Vol.2017-DBS-165 No.32
Vol.2017-IFAT-128 No.32
2017/9/20

Lyric Jumper：アーティストごとの歌詞トピックの傾向に基づく歌詞探索サービス

佃 洸摂[1,a)]　石田 啓介[1,b)]　後藤 真孝[1,c)]

概要：各アーティストには，「恋愛」や「友情」といったトピックの歌詞が書かれやすい，といった傾向が存在する．そのような傾向を考慮することで，歌詞のトピックに基づくアーティストの選択や，好みのアーティストと歌詞のトピックの傾向が類似した未知のアーティストの推薦といった新しいアプリケーションが実現できるようになる．本稿では，アーティストごとの歌詞トピックの傾向を推定するためのモデルを提案する．提案モデルでは，各アーティストがトピックの分布を持ち，その分布に応じて各歌詞にひとつのトピックが割り当てられる．歌詞データセットを用いた実験により，従来研究において歌詞トピックを推定する主要な手法である latent Dirichlet allocation（LDA）よりも提案モデルが優れていることを定量的に示した．さらに，提案モデルを 3,722 アーティストの 147,990 件の歌詞に適用することで，歌詞のトピックに基づく歌詞探索サービス「Lyric Jumper」を実装・公開した．Lyric Jumper ではアーティストごとのトピックの好みの可視化や，トピックの類似度に基づくアーティストの推薦などの機能が提供される．Lyric Jumper にアクセスした 12,353 ユーザの操作ログを分析することで，トピックと関連する歌詞のフレーズ推薦機能が特に有用であることを示した．

1. はじめに

あるアーティストは「恋愛」に関する歌詞をよく歌い，別のアーティストは「友情」に関する歌詞をよく歌う，といったように，各アーティストには固有の歌詞の傾向が存在する．人々が音楽を聴くときは，ジャンルやムード，メロディやリズムといった音響的な特徴量だけでなく，歌詞のトピックも考慮してアーティストを選択する [1], [2]．しかし，音楽情報検索の分野では，歌詞のトピックが持つ可能性を十分に活用できているとは言えない．たとえば，歌詞のトピックに応じてアーティストや楽曲を選択したり，ユーザの好みのアーティストと歌詞のトピックの傾向が類似する未知のアーティストを推薦したりすることは難しい．本研究の目的は，楽曲の歌詞のトピックを考慮した音楽情報検索を実現することである．

歌詞に基づく音楽情報検索のアプローチのひとつは，歌詞内の語を直接使用することである．これにより，ユーザが任意の単語をクエリとして入力したり [3], [4]，現在聴いている歌詞と同じフレーズを持つ別の歌詞を発見したり [5] することが可能になる．別のアプローチは，トピックモデルを用いて歌詞の潜在的な意味を使用することである．トピックモデルにおけるトピックは単語の確率分布によって表され，トピックの意味はその分布に応じて解釈される．音楽情報検索では，トピックモデルとして latent Dirichlet allocation（LDA）[6] が主に用いられる．LDA では各歌詞がトピックの確率分布を持ち，歌詞内の単語ごとにひとつのトピックを割り当てる．LDA はアーティストごとの楽曲集合を考慮しておらず，トピックに対するアーティストの好みを明示的に反映したモデルにはなっていない．

それに対して本稿では，アーティストのトピックに対する好みを考慮したトピックモデルを提案する．提案モデルでは，各アーティストがトピックの確率分布（アーティストのトピックに対する好みを表す）を持つ．また，歌詞を書き始める前にひとつ主題を決めることが一般的である [7], [8] ことから，各歌詞にひとつのトピックを割り当てる（以下「歌詞トピック」と呼ぶ）．つまり，ある歌詞にトピック k が割り当てられると，その歌詞内の全ての単語にもトピック k が割り当てられる．さらに，歌詞内の全ての単語がトピックと関連しているとは限らないため，トピックとは無関係な背景後の確率分布も扱う．

提案モデルを用いて，歌詞探索サービス *Lyric Jumper* [*1] を実装し公開した．Lyric Jumper は，歌詞のトピックを考慮することで，ユーザがより柔軟な方法で歌詞を探索し楽曲を楽しめることを目指している．Lyric Jumper では 20 個のトピックの推定結果に基づき，各歌詞にいずれかのトピックが割り当てられる．サービス上では，アーティストごとの歌詞トピックの傾向の可視化やトピックに基づくアーティストのランキング，歌詞トピックの傾向の類似度に基づくアーティスト推薦などの機能が提供される．

本研究の主な貢献を以下に述べる．

- アーティストの歌詞トピックに対する好みを考慮し，

[1] 国立研究開発法人 産業技術総合研究所
[a)] k.tsukuda@aist.go.jp
[b)] ksuke-ishida@aist.go.jp
[c)] m.goto@aist.go.jp

[*1] https://lyric-jumper.petitlyrics.com

各歌詞はひとつのトピックを持つという仮定に基づいて歌詞のモデル化を行った（3 章）．

- 歌詞の配信企業により提供された歌詞データを用いて，既存研究で主に使用されていた LDA よりも提案モデルが優れていることを、perplexity の観点から定量的に示した（4 章）．
- 提案モデルを用いて，歌詞トピックに基づく歌詞探索が可能な web サービス Lyric Jumper を実装し公開した．Lyric Jumper 上での 12,000 人以上のユーザの操作ログを用いて，Lyric Jumper がユーザの探索行動に与える影響を分析した（5 章）．

2. 関連研究

2.1 歌詞に基づく楽曲検索システム

Müller ら [4] は楽曲と歌詞のペアに対して，歌詞に対応する楽曲中の歌唱部分を割り当てる手法を提案し，歌詞の一部をクエリとして入力できる楽曲検索システムを提案した．ユーザが検索結果中の楽曲を選択すると，クエリの歌詞を含む箇所から音楽が再生される．ユーザが歌った歌詞と一致する楽曲の検索も多数取り組まれている [9], [10]．Fujihara ら [5] は歌詞中のフレーズに基づいて歌詞間にハイパーリンクを貼る「Music Web」という概念を提唱した．これにより，ユーザが楽曲を聴いている最中に，リンクされたフレーズをクリックすることで同じフレーズを持つ別の楽曲に遷移できる．楽曲集合を俯瞰するアプローチのひとつとして可視化の研究も行われてきた．SongWords [11] は，自己組織化マップを用いて歌詞とタグを二次元平面上に表示するテーブルトップコンピュータ向けのシステムである．Lyricon [12] は，ユーザが直感的に歌詞の内容を理解できるように，歌詞中の単語と適合するアイコンを歌詞に沿って表示するシステムである．

これらの研究では，歌詞中の単語を直接用いているが，我々は歌詞から推定されるトピックを用いるため，歌詞の潜在的な意味に基づいてユーザが歌詞を探索できるという優位性がある．

2.2 トピックに基づく歌詞分析およびシステム

トピックモデルは歌詞の潜在的な意味を考慮できるため，歌詞の分析 [13], [14], [15]，歌詞検索システム [16]，音楽プレーヤー [17] などの研究で使用されてきた．これらの研究では，トピックモデルとして LDA が使用されている．それに対して我々は，各アーティストがトピックの確率分布を持ち，各歌詞にひとつトピックが割り当てられるモデルを提案する．4 章で示すように，歌詞の生成過程を表すモデルとして，提案モデルは LDA を上回る精度を持つため，提案モデルを用いることで既存研究の歌詞分析やシステムを改善できる可能性がある．

本研究と最も関連のある研究は Kleedorfer ら [18] の研究

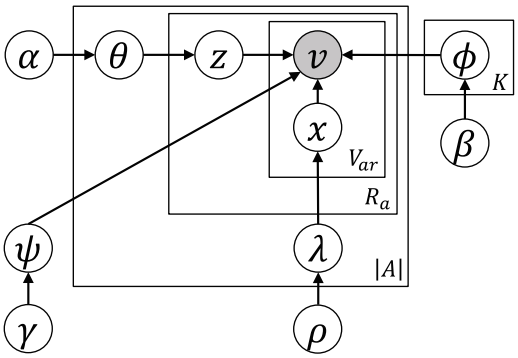

図 1 提案モデルのグラフィカルモデル．

である．彼らは歌詞集合に対して非負値行列分解を適用して歌詞をクラスタリングし，各クラスタをトピックのようにみなして人手で名前を付けている．本研究ではアーティストの歌詞トピックに対する好みを考慮している点が彼らの研究とは異なる．これにより，ユーザがアーティストとトピックの関係に基づいて好みの歌詞を発見することが可能になる．さらに，我々は新しいモデルを提案しただけでなく，誰もが歌詞の実データを探索できるように，提案モデルを基に web サービスも実装し公開している．

3. モデル

3.1 記号の定義

歌詞データセットが与えられたとき，A をデータセット内のアーティスト集合とする．R_a をアーティスト $a \in A$ の楽曲数とすると，a の楽曲集合は $\{S_{ar}\}_{r=1}^{R_a}$ と表される．S_{ar} は a の r 番目の楽曲を表す．また，V_{ar} を楽曲 S_{ar} の歌詞内の単語数とすると，$S_{ar} = \{v_{arj}\}_{j=1}^{V_{ar}}$ と表される．v_{arj} は S_{ar} の j 番目の単語である．したがって，すべての歌詞内のすべての単語は $D = \{\{\{v_{arj}\}_{j=1}^{V_{ar}}\}_{r=1}^{R_a}\}_{a \in A}$ となる．

3.2 提案モデル

既存研究の中で，歌詞に LDA を適用する有用性は報告されてきたが [13], [14], [15], [16], [17]，LDA の歌詞生成過程ではアーティストの情報が考慮されていない．本研究では，あるアーティストは「恋愛」というトピックに関する歌詞の楽曲が多く，別のアーティストは「人生」というトピックに関する歌詞の楽曲が多いといったように，各アーティストには歌詞のトピックに対する固有の好みが存在すると仮定する．

この仮定に基づき，アーティストのトピックに対する好みを考慮した歌詞生成モデルを提案する．図 1 に提案モデルのグラフィカルモデルを示す．図中の色付きの円は観測変数，白地の円は未知変数を表し，K はトピック数である．提案モデルでは，各アーティストはトピックの確率分布 θ を持つ．歌詞を書き始める前に歌詞の主題（すなわちトピック）を決めることが一般的であることから [7], [8]，各歌詞は θ から生成されたひとつのトピック z を持つと仮定する．しかし，歌詞中の全ての単語がトピックと関連

Vol.2017-DBS-165 No.32
Vol.2017-IFAT-128 No.32
2017/9/20

しているとは限らない．たとえば，「これ」や「事」などの単語は多くの歌詞に出現するが，固有のトピックとの関連度が高いとは言えない．そこで，提案モデルでは背景語の存在を考慮する．図 1 中で ψ が背景語の確率分布を表し，いずれのトピックとも関連度の低い単語が高い生起確率を持つ．各アーティストはベルヌーイ分布 λ を持ち，この分布に応じてトピックと背景語のどちらからどの程度単語が生成されるかが決まる．つまり，アーティスト a が歌詞中の単語を生成するとき，確率 λ_{a0} $(x=0)$ でトピックの単語確率分布 ϕ から単語が生成され，確率 λ_{a1} $(x=1)$ で背景語の単語確率分布 ψ から単語が生成される．ここで，$\lambda_{a0}+\lambda_{a1}=1$ である．提案モデルの歌詞生成過程を Algorithm 1 に示す．

3.3 パラメータ推定

崩壊型ギブスサンプリング [19] により提案モデルのパラメータを推定する．パラメータ θ，ϕ，ψ の事前分布としてディリクレ分布を，λ の事前分布としてベータ分布を用いる．これらの事前分布は各パラメータの共役事前分布であるため，パラメータに関して周辺化を行える．$\mathbf{\Theta}=\{\theta_a\}_{a\in A}$，$\mathbf{\Phi}=\{\phi_k\}_{k=1}^{K}$，$\mathbf{\Lambda}=\{\lambda_a\}_{a\in A}$ とすると，全ての歌詞中の全単語データ D，潜在変数 $Z=\{\{z_{ar}\}_{r=1}^{R_a}\}_{a\in A}$ および $X=\{\{\{x_{arj}\}_{j=1}^{V_{ar}}\}_{r=1}^{R_a}\}_{a\in A}$ の周辺化された同時分布は次式により解析的に計算できる．

$$
\begin{aligned}
&P(D,Z,X|\alpha,\beta,\gamma,\rho)\\
&=\iiiint P(D,Z,X|\mathbf{\Theta},\mathbf{\Phi},\psi,\mathbf{\Lambda})P(\mathbf{\Theta}|\alpha)\\
&\qquad\times P(\mathbf{\Phi}|\beta)P(\psi|\gamma)P(\mathbf{\Lambda}|\rho)d\mathbf{\Theta}d\mathbf{\Phi}d\psi d\mathbf{\Lambda}\\
&\propto\prod_{a\in A}\frac{\Gamma(\rho+N_{a0})\Gamma(\rho+N_{a1})}{\Gamma(2\rho+N_a)}\frac{\prod_{v\in V}\Gamma(N_{1v}+\gamma)}{\Gamma(N_1+\gamma|V|)}\\
&\times\prod_{k=1}^{K}\frac{\prod_{v\in V}\Gamma(N_{kv}+\beta)}{\Gamma(N_k+\beta|V|)}\prod_{a\in A}\frac{\prod_{k=1}^{K}\Gamma(R_{ak}+\alpha)}{\Gamma(R_a+\alpha K)}. \quad (1)
\end{aligned}
$$

N_{a0}，N_{a1} はそれぞれ，アーティスト a の全ての歌詞の中で $x=0$，$x=1$ が割り当てられた単語数を表す（$N_a=N_{a0}+N_{a1}$）．N_{1v} は単語 v に $x=1$ が割り当てられた回数である（$N_1=\sum_{v\in V}N_{1v}$）．V は D 内の語彙集合を表す．N_{kv} は単語 v が $x=0$ の下でトピック k から生成された回数である（$N_k=\sum_{v\in V}N_{kv}$）．R_{ak} はアーティスト a の歌詞の中でトピック k が割り当てられた歌詞の数を表す（$R_a=\sum_{k=1}^{K}R_{ak}$）．式（1）に基づき，z_{ar} および x_{arj} をサンプリングする．十分な回数のイテレーションの後，サンプリングされた値を用いてパラメータの値を求める．紙面の都合上，詳細は割愛する．

4. 評価実験

本章では，次の疑問に答えるために定量的な評価を行った：アーティストの歌詞に対する好みを考慮することは歌詞のモデル化を行う際に有用か．

Algorithm 1 提案モデルの歌詞生成過程．

```
for each topic k ∈ {1,···,K} do
  Draw φ_k ~ Dirichlet(β)
end for
Draw ψ ~ Dirichlet(γ)
for each artist a in A do
  Draw θ_a ~ Dirichlet(α)
  Draw λ_a ~ Beta(ρ).
  for each song S_ar do
    Draw a topic z_ar ~ Multinomial(θ_a)
    for each word v_arj in S_ar do
      Draw switch x ~ Bernoulli(λ_a)
      if x = 0 then
        Draw a word v_arj ~ Multinomial(φ_{z_ar})
      else if x = 1 then
        Draw a word v_arj ~ Multinomial(ψ)
      end if
    end for
  end for
end for
```

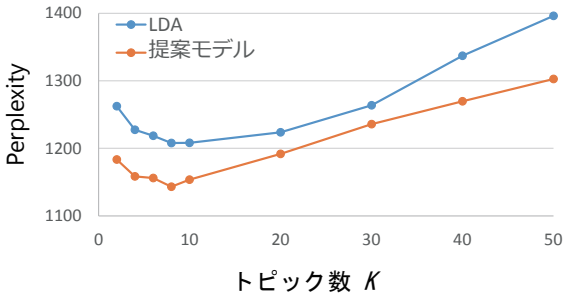

図 2　LDA と提案モデルの perplexity の比較結果．

[データセット] 本実験では，商用的に歌詞の配信を行っている企業から提供された歌詞データを使用した．データセットには，歌詞ごとの曲名とアーティスト名も含まれる．評価には，2016 年 12 月末時点における楽曲数の上位 1,000 アーティストの歌詞，計 93,716 件を使用した．形態素解析エンジン MeCab [20] を用いて，各歌詞から日本語の名詞を抽出した．ただし，出現する歌詞の数が 10 未満の名詞は除いた．提案モデルは言語に非依存であるが，5 章で述べる Lyric Jumper のユーザにとって，抽出されたトピックの理解しやすさを考慮して，日本語のみを使用した．各歌詞から 80%の名詞を学習データとしてランダムに選択し，残りの 20%の名詞をテストデータに使用した．

[実験設定] LDA および提案モデルでは，既存研究に倣い，ハイパーパラメータの値を $\alpha=\frac{1}{K}$，$\beta=\frac{50}{|V|}$ とした．これに加えて，提案モデルでは $\gamma=\frac{50}{|V|}$，$\rho=0.5$ とした．LDA と提案モデルの精度を比較するため，トピックモデルの精度を評価する際の指標として広く使用されている perplexity を使用した [6]．モデルの精度が高いほど perplexity の値は小さくなる．トピック数は $K=2$，4，6，8，10，20，30，40，50 とし，各値に対する perplexity を比較した．

[結果] 結果を図 2 に示す．トピック数に関わらず，提案

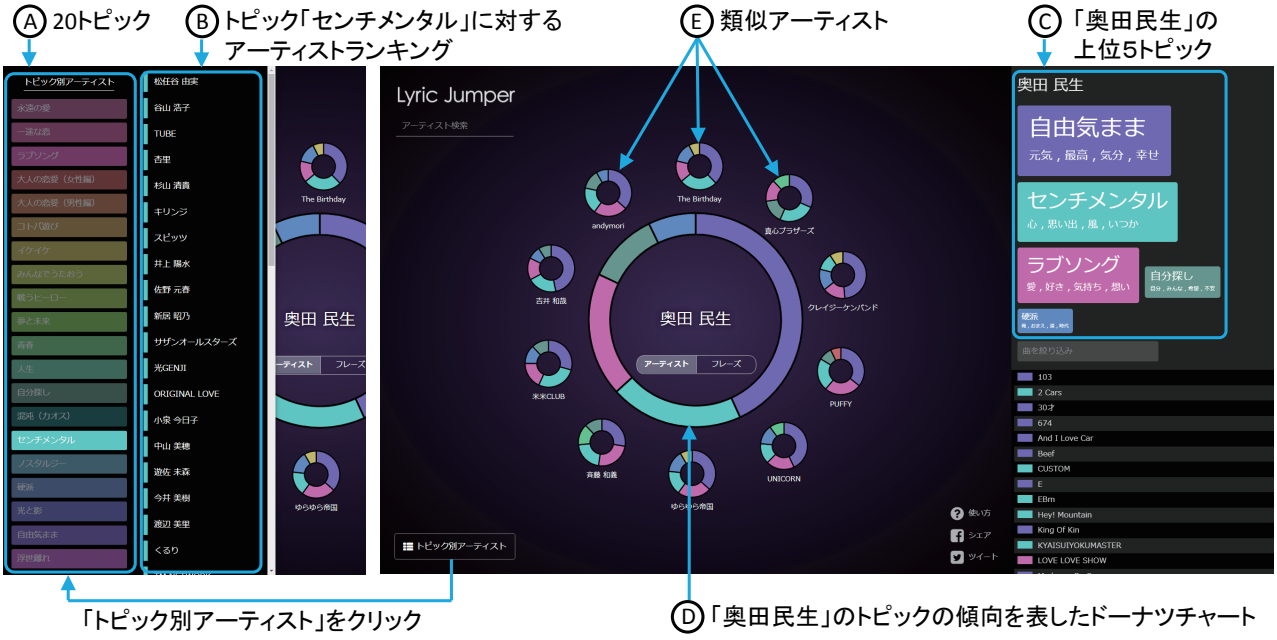

図 3　Lyric Jumper の概観.

モデルは LDA を上回った．いずれのモデルでも，トピック数が 8 のときに perplexity の値は最小となった．トピック数が大きくなるほど，2 モデル間の perplexity の差も大きくなる傾向が見られた．これらの結果から，提案モデルは LDA よりも優れたモデルであり，アーティストのトピックに対する好みを考慮することは歌詞のモデル化を行う際に有用であるといえる．

5. Lyric Jumper

提案モデルを用いて，誰もが web ブラウザ上で利用可能な歌詞探索サービス「Lyric Jumper」を実装し公開した．本章では Lyric Jumper の実装および機能について述べたあと，web サービス上のユーザのログ分析結果を述べる．

5.1 実装

Lyric Jumper を実装するにあたり，歌詞データは 4 章で述べた企業から提供されたものを使用した．2016 年 12 月末時点で利用可能な全ての歌詞を使用し，各歌詞から日本語の名詞を抽出した．トピックの質を保障するため，登録されている楽曲が 10 曲未満のアーティストおよび，10 件未満の歌詞にしか出現しない名詞は除いた．最終的に，3,722 アーティストの 147,990 曲の歌詞データを使用した．

トピック数に関しては，トピック数が少なすぎると，ユーザは Lyric Jumper の使用にすぐに飽きると考えられ，多すぎると，類似したトピックが多数生成され，ユーザがトピック間の差異を把握するのが難しくなると考えられる．このような理由から，様々なトピック数の結果を比較したうえで，Lyric Jumper のトピック数は 20 とした．提案モデルで 20 個のトピックを求めた後，ユーザが各トピックの特徴を容易に理解できるよう，人手で各トピックの名前をラベル付けした．トピック名には「センチメンタル」「青春」「自分探し」などが含まれる．5 つのトピックは「恋愛」に関するものであったが，Lyric Jumper ではそれらを「永遠の愛」「一途な愛」「ラブソング」「大人の恋愛（女性編）」「大人の恋愛（男性編）」と表現している．

5.2 機能

5.2.1 アーティストランキング機能

Lyric Jumper では図 3 Ⓐのように 20 個のトピック名を表示する．トピックを選択することで，そのトピックと関連の強いアーティストが最大 100 件表示される（図 3 Ⓑ）．これによりユーザは，興味のあるトピックと関連のある多くのアーティストを見ることができ，また意外なアーティストの選択トピックとの関連性も知ることができる．

トピック k が選択されたとき，トピック k とより強く関連したアーティストほど上位に表示されるように，アーティスト a のトピック分布 θ_a におけるトピック k の順位を第一キー（小さいほど良い），トピック k に割り当てられた楽曲数 R_{ak} を第二キー（大きいほど良い）として全アーティストをランキングし，上位 100 アーティストをユーザに表示する．5.2.2 項で述べるように，Lyric Jumper では各アーティストの上位 5 トピックを表示するため，θ_a におけるトピック k の順位が 5 位より下のアーティストはランキングに含まない．

5.2.2 トピック傾向表示機能

ユーザがアーティスト a を選択すると，a のトピックの傾向を表示する．具体的には，θ_a において生起確率の上位 5 トピックの名前が表示される（図 3 Ⓒ）．トピックの生起確率が高いほどトピック名は大きく表示される．これにより，ユーザは選択したアーティストのトピックの傾向を

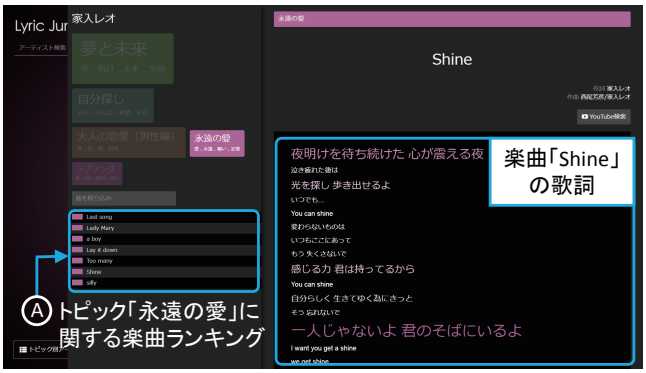

図 4　歌詞フレーズの強調表示.

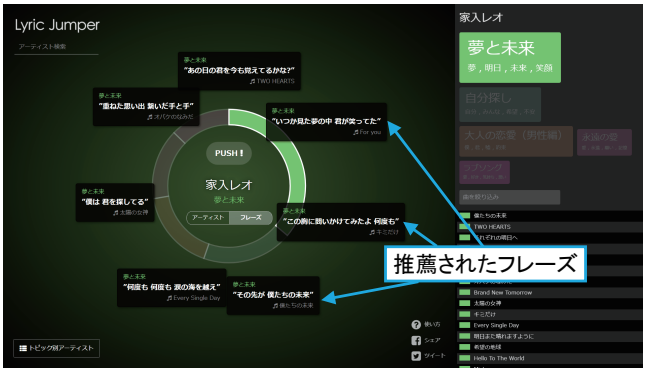

図 5　フレーズ推薦.

把握できるだけでなく，意外なトピックも関連していることを把握できる．ユーザがトピックの意味をより容易に理解できるように，各トピックの単語生起確率分布 ϕ からトピックに特徴的な 4 つの単語を人手で選択し，トピック名の下に表示している．さらに Lyric Jumper では，円周が上位 5 トピックの割合に応じて分割されたドーナツチャートも用いてトピックの傾向を表示している（図 3 Ⓓ）．

5.2.3　類似アーティスト推薦機能

音楽情報検索において，類似アーティストは重要な情報のひとつであることから [2]，Lyric Jumper ではユーザが選択したアーティストとのトピック類似度に基づいて 10 アーティストを推薦する（図 3 Ⓔ）．10 アーティストは，ポピュラーな 8 アーティストとマイナーな 2 アーティストから構成される．ポピュラーなアーティストだけでなくマイナーなアーティストも表示することで，ユーザにとって馴染みはないが，興味のあるアーティストと関連のあるアーティストの曲を聴いてもらうことを狙っている．推薦されたアーティストのグラフを選択することで，そのアーティストのドーナツチャートが中心となった画面に遷移する．

ユーザが選択したアーティスト a が与えられると，a と $a' \in A \setminus \{a\}$ の類似度を θ_a と $\theta_{a'}$ の JS ダイバージェンスの逆数により求める．アーティスト a と各アーティストの類似度を求めた後，データセット内の楽曲数が 100 曲以上のアーティストの中から類似度の上位 8 組をポピュラーなアーティスト，100 曲未満の中から類似度の上位 2 組をマイナーなアーティストとして選択し，計 10 アーティストをユーザに推薦する．

5.2.4　歌詞フレーズの強調表示機能

ユーザがアーティストのトピックを選択すると，アーティストの楽曲の中でそのトピックに割り当てられた楽曲のリストが表示される（楽曲のランキング方法については 5.2.5 項で述べる）．楽曲のタイトルを選択すると，その楽曲の歌詞が表示される（図 4）．歌詞の中でトピックと関連の強い行 [*2] ほど，大きなフォントかつ濃い色で表示される．これによりユーザは，「歌詞のサビ部分が特にトピックと関連が強い」のように，歌詞の特徴を容易に理解できるようになる．「YouTube 検索」ボタンを押すと，YouTube [*3] でアーティスト名と楽曲名を AND 検索した検索結果が表示され，Lyric Jumper 上で動画を視聴できる．

歌詞の各行とトピック k の関連度を求めるために，まず ϕ_k を用いて，各名詞のスコアを計算する．$rank(k, v)$ を ϕ_k 内での名詞 v の生起確率の高さの順位とする．$rank(k, v) \leq 100$ であれば $w\_rel(k, v) = 101 - rank(k, v)$，それ以外であれば $w\_rel(k, v) = 0$ とする．行 l は $n(\geq 0)$ 個の名詞を含み，$l = (v_1, \cdots, v_n)$ と表せ，行 l とトピック k の関連度を $l\_rel(k, l) = \sum_{i=1}^{n} w\_rel(k, v_i)$ により求める．歌詞内の全ての行のスコアを計算したら，各行のスコアが $[0, 1]$ に収まるよう min-max 正規化を行う．文字サイズと文字の色はスコアが 0 の行の 16pt・#FFFFFF からスコアが 1 の行の 36pt・トピックの色まで線形に変化させる．

5.2.5　アーティストの楽曲ランキング機能

5.2.4 項で述べたように，ユーザがアーティストのトピックを選択すると，そのトピックに割り当てられた楽曲のランキングが表示される（図 4 Ⓐ）．トピックと関連の強い曲にユーザが容易にアクセスできるよう，トピックとの関連度が高い順に楽曲が順位付けされる．

楽曲 s とトピック k の関連度は $s\_rel(k, s) = \frac{1}{|L_s|} \sum_{l \in L_s} l\_rel(k, l)$ により求める．L_s は楽曲 s の歌詞内の全ての行のリストである．つまり，s と k の関連度は s の歌詞の各行と k の関連度の平均値により求められる．

5.2.6　フレーズ推薦機能

アーティストのトピックを選択後，「フレーズ」ボタンをクリックすると，アーティストの楽曲の歌詞の中でそのトピックと関連のあるフレーズが推薦される（図 5）．さらに，「PUSH!」ボタンを押すたびに新しいフレーズが推薦される．これによりユーザは，同じトピックに関する様々な表現に触れることができる．フレーズを選択すると，そのフレーズを含む歌詞が 5.2.4 項と同様に表示される．

アーティスト a とトピック k が与えられると，以下の方法で推薦フレーズを選択する．i 番目のループ（$i = 1, 2, \cdots$）

[*2] 本稿では「フレーズ」と「行」を同じ意味で用いる．

[*3] https://www.youtube.com/

表 1　各機能の使用回数.

機能	PC	スマートフォン
アーティストランキング機能	2,092	30,295
類似アーティスト推薦機能	1,706	4,016
アーティスト楽曲ランキング機能	5,399	14,665
フレーズ推薦機能	4,997	253,430

では，アーティスト a の楽曲で $s\_rel(k,s)$ の値が高い順に，各楽曲の歌詞の i 番目に $l\_rel(k,l)$ の値が高い行を抽出する．抽出された行数が 100 件になるまで，このループを繰り返す．ユーザが Lyric Jumper を使用するたびに異なるフレーズに触れられるようにするため，抽出した 100 件のフレーズをランダムな順番で推薦する．

5.3　ログ解析

我々は Lyric Jumper を無料で使用可能な web サービスとして 2017 年 2 月 21 日に公開した．Lyric Jumper 上でのユーザの行動を分析するため，2 月 21 日から 3 月 22 日の 30 日間に渡る全ユーザの操作ログを収集した．PC とスマートフォンのユニークユーザ数はそれぞれ 1,288 と 11,065 であった．各機能の使用頻度を表 1 に示す．まず，アーティストランキング機能とアーティスト楽曲ランキング機能の使用頻度が高いことがわかる．この結果は，トピックに基づくアーティストや楽曲の探索がユーザの興味を喚起した可能性を示している．また，特にスマートフォンユーザの間では，フレーズ推薦機能が頻繁に使用されており，「PUSH!」ボタンは 253,430 回押された．このことから，トピックに関するフレーズを基に歌詞を探したいというニーズの高さがうかがえる．上記の機能に比べ，類似アーティスト推薦機能はそれほど頻繁には使用されなかった．類似アーティストに基づく歌詞探索を促進するためには，より洗練されたインタフェースの提案が必要であると考えている．

6.　まとめ

本稿ではアーティストの歌詞トピックに対する好みを考慮したモデルを提案した．実験により，トピックの数に関わらず，提案モデルが従来モデルの LDA を上回る精度であることを perplexity を用いて定量的に示した．また，3,722 アーティストの 147,990 件の歌詞に提案モデルを適用することで，歌詞探索サービス Lyric Jumper を実装し公開した．ログ分析の結果，選択されたトピックと関連の強い歌詞のフレーズを推薦する機能が特に高い頻度で使用されていることを明らかにした．

提案モデルは言語に非依存であるため，提案モデルを英語歌詞に適用し，洋楽版の Lyric Jumper を公開する予定である．また，提案モデルによって得られたトピックを，音響特徴量やタグなどと組み合わせることで，より柔軟にユーザの検索意図を反映した楽曲の探索システムの提案も行っていきたい．

謝辞 歌詞データを提供していただいた株式会社シンクパワーに感謝する．本研究の一部は JST ACCEL (JPMJAC1602) の支援を受けた．

参考文献

[1] D. Bainbridge *et al.*: “How people describe their music information needs: A grounded theory analysis of music queries”, ISMIR, pp. 221–222 (2003).
[2] J. H. Lee and J. S. Downie: “Survey of music information needs, uses, and seeking behaviours: Preliminary findings”, ISMIR, pp. 989–992 (2004).
[3] E. Brochu and N. de Freitas: “”Name that song!” a probabilistic approach to querying on music and text”, NIPS, pp. 1505–1512 (2002).
[4] M. Müller *et al.*: “Lyrics-based audio retrieval and multimodal navigation in music collections”, ECDL, pp. 112–123 (2007).
[5] H. Fujihara *et al.*: “Hyperlinking lyrics: A method for creating hyperlinks between phrases in song lyrics”, ISMIR, pp. 281–286 (2008).
[6] E. M. Airoldi *et al.*: “Mixed membership stochastic blockmodels”, Journal of Machine Learning Research, **9**, pp. 1981–2014 (2008).
[7] M. Baxter: “Voices of resistance, voices of transcendence: Musicians as models of the poetic - political imagination”, International Journal of Education & the Arts, **11**, pp. 1–24 (2010).
[8] J. M. Toivanen *et al.*: “Automatical composition of lyrical songs”, ICCC, pp. 87–91 (2013).
[9] T. Hosoya *et al.*: “Lyrics recognition from a singing voice based on finite state automaton for music information retrieval”, ISMIR, pp. 532–535 (2005).
[10] C. Wang *et al.*: “An improved query by singing/humming system using melody and lyrics information”, ISMIR, pp. 45–50 (2010).
[11] D. Baur *et al.*: “SongWords: Exploring music collections through lyrics”, ISMIR, pp. 531–536 (2010).
[12] W. Machida and T. Itoh: “Lyricon: A visual music selection interface featuring multiple icons”, IV, pp. 145–150 (2011).
[13] C. Johnson-Roberson and M. Johnson-Roberson: “Temporal and regional variation in rap lyrics”, NIPSW (2013).
[14] J. Ren *et al.*: “What makes a music track popular in online social networks?”, WWW, pp. 95–96 (2016).
[15] G. Sharma and M. N. Murty: “Mining sentiments from songs using latent Dirichlet allocation”, IDA, pp. 328–339 (2011).
[16] S. Sasaki *et al.*: “LyricsRadar: A lyrics retrieval system based on latent topics of lyrics”, ISMIR, pp. 585–590 (2014).
[17] T. Nakano and M. Goto: “LyricListPlayer: A consecutive-query-by-playback interface for retrieving similar word sequences from different song lyrics”, SMC, pp. 344–349 (2016).
[18] F. Kleedorfer *et al.*: “Oh oh oh whoah! towards automatic topic detection in song lyrics”, ISMIR, pp. 287–292 (2008).
[19] T. L. Griffiths and M. Steyvers: “Finding scientific topics”, Proceedings of the National Academy of Sciences, **101**, Suppl. 1, pp. 5228–5235 (2004).
[20] T. Kudo *et al.*: “Applying conditional random fields to Japanese morphological analysis”, EMNLP, pp. 230–237 (2004).