Vol.2025-MUS-142 No.11
2025/3/6

レイヤー構造に基づく楽曲推薦手法の提案と音楽発掘サービスKiiteへの応用

佃 洸摂[1,a)]　石田 啓介[1]　高橋 卓見[1]　濱崎 雅弘[1,b)]　後藤 真孝[1,c)]

概要：ユーザに対して楽曲を推薦する際には，そのユーザと好みが類似したユーザが好きな楽曲や，そのユーザが好きな楽曲と音響特徴量が類似した楽曲など，様々な種類の関係に基づく推薦が考えられる．本稿では，そうした複数種類の関係を同時に考慮可能な推薦手法を提案する．提案手法では，一つの関係を一つのレイヤーで表現し，各レイヤーは，ユーザや楽曲などに対応する多次元ベクトルのノード同士がそのレイヤーの関係を表すエッジで結ばれたグラフを持っている．さらに，各レイヤーに対して任意の重みを割り当てることができ，そのうえで全レイヤーのノードのベクトルの値を同時に最適化することで，大きな重みを割り当てたレイヤーに対応する関係を重視した推薦結果の生成を可能とする．我々は「ユーザ同士の楽曲の好みの類似性」「楽曲同士の音響特徴量の類似性」「楽曲同士の創作クリエータの共通性」の3種類の関係を考慮した楽曲推薦を実現し，音楽発掘サービス「Kiite」で実証実験を行った．Kiiteでは，上記の3つの関係のどれをどの程度重視するかに応じて4種類の推薦結果が表示され，ユーザはその4種類の推薦結果を自由に切り替えながら，楽曲の聴取やお気に入りへの追加などが行える点に大きな特徴がある．本稿では，4種類の推薦結果を通じてユーザがどのようなインタラクションを行ったかを，2年以上に渡るKiiteの利用ログを用いて分析した．

1. はじめに

音楽配信サービスや動画共有サービスを通じて膨大な数の楽曲に誰もがアクセスできるようになった．そうした膨大な楽曲の中からユーザが自分の好みに合った楽曲を自力で見つけるのは容易ではないため，楽曲に関する多くのサービスでは，ユーザの好みを推定したうえでユーザに楽曲を推薦する仕組みが導入されている．ユーザの好みを推定する際は「ユーザ同士の楽曲の好みの類似性」や「楽曲同士の音響特徴量の類似性」など複数の関係を考慮するのが一般的であり [1,2]，複数の関係を考慮しながらより高い精度でユーザの好みを推定して楽曲等のアイテムを推薦する手法の開発に関する研究も盛んに取り組まれている [3,4]．

ユーザに対して推薦システムを提供するうえでは，推薦精度の高さだけでなく，推薦の透明性の高さも備えることの重要性が提唱されている [5,6]．透明性を高めることで，ユーザの推薦システムに対する信頼性が高まることが知られており [5–7]，それがサービスの継続的な利用につながると期待されるためである．推薦システムの透明性を高めるための有効な手段として，推薦理由の提示があげられる [8,9]．推薦理由を提示する際は，例えば楽曲推薦であれば，推薦された各楽曲に対して「あなたと好みが似たユーザが好む楽曲です」や「あなたの好きな楽曲と曲調が似た楽曲です」のように，推薦時に考慮される関係を提示するのが主要なアプローチの一つである．

推薦理由の提示により推薦システムの透明性は高まるものの，推薦された各アイテムがどの関係を理由として推薦されるかは推薦システムにより決められる．そのため，例えばあるユーザに推薦された楽曲の大半に対して「あなたと好みが似たユーザが好む楽曲です」という推薦理由が提示された場合に，このユーザが「『あなたの好きな楽曲と曲調が似た楽曲です』という推薦理由に基づく推薦結果をもっと表示してほしい」と思っても，そうした意図を推薦システムに反映することはできない．特定の推薦理由に基づく推薦結果をユーザの意図に応じて表示できるような「操作性」を推薦システムに取り入れることができれば，推薦結果に対するより能動的なインタラクションを通じて，ユーザは推薦システムに対する透明性を感じながら，自分の好みに合った楽曲を効率的に見つけられることが期待できる．

こうした背景の元，本稿では透明性と操作性を備えた推薦システムの実現を目的とする．そのためにまず，推薦に

[1] 産業技術総合研究所
[a)] k.tsukuda@aist.go.jp
[b)] masahiro.hamasaki@aist.go.jp
[c)] m.goto@aist.go.jp

おける様々な関係を考慮しつつ, 特定の関係を重視した推薦結果を生成するための推薦手法を提案する．既存の Graph Convolution Network (GCN) を用いた推薦手法 [10,11] を基にしているが，提案手法では新たにレイヤー構造を導入し，一つの関係が一つの「レイヤー」により表現される工夫をしている．各レイヤーはグラフを持ち，ユーザや楽曲が多次元ベクトルのノードとなり，レイヤーに対応する関係を持つノード間にエッジが張られる．提案手法では各レイヤーに対して任意の重みを割り当てることができ，その重みに応じて全レイヤーのノードのベクトルの値を GCN の要領で最適化する．これにより，大きな重みを割り当てたレイヤーに対応する関係を重視した推薦結果の生成を実現する．

提案手法を用いて，我々は音楽発掘サービス「Kiite」[*1] 上で，透明性と操作性を備えた楽曲推薦システムを実装した．Kiite では「ユーザ同士の楽曲の好みの類似性」「楽曲同士の音響特徴量の類似性」「楽曲同士の創作クリエータの共通性」の 3 つの関係性を考慮しており，それぞれの関係を重視した 3 種類の推薦結果に加えて，全ての関係をバランス良く考慮した推薦結果の，計 4 種類の推薦結果が表示される．ユーザはその 4 種類の推薦結果を自由に切り替える操作をしながら，楽曲の聴取や推薦結果中の楽曲に対する適合・不適合のフィードバックなどを行える．

本稿では，4 種類の推薦結果を通じてユーザが推薦システムとどのようなインタラクションを行ったかを，2 年以上に渡る Kiite の利用ログを用いて分析した．分析の結果，4 種類の推薦結果のうち，より多くの種類の推薦結果を閲覧するユーザほど，推薦システムとのインタラクションを活発に行っていることが明らかになった．また，ユーザによって 4 種類の推薦結果の中でどの種類を好むかの違いはあったものの，40%近いユーザは複数種類の推薦結果を通じて好みの楽曲を発見しており，複数種類の推薦結果をユーザが自由に切り替えられるように操作性を持たせることの意義が示された．

2. 関連研究

推薦システムにおいて，ユーザに対してアイテムが推薦された理由を提示することで，ユーザが効率的にアイテムを探せるだけでなく [12,13]，推薦システムの透明性が高まり [7,14]，ユーザの推薦システムに対する信頼性の向上にも寄与することも示されてきた [7,13,15]．そうした重要性を踏まえて，情報推薦分野では推薦理由を提示する手法やインタフェースがこれまでに多数提案されてきた [8,9]．推薦理由を提示する際のアプローチは様々であるが，一例として，推薦システムが推定したユーザの好みを提示するアプローチがあげられる [12,16,17]．例えば Balog ら [17] は，映画に付与されたタグ情報を用いてユーザの好みを推定したうえで，ユーザの好みを説明する文章を生成し，推薦結果のリストと共にユーザに提示する手法を提案した．

推薦される各アイテムに対して推薦理由を提示するのも主要なアプローチの一つである [7,13–15,18,19]．具体的には，推薦されたアイテムに対する，自身と好みが似たユーザの評価を提示する方法 [18] や，自身が好むアイテムとの特徴量の一致度合いを提示する方法 [15] などが提案されている．また，推薦される各アイテムに対して，ユーザやアイテムに関する複数の関係に基づく推薦理由を提示する手法も提案されている [20–22]．例えば Kouki ら [21] の研究では，「ユーザ同士の好みの類似性」「アイテム同士のメタ情報の類似性」「アイテムの人気度」など 7 種類の関係を考慮し，推薦された各アイテムに対して，最大でその 7 種類全ての関係に基づく 7 個の推薦理由が提示される．どのような関係の推薦理由を好むかは個人差があるため [19,21]，様々な関係に基づく推薦理由を提示することは，そうした個人差への対応策の一つになりうる．しかし，単に推薦理由を提示するだけでは，例えばユーザが「『ユーザ同士の好みの類似性』に基づく推薦結果をもっと見たい」と思っても，そうした意図を反映した推薦結果を表示することはできない．

こうした問題の解決策として，複数の関係を考慮した推薦結果を生成し，かつどの関係をどの程度重視するかをユーザ自身が調整できるように，スライダーのインタフェースを用いて推薦理由の提示に操作性を持たせた推薦システムの研究があげられる [23–25]．Bostandjiev ら [23] の研究では，「ユーザの友人との好みの類似性」「アイテム間のメタ情報の類似性」「専門家が興味を持つアイテムの共通性」という 3 つの関係を考慮している．関係ごとに独立した推薦手法を用いて推薦結果を生成したうえで，各関係に対応した 3 つのスライダーをユーザが動かすことで，どの関係をどの程度重視するかをユーザが決められる．そして，各スライダーの重みに応じて各関係の推薦結果を統合して一つの推薦結果を生成し，ユーザに表示する．スライダーのインタフェースを用いた他の研究 [24,25] でも，同様のアプローチが取られている．スライダーを用いることで，各関係の重みを細かく調整できるという利点があり，ユーザにとって既知のアイテムを探す際にはユーザが頻繁にスライダーを使用したが，その一方で，ユーザにとって未知でありかつ興味を引かれるアイテムを探す際にはスライダーはほとんど使用されなかったという実験結果が報告されている [25]．

本研究では，複数の関係を考慮した推薦結果を表示しながら，ユーザが自身にとって未知でありかつ好みに合う楽曲を探すための推薦システムを実現することを目的としている．それにあたり，先行研究の知見 [25] に基づき，ユー

[*1] https://kiite.jp

ザがスライダーを用いて各関係の重みを調整するのではなく，それぞれの関係を重視した推薦結果をユーザが自由に切り替えながら閲覧できるインタフェースにすることで，スライダーに比べてユーザの操作の負荷を減らし，より手軽に利用できるような推薦システムを実装した．また，複数の関係を考慮しながら推薦システムに操作性を持たせる場合，前述の通り関係ごとに独立した推薦手法を用いることが一般的である．それに対して我々は，複数の関係を扱えるグラフに基づく推薦手法の推薦精度の高さ [4] を活かし，グラフとレイヤー構造を組み合わせることで特定の関係を重視した推薦結果を生成可能な推薦手法を提案する．さらに，推薦理由を提示する際に複数の関係を考慮して操作性を持たせた推薦システムは，数十名程度が 1 日から数日間使用するような小規模で短期的なユーザ実験によって評価が行われてきた [23–25]．それに対して我々は，提案する推薦システムを Web サービスの機能として実装することで 2 年以上に渡ってログを収集し，そこに含まれる 3,413 名分の推薦システムの利用ログを分析した．このように，透明性と操作性を備えた推薦システムの利用のされ方を長期間かつ大規模に調査して明らかにする点にも本研究の学術的価値がある．

3. レイヤー構造に基づく楽曲推薦手法

本章では，推薦における様々な関係を考慮しつつ，特定の関係を重視した推薦結果を生成するための推薦手法について説明する．

3.1 レイヤー上のグラフ作成（図 1 (a)）

提案手法では，推薦の際に考慮する様々な関係をグラフで表現する．関係ごとにグラフを作成してそれをレイヤーとみなすことで，特定のレイヤーを重視した推薦結果の生成を実現する．レイヤーの集合を L とし，レイヤー $l \in L$ 上のグラフを $G_l = (V_l, E_l)$ で表す．G_l は無向グラフであり，V_l と E_l はそれぞれ G_l のノード集合とエッジ集合である．例えば「ユーザ同士の楽曲の好みの類似性」という関係に対応するレイヤー l_1 上のグラフ G_{l_1} であれば，$V_{l_1} = U \cup S$（U と S はそれぞれ全ユーザと全楽曲の集合）であり，E_{l_1} は任意のユーザ $u \in U$ と u が好む各楽曲 $s \in S$ の間に張られたエッジの集合である．ユーザ u と楽曲 s の間に張られたエッジであれば $\{u, s\}$ のように表記する．

グラフに含まれるノードの集合は，レイヤー間で異なっていてよい．上記の関係に加えて，「楽曲同士の音響特徴量の類似性」という関係を考慮する場合，対応するレイヤー l_2 上のグラフ G_{l_2} では $V_{l_2} = S$ であり，E_{l_2} は任意の楽曲 $s \in S$ と類似している各楽曲 $s' \in S$ の間に張られたエッジの集合である．さらに，「楽曲同士の創作クリエータの共通性」という関係も考慮する場合，対応するレイヤー l_3 上のグラフ G_{l_3} では $V_{l_3} = C \cup S$（C は全クリエータの集合）であり，E_{l_3} は任意のクリエータ $c \in C$ と c が創作した各楽曲 $s \in S$ の間に張られたエッジの集合となる[*2]．

3.2 グラフに基づくノードのベクトル計算（図 1 (b)）

提案手法では，GCN に基づく推薦手法 [4] と同様に，グラフ上の各ノードを d 次元ベクトル $\boldsymbol{e}^0 \in \mathbb{R}^d$ で表す．このとき，異なるレイヤーに存在する同一ノードは同一のベクトルで表現される．例えば，3 つのレイヤー l_1，l_2，l_3 にそれぞれ存在する楽曲 s は，同一のベクトル $\boldsymbol{e}^0_s$ で表現される．そのうえで，メッセージパッシングのアーキテクチャ [26,27] を用いて，グラフ上で K ホップ以内に存在するノード集合に基づいて，各ノードのベクトルの値を計算する．以下ではまず，1 ホップ以内のノードに基づくベクトルの計算について説明し，次に k（$1 \leq k \leq K$）ホップへの一般化の方法を説明する．最後に，各ノードのベクトルの値を用いた推薦スコアの計算方法を述べる．

3.2.1 1 ホップ以内のノードに基づくベクトルの計算

ノードのベクトルの値を計算するにあたり，まず各レイヤー $l \in L$ に対して任意の重み $w_l \in \mathbb{R}^+$ を割り当てる．このとき，推薦の際により重視したい関係に対応するレイヤーの重みを大きくする．レイヤー l 上のグラフ G_l におけるノード $n \in V_l$ の，隣接ノード $t \in V_l$ と w_l のペアの集合を $N^l_n = \{(t, w_l) \mid \{n, t\} \in E_l\}$ とし，全レイヤーにおけるそれらのペアの集合を $N_n = \bigcup_{l \in L} N^l_n$ とする．そのうえで，各レイヤー上のグラフにおけるノード n の 1 ホップ先のノードに基づく n のベクトル $\boldsymbol{e}^0_{N_n}$ を次式で求める．

$$\boldsymbol{e}^0_{N_n} = \sum_{(t,w_l) \in N_n} \pi(n, t, w_l) \boldsymbol{e}^0_t, \quad \text{where} \quad \pi(n, t, w_l) = \frac{w_l}{\sum_{(t', w'_l) \in N_n} w'_l}. \tag{1}$$

これはつまり，ノード n の隣接ノードのベクトルの重み付き和を求めており，重みの大きいレイヤーのグラフ中の隣接ノードほど $\boldsymbol{e}^0_{N_n}$ の値に与える影響が大きくなる．GCN に基づく推薦手法では，Attention 機構を用いることで，各ノードがどの関係をどの程度重視しているかを自動的に計算する手法が提案されているが [10]，本手法では各関係の重要度を推薦システムの提供者が指定することで，特定の関係を重視した推薦結果の生成を可能としている．最後に，ノード n 自身も含む，n の 1 ホップ以内のノードに基づく n のベクトル $\boldsymbol{e}^1_n$ を次式で求める．

$$\boldsymbol{e}^1_n = \mathrm{LeakyReLU}\left(\boldsymbol{W}^0 \left(\boldsymbol{e}^0_n + \boldsymbol{e}^0_{N_n}\right)\right). \tag{2}$$

[*2] 本章ではこれ以降も，本節で述べた 3 つの関係（「ユーザ同士の楽曲の好みの類似性」「楽曲同士の創作クリエータの共通性」「楽曲同士の創作クリエータの共通性」）とそれに対応するレイヤー（l_1，l_2，l_3）を例として用いる．

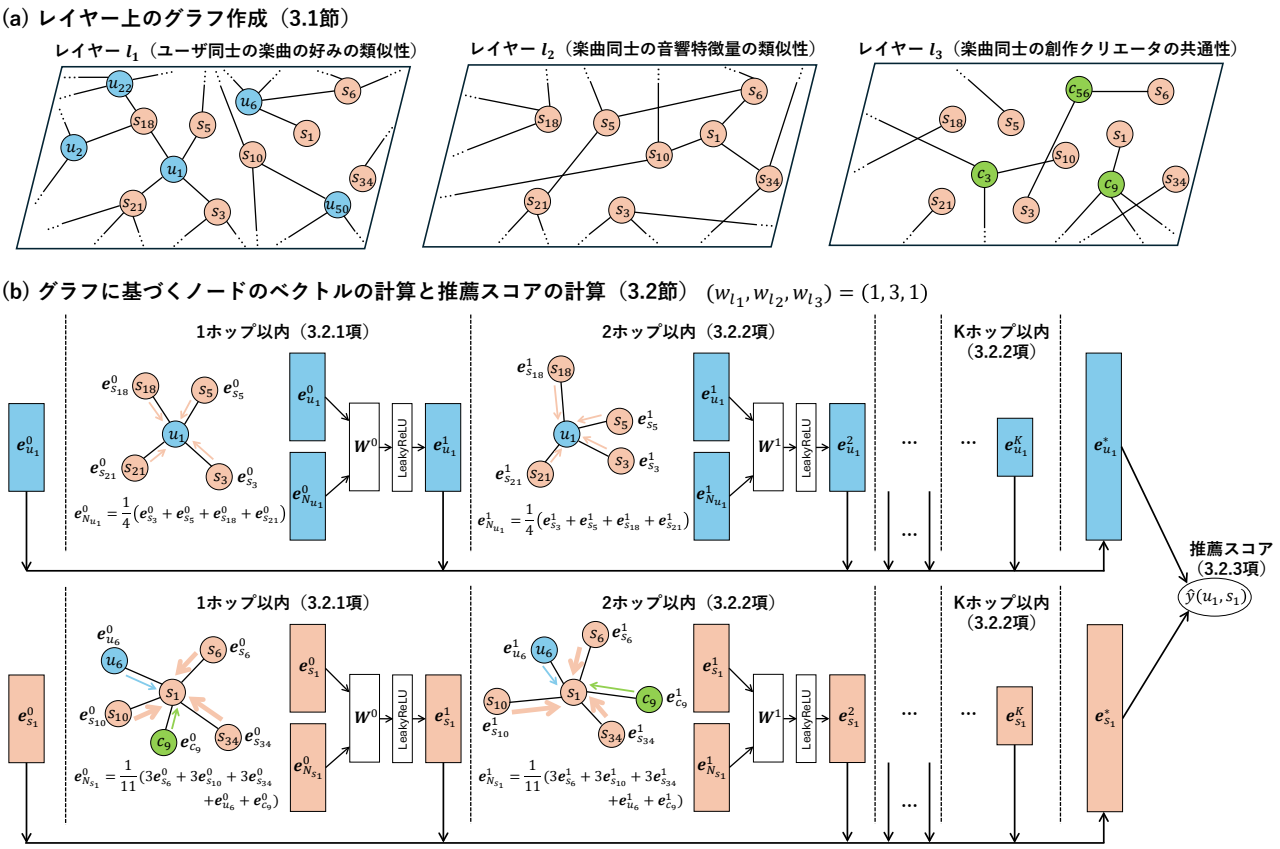

図 1　提案手法の概要.

これは GCN においてよく用いられる，2 つのベクトル（ここでは $\boldsymbol{e}_n^0$ と $\boldsymbol{e}_{N_n}^0$）の統合方法のひとつである [28]. $\boldsymbol{W}^0 \in \mathbb{R}^{d' \times d}$（$d'$ は $\boldsymbol{e}_n^1$ の次元数であり，一般的には d と同じ値や $\frac{d}{2}$ といった値が用いられる）は学習の対象となるパラメータ行列であり，LeakyReLU [29] を用いることで，非線形性を持たせている.

3.2.2　k ホップ以内のノードに基づくベクトルの計算

ノード n から 2 ホップ以内のノードに基づいて n のベクトルの値を計算する場合，$\boldsymbol{e}_{N_n}^0$ と同様に，次式により $\boldsymbol{e}_{N_n}^1$ を求める.

$$\boldsymbol{e}_{N_n}^1 = \sum_{(t,w_l) \in N_n} \pi(n, t, w_l) \boldsymbol{e}_t^1. \tag{3}$$

$\boldsymbol{e}_t^1$ は，ノード t の 1 ホップ先のノードを反映したベクトルであるため，$e_{N_n}^1$ はノード n の 2 ホップ先（n の 1 ホップ先である t の 1 ホップ先）までのノードを反映したベクトルになっている．これを用いて，n の 2 ホップ以内のノードに基づく n のベクトルは次式で計算される.

$$\boldsymbol{e}_n^2 = \mathrm{LeakyReLU}\left(\boldsymbol{W}^1 \left(\boldsymbol{e}_n^1 + \boldsymbol{e}_{N_n}^1\right)\right). \tag{4}$$

上記を一般化すると，ノード n の k ホップ以内のノードに基づく n のベクトルは次式により得られる.

$$\boldsymbol{e}_n^k = \mathrm{LeakyReLU}\left(\boldsymbol{W}^{k-1} \left(\boldsymbol{e}_n^{k-1} + \boldsymbol{e}_{N_n}^{k-1}\right)\right). \tag{5}$$

3.2.3　推薦スコアの計算（図 1 (b)）

ユーザ u に対する楽曲 s の推薦スコア $\hat{y}(u, s)$ を計算するために，全レイヤーに含まれる各ノードの K ホップ以内のノードを考慮した際の，u と s の最終的なベクトル $\boldsymbol{e}_u^*$ と $\boldsymbol{e}_s^*$ を次式により求める.

$$\boldsymbol{e}_u^* = \boldsymbol{e}_u^0 || \cdots || \boldsymbol{e}_u^K. \tag{6}$$

$$\boldsymbol{e}_s^* = \boldsymbol{e}_s^0 || \cdots || \boldsymbol{e}_s^K. \tag{7}$$

|| はベクトルの連結を表し，GCN において一般的に用いられる，異なるホップ数に基づいて計算されたノードベクトルの集約方法のひとつである [27]．推薦スコア $\hat{y}(u, s)$ は，これら 2 つのベクトルの内積により求める.

$$\hat{y}(u, s) = \boldsymbol{e}_u^{*\top} \boldsymbol{e}_s^*. \tag{8}$$

3.3　パラメータ学習

パラメータの学習には Bayesian Personalized Ranking (BPR) [30] を用いる．BPR では，パラメータの最適化をする際に用いる学習データ D は次式のように定義される.

$$D = \{(u, s, s') \mid u \in U \wedge s \in S_u^+ \wedge s' \in S \setminus S_u^+\}. \tag{9}$$

S_u^+ はユーザ u が好むアイテム（楽曲）の集合であり，三つ組 (u, s, s') はユーザ u がアイテム s' よりもアイテム s の方を好んでいることを意味する．D に基づく損失関数は次式で与えられる.

$$L = \sum_{(u,s,s')\in D} -\ln \sigma(\hat{y}(u,s) - \hat{y}(u,s')) + \lambda \|\boldsymbol{\Theta}\|_2^2. \quad (10)$$

σ はシグモイド関数であり，$\boldsymbol{\Theta} = \{\boldsymbol{E}, \boldsymbol{W}^0, \cdots, \boldsymbol{W}^{K-1}\}$ はモデルの全パラメータを，λ はハイパーパラメータを表す．$\boldsymbol{E}$ はレイヤーに含まれる全てのノードのベクトルを含む行列であり、3.1 節の例であれば

$$\boldsymbol{E} = \left[\boldsymbol{e}^0_{u_1}, \cdots, \boldsymbol{e}^0_{u_{|U|}}, \boldsymbol{e}^0_{s_1}, \cdots, \boldsymbol{e}^0_{s_{|S|}}, \boldsymbol{e}^0_{c_1}, \cdots, \boldsymbol{e}^0_{c_{|C|}}\right] \quad (11)$$

のように表される．

4. 音楽発掘サービス Kiite への応用

提案手法に基づいて，音楽発掘サービス Kiite において，透明性と操作性を備えた推薦システムを実装した．本章では，まず Kiite の概要を説明し，その次に Kiite 上のデータに対して提案手法を適用して推薦モデルの学習を行う方法を述べる．そして，推薦モデルを用いて Kiite 上に実装した推薦システムを説明する．本章ではパソコンで Kiite を利用する際のスクリーンショットを用いて推薦システムの説明を行うが，Kiite にはスマートフォン専用のインタフェースも用意されており，パソコンと同等の操作がスマートフォン上でも行える．

4.1 Kiite の概要

Kiite で再生可能な楽曲は，動画コミュニティサービス「ニコニコ動画」[*3]上の歌声合成楽曲を対象としている．ニコニコ動画では，VOCALOID [31] などの歌声合成ソフトウェアを用いて，オリジナル楽曲を創作して公開するという活動がアマチュアやプロを問わず活発である．2025 年 2 月の時点で，52 万曲以上のそうした楽曲を Kiite 上で再生できる．ニコニコ動画では全ての楽曲は音楽動画として公開されており，Kiite 上で楽曲を聴く際は動画視聴用プレーヤ（ニコニコ動画が提供する外部プレーヤ）を用いて音楽動画が再生される．Kiite にユーザ登録をすることで，Kiite のユーザはアイコンの設定，お気に入りリストへの楽曲の登録，複数のプレイリストの作成，他のユーザが作成したプレイリストの聴取などが可能になる．

4.2 推薦モデルの学習

Kiite ではユーザは様々な機能を通じて楽曲を聴くことができる．我々はユーザがお気に入りリストやプレイリストに追加した楽曲を，そのユーザの好みに合った楽曲集合（すなわち，3.3 節の S_u^+）とした．楽曲推薦においては，3.1 節で例としてあげた 3 つの関係を用いた．「ユーザ同士の楽曲の好みの類似性」の関係に対応するレイヤー l_1 上のグラフでは，ユーザ $u \in U$ のノードと，S_u^+ に含まれる各楽曲のノードの間にエッジを張った．「楽曲同士の音響特徴量の類似性」の関係に対応するレイヤー l_2 上のグラフでは，音響特徴量に基づいて楽曲間の類似度を計算し，楽曲 $s \in S$ のノードと，s との類似度が上位 5 件の各楽曲のノードの間にエッジを張った．「楽曲同士の創作クリエータの共通性」の関係に対応するレイヤー l_3 上のグラフでは，楽曲のメタデータに基づいて，クリエータ $c \in C$ のノードと c が創作した各楽曲のノードの間にエッジを張った．

各レイヤーの重みは，3 つの関係をバランス良く重視して $(w_{l_1}, w_{l_2}, w_{l_3}) = (1,1,1)$ としたものに加えて，3 つの関係のうち一つを重視して $(w_{l_1}, w_{l_2}, w_{l_3})$ を $(100,1,1)$, $(1,100,1)$, $(1,1,100)$ とした 3 つの，計 4 つを用いた．すなわち，レイヤーの重みの組み合わせに応じて 4 つの異なる推薦モデルが作られるが，4 つのいずれの場合でも，学習に用いるデータ，レイヤーの重みを除くハイパーパラメータ，パラメータの学習方法は共通である．ハイパーパラメータに関しては，ノードのベクトル $\boldsymbol{e}^0$ の次元数 d を 64，各ノードで考慮する最長のホップ数 K を 3 とし，W^0，W^1，W^2 はそれぞれ $\mathbb{R}^{64\times64}$，$\mathbb{R}^{32\times64}$，$\mathbb{R}^{16\times32}$ 上の行列とした．つまり，式 (6) の $\boldsymbol{e}^*_u$ と式 (7) の $\boldsymbol{e}^*_s$ はいずれも $64+64+32+16 = 176$ 次元のベクトルで表される．パラメータの学習には mini-batch Adam optimizer [32] を用いた．

4.3 推薦システム（デフォルト推薦エンジン）

本節で述べる推薦システムは，ユーザが日頃 Kiite 上で聴く楽曲に基づいてユーザの好みを学習して推薦結果を生成しており，Kiite では「デフォルト推薦エンジン」と呼んでいる．学習により得られた 4 種類の推薦モデルを用いて Kiite に実装したデフォルト推薦エンジンのインタフェースを図 2 に示す．Kiite では，ユーザにとっての分かりやすさを考慮して，4 種類の各推薦モデルのことを「推薦モード」と呼んでいる．さらに，各推薦モードをユーザが容易に区別できるように，3 つの関係をバランス良く重視した推薦モード（$(w_{l_1}, w_{l_2}, w_{l_3}) = (1,1,1)$）を「バランス重視」，「ユーザ同士の楽曲の好みの類似性」を重視した推薦モード（$(w_{l_1}, w_{l_2}, w_{l_3}) = (100,1,1)$）を「人気度重視」，「楽曲同士の音響特徴量の類似性」を重視した推薦モード（$(w_{l_1}, w_{l_2}, w_{l_3}) = (1,100,1)$）を「楽曲類似度重視」，「楽曲同士の創作クリエータの共通性」を重視した推薦モード（$(w_{l_1}, w_{l_2}, w_{l_3}) = (1,1,100)$）を「クリエイター重視」と表現している[*4]．

ユーザが初めて推薦システムにアクセスした際は，「バランス重視」の推薦結果が表示される．「推薦モード切替パネル」（図 2 Ⓐ）には 4 つの推薦モードの名前が書かれた

[*3] https://www.nicovideo.jp

[*4] 楽曲に付与されたタグを利用するなどして，さらにレイヤーを増やすことも可能であるが，ユーザに提示する推薦理由の数は 4 個程度が望ましいことが知られているため [21]，Kiite ではこの 4 つの推薦モードを採用した．

図 2 デフォルト推薦エンジンのインタフェース.

図 3 推薦モード切替パネルの説明.

トグルボタンが表示され，ユーザが閲覧したい推薦モードのボタンを選択すると，その推薦モードでの推薦結果が表示される．「推薦モード切替パネルの説明」というテキスト（Ⓑ）を選択すると，各推薦モードの説明がポップアップで表示され（図 3），各推薦モードで表示される推薦結果の違いをユーザが直感的に理解できるようにして，推薦システムにおける透明性の高さを実現している．そのうえでユーザは，推薦モード間での推薦結果の違いを見比べたり，自身の好みの推薦モードに切り替えて楽曲を聴いたりでき，これにより推薦システムにおける操作性を実現している．ユーザが Kiite の推薦エンジンのページに再度アクセスすると，前回のアクセス時に最後に選択した推薦モードで推薦結果が表示される．これによりユーザは，自身の好みの推薦モードを見つけたときなどに，毎回手動で推薦モードを切り替えることなく，好みの推薦モードの推薦結果を手軽に確認できる．

推薦結果には最大で 100 件の楽曲が表示され，楽曲のサムネイルやタイトルをクリックすると楽曲が再生される．楽曲が気に入れば，「お気に入り」ボタン（図 2 Ⓒ）を押してお気に入りリストに楽曲を追加したり，「プレイリスト」ボタン（Ⓓ）を押して自身が作成した任意のプレイリストに楽曲を追加したりできる．また，楽曲が自身の好みに合っているか否かを推薦システムにフィードバックするための「合っている」（サムアップ）ボタンと「合ってない」（サムダウン）ボタン（Ⓔ）もあり，これらのボタンが押された楽曲の情報は，後述する推薦モデルの更新時に使用される．推薦された各楽曲は最初から最後まで聴くことができるだけでなく，サビの自動解析技術 [33] を用いることで，各楽曲のサビから 30 秒や 60 秒などの一定時間再生されたら，自動的に次の楽曲が再生されるような聴き方もできるなど，推薦された楽曲を効率的にチェックする手段が提供されている．

Kiite では，推薦楽曲に対するユーザのインタラクションを元に，1 日に一度，デフォルト推薦エンジンの 4 つの各推薦モデルが更新される．ユーザがインタラクションを行った楽曲のうち，お気に入りリストに追加された楽曲，プレイリストに追加された楽曲，「合っている」ボタンが押された楽曲は正例，「合ってない」ボタンが押された楽曲は負例として用いる．ユーザ u の正例の各楽曲は S_u^+ に追加され，「ユーザ同士の楽曲の好みの類似性」の関係に対応するレイヤー上のグラフで，u のノードと正例の各楽曲の間に新たにエッジを張る．これにより，式 (1) でユーザ u の 1 ホップ先のノードに基づいて u のベクトルを計算する際などに，新たにエッジが張られた楽曲のノードのベクトル

図 4 図 2 の「推薦エンジンを選ぶ」(Ⓕ) をクリックした際に表示される画面.

が考慮されることになる．一方で負例は，BPR において式 (10) で損失関数を計算する際に考慮される．具体的には，BPR では一般的に，式 (10) 中の楽曲 s' を $S \setminus S_u^+$ の中からランダムにサンプリングするが，Kiite では一定の確率でユーザ u がサムダウンした負例の楽曲集合から楽曲 s' をサンプリングする．これにより，モデルのパラメータがより効率的に学習されることが期待できる．

推薦モデルを更新する際は，パラメータを一から学習し直すのではなく，前日に得られたパラメータを更新する方法を取っている．前日には学習データに含まれていなかった，新規登録したユーザ，新しく創作された楽曲，初めて楽曲を創作したクリエータが存在する場合，それらも含めてパラメータを更新する必要がある．その際，そうした新規データのベクトルの値をランダムに決めるよりも効率的にベクトルの値が求められるように，新規楽曲 s であれば s と音響特徴量の類似度が上位 10 件の楽曲の平均ベクトル，新規ユーザ u であれば S_u^+ に含まれる楽曲の平均ベクトル，新規クリエータ c であれば c が創作した楽曲の平均ベクトルをそれぞれの初期値として用いている．

4.4 カスタム推薦エンジン

Kiite では 4.3 節で述べたデフォルト推薦エンジンに加えて，ユーザの気分や状況に応じた楽曲の推薦が可能な「カスタム推薦エンジン」と呼ばれる推薦システムも提供している．デフォルト推薦エンジンは Kiite にアカウントを持つユーザに対して自動的に一つ作られるのに対して，カスタム推薦エンジンはユーザが自分で作ることができる．カスタム推薦エンジンを作りたいユーザは，図 2 Ⓕの「推薦エンジンを選ぶ」をクリックし，遷移先の画面で「新しいカスタム推薦エンジンを作成」(図 4 Ⓖ) をクリックする．作成用の画面が表示されたら，まずはそのカスタム推薦エンジンの名前を入力する (図 5 Ⓘ)．「リラックスできる曲」や「勉強中の BGM 用」など，そのカスタム推薦エンジンを使用する際の気分や状況などに応じた任意の名前を入力できる．次に，「シード楽曲」と呼ばれる楽曲を 1

図 5 推薦モード切替パネルの説明.

曲以上指定する．例えば「リラックスできる曲」という名前を付けたカスタム推薦エンジンであれば，ユーザは自分が既に知っている「リラックスできる曲」に適した楽曲をシード楽曲として指定することで，カスタム推薦エンジンにそのシード楽曲を参考にした推薦結果を生成させることができる．シード楽曲を指定する際は，自身が過去に作成したプレイリスト中の楽曲をまとめて指定する (Ⓙ) 他，キーワード検索を用いて特定の楽曲を検索して指定することもできる (Ⓚ)．シード楽曲を指定して「作成」ボタン (Ⓛ) を押すと，1 時間以内にそのカスタム推薦エンジン用の推薦結果が生成される．

各ユーザはカスタム推薦エンジンを何個でも作成でき，「推薦エンジンを選ぶ」(図 2 Ⓕ) をクリックすると，自身がこれまでに作成したカスタム推薦エンジンの一覧が表示される．図 4 では，3 件のカスタム推薦エンジンが作成されている (Ⓗ)．この中からカスタム推薦エンジンを選択すると，そのカスタム推薦エンジンの推薦結果が表示される．カスタム推薦エンジンを選択した後のインタフェースはデフォルト推薦エンジンと同様で，楽曲の再生に加えて，4 つの推薦モードの切り替えや，推薦された楽曲に対するフィードバックなどができる．

カスタム推薦エンジンでも，3 章の提案手法を用いて推薦結果を生成しているが，デフォルト推薦エンジンにおける「ユーザ同士の楽曲の好みの類似性」の関係の代わりに「カスタム推薦エンジン同士の楽曲の好みの類似性」という関係を考慮している点が異なる．この関係に対応するレイヤー上のグラフでは，カスタム推薦エンジンと楽曲がノードとして存在し，カスタム推薦エンジンのノードと，そのカスタム推薦エンジンの各シード楽曲のノードの間にエッ

ジが張られる．「楽曲同士の類似性」と「楽曲同士の創作クリエータの共通性」の関係を考慮する点はデフォルト推薦エンジンと共通しており，この 3 つのレイヤーに基づいて推薦モデルを学習する[*5]．

カスタム推薦エンジンの推薦結果に対するユーザのインタラクションを通じて正例と負例の楽曲集合が得られたら，デフォルト推薦エンジンと同様に正例は「カスタム推薦エンジン同士の楽曲の好みの類似性」の関係に対応するレイヤー上のグラフの拡張に，負例は BPR における楽曲のサンプリングに使用し，カスタム推薦エンジンを更新する．

5. 分析

我々は 4 章で述べた推薦システムを，2022 年 11 月 22 日に Kiite 上で公開した．本章では，透明性と操作性を備えた推薦システム上でのユーザのインタラクションを分析する．そのために，2022 年 11 月 22 日から 2024 年 12 月 24 日までの 2 年以上に渡るユーザの利用ログを用いた[*6]．

5.1 分析対象の推薦エンジン

デフォルト推薦エンジンに関する分析では，分析期間中に楽曲が 1 曲以上再生された 3,264 個の推薦エンジン（すなわち 3,264 名分のデフォルト推薦エンジン）を対象とした．カスタム推薦エンジンの分析でも同様に，分析期間中に楽曲が 1 曲以上再生された 1,332 個の推薦エンジンを対象とした．この 1,332 個のカスタム推薦エンジンは 722 名のユーザによって作成され，そのうち 2 個以上のカスタム推薦エンジンを作成したユーザは 253 名（35.04%），3 個以上は 105 名（14.54%），5 個以上は 40 名（5.54%）のような分布になっており，最も多いユーザは 41 個のカスタム推薦エンジンを作成していた．デフォルト推薦エンジンとカスタム推薦エンジンを合わせた場合の，分析対象となったユニークユーザ数は 3,413 名であった．

5.2 推薦モードの切り替え

3,264 個のデフォルト推薦エンジンのうち，初めてデフォルト推薦エンジンにアクセスした際に表示される「バランス重視」の推薦モードから別の推薦モードに一度以上切り替えられたのは 53.95%にあたる 1,761 個であった．カスタム推薦エンジンでは，1,332 個のうち 69.59%にあたる 927 個の推薦エンジンで，推薦モードが「バランス重視」から一度以上切り替えられていた．デフォルト推薦エンジンは全てのユーザに対して自動的に作られるため，楽曲を推薦されることに強い興味がなく，最初に表示される「バランス重視」の推薦モードだけを使用しているユーザや推薦システムに少しだけ触れて使わなくなったユーザ，また推薦モードが切り替えられることに気づいていないユーザなども含まれている可能性があり，53.95%という割合にとどまった．それに対して，カスタム推薦エンジンはユーザが能動的に作る必要があり，楽曲推薦に興味を持つユーザが利用する傾向にあるため，自身の好みに合う推薦結果を探索するために，積極的に推薦モードを切り替え，それが 69.59%という，デフォルト推薦エンジンよりも高い割合につながったと考えられる．

表 1 m 種類の推薦モードが使用された推薦エンジンの個数

推薦エンジン	$m=1$	$m=2$	$m=3$	$m=4$
デフォルト推薦エンジン	1,503	533	404	824
カスタム推薦エンジン	405	412	195	320

次に，デフォルト推薦エンジンとカスタム推薦エンジンのそれぞれで，使用した推薦モードの種類数の分布を調べた結果を表 1 に示す．種類数が 1 というのは，推薦モードの切り替えを一度も行っておらず，「バランス重視」の推薦モードのみを使っていることを意味する．2 種類以上の推薦モードが使用された場合は，デフォルト推薦エンジンでは 4 種類の推薦モード，つまり全ての推薦モードの推薦結果を閲覧した推薦エンジンの数が最も多かった．一方でカスタム推薦エンジンでは，2 種類の推薦モードを使用した推薦エンジンの数が全体を通して最も多かった．このような違いが生じた理由として，デフォルト推薦エンジンはユーザのサービス登録時に一つ作られ，その後ずっと使い続けるものであり，一般的にはカスタム推薦エンジンよりも長い期間に渡って使用することから，4 つの推薦モードによる推薦結果の違いを見比べようと思う機会が多い可能性があげられる．カスタム推薦エンジンで 2 種類を使用したものが最も多いのは，デフォルト推薦エンジンを使用したり，カスタム推薦エンジンを作って使用したりした経験から，ユーザが自身の好みに合う推薦モードを「人気度重視」「楽曲類似度重視」「クリエイター重視」の中から一つ見つけ，新たなカスタム推薦エンジンを作った際に，その好みの推薦モードでのみ推薦エンジンを使用しているユーザが一定数含まれるためであると思われる．

最後に，使用した推薦モードの種類数による，推薦結果の楽曲に対するユーザのインタラクションの頻度の違いを分析した．対象としたインタラクションは「楽曲再生」「お気に入りリストへの追加」「プレイリストへの追加」「サムアップ（「合っている」ボタンのクリック）」「サムダウン（「合ってない」ボタンのクリック）」の 5 種類である．デフォルト推薦エンジンとカスタム推薦エンジンに関する分析結果を表 2 と表 3 にそれぞれ示す．例えば表 2 であれば，インタラクションが「楽曲再生」，種類数が「2」のセルには「51.96 (144.23)」と記述されているが，これは，使用した推薦モードの種類数が 2 であった 533 件のデフォル

[*5] すなわち，Kiite 上にはデフォルト推薦エンジン用の推薦モデルが 4 個と，カスタム推薦エンジン用の推薦モデルが 4 個の，計 8 個の推薦モデルが存在する．
[*6] Kiite の利用規約には，ユーザの利用ログを学術研究の目的で使用できることが記述されている．

表 2 m 種類の推薦モードが使用されたデフォルト推薦エンジンにおける，各インタラクションの頻度の平均値（括弧内の数値は標準偏差）．

インタラクション	$m=1$	$m=2$	$m=3$	$m=4$
楽曲再生	29.78 (90.55)	51.96 (144.23)	96.64 (262.5)	263.55 (1010.40)
お気に入りリストへの追加	3.46 (13.43)	10.26 (39.95)	13.85 (53.40)	37.00 (145.03)
プレイリストへの追加	1.14 (5.57)	3.38 (12.34)	5.75 (40.06)	10.89 (48.56)
サムアップ	7.75 (32.15)	13.07 (37.53)	21.04 (72.42)	62.47 (244.01)
サムダウン	7.14 (40.90)	9.87 (52.33)	27.95 (221.34)	91.41 (661.26)

表 3 m 種類の推薦モードが使用されたカスタム推薦エンジンにおける，各インタラクションの頻度の平均値（括弧内の数値は標準偏差）．

インタラクション	$m=1$	$m=2$	$m=3$	$m=4$
楽曲再生	27.82 (59.75)	73.71 (132.13)	111.19 (403.53)	871.26 (5636.19)
お気に入りリストへの追加	4.82 (15.30)	13.42 (37.95)	12.69 (32.32)	50.27 (229.43)
プレイリストへの追加	2.08 (5.64)	6.03 (16.30)	4.63 (11.92)	13.94 (42.05)
サムアップ	5.58 (17.08)	15.71 (28.19)	15.53 (28.59)	103.46 (578.79)
サムダウン	5.72 (32.73)	29.30 (87.93)	32.71 (132.17)	668.65 (3796.75)

ト推薦エンジンにおいて，楽曲再生をした回数の平均値が 51.96 で，標準偏差が 144.23 であることを意味する．

表 2 のデフォルト推薦エンジンでは，いずれのインタラクションでも，種類数が増えるにつれて，インタラクションの頻度の平均値が増加していた．つまり，多くの推薦モードを使うユーザほど，より積極的に推薦エンジンとインタラクションを行い，好みに合う楽曲（お気に入りリストに追加した楽曲，プレイリストに追加した楽曲）をより多く見つけられていることが明らかになった．表 3 のカスタム推薦エンジンでは，「お気に入りリストへの追加」「プレイリストへの追加」「サムアップ」に関しては，種類数が 2 の方が，種類数が 3 の場合よりも平均値が高かった．このことは，先ほど述べた，特定の好みの推薦モードのみを通じて，カスタム推薦エンジンを積極的に使用しているユーザが一定数いることの根拠となりうる．

表 3 においても，4 種類全ての推薦モードを使用した場合が，いずれのインタラクションでも平均値が最大である点は表 2 と共通していた．この結果から，例えば 1 種類の推薦モードしか使用していないユーザに対して，複数の推薦モードを使ってみるよう促すメッセージを Kiite 上で表示するといった応用が考えられる．そのメッセージを見て複数の推薦モードを使用したユーザのインタラクションの頻度が高くなれば，推薦理由を提示する際に推薦システムに操作性を持たせてユーザに使ってもらうことと，インタラクションの高さとの間に因果関係があることが示せる点で有用な知見となるため，興味深い研究課題である．

表 3 において，特に種類数が 4 の場合にサムダウンの平均値が 668.65 と高くなっているが，これは一部のカスタム推薦エンジンにおいてユーザが大量のサムダウンをしていたためである．実際，最もサムダウンの多かったカスタム推薦エンジンでは，53,726 回ものサムダウンが行われていた．しかし，それと同時に，そのカスタム推薦エンジンではお気に入りリストへの追加も 3,197 回行われていた[*7]．これは，このカスタム推薦エンジンを作ったユーザは，たった一つのカスタム推薦エンジンを通じて 3,197 曲もの好みに合う新たな楽曲に出会えたことを意味している．したがって，サムダウンが多いからといって，推薦エンジンの推薦精度が低いというわけではなく，このユーザはサムダウンのフィードバックをすることが推薦精度を高めることにつながると考え，大量のサムダウンをするに至ったと思われる．

5.3 推薦モード内でのインタラクション

本節では，推薦モード内で行われる各インタラクションを分析することで，複数種類の推薦モードを提供し，かつそれが切り替えられるように操作性を持たせることに意義があるかを検証する．ユーザがインタラクションを行う際の推薦モードの使い分け方を分析したいので，一定程度以上インタラクションが行われている推薦エンジンを対象とした．具体的には，「お気に入りリストへの追加」「プレイリストへの追加」「サムアップ」「サムダウン」の 4 種類のインタラクションの合計回数が 10 回以上の推薦エンジンを対象とした．その結果，デフォルト推薦エンジンは 1,288 個，カスタム推薦エンジンは 745 個が分析の対象となった．

まず，4 種類の各推薦モードにおいて，5.2 節で対象とした 5 種類の各インタラクションが 1 回以上行われた推薦エンジンの数と，インタラクションの頻度を求めた．デフォルト推薦エンジンに関する結果を表 4 に，カスタム推薦エンジンに関する結果を表 5 に示す．デフォルト推薦エンジンでは，いずれのインタラクションでも，インタラクションが行われた推薦エンジンの数は「バランス重視」「楽曲類似度重視」「人気度重視」「クリエイター重視」の順に

[*7] 楽曲再生は 92,383 回，プレイリストへの追加は 362 回，サムアップは 9,776 回であった．

表 4 各推薦モードにおいて各インタラクションを行ったデフォルト推薦エンジンの個数（括弧内の数値はインタラクションの頻度）.

インタラクション	バランス重視	人気度重視	楽曲類似度重視	クリエイター重視
楽曲再生	1,180 (138,891)	601 (53,019)	750 (90,888)	442 (21,084)
お気に入りリストへの追加	935 (20,739)	389 (7,814)	548 (13,621)	233 (2,765)
プレイリストへの追加	505 (7,069)	191 (2,767)	291 (3,651)	93 (707)
サムアップ	970 (39,477)	430 (15,704)	612 (18,067)	293 (4,197)
サムダウン	638 (41,560)	244 (17,885)	377 (35,813)	174 (7,165)

表 5 各推薦モードにおいて各インタラクションを行ったカスタム推薦エンジンの個数（括弧内の数値はインタラクションの頻度）.

インタラクション	バランス重視	人気度重視	楽曲類似度重視	クリエイター重視
楽曲再生	594 (96,541)	260 (70,540)	539 (121,156)	185 (42,482)
お気に入りリストへの追加	415 (9,329)	160 (4,052)	403 (10,297)	93 (1,962)
プレイリストへの追加	294 (3,400)	101 (1,000)	312 (3,613)	58 (445)
サムアップ	486 (14,495)	219 (8,939)	483 (15,516)	130 (5,329)
サムダウン	383 (67,913)	148 (57,156)	396 (71,392)	103 (38,059)

表 6 各インタラクションにおいて各推薦モードを最も多く使用したデフォルト推薦エンジンの個数.

インタラクション	バランス重視	人気度重視	楽曲類似度重視	クリエイター重視
楽曲再生	799	127	300	37
お気に入りリストへの追加	621	125	217	24
プレイリストへの追加	280	73	122	13
サムアップ	698	118	265	31
サムダウン	398	55	151	20

表 7 各インタラクションにおいて各推薦モードを最も多く使用したカスタム推薦エンジンの個数.

インタラクション	バランス重視	人気度重視	楽曲類似度重視	クリエイター重視
楽曲再生	323	67	334	10
お気に入りリストへの追加	231	55	231	13
プレイリストへの追加	166	33	155	9
サムアップ	266	83	289	10
サムダウン	200	38	235	13

多かった．カスタム推薦エンジンでは，いずれのインタラクションでも「クリエイター重視」を使用した推薦エンジンの数が最も少なく，次いで「人気度重視」が少なかった点はデフォルト推薦エンジンと共通していた．しかし，デフォルト推薦エンジンとは異なり，カスタム推薦エンジンでは「プレイリストへの追加」と「サムダウン」では「楽曲類似度重視」が最も多かった．さらに，それ以外の「楽曲再生」「お気に入りリストへの追加」「サムアップ」でも，デフォルト推薦エンジンに比べると，カスタム推薦エンジンでは「バランス重視」と「楽曲類似度重視」の差は小さかった．この結果は，カスタム推薦エンジンにおける「楽曲類似度重視」の推薦モードの需要の高さを示している．カスタム推薦エンジンでは「静かなピアノ曲」や「バラード系」のように，楽曲の音に関係する名前が付けられたカスタム推薦エンジンが一定数作られており，そうした推薦エンジンでは「楽曲類似度重視」の推薦モードが使用されることが多いのが，需要の高さの一因ではないかと推測される．「音に関するカスタム推薦エンジン」や「状況に関するカスタム推薦エンジン」のように，名前に応じたカテゴリを作成したうえでカスタム推薦エンジンをカテゴリに分類し，カテゴリごとに表 5 のような結果を求めて比較することで，この推測の妥当性が検証できるため，より詳細な分析として今後取り組みたい．

次に，ユーザが最も好んで使用した推薦モードを分析するために，各インタラクションにおいて各推薦エンジンによって最も使用された推薦モードを求めた．例えば，あるデフォルト推薦エンジンで，「楽曲再生」を行った回数が「バランス重視」で 22 回，「人気度重視」で 39 回，「楽曲類似度重視」で 24 回，「クリエイター重視」で 16 回となっていたら，この推薦エンジンにおいて「楽曲再生」で最も使用された推薦モードは「人気度重視」となる．このとき，最も使用された推薦モードであっても，その使用回数が 1 回など少なければその推薦モードが「楽曲再生」をするために好んで使用されているとは十分に言えないので，最も使用された推薦モードの使用回数が 3 回以上の推薦エンジンのみを対象とした．デフォルト推薦エンジンとカスタム推薦エンジンに関する結果を表 6 と表 7 にそれぞれ示す．デフォルト推薦エンジンでは，表 4 と同様に，全てのインタラクションにおいて「バランス重視」が最も好んで使用され，次いで「楽曲類似度重視」「人気度重視」「クリエイター重視」の順であった．カスタム推薦エンジンでも表 5 と同様に，「楽曲類似度重視」が好んで使用される傾向にあった．デフォルト推薦エンジンとカスタム推薦エンジンのいずれでも，「人気度重視」と「クリエイター重視」が最も使用される頻度は相対的には低かったものの，これらの推薦モードを最も好むユーザが存在することは示された．したがって，複数の推薦モードから，ユーザが自身の閲覧したい推薦結果を選択できるようにすることには意義があ

表 8 各インタラクションにおいて推薦モードを m 個併用したデフォルト推薦エンジンの個数.

インタラクション	m				$m \geq 2$ の推薦エンジンの割合
	1	2	3	4	
楽曲再生	579	279	155	237	53.68%
お気に入りリストへの追加	586	216	96	73	39.65%
プレイリストへの追加	334	83	36	26	30.27%
サムアップ	613	227	116	135	43.81%
サムダウン	367	108	46	93	40.23%

表 9 各インタラクションにおいて推薦モードを m 個併用したカスタム推薦エンジンの個数.

インタラクション	m				$m \geq 2$ の推薦エンジンの割合
	1	2	3	4	
楽曲再生	341	206	89	93	53.22%
お気に入りリストへの追加	328	121	34	35	36.68%
プレイリストへの追加	248	71	22	14	30.14%
サムアップ	356	164	56	63	44.29%
サムダウン	266	125	35	58	45.04%

ると言える.

最後に，推薦エンジンの使用ユーザが複数の推薦モードを併用しているかを分析するために，各インタラクションにおいて各推薦エンジンによって一定回数（本分析では 3 回とした）以上使用された推薦モードを求めた．例えば，あるデフォルト推薦エンジンで，「楽曲再生」を行った回数が「バランス重視」で 1 回，「人気度重視」で 2 回，「楽曲類似度重視」で 9 回，「クリエイター重視」で 5 回となっていたとすると，この推薦エンジンでは 3 回以上使用された「楽曲類似度重視」「クリエイター重視」の 2 種類の推薦モードが「楽曲再生」で併用されたとみなす．デフォルト推薦エンジンに関する結果を表 8 に，カスタム推薦エンジンに関する結果を表 9 に示す．いずれの場合も，全てのインタラクションにおいて，30%以上の推薦エンジンで 2 つ以上の推薦モードが併用されていた．「楽曲再生」「サムアップ」「サムダウン」に関しては，その割合は 40%以上とさらに高くなる．表 6 と表 7 では推薦エンジンによって好んで使用される推薦モードに違いがあることが示されたが，表 8 と表 9 の結果から，ユーザは状況に応じて複数の推薦モードを併用していることも明らかになった．したがって，各推薦エンジンで最も好まれる推薦モードだけをユーザが使用できるようにすれば良いわけではなく，複数の推薦モードをユーザが自由に切り替えられるように，操作性を持たせることの意義がこの分析から示された.

6. おわりに

本稿では，透明性と操作性を備えた推薦システムの実現に取り組んだ．本研究の主な貢献は以下のようにまとめられる.

- 推薦における様々な関係を考慮しつつ，特定の関係を重視した推薦結果を生成するために，GCN に基づく推薦手法を提案した.
- 提案手法を元に，4 つの推薦モードを切り替え可能な推薦システムを実装し，誰もが利用できる形で音楽発掘サービス Kiite 上で公開した.
- Kiite の推薦システムの 2 年以上に渡る利用ログを分析し，複数の推薦理由に基づく推薦結果をユーザが自由に切り替えられるように，操作性を持たせることの意義を示した.

本稿では Kiite の利用ログに基づいて推薦モード切り替え機能の有用性を検証したが，我々は別途，その提案した切り替え機能がユーザの推薦システムに対する信頼にどういう影響を与えるかを調査するためのユーザ実験も実施している．その結果，推薦モードの切り替えができることが推薦システムに対する信頼性の向上に繋がることなども明らかになっている [34]．今後も，大規模な利用ログに基づく工学的な側面からの評価と，ユーザ実験に基づく心理学的な側面からの評価を併用しながら，提案する推薦システムの有用性をより深く検証したい.

Kiite では VOCALOID 楽曲を対象にしているが，VOCALOID の楽曲のジャンルは多様であることを踏まえると，我々が提案した推薦システムは幅広い楽曲に有効であると考えられる．また，提案した推薦手法はグラフで表現できる任意のデータを対象に推薦結果を生成できるため，音楽以外のドメインでも同様の推薦システムを実現可能である．多様なドメインにおいて，そうした透明性と操作性を備えた推薦システムが実現されて普及していくことを期待する.

謝辞 本研究の一部は JST CREST JPMJCR20D4 の支援を受けた.

参考文献

[1] Jacobson, K., Murali, V., Newett, E., Whitman, B. and Yon, R.: Music personalization at Spotify, *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, p. 373 (2016).
[2] Bontempelli, T., Chapus, B., Rigaud, F., Morlon, M., Lorant, M. and Salha-Galvan, G.: Flow Moods: Recommending music by moods on Deezer, *Proceedings of the 16th ACM Conference on Recommender Systems*, RecSys '22, pp. 452–455 (2022).
[3] Schedl, M.: Deep learning in music recommendation systems, *Frontiers in Applied Mathematics and Statistics*, Vol. 5, pp. 1–9 (2019).
[4] Wu, S., Sun, F., Zhang, W., Xie, X. and Cui, B.: Graph neural networks in recommender systems: A survey, *ACM Computing Surveys*, Vol. 55, No. 5, pp. 1–37 (2022).
[5] Sinha, R. and Swearingen, K.: The role of transparency in recommender systems, *Proceedings of the CHI 2002 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '02, pp. 830–831 (2002).
[6] Gedikli, F., Jannach, D. and Ge, M.: How should I explain? A comparison of different explanation types for recommender systems, *International Journal of Human-Computer Studies*, Vol. 72, No. 4, pp. 367–382

(2014).
[7] Ma, B., Lu, M., Taniguchi, Y. and Konomi, S.: CourseQ: The impact of visual and interactive course recommendation in university environments, *Research and Practice in Technology Enhanced Learning*, Vol. 16, No. 18, pp. 1–24 (2021).
[8] Zhang, Y. and Chen, X.: Explainable recommendation: A survey and new perspectives, *Foundations and Trends® in Information Retrieval*, Vol. 14, No. 1, pp. 1–101 (2020).
[9] Chatti, M. A., Guesmi, M. and Muslim, A.: Visualization for recommendation explainability: A survey and new perspectives, *ACM Transactions on Interactive Intelligent Systems*, Vol. 14, No. 3, pp. 1–40 (2024).
[10] Wang, X., He, X., Cao, Y., Liu, M. and Chua, T.-S.: KGAT: Knowledge graph attention network for recommendation, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pp. 950–958 (2019).
[11] He, X., Deng, K., Wang, X., Li, Y., Zhang, Y. and Wang, M.: LightGCN: Simplifying and powering graph convolution network for recommendation, *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, pp. 639–648 (2020).
[12] Vig, J., Sen, S. and Riedl, J.: Tagsplanations: Explaining recommendations using tags, *Proceedings of the 14th International Conference on Intelligent User Interfaces*, IUI '09, pp. 47–56 (2009).
[13] Millecamp, M., Htun, N. N., Conati, C. and Verbert, K.: To explain or not to explain: The effects of personal characteristics when explaining music recommendations, *Proceedings of the 24th International Conference on Intelligent User Interfaces*, IUI '19, pp. 397–407 (2019).
[14] Jin, Y., Seipp, K., Duval, E. and Verbert, K.: Go with the flow: Effects of transparency and user control on targeted advertising using flow charts, *Proceedings of the 13th International Working Conference on Advanced Visual Interfaces*, AVI '16, pp. 68–75 (2016).
[15] Dominguez, V., Messina, P., Donoso-Guzmán, I. and Parra, D.: The effect of explanations and algorithmic accuracy on visual recommender systems of artistic images, *Proceedings of the 24th International Conference on Intelligent User Interfaces*, IUI '19, pp. 408–416 (2019).
[16] He, X., Chen, T., Kan, M.-Y. and Chen, X.: TriRank: Review-aware explainable recommendation by modeling aspects, *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pp. 1661–1670 (2015).
[17] Balog, K., Radlinski, F. and Arakelyan, S.: Transparent, scrutable and explainable user models for personalized recommendation, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '19, pp. 265–274 (2019).
[18] Herlocker, J. L., Konstan, J. A. and Riedl, J.: Explaining collaborative filtering recommendations, *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work*, CSCW '00, pp. 241–250 (2000).
[19] Tsukuda, K. and Goto, M.: Explainable recommendation for repeat consumption, *Proceedings of the 14th ACM Conference on Recommender Systems*, RecSys '20, pp. 462–467 (2020).
[20] Kouki, P., Fakhraei, S., Foulds, J., Eirinaki, M. and Getoor, L.: HyPER: A flexible and extensible probabilistic framework for hybrid recommender systems, *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, pp. 99–106 (2015).
[21] Kouki, P., Schaffer, J., Pujara, J., O'Donovan, J. and Getoor, L.: Personalized explanations for hybrid recommender systems, *Proceedings of the 24th International Conference on Intelligent User Interfaces*, IUI '19, pp. 379–390 (2019).
[22] Tsukuda, K. and Goto, M.: DualDiv: Diversifying items and explanation styles in explainable hybrid recommendation, *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, pp. 398–402 (2019).
[23] Bostandjiev, S., O'Donovan, J. and Höllerer, T.: TasteWeights: A visual interactive hybrid recommender system, *Proceedings of the 6th ACM Conference on Recommender Systems*, RecSys '12, pp. 35–42 (2012).
[24] Parra, D., Brusilovsky, P. and Trattner, C.: See what you want to see: Visual user-driven approach for hybrid recommendation, *Proceedings of the 19th International Conference on Intelligent User Interfaces*, IUI '14, pp. 235–240 (2014).
[25] Tsai, C.-H. and Brusilovsky, P.: Providing control and transparency in a social recommender system for academic conferences, *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*, UMAP '17, pp. 313–317 (2017).
[26] Hamilton, W. L., Ying, R. and Leskovec, J.: Inductive representation learning on large graphs, *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS '17, pp. 1025–1035 (2017).
[27] Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K. and Jegelka, S.: Representation learning on graphs with jumping knowledge networks, *Proceedings of the 35th International Conference on Machine Learning*, ICML '18, pp. 5453–5462 (2018).
[28] Kipf, T. N. and Welling, M.: Semi-supervised classification with graph convolutional networks, *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17, pp. 1–14 (2017).
[29] Maas, A. L., Hannun, A. Y. and Ng, A. Y.: Rectifier nonlinearities improve neural network acoustic models, *Proceedings of the 30th International Conference on Machine Learning*, ICML '13, pp. 1–6 (2013).
[30] Rendle, S., Freudenthaler, C., Gantner, Z. and Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback, *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, UAI '09, pp. 452–461 (2009).
[31] Kenmochi, H. and Ohshita, H.: VOCALOID - commercial singing synthesizer based on sample concatenation, *Proceedings of the 8th Annual Conference of the International Speech Communication Association*, INTERSPEECH 2007, pp. 4009–4010 (2007).
[32] Kingma, D. P. and Ba, J.: Adam: A method for stochastic optimization, *3rd International Conference on Learning Representations*, ICLR (2015).
[33] Goto, M.: A chorus section detection method for musical audio signals and its application to a music listening station, *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 14, No. 5, pp. 1783–1794 (2006).
[34] 天井里咲，徳山陽祐，佃　洸摂，濱崎雅弘，後藤真孝，土方嘉徳：推薦エンジンの切り替え機能が信頼形成に与える影響を明らかにするための心理学実験，第 4 回計算社会科学会大会，CSSJ '25, pp. 1–8 (2025).