楽器で演奏された単旋律フレーズをクエリとした楽曲検索システム

retrieval system using monophonic phrase queries played on musical instruments

松下陸
Riku Matsushita
法政大学情報科学部ディジタルメディア学科
Email: riku.matsushita.7a@stu.hosei.ac.jp

Abstract

Currently, there is a large amount of digital music available on the Internet. In the field of music retrieval, an important issue is to accurately and quickly retrieve the music consumer's target music from such a large music database. In this paper, We propose a new music retrieval system that can retrieve music from a music database using monophonic melody phrases played on musical instruments. In this system, melody phrases were represented as strings and used as training data for fastText. After learning, search for phrases with high cosine similarity to the input phrase using distributed representation. The experimental results showed that the proposed method achieved comparable values of MRR and Top-10 hit rate to those of the conventional method. However, the proposed method was constructed using a much larger database of approximately 42,000 songs, compared to the conventional studies that utilized a database of only about 4,400 songs. This suggests that the proposed method is superior as a search system, given its larger scale and broader coverage of the music database.

1 まえがき

楽曲検索システムにおいて膨大なデジタル楽曲データベースの中からユーザーが求める楽曲を適切に検索することが重要な課題である。 既存の楽曲検索方法としては楽曲名やアーティスト名などメタデータによる検索が主流である。しかし現在ではその他にも・周囲で流れる音楽・ハミング (鼻歌)・口笛・歌う等のような入力による検索方法が存在する。これらの検索システムは楽曲メタデータがわからなかったり、思い出せなかったりした場合に歌うだけで簡単に直感的な操作で楽曲検索ができることから非常に有用である。このような状況は楽器演奏者にもあるが、歌やハミングをクエリとしたシステムでは楽器音声が想定されていないため検索が望んだ結果を出さない。

本研究では楽器の演奏フレーズをクエリとして、そのクエリに類似したフレーズを検索し得られた類似フレーズを用いて楽曲検索を行うシステムを提案する。また、楽器で楽曲検索が可能であることを示すため和声楽器ではなく基本的に単音でメロディを演奏するエレキベースに着目した。

2 関連研究

2.1 ハミング検索

ハミング検索システムと呼ばれるユーザーの鼻歌をクエリとしてメロディーを検索することができるコンテンツベースの音楽検索手法がある。従来 [1] では音楽信号から FTANet と呼ばれるニューラルネットワークを用いて主旋律の抽出を行いクエリに変換する、そして 3 つの n-gram アルゴリズムと Mode Normalized Frequency を組み合わせた UnifiedAlgorithm により入力フレーズと楽曲データベースとのクエリマッチングを行い類似度が高い楽曲を検索結果としていた。システムの評価実験では 4,400 曲ほどのデータベースを用いて行われ、ベースラインシステムよりも評価指標である MRR,top-n hit rate が上回る結果を得たが、UnifiedAlgorithm を使用した事により検索にかかる処理時間が大きく増加してしまった。そのため、より大規模なデータベースへの応用は困難である。

それに対し本研究では、検索の精度向上に主眼を置き、メロディ抽出の精度が検索結果に影響を与えないよう、約 4.2 万曲からなる大規模な MIDI データセットを使用した。MIDI は各トラックが独立しているため特定のトラックを抽出するのに適しているからである。そしてクエリマッチングにおいてコサイン類似度を用いることで検索にかかる時間を大幅に短縮しつつ、fastText[2] と呼ばれる単語の内部構造を考慮し、高速にテキスト分類や単語埋め込みを生成するニューラルネットワークを利用することで検索の高精度かつ計算の高速化を図る。本研究により、高度な楽曲検索が可能となり、音楽情報処理分野における実用的な応用が期待される。

2.2 melody2vec

melody2vec[3] は、音楽のメロディー情報を分散表現として表現する手法の一つで、word2vec[4] のアルゴリズムをベースにしています。melody2vec では、音符の系列をテキストの単語列に置き換え、その系列から分散表現を学習することで、音楽的な意味や関係性を捉えることができます。楽曲データを前処理し、適切な単位に分割して、それぞれを表すベクトルを生成することで、楽曲の単語ベクトルを作成します。これらの単語ベクトルを用いて、楽曲の特徴を表現することができます。しかし word2vec は学習データセット中に存在しない単語である未知語に対しては対応できません。つまり、学習データセット中に存在しない単語を入力すると、その単語の分散表現を得ることができず、類似したフレーズの検索ができなくなる。

一般的にハミング検索においてもユーザーは誤った入力をする可能性が高いため誤ったフレーズ入力に対応しなければならない。そこで本研究では誤った入力に対応するため FastText を使用した。fastText は、各単語を n-gram に分割して表現し、その n-gram を学習することで単語の分散表現を獲得する。この手法により、未知語でもその n-gram の組み合わせが辞書内に存在する単語と似ている場合、適切な分散表現を生成することができる。つまり、辞書に存在しない単語でも、その構成要素である n-gram を考慮することで適切な分散表現を生成し、類似性を計算することができるため、未知語に対応できる。本研究では 4 つの音符を 1 つのグループとしてテキスト表現化し、これを fastText の単語として学習を行った。グループ化は、4 つの音符をまとめて 1 つの単位とし、1 つずつずらしながら実施した。これにより楽曲内のフレーズの漏れを防ぎつつ、

Supervisor: Prof. Katunobu Itou

楽曲の構造的な特徴を捉えることができるという利点がある。

2.3 fastText

単語の分散表現を学習するモデルとして word2vec の skip-gram モデルをベースとする fastText がある。skip-gram はとある文中の単語からその周辺の単語を予測する教師あり学習モデルである。fastText の skip-gram は文章中のある単語とその単語の n-gram とすることで単語の内部構造を意識し学習ができる。例えば、”where”という単語は 3-gram の場合、”¡wh”,”whe”,”her”,”ere”,”er¿”の 5 つに分けられる。fastText ではこの 5 つの subword 情報と”where”のベクトルを足し合わせることで分散表現を生成する。ここで同じ疑問詞である”when”という単語について考えると subword が 2 つ共通しており”where”と”when”が似ていることを数式的に表現できる。さらに、subword 情報を用いた学習により Word2vec ではできなかった未知語に対しても分散表現を得ることができる。またこのモデルは意味類推や文法理解についても従来の手法より精度が改善されている。単語分散表現の次元数は、従来研究で最も結果が良かった 100 次元を採用した。

3 提案手法

本研究では、fastText を用いてメロディフレーズの分散表現を事前に学習し、データベース中の楽曲を表す楽曲ベクトル、クエリを表す入力フレーズベクトルの生成に用いた。また、学習のためにメロディのテキスト化とセグメンテーションを行った。セグメンテーションにより切り分けられたフレーズが fastText における単語にあたる。楽曲ベクトルは単語分散表現を用いて SWEM と呼ばれる手法により計算し保存する。SWEM は単語の分散表現から文章レベルのベクトルを生成が d 着る手法である。fastText の事前学習にはインターネット上で最大規模の MIDI データセットである Lakh MIDI dataset[5] を使用した。オンライン時には、同様の手法を用いて入力フレーズからベクトルを生成し、事前に計算され保存されている各楽曲ベクトルとの間でコサイン類似度に基づくクエリマッチングを行う。コサイン類似度は高速に計算が可能であることから、従来の研究に比べて検索にかかる時間を大幅に削減することが期待される。また、fastText は学習の過程で単語を subword 情報を用いることで、従来のフレーズ検索で用いられていた word2vec での問題であったユーザーによる未知フレーズ入力にも対応し、音符の音高や音価の内部情報も考慮可能である。

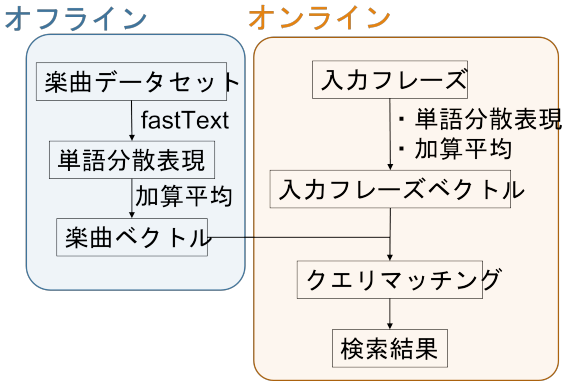

図 1. Overview of this system

3.1 データの準備

学習に用いた Lakh MIDI dataset は、176,581 曲分の MIDI データによって構成されておりメロディ分析を行うことができる最大の楽曲データセットである。MIDI ファイルは各楽器の演奏情報がトラックに分かれている。したがって音源分離のような前処理を必要とせず、ベーストラックを発見できればベースパート抽出が容易に可能であるので大規模な学習データが準備できる。しかし、Lakh MIDI dataset は web 上からクローリングされた MIDI ファイルで構成されているため MIDI ファイル中のどのトラックがベースパートに該当しているのか特定するための規則はない。そこでベーストラック特定のために各トラックの演奏楽器を指定するためのプログラムナンバーを用いた。ベーストラックのプログラムナンバーは”Bass”という単語を含む場合が多い。

本研究ではすべての楽曲のトラック中のプログラムナンバーを小文字化し、その中で”bass”という単語を含む楽曲からベーストラックを抽出した。その結果、全 176,581 曲分のデータの中から 42,829 曲分のベーストラックを抽出することができた。このようにして得られた MIDI データに対し音符情報の書き換えとセグメンテーションによる前処理を行った。

3.2 音符の挿入と書き換え

Lakh MIDI dataset から抽出したベーストラックにおいても不明瞭なメロディをもとの楽譜に近づけるために音長の調整と休符の挿入を行った。

まず MIDI の note はその音が鳴る開始時刻と終了時刻、音高 (pitch) と強さ (velocity) の 4 つの情報を保持している。MIDI には休符の note が存在しないため、本研究では休符を表す note は pitch を 999、velocity を 0 として挿入を行った。休符は 8 分音符以上の無音区間を休符とみなした。8 分音符を挿入することで、不自然な長さの休符が見られなくなった。

また、音価は開始時刻から終了時刻までの時間から単純には求めることができない。それは終了時刻後も余韻として音が鳴っているからである。私の研究では例として開始時刻から終了時刻までが少し 4 分音符分の長さ以上となったら 2 分音符、8 分音符以上なら 4 分音符のように整えた。

3.3 セグメンテーション

melody2vec を学習させるため word2vec における 1 単語を連続する 4 つの音符情報を 1 つのグループとした。

例) 「E1:1/4-E1:1/4-E:1/4-R:1/4」

この例では、E1(ミ) の音高の 4 分音符を 3 回鳴らした後に四分休符がくるフレーズを表現している。セグメンテーションを抽出されたベーストラック 42,829 曲に対して行った結果、5,180,014 個のフレーズが得られた。これを melody2vec の学習に用いることで各フレーズの分散表現を獲得する。

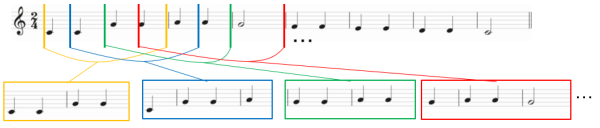

図 2. Segmentation Example

3.4 SWEM

セグメンテーショにより分けられた単語 (メロディフレーズ) は単語分散表現を用いることでベクトルを得られるが、楽曲自体のベクトルを表すことができていない。そこで、SWEM と呼ばれる手法を使用し、文章ベクトルを生成した。ここでの文章ベクトルは本研究における楽曲を表すベクトルである楽曲ベクトルにあたる。SWEM には 4 種類の方法が存在し、一つ目は SWEM-aver と呼ばれる、文章中の単語のベクトルの加算平均で生成する手法。2 つ目は SWEM-max と呼ばれる、各次元の最大値をとりベクトル生成する手法。3 つ目は SWEM-aver と SWEM-max を結合した SWEM-concat と呼ばれる手法。最後に、n-gram のように固定長のウィンドウで average-pooling した結果に対して max pooling する SWEM-hier である。本研究では最も単純な SWEM-aver を使用し楽曲を表すベクトルを生成した。

3.5 フレーズ類似度

SWEM によるベクトル生成後、入力フレーズと各楽曲ベクトルとのコサイン類似度を計算しマッチングを行った。コサイン類似度は、ベクトルの内積とノルムの計算に基づいているため、計算量が比較的少なく、高速な演算が可能である。このため、大規模なデータセットに対しても効率的に類似度を算出することができる。

コサイン類似度は、2 つのベクトルがどれほど似ているかを表す尺度であり、値は 0 から 1 の範囲をとり、1 に近いほど 2 つのベクトルが似ていることになる。具体的には、2 つのベクトル A と B のコサイン類似度は以下の式で定義される。

$$\cos(\mathbf{A},\mathbf{B}) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}$$

ここで、$\mathbf{A}\cdot\mathbf{B}$ はベクトル $\mathbf{A}$ と $\mathbf{B}$ の内積を表し、$|\mathbf{A}|$ と $|\mathbf{B}|$ はそれぞれ $\mathbf{A}$ と $\mathbf{B}$ のノルムを表す。

また、単語の分散表現を用いフレーズ検索した例を以下に示す。例として図 3 の類似フレーズを検索する。

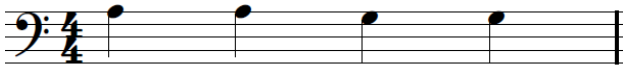

図 3. Example of input phrase

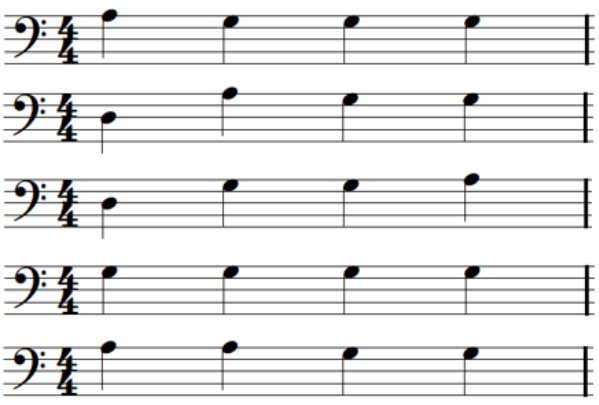

図 4. Similarity #1

図 3 の類似フレーズとして検索結果上位 5 つのフレーズは図 4 という結果になった。上から類似度 1 位から 5 位を示している。

これらから類似フレーズとして上位に表れているのは入力フレーズの一つの音符の音の高さが違うだけということが多く実際に類似していることが確認された。

3.6 楽曲のマッチング

本研究では 100 次元の実数値ベクトルで表現された入力フレーズと各楽曲とのコサイン類似度が大きい楽曲を上位にランキングすることで検索結果を決定した。これはコサイン類似度が大きいほど、楽曲中に入力フレーズに近いメロディを含んでいると考えられるからである。

データベース中の楽曲のベクトルは事前に求め保存し、入力フレーズのベクトルはリアルタイムで計算を行う。そして類似度を計算した際、入力フレーズを含む楽曲がランキング上位に現れることが良いマッチングであるといえる。

4 実験

4.1 実験 1

実験には、ランダムに選んだ 200 曲の各楽曲からある 10 秒から 15 秒程度のフレーズを MIDI から抜き出し、入力フレーズとして検索を行い、その検索結果を用いて検索システムの性能を MRR,Top-10 hit rate で評価し、同じメロディを入力として楽曲検索する従来のハミングシステムと比較した。MRR、Top-10 hit rate はともに 0 から 1 の範囲の実数値をとり、1 に近いほど目的の楽曲が検索結果のランキング上位にヒットすることを表す。

$$\mathrm{MRR} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\mathrm{rank}_i}$$

$$\text{Top-N hit rate} = \frac{\text{N 番目までに正しい結果が含まれるクエリ数}}{\text{全体のクエリ数}}$$

この実験の結果、MRR は 0.33、Top-10 hit rate は 0.69 と従来システムと同程度の値であり、従来と同等の性能であった。ここで MRR が 0.33 であることは正解楽曲がおよそ上位 3 位までには出現することを表している。また、Top-10 hit rate が 0.69 であることは検索結果上位 10 件に正解楽曲が 69% で存在していることを示す。

表 1. MRR and Top-10 hit rate of experiment1

	MRR	Top-10 hit rate
Proposed	0.33	0.69
Baseline SWEM-aver	0.33	0.73

4.2 実験 2

2 つ目の実験として、実際に楽器で演奏されたフレーズを MIDI に変換しクエリとして検索を行った。楽器にはエレキベースを選び、演奏による音声信号を MIDI に変換する sonuus G2M V2 を用い MIDI を録音した。しかし、事前調査によりこの機器は G1 以下の低音やあまりにも早い演奏による MIDI 変換には対応していないことが分かった。機器による MIDI 変換の影響をなるべく抑えるため、BPM200 などの速い楽曲に対しては BPM を 100 前後に落とし、G1 を含むそれよりも低い音を使用するフレーズの場合は一旦キーを上げた状態で録音し、MIDI 変換後、元のキーにトランスポーズを行った。今回は 5

曲を聞きその中の各楽曲 10 秒から 15 秒程度の 1 フレーズを演奏し、その MIDI をクエリとした。

この実験の結果は、MRR が 0.09、上位 10 曲に正解楽曲が現れず Top-10 hit rate が 0.00 となった。

表 2. MRR and Top-10 hit rate of experiment2

	MRR	Top-10 hit rate
Proposed	0.33	0.69
SWEM-aver	0.09	0.00

4.3 考察

実験 1 による MIDI 楽曲のある部分をそのまま抽出したクエリを用いた検索では従来システムと同等の性能を得ることができたが実験 2 による楽器演奏を MIDI へ変換したクエリを用いた検索では MRR,Top-10 hit rate が共に大きく低い値となった。そのため実験 2 で用いた MIDI クエリに問題があると考える。本研究で使用した MIDI クエリは自分の耳で聞いたときには何となく正解楽曲のある部分に似ていると感じることができている。実際に DAW ソフトを用いてピアノロールを確認すると、図 5 のように音符の配置の形状はおおよそ似ている。しかし、クエリで 1 音として演奏している音符が正解楽曲では 2 回に分けて演奏されている部分や、単純に音高がずれている音符が存在する。こういったこれにより、テキスト化を行う際に音高や音価のずれが生じマッチングに影響を与えていると考える。また、楽器を演奏する際にゴーストノートという演奏

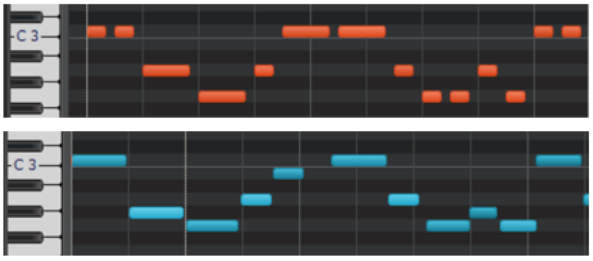

図 5. Piano roll of query and correct music: top is correct music, bottom is query

技法がある。ゴーストノートを用いることで、リズムキープやグルーブ感を生むことができる。ゴーストノートはごく短い時間に発生させる音高のない音であり、エレキベースではよく多用される。図**??**はゴーストノートを用いて演奏されたクエリとその正解楽曲の一部である。クエリのピアノロールにはところどころに小さな音符が現れている。MIDI には休符と同様に、ゴーストノートの音符は存在しないためこの技法による演奏は本手法とは相性が悪いと考える。しかし、うまくゴーストノート部分の音符情報を削除することでより上位に検索結果が現れると考える。

5 あとがき

本研究では、MIDI の演奏フレーズを用いた楽曲検索システムを提案した。従来、クエリマッチングにかかっていた計算時間を改善するためにコサイン類似度を用いた。そのため、データベース中の楽曲と入力フレーズをテキスト化し word2vec の一種である Fasttext の事前学習により生成された単語分散表現から SWEM を用いて文章レベルベクトルをそれぞれ生成した。検索結果はコサイン類似度が大きい楽曲を上位にランキングし決定した。

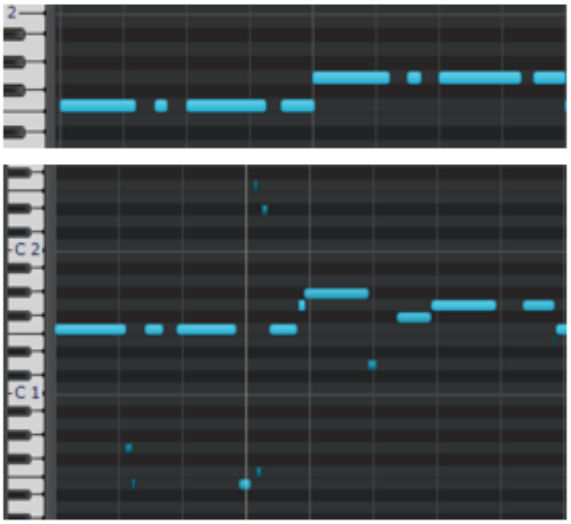

図 6. Queries with many ghost notes and their correct parts

実験の結果、演奏による検索結果は悪い実験では評価指標である MRR と Top-10hit rate において従来システムと同等の性能を示したが、本研究では従来の約 10 倍の規模である約 5 万曲の楽曲をデータベースに使用している。そのため、本システムが従来システムより検索システムとして優れていると考えられる。

また、本研究では入力フレーズやデータベース中の楽曲のクエリ生成の際、含んでいる単語の分散表現の加算平均をとる SWEM-aver を用いた。他の研究で最も成果を上げている SWEM-hier との比較も今後行っていく。

参考文献

[1] Muhammad Ulfi, Rila Mandala, ”Improving Query by Humming System using Frequency-Temporal Attention Network and Partial Query Matching”, 2022 9th International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA).

[2] Piotr Bojanowski, Edouard Grave, Armand Joulin, Tomas Mikolov, ”Enriching Word Vectors with Subword Information”, https://doi.org/10.48550/arXiv.1607.04606

[3] Tatsunori Hirai, Shun Sawada, ”Melody2Vec: Distributed Representations of Melodic Phrases based on Melody Segmentation”, Journal of Information Processing Vol.27 278–286 (Mar. 2019)

[4] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J.: Distributed Representations of Words and Phrases and Their Compositionality, Advances in neural information processing systems, pp. 3111–3119 (2013).

[5] Colin Raffel. Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching. PhD thesis, Columbia University, 2016.

[6] D. Shen, G. Wang, W. Wang, M. R. Min, Q. Su, Y. Zhang, et al., ”Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms”, CoRR, 2018.