CA-001

自然言語を含む文書の埋め込みベクトル分析のための位相的データ解析手法

佐藤 哲 †

パーソルキャリア株式会社 †

1 はじめに

自然言語や記号，数式で構成された文書を機械学習で処理できるようにするためには，文字列で構成されている文書を数値で構成されているベクトルに変換する埋め込みベクトル作成の技術が必要である．文書に対する埋め込みベクトル作成の技術は，近年ではニューラルネットワークの言語モデルを利用し，文書データをトークン化しニューラルネットワークに入力し，その出力をベクトルとして利用する方法が主流である．しかし，言語モデルの出力は変数の数が多い高次元ベクトルにより構成されており，扱いが難しい．そこで本研究では，高次元ベクトルに対し位相的データ解析を適用することで効果的に特徴を抽出する方法を提案する．高次元の埋め込みベクトルに対し位相的データ解析を用いてベクトル同士の類似度を測定する実験を行った結果，本研究で用いたタスクについては，埋め込みベクトルをそのまま用いるよりも精度が 10%以上向上することを確認した．

2 言語モデルによる文書データのベクトル化

自然言語を含む文書データをベクトル化する手法は，数多く提案されている．例えば，(1) トークン（単語）の埋め込みベクトルに基づく手法（Sentence Bert[1] など）や (2) ニューラルネットワークの次元圧縮に基づく手法（Universal Sentence Encoder[2] など）が有名である．本研究では，トークン単位ではなく，文書データに含まれるトークン全体から情報を抽出して文書データの埋め込みベクトルを求める (2) に近いアプローチを取る．

本研究では，大規模言語モデルである RWKV モデル [3] を用いて文書データをベクトル化する手法について検討する．RWKV モデルは， Transformer[4] の特徴を取り入れた Recurrent Neural Network (以下，RNN) で，Transformer のモデル性能と処理の並列性を持ちつつ，RNN のような低コストの処理を達成するとされるアーキテクチャで，Apache License で公開されている††．図 1 は，RWKV モデルのアーキテクチャを説明する概要図の一部である．$t = 0$ で「My」というトークンが入力されると，ニューラルネットワークの各層で演算処理され，その結果は $t = 1$ のレイヤーで使用されると共に，$t = 0$ のレイヤーの出力としてトークンの出現確率が出力される．図 1 の LM Head のブロックは，ニューラルネットワークの出力から Softmax 関数により確率を計算しており，ニューラルネットワークの出力自体は入力されたトークン列の固定長ベクトル埋め込みとみなすことができる．標準的な RWKV モデルでは 50277 次元のベクトルが得られる．

一般に，大規模言語モデルによる文書データの埋め込みベクトルは次元が高く分析が難しい．埋め込みベクトルの使い方は，埋め込みベクトル同士のコサイン類似度を用いて比較することが標準的であるが，(1) 結果がスカラー量であるため，表現能力に限界がある，(2) 高次元空間ではベクトルの分布に偏りが発生するため，計算された類似度にも偏りが発生する，などの問題がある．そこで本研究では，位相的データ解析の導入によりそれらの問題の解決を試みる．

3 位相的データ解析による特徴抽出

文書データを数値化した埋め込みベクトルは，単純な時系列データとは異なる構造を持つと考えられる．単純な時系列データは，時間発展によって変化するデータであり，変化の要因は主に時間軸を中心として複数の要素が影響する．一方で文書データは，言語による文法やコンテキスト，文のテーマなど多くの拘束条件によりパラメータが束縛されており，複雑な構造を持っている．単調増加などではない複雑な構造を持ったデータに対しては，幾何的な構造を抽出する手法の適用が効果的である．本研究では，文書データに対し位相的データ解析を適用し，特徴の抽出を試みる．

3.1 Sliding Window Embedding によるシーケンシャルデータの 3 次元埋め込み

RNN のような入力データ系列を逐次的に処理するアーキテクチャでは，入力データの近傍のデータとの関連性を抽出することが効果的である．時間的に前後

Topological Data Analysis for Embedding Vector Analysis of Documents with Natural Language

†Tetsu R. Satoh, PERSOL CAREER CO., LTD.

†† https://github.com/BlinkDL/RWKV-LM

のデータの変化具合を抽出する手法としては，Sliding Window Embedding が一般的である [5]．Sliding Window Embedding では，入力データを一定のウィンドウ幅で抽出することにより，ウィンドウ幅の次元のデータ系列を生成する手法である．入力データが一次元データ系列

$$x_k, x_{k+1}, x_{k+2}, x_{k+3}, \cdots$$

で，ここからウィンドウ幅 2 でデータを抽出する処理を行えば，

$$(x_k, x_{k+1}), (x_{k+1}, x_{k+2}), (x_{k+2}, x_{k+3}), \cdots$$

という 2 次元ベクトルのデータ系列を得ることができる．この変換は，データ長が無限であってもデータの値域が有限である場合，有限の空間に埋め込むことが可能であり，有限の範囲内での構造を抽出することに適している．図 2 では，1 次元のデータ系列を 2 次元に埋め込んだ例を示している．本研究では，ウィンドウ幅 3 でデータを抽出し，3 次元空間にデータ系列を埋め込む処理を採用している．

3.2 パーシステンスホモロジーによる 3 次元データの 2 次元埋め込み

本研究では Sliding Window Embedding により有限の空間に時系列データを埋め込んでおり，有限の空間内で幾何的な構造を抽出するために位相的データ解析におけるパーシステンスホモロジーを利用する．パーシステンスホモロジーを使うことにより，任意の次元のデータを 2 次元のパーシステンス図として表すことができる [6].

パーシステンス図を求めるアルゴリズムは以下である：

(1) 各データ x_i に対し，データの次元 n の空間の中でデータを中心とする半径 r 球 B_r を配置する

$$B_r(x_i) = \{x \in R^n | ||x - x_i|| < r\}$$

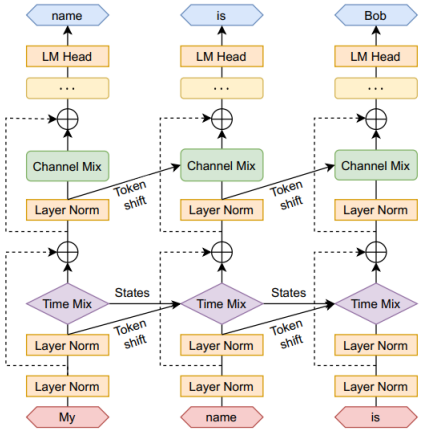


Figure 3: RWKV architecture for language modelling.

図 1: RWKV モデルのアーキテクチャ．文献 [3] より引用

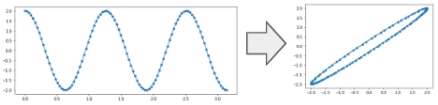


図 2: Sliding Window Embedding による埋め込み例

図 3: フィルトレーションの例

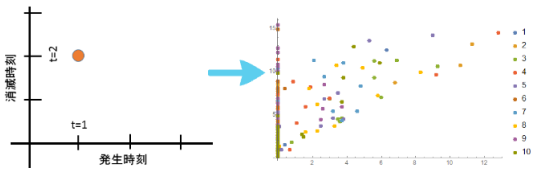


図 4: パーシステンス図の例

(2) 半径 r を増大させ，球同士の交差点により発生する位相的特徴量の穴を抽出し，穴の発生時刻と消滅時刻を記録する（フィルトレーション，図 3）

(3) 発生時刻を x 座標，消滅時刻を y 座標とするパーシステンス図を生成し，2 次元ベクトル群とする（図 4）

3.3 位相的データ解析をを利用した文書データからの特徴抽出

本研究での位相的データ解析の技術を利用した文書データからの特徴抽出は，次のようになる：

(1) 文書データをトークン化し，RWKV モデルに入力することで 50277 次元のベクトルを得る

(2) 50277 次元のベクトルの成分を一つのデータ系列として Sliding Window Embedding を適用し，50275 個の 3 次元ベクトル系列を得る

(3) 3 次元ベクトル系列に対しパーシステンスホモロジーを計算し，2 次元ベクトル列を得る

このアルゴリズムにより，文書データを 2 次元ベクトル列で表すことができる．

4 実験

文書データとして，グーテンベルクプロジェクトのデータ配布サイト https://www.gutenberg.org/にて，1) Lewis Carroll, 2) Einstein, 3) Lovecraft で検索し，その中で特徴的な 5 作品ずつを，各代表する文書として実験をする．1) は幻想文学，2) は科学啓蒙書，3) は幻想怪奇文学と分類される．文書データの分類方法は，著者や分野，言語，執筆ジャンルなど様々な考え方があり得るが，ここでは幻想文学・科学

表 1: 各類似度計算法に基づくスコア計算結果

トークン数	1024	2048	4096
コサイン類似度	0.4578	0.4678	0.3689
JFIP	0.4933	0.4844	0.5022
TDA+コサイン類似度	0.5022	0.4756	0.5111
TDA+JFIP	0.4844	0.4667	0.5644
bottleneck 距離+Gauss カーネル	0.52	0.5911	0.5111

啓蒙書・幻想怪奇文学の分類の推定結果により判定する指標を使う．すなわち，2 つの文書を (i, j) に対し，それぞれの文書が属するクラスタを $(cl(i), cl(j))$，推定された類似度を S_{ij} としたとき，

$$hit_{ij} = \begin{cases} 1, & \text{if } k = l \text{ and } S_{ij} \geq 0.5 \\ 0, & \text{if } k = l \text{ and } S_{ij} < 0.5 \\ 1, & \text{if } k \neq l \text{ and } S_{ij} < 0.5 \\ 0, & \text{if } k \neq l \text{ and } S_{ij} \geq 0.5 \end{cases} \quad (1)$$

とし，

$$\text{スコア} = \sum_{i=0}^{n} \sum_{j=0}^{n} hit_{ij}/n^2 \quad (2)$$

として，スコアが高いほど性能が良いとする．

類似度の計算方法は，以下のような手法を用いる：

- RWKV モデルから出力された 50277 次元のベクトル同士にコサイン類似度を計算する
- RWKV モデルから出力された 50277 次元のベクトル同士に JFIP 類似度 [7] を計算する
- RWKV モデルから出力された 50277 次元のベクトルからパーシステンスホモロジーを計算して得られた 2 次元ベクトル系列を，1 次元に変換してコサイン類似度を計算する
- RWKV モデルから出力された 50277 次元のベクトルからパーシステンスホモロジーを計算して得られた 2 次元ベクトル系列に，JFIP 類似度を計算する
- RWKV モデルから出力された 50277 次元のベクトルからパーシステンスホモロジーを計算して得られた 2 次元ベクトル系列に，bottleneck 距離 [8] を計算し，Gauss カーネルにより類似度に変換する

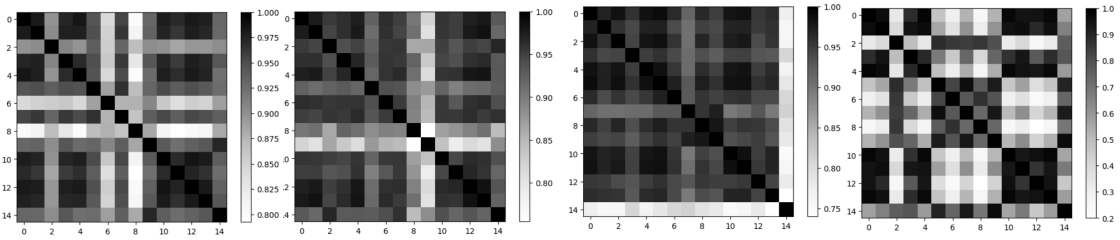
パーシステンスホモロジーの計算によって得られた 2 次元ベクトル系列に含まれるベクトルは，全てが重要な情報を持つのではない．穴の発生時刻，消滅時刻により定義される 2 次元ベクトル (x, y) は，定義により直線 $y = x$ のグラフより上にあり，かつ直線 $y = x$ からの距離が小さいベクトルは重要度が低く，距離が大きいベクトルは重要度が高いという性質があるため，直線 $y = x$ からの距離によりソートし，距離が大きい 10 個以外は捨てる処理をしている．また，bottleneck 距離を類似度行列に変換する際の Gauss カーネルの分散に対応するパラメータは 3.0 とした．

表 1 に，スコア計算の結果を示す．実験では，文書データをトークン化し，先頭から 1024 トークン，2048 トークン，4096 トークンとトークン長を変化させ，各類似度を計算した．下線で示されたスコアは，各トークンサイズのデータに対し最もスコアが高かった結果を示している．結果より，RWKV モデルから出力されたベクトルをそのまま利用するよりも，位相的データ解析によりパーシステンスホモロジーを計算して得られた 2 次元ベクトル系列を利用した方が精度が高いことが確認できる．ただし，式 (1)，式 (2) により定義されるスコアは，類似度 S_{ij}| が全て 0 となるケースや全て 1 となるケースでは全て 0 となるケースの方がスコアが高くなるように設計されているため，評価には注意が必要である．図 5a〜図 5o には，スコア計算に用いた類似度行列の濃淡画像を示す．カッコの中の数字は，入力されたトークン数である．RWKV モデルから出力されたベクトルをそのまま利用した場合は類似度が 0.8 以上と高く計算されている場合が多いのに対し，位相的データ解析を利用した場合は類似度が 0 から 1 の間に分散している傾向があることが分かる．

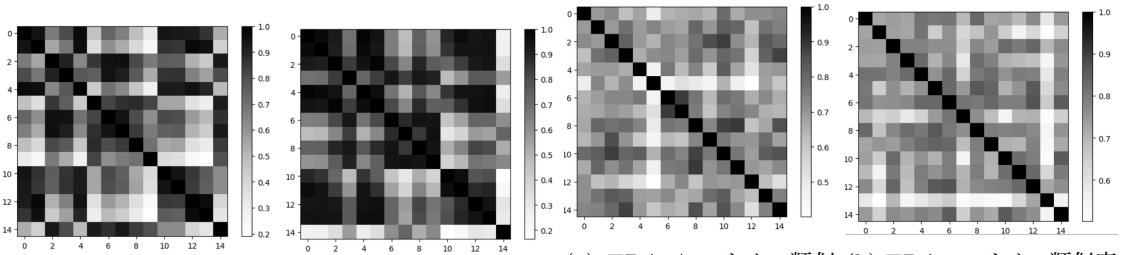
以上の実験は，パーシステンスホモロジーの計算には homcloud[9] を，Sliding Window Embedding の計算には [10] を用いた．また，RWKV モデルによる推論を含めた全ての実装は，Google Cloud Compute Engine の a2-highgpu-1g インスタンスモデル（NVIDIA A100 40G）上で python3.7 により実行した．

5 おわりに

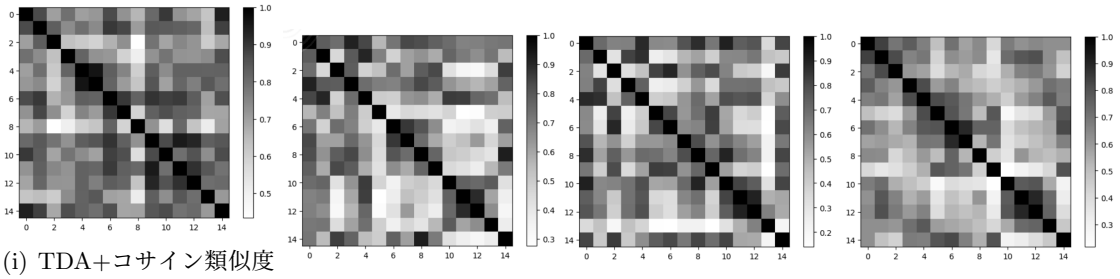
本研究では，言語モデルを利用し，自然言語を含む文書データから埋め込みベクトルを求め，埋め込みベクトルに位相的データ解析を適用することで効果的に特徴量を抽出する手法を提案した．位相的データ解析を利用して抽出した特徴量ベクトルを類似度計算に用いることにより，精度が向上することを確認した．



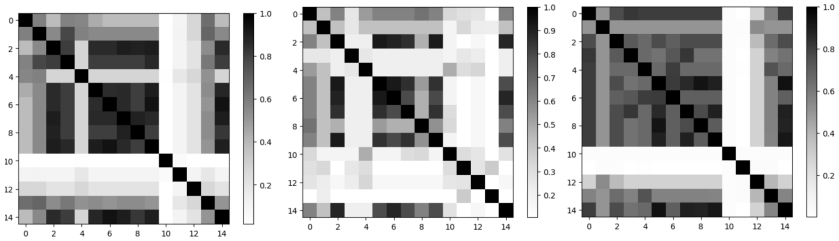
(a) コサイン類似度 (1024) (b) コサイン類似度 (2048) (c) コサイン類似度 (4096) (d) JFIP(1024)



(e) JFIP(2048) (f) JFIP(4096) (g) TDA ＋コサイン類似度 (1024) (h) TDA+コサイン類似度 (2048)



(i) TDA+コサイン類似度 (4096) (j) TDA ＋ JFIP(1024) (k) TDA+JFIP(2048) (l) TDA+JFIP(4096)



(m) bottleneck 距離＋ Gauss カーネル (1024) (n) bottleneck 距離＋ Gauss カーネル (2048) (o) bottleneck 距離＋ Gauss カーネル (4096)

参考文献

[1] N. Reimers and I. Gurevych, Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, arXiv 1908.10084, 2019

[2] , Universal Sentence Encoder, D. Cer el. al, arXiv 1803.11175, 2018.

[3] B. Peng et. al., RWKV: Reinventing RNNs for the Transformer Era, arXiv 2305.13048, 2023.

[4] A. Vaswani, et. al, Attention Is All You Need, arXiv 1706.03762, 2017.

[5] J. Perea and J. Harer, Sliding Windows and Persistence: An Application of Topological Methods to Signal Analysis, arXiv 1307.6188, 2013.

[6] 平岡裕章, 位相的データ解析とパーシステントホモロジー, 数学, Vol. 68, pp. 361–380, 2016.

[7] , B. Fernando and S. Herath, Anticipating human actions by correlating past with the future with Jaccard similarity measures, arXiv 2105.12414, 2021.

[8] , D. Cohen-Steiner and H. Edelsbrunner and J. Harer, J. Stability of Persistence Diagrams. Discrete Comput Geom 37, pp. 103—120, 2007.

[9] I. Obayashi and T. Nakamura and Y.i Hiraoka, Persistent Homology Analysis for Materials Research and Persistent Homology Software: HomCloud, J. Phys. Soc. Jpn., Vol. 91, No. 9, arXiv 2112.03610, 2022.

[10] , G. Tauzin el. al., giotto-tda: A Topological Data Analysis Toolkit for Machine Learning and Data Exploration, arXiv 2004.02551, 2021.