案 手 法

この章では、高い Velocity を持つニュースサービスの上で埋め込み表現を獲得する手法を提案する。

4.1　オンライン学習

ニュースサービスでは、日々新しい記事が入稿されるため、比較的早い段階で記事に対する埋め込み表現を獲得する必要がある。さらに、新しい記事に対してバッチ処理によって逐一新しい埋め込み表現を獲得する場合は、長期間使える埋め込み表現ではなくなってしまう。以上の制約から、新しい記事が入稿されユーザーのフィードバックが得られた際に、すでに学習されたモデルに対して新たな記事を学習対象に加え、学習を行うオンライン学習手法を提案する。処理の全体は Alogrithm 3 に示す。

Algorithm 1: オンライン学習

```
model = loadLatestModel()
model = updateVocabulary(recentUserClickPairs, model)
deleteTargetIDs = getDeleteTargetIDsFromDB()
model = deleteVocabulary(deleteTargetIDs, model)
model.train()
model.save()
```

ポアンカレ埋め込みのオンライン学習の際に、モデルで管理するのは下記の変数である。これらの変数を用いたオンライン学習は 2 つの処理に分けられる。

vocab　学習データにおけるグラフの頂点の出現回数とインデックスを管理する辞書

vector　グラフの頂点の特徴ベクトルの辞書

index2word　グラフのインデックスからノード名を引くためのリスト

nodeRelations　グラフの頂点から他の頂点に伸びる辺を管理するリストの辞書

allRelations　グラフのすべての辺を管理するリスト

1 つ目がモデルに新たな学習データを追加する処理である。新たに入力された学習データに関して、グラフを構築する。すでに存在する頂点の入力に対しては既存の特徴ベクトルを保持し、新規の頂点に関しては特徴ベクトルをランダムに初期化する。具体的な処理はアルゴリズムは Alogrithm 2 である。

2 つ目が不要なノードをモデルから除去する処理である。モデルに新規記事を加え続けることでモデルが肥大化し、新しい記事に対する学習が阻害される懸念がある。そのため一定期間以上入稿が前の記事に対しては、学習モデルから記事を削除する処理を行う。削除対象の頂点に関しては、グラフのノードから削除し、特徴ベクトルも削除する。非削除対象の頂点に関しては、グラフのインデックスを貼り直した上で、特徴ベクトルは保持する。

学習全体の流れは Algorithm1 の通りである。

5 実　　験

以下を明らかにするための 3 つ実験を行った。

- 双曲空間における埋め込みは他の埋め込み手法に比べてどれほどの精度か

Algorithm 2: モデル辞書の追加

```
Input: recentUserClickPairs, model
model: output
prevIndex2wordLen = len(model.vocab)
# 辞書の更新
foreach relation in recentUserClickPairs do
    foreach item in relation do
        if item in model.vocab then
            model.vocab[item].count += 1
        end
        else
            model.vocab[item] = Vocab(count=1,
              index=len(model.index2word))
            model.index2word.append(item)
        end
        node1, node2 = relation
        node1Index, node2Index = model.vocab[node1].index,
          model.vocab[node2].index
        model.nodeRelations[node1Index].add(node2Index)
        model.allRelation.append( (node1Index, node2Index) )
    end
end
# ベクトルの追加
shape = (len(model.vocab) - prevIndex2wordLen, model.size)
v = random.uniform(0, 1.0, shape)
model.vector = concatenate([model.vector, v])
```

Algorithm 3: モデル辞書の削減

```
Input: deleteTargetArticleIDs, model
Output: model
preIndex2word = model.index2word
# 学習済みベクトルを退避 trainedVectors = dict()
for index, key in enumerate(model.index2word) do
    trainedVectors[key] = model.vector[index]
end
deleteIndices = []
# 削除対象のノードを辞書から削除
for articleID in deleteTargetArticleIDs do
    if articleID in model.vocab.keys() then
        deleteIndices.append(model.vocab[articleID].index)
        del model.vocab[articleID]
    end
end
# 削除対象のノードのベクトルを削除
for index in sorted(deleteIndices, reverse=True) do
    del model.index2word[index]
    np.delete(model.vector, index, 0)
end
# 辞書の再構築;
for index, key in enumerate(model.index2word) do
    model.vocab[key] = Vocab(count=model.vocab[key].count,
      index=index)
end
# 退避した学習済みベクトルを再挿入
for index, key in enumerate(model.index2word) do
    model.vocab.vector[index] = trainedVectors[key]
end
# グラフの頂点インデックスから delete 対象を入れ替え
for preIndex in model.nodeRelations.keys() do
    if preIndex in DeleteIndices or preIndex >=
      len(preIndex2word) then
        del model.nodeRelations[preIndex]
    end
end
nodeRelations = dict(set)
for preIndex in model.nodeRelations.keys() do
    nodeIndexRelations = set()
    for preIndex in model.nodeRelations.keys() do
        if nodeIndex < len(preIndex2word) then
            word = preIndex2word[nodeIndex]
            nodeIndexRelations.add(model.vocab[word].index)
        end
        nodeRelations[model.vocab[preIndex2word[preIndex]].index]
          = nodeIndexRelations
    end
end
model.nodeRelations = nodeRelations
# グラフの辺インデックスから delete 対象を入れ替え
allRelations = []
for (r1, r2) in model.allRelations do
    if both (r1, r2) not in deleteIndices and both (r1, r2) <
      len(preIndex2word) then
        allRelations.append((model.vocab[preIndex2word[r1],
          model.vocab[preIndex2word[r2]))
    end
end
model.allRelations = allRelations
return model
```

サービス名	特徴
ルクラ	2017 年サービス開始、女性向けサービス
ニュースパス	2016 年サービス開始、40-50 代がメイン層
グノシー	2012 年サービス開始、エンタメ領域に強み

表 1　各種サービスの特徴

- オンライン学習を用いることでどれほど学習が効率的に行われるか。またどれほど長期的に使える埋め込み表現になっているか
- 実サービスに提案手法を適用した際に、サービス KPI に与える影響はどれくらいか

5.1 オフライン実験

5.1.1 データセット

オフライン実験では、株式会社グノシーにおけるルクラ、ニュースパス、グノシーの 3 種のサービスを用いた。各種サービスの特徴は表 1 である。これらのサービスに対し、2019 年 9 月 2 日のサンプリングした 1000000 行の行動ログを実験に用いた。

5.1.2 オフラインメトリクス

学習の精度と学習速度を測定するために、2 つのランキングメトリクスを用いた。一つは、nDCG であり、もう一つは MAP である。

$$nDCG = \frac{1}{|S|}\sum_{s\in S}\frac{\sum_i c_{s,\pi(i)}/log_2(\pi(i)+1)}{max_\pi \sum_i c_{s,\pi(i)}/log_2(\pi(i)+1)}$$

$$MAP = \frac{1}{|S|}\sum_{s\in S}\frac{1}{|\{i|c_{s,i}=1\}|}\sum_{j=1}^{|S|} c_{s,j}\frac{|\{(i,j)|i \leqq j, c_{s,i}=1\}|}{j} \tag{2}$$

5.1.3 精度の比較

各種手法についての精度の比較実験はパーソナライゼーションの設定で行った。まず記事ごとにベクトルを生成し、ユーザーがクリックした記事の平均ベクトルをユーザーのベクトルとする。次にユーザーベクトルと推薦候補記事の記事ベクトルの類似度が大きい順に記事を 20 個選ぶ。最後に選定された記事に対して nDCG@20 をクリックを元に算出する。

比較対象の手法は、タイトルを用いたコンテンツベースの手法とユーザーのフィードバックを協調フィルタリングベースの手法に大きく分けられる。コンテンツベースの手法は、単語を word2vec によって学習させ、記事のタイトルを単語に分割した際のベクトル平均を用いる手法と、そのタイトルベクトルを RNN に入力する手法である。協調フィルタリングベースの手法はユーザーのクリック履歴に対して記事を単語とみなして記事の ID を word2vec で学習させる手法と、記事のクリック履歴の ID 自体を RNN に入力する 2 つの手法を用いた。

双曲空間における埋め込みの非オンライン学習時の精度と各種埋め込み手法の比較については、評価の際にテストデータの作り方を 2 通り計測した。1 つ目が実際にユーザーに閲覧のあった記事を並び替えて評価する方法、もう 1 つが、実際にユーザーに閲覧があるとは限らない記事も含めて評価する方法である。このようにオフライン評価には各種のバイアスが存在するため、複数の方法で評価を行っている。

5.1.4 学習の効率性

学習の効率性については以下の通り検証を行った。まず、データセットに含まれる推薦アイテム I を A, B, C の 3 つに分割する。ここで、$I = A \cup B \cup C$ である。次に、データセットに含まれる推薦アイテムのペア $E = \{(u,v)|u \in I, v \in I\}$ を $AB = \{(u,v)|u \in A, v \in B\}$, $BC = \{(u,v)|u \in B, v \in C\}$ に分ける。これらの組み合わせを用いて、バッチ学習の学習速度を測定するためには $AB \cup BC$ を一度に学習する。オンライン学習の学習速度を測定するために AB を学習させた後、$AB \cup BC$ を学習させる。モデルの肥大化を抑えるために辞書を削除したときの学習速度を測定するために、$AB \cup BC$ から AB を削除した後、BC を学習させた。

5.1.5 結果と考察

双曲空間における埋め込みの非オンライン学習時の精度と各種埋め込み手法の比較結果が表 3 である。

nDCG@20	ルクラ	ニュースパス	グノシー
Title-based word2vec	0.157	0.000	0.219
Title-based GRU	0.212	0.007	0.152
ID-based word2vec	0.368	0.210	0.344
ID-based poincare	0.379	0.363	0.327
ID-based GRU	0.316	0.007	0.152

表 2　Not-Rerank: 各種埋め込み手法の比較

nDCG@20	ルクラ	ニュースパス	グノシー
Title-based word2vec	0.554	0.526	0.561
Title-based GRU	0.551	0.544	0.557
ID-based word2vec	0.535	0.555	0.556
ID-based Poincare	0.548	0.551	0.574
ID-based GRU	0.521	0.567	0.543

表 3　Rerank: 各種埋め込み手法の比較

タイトルの埋め込み表現を用いたコンテンツベースの手法が上位の精度となっている。これは、このログを生み出す元となったロジックがタイトルベースの埋め込み表現を用いているため、タイトルベースの類似度が高い記事ほど上位に上がりやすくポジションバイアスが発生していることに起因していると考えられる。

図 3 は学習 epoch に対する MAP を示している。オンライン学習はバッチ学習よりも学習序盤から終盤までの MAP が高いことがわかる。これは、事前の埋め込み学習を差分更新していることに起因する。

図 4 は学習モデルのサイズを制限する場合としない場合の学習 epoch に対する MAP を示している。学習モデルを制限する場合の MAP が制限しない場合の MAP よりも学習後半にかけて精度が上回っている。これは、学習対象数が小さいほうが効率的な学習を行えることを示している。そのため、オンライン学習する際に、過去の推薦アイテムを学習対象から外すことで新しい推薦アイテムの学習が促進できることが分かった。

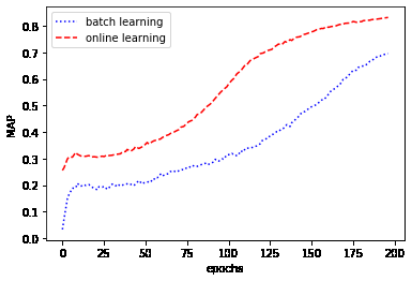

図 3　バッチ学習とオンライン学習の効率の差異

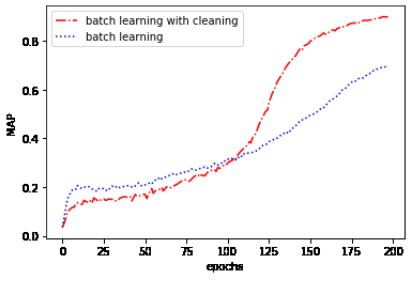

図 4　モデルの削減を行う際の学習効率の差異

5.2　A/B テスト

2020 年 2 月 5 日から 2020 年 2 月 10 日にかけて、A/B テストをルクラのサービス上で行った。ユーザー体験を損なわないよう、提案手法に対して 5%のトラフィックを適用した。本節では A/B テストの結果を報告する。

5.2.1　設　　定

ルクラでは、ユーザーが閲覧した記事の下に関連記事を掲出している。この関連記事のロジックは、タイトルベクトルの類似度と記事の Click Through Rate を用いたスコアリングが使用されている。この既存ロジックを control と呼び、ポアンカレ埋め込みによる類似度を新たに追加でスコアリングに用いたロジックを treatment と呼ぶ。

5.2.2　モデルと学習

ポアンカレ埋め込みを行うために、ユーザーの最新のセッションログと一つ前のセッションログを用いて学習を行った。現在のセッションの記事に対して記事のペアを取り、更に一つ前のセッションログと最新のセッションログの記事のペアを生成し、学習を行った。時間の経過とともに 1 日で約 10000 記事が入稿され、逐次蓄積されるユーザーログに対して、15 分間隔でモデルの差分学習を行った。ポアンカレ埋め込みは **10** 次元で行った。

5.2.3　オンラインメトリクス

下記リストに A/B テストで計測したメトリクスを記す。

Clicks　ユーザーのクリックの総数

Click Through Rate　ユーザーのクリックの総数をユーザーの記事閲覧数で割った値。この値が小さい場合、ユーザーは記事閲覧よりも記事を探す方に行動が寄っていることになる。

Clicks / Unique Users　ユーザーのクリックの総数をユニークユーザー数で割った値。バケット間の群によるユーザー数のバイアスを除去するために、ユニークユーザー数で割っている。

Duraion / Unique Users　ユーザーの関連記事総閲覧時間をユニークユーザー数で割った値。記事閲覧総時間は、関連記事をクリックしたあとで記事を閲覧している時間を指す。

5.2.4　結果と考察

表 4 に A/B テストの結果のサマリーを記す。すべてのメトリクスで treatment バケットの数値が有意に向上していた。クリック総数の向上のみならず、一人あたりの記事閲覧数が大幅に改善していた。これは、control バケットのロジックがタイトルのコンテンツ情報と CTR のみを使っているのに対し、treatment バケットに協調フィルタリングベースのポアンカレ埋め込みを新たに組み込むことで、ユーザーが事前にもっていない知識を含む記事が推薦されやすくなり、結果として記事閲覧時間が伸びたと考えられる。

Metrics	Percent Lift
Click Through Rate	+2.58 %
Clicks	+3.79 %
Clicks / Unique Users	+3.92 %
Duration / Unique Users	+10.23 %

表 4　Treatment vs Control

過去に学習されたが現在のモデルには含まれない記事に対するメトリクスと、一度もまだ学習されてない記事のメトリクスを比較した結果が表 5 である。バッチ学習とは異なりモデルを差分更新しているため、過去に一度は学習されたが現在はモデルから除外された記事 (Out of Model) に関しても、一度も学習が行われていない記事 (Never Trained) よりも平均的に精度が高いことが確認できる。一度も学習されていない記事に対してはポアンカレ埋め込みを使用せずにスコアリングを行った。各レスポンスのカバレッジは表 6 に記した。なおこのトラッキングは推薦システムがレスポンスを返却する際にどの素性を使用したかをロギングして実現している。

Metrics	vs Never Trained	vs In Model
Click Through Rate	+20.44 %	-39.2 %
Clicks / Unique Users	+50.34 %	-73.1 %

表 5　Out of Model に関する精度

	Percent
In Model	83.84 %
Out of Model	12.50 %
Never Trained	3.66 %

表 6　ポアンカレ埋め込みを用いたレスポンスカバレッジ

本実験は、タイトルベクトルが 300 次元であるのに対し、ポアンカレ埋め込みを 10 次元で行った。そのため、新たにポアンカレ埋め込みをスコアリングに活用する際に、システムパフォーマンスの悪化を招くこともなかった。ニュースの推薦システムのみならず、あらゆる推薦システムに対しても同様に次元の小さいポアンカレ埋め込みは活用が容易であると考えられる。

6 まとめと今後の課題

推薦対象のアイテムの入れ替わりが激しい高い Velocity を持つサービスでは、従来の協調フィルタリングベースの埋め込み表現の獲得が困難であり、そのため推薦精度の向上が限定的であった。

そこで、本研究ではまずニュースサービスにおけるユーザーの行動ログが双曲空間での埋め込みに適した性質をもつことを確認した上で、ポアンカレ埋め込みによる精度を検証した。次に、実際のニュースサービスに適用するために、モデルのオンライン学習手法を提案し、オフライン実験によって学習の効率性の向上を確認した。最後に、実際のニュースアプリケーションに提案手法を組み込み、A/B テストを行った。結果として、今まで獲得が困難であった精度の高い協調フィルタリングベースの埋め込み表現が推薦対象の類似度計算に活用できるようになり、クリック数や滞在時間が向上することを確認した。

本研究では、ユーザーのフィードバックログを用いたポアンカレ埋め込みを協調フィルタリングベースで行ったが、今後はナレッジベースをポアンカレ埋め込みすることで、コンテンツベースの精度の高い推薦手法を確立することが課題である。またコールドスタート問題に対して、ユーグリット空間のコンテンツベクトルを双曲空間の協調フィルタリングベースのベクトルに精度良く変換し、初期ベクトルとして利用するためのメタ学習を行うことが課題である。さらにオープンデータに対して実験を行いニュース推薦システム以外のシステムへの活用の可能性を模索する。

文　献

[1] M. An, F. Wu, C. Wu, K. Zhang, Z. Liu, and X. Xie, “Neural News Recommendation with Long- and Short-term User Representations .” In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics., pp. 336-345, 2019.

[2] O. Barkan and N. Koenigstein, “Item2Vec: Neural Item Embedding for Collaborative Filtering.” arXiv Preprint arXiv:1603.04259 (2016).

[3] O. Barkan, N. Koenigstein and E. Yogev, O. Katz, “CB2CF: a neural multiview content-to-collaborative filtering model for completely cold item recommendations.” In Proceedings of the 13th ACM Conference on Recommender Systems., pp. 228–236, 2019.

[4] W. Fan., Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang and D. Yin, “Graph Neural Networks for Social Recommendation.” In Proceedings of the 2019 World Wide Web Conference on World Wide Web.ACM, pp. 417–426, 2019.

[5] M. Grbovic and H. Cheng, “Real-time Personalization using Embeddings for Search Ranking at Airbnb.” In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 311–320, 2018.

[6] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, “ Session-based recommendations with recurrent neural networks. ” arXiv preprint arXiv:1511.06939 (2015).

[7] L. Li, W. Chu, J. Langford, and R. E. Schapire, “A Contextual-Bandit Approach to Personalized News Article Recommendation.” In Proceedings of the 2010 World Wide Web Conference on World Wide Web.ACM, pp. 661–670, 2010.

[8] M. Nickel and D. Kiela, “Poincaré Embeddings for Learning Hierarchical Representations.” In Proceedings of the 2017 Neural Information Processing Systems., pp. 6338–6347, 2017.

[9] S. Okura, Y. Tagami, S. Ono, and A. Tajima, “Embedding-based news recommendation for millions of users.” In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.ACM, pp. 1933–1942, 2017.

[10] B. Perozzi, R. Al-Rfou, and S. Skiena, “DeepWalk: Online Learning of Social Representations.” In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 701–710, 2014.

[11] E. Ravasz and A. Barabasi, “Hierarchical organization in complex networks.” Physical Review E 67:026112 (2003).

[12] H. Wang, F. Zhang, X. Xie and M. Guo, “DKN: Deep Knowledge-Aware Network for News Recommendation.” In Proceedings of the 2018 World Wide Web Conference on World Wide Web.ACM, pp. 1835–1844, 2018.

[13] G. Zheng , F. Zhang , Z. Zheng, Y. Xiang, N. J. Yuan, X. Xie and Z. Li, “DRN: A Deep Reinforcement Learning Framework for News Recommendation.” In Proceedings of the 2018 World Wide Web Conference on World Wide Web.ACM, pp. 167–176, 2018.