ポワンカレ空間における特徴ベクトル埋め込みの実装

谷聖一 研究室　須永光葵
Mitsuki Sunaga

概要

単語の意味や語彙体系を機械的に扱う様々な手法が提案されている. 分散表現は, 最近, 重要性を増している手法の 1 つであり, 単語の全体集合を数学的空間とみなし, 関連のある語同士は近く関連が少ない語同士は遠くなるような単語のベクトル表現である．これまで, 十分な表現力を持たせるには, 200 次元程度のユークリッド空間に埋め込むのが一般的であった．この次元が減少すると, 計算や学習に必要な計算資源が減少することが期待される．近年, 分散表現をポワンカレ空間に埋め込むことで, 200 次元よりはるかに低次元での表現獲得に成功したと主張がなされた [Maximilian Nickel and Douwe Kiela (2017)，Poincaré Embeddings for Learning Hierarchical Representations，https://arxiv.org/abs/1705.08039]. 本演習では, この手法の実装に取り組んだ.

1　はじめに

言葉を話すロボットはここ数年で馴染みのあるものになっているのではないだろうか．特に Apple 社の Siri や SoftBank 社のペッパーくんなどに代表される,「対話可能な AI」は今も研究が盛んな分野である．機械に自然言語を理解させるという目的のためには，主に「形態素解析」「構文解析」「意味解析」「文脈解析」の流れを汲むが，ここでは「意味解析」の中の語彙体系 · 単語理解について取り扱う.

まず単語の意味とは何だろうか．イギリスの言語学者 John Rupert Firth は,「意味とはコンテクストにおける機能である」と考えた ([3])．つまり，単語それ自身に意味があるのではなく，周辺の語との関係によりその文中における尤もらしい「意味」が生まれる．単語の意味を考える様々な手法においてこの言葉が体現されている．80 年代，単語を分割して意味を捉える方法が考案された ([5])．例えば「幸せ」という単語を「誰かを無表情から笑顔にするような感情」と表現し以下のように表すとする．幸せ → 感情 (誰か (無表情 → 笑顔)) このようにして意味が表せたように思うが，今度は「感情」や「笑顔」の意味が必要になり何を基準にすべきか明確でない．90 年代には，アメリカの言語学者 Z.S.Harris により「共起」の概念が提唱され，共起表現と大規模コーパスを利用して高い精度で単語の曖昧さを解消した ([6])．共起数を要素とする共起行列も考えられたが，単語数が増えるにつれ次元も上がり疎な行列になるので頑健性がない．次に考えられたのが分散表現である．これは単語を高次元の実数ベクトルで表現する技術である．分散表現を象徴する有名な例で，王様 − 男 + 女 = 女王様がある．ベクトルの演算が意味の演算のようになっている．文章は本 · ネット記事など世の中に溢れているため，それらを学習データとし，単語の分散表現を求めるツールとして word2vec[4] が有名である.

論文 [1] では，これまでユークリッド空間に埋め込んでいたベクトルをポワンカレ空間という空間に埋め込むことにより，精度が飛躍的に上昇したという．単に精度が上がっただけでなく，埋め込む次元が大幅に削減され，わずか 5 次元で従来の精度を上回ったという．本演習ではこの論文 [1] の手法の実装を試みた.

2　Word2Vec

2.1　概要

Word2Vec は 2013 年に Google 社の Tomas Mikolov 氏らにより提案された自然言語処理の手法である．学習により 200 次元ほどの単語の分散表現が獲得でき，類似語の算出や単語ベクトル演算結果を返す．オリジナルのリポジトリにはデモ用のテキストファイルが用意されているが，自身でデータを用意する際の注意点を挙げる．英文であれば単語ごとにスペースがあるが日本語にはないため,「は」や「の」などはある単語に含まれる一文字であるか助詞であるかがそのままではわからない．ゆえに文を分かち書き (文節分け) する必要がある．そのためにオープンソースの日本語形態素解析ソフトの MeCab がよく用いられる．MeCab について詳しくは [2] を参照されたい．実装するにあたり，まず Word2Vec の基本的な動作を確認する．Word2Vec は大きく分ければ，Skip-gram と CBoW(Countinuous Bag-of-Words) という学習アルゴリズムから生成されるモデルを用途により使い分ける．簡単に言えば Skip-gram はある 1 単語から類似 (関連) する単語を，CBoW はいくつかの単語から連想される 1 単語を求めるイメージである．今回は

Skip-gram のみ実装を行うため，Skip-gram の動作を確認する．以下 2.2 節でツールの基本事項をまとめ，2.3 節で数学的な解説を行う．

2.2 ツールについて

2.2.1 学習目的

類似語の算出が可能と述べたが，近しい単語 · 関連する単語の基準を決めなくてはいけない．基本的には共起表現の考え方であり，注目する単語の周辺何単語かを関連語とみなし学習する．(コードではウィンドウサイズで指定する．) 共起する単語対の内積を大きく，乱択された単語対の内積は小さくなるように最適化を行う．つまり関連語の出現確率を最大化させるような分散表現を求めることが Word2Vec の学習目的である．

2.2.2 ニューラルネットワーク

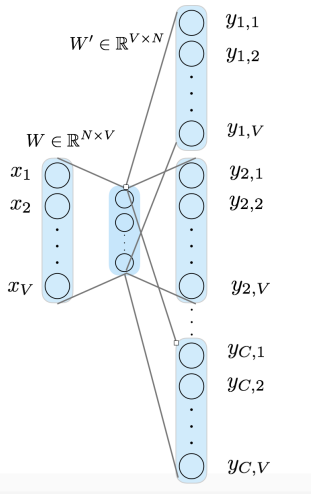

図 1: ニューラルネットワーク

以下で使う用語を定義する．

V

単語辞書．単語数は入力データの重複をなくした数一般に日常会話で使われる英単語は 3000 語，日本語単語は 10000 語と言われる

$\boldsymbol{v_i}$

$\boldsymbol{v_i} \in V, 1 < i < |V|$
各単語のベクトル (分散表現) 初期値は乱数で与えられる

N

分散表現の次元

C

コンテキスト数．関連語の数

W

入力層から隠れ層への重み
単語ベクトルを横に並べた $N \times V$ 行列
$W = [\boldsymbol{v_1}, \boldsymbol{v_2}, \boldsymbol{v_3}, ..., \boldsymbol{v_{|V|}}]$

W'

隠れ層から出力層への重み
単語ベクトルを転置して並べた $V \times N$ 行列
(ただし縦ベクトル)
$W' = [\boldsymbol{v_1}^{\mathrm{T}}, \boldsymbol{v_2}^{\mathrm{T}}, \boldsymbol{v_3}^{\mathrm{T}}, ..., \boldsymbol{v_{|V|}}^{\mathrm{T}}]$

各層の簡単な解説を以下に記述する．

入力層

入力値:one-hot-vector

隠れ層

重み:W
入力値は one-hot-vector であることから，注目単語自身のベクトルが出てくる

出力層

重み:W'
活性化関数:softmax 関数
出力:関連語の出現確率

上記の通り，構造はかなり単純である．学習の肝は重みの更新にある．

2.2.3 最適なパラメータ

関連単語の出現確率は w_t を注目語 (target)，w_a を関連語 (around)，w_v を全単語 (V の元)，v_{w_i} を $w_i \in V$ のベクトルとすると

$$P\left(w_t|w_a\right) = \frac{\exp\left(v_{w_t}^{\mathrm{T}} \cdot v_{w_a}\right)}{\sum_{w_v \in V} \exp\left(v_{w_v}^{\mathrm{T}} \cdot v_{w_t}\right)}$$

ここで分母を見ると全単語との内積を計算しなくてはいけないことがわかるが，前述した通り日常会話で使われるものだけで 3000 語を越えるため，計算に膨大な時間がかかる．そのための技術としてネガティブサンプリングと階層的ソフトマックスが挙げられる．(Word2Vec はデフォルトでネガティブサンプリング付きの Skip-gram を用いている．)

ネガティブサンプリング

本来，注目語から予想される単語は何かというフェイクタスクに答えるためのアルゴリズムであるため全単語との共起確率を求める必要があるが，何かという漠然な問いではなく○○か？と具体的な質問にするとしよう．そうすればほとんどイエス

かノーで答えらる単純な問いになる．もちろん意味のある学習にするため○○に当たる単語は注目語に関連すると思われる単語にする必要があるが，それだけでは常に答えがイエスになってしまう．そこで確率分布に従い単語を乱択し，それらとの内積は小さくなるようにし目的関数の値を最大化する．このサンプリングによるパラメータの更新の極限は softmax 関数の極限に近接することに加え，計算量も乱択したノイズ単語数に比例する．一般的にこのノイズ単語は学習データが十分にあり，関連語が 10 語前後であれば 5〜20 語程度で十分な精度が出ると言われている．

階層的ソフトマックス

全単語を木構造に埋め込むことにより，ある単語がターゲット予測される確率は根からその単語に至るパス上の分岐点に対する確率の総乗のみを計算すれば良い．木に埋め込むには，語彙の上位関係が定義された辞書やハフマン木を用いたりするが新しい単語を加えるには木を再構築する必要がある．

3　先行研究について

本節では，先行研究 [1] の主張を確認する．簡潔に述べると，通常の学習で得られている分散表現はユークリッド空間における実ベクトルであるが，埋め込む空間を変えることにより低次元で同等以上の表現を得られるということである．学習層を厚くしたりするのではなく，表現そのものの性質をうまく捉え利用するアプローチである．報告者自身実装に取り組んだが，値の調整や関数をコードに起こすのはかなり難しく，テストデータで検証してみたがあまり期待した結果は得られなかった．そこは今後改善を試みることとするが基本的な数学的考察はさほど難しくはないと思われるため，この報告書においてはポワンカレ空間の定義や用いることの利点・従来の関数との違いなどを次節で解説する．

4　本演習について

4.1　ポワンカレ空間

非ユークリッド幾何学における双曲幾何のモデルの 1 つである．非ユークリッド幾何学とはユークリッドの第 5 公準，平行線の公理を満たさない幾何を考察する分野の総称で，例えば「三角形の内角の和が 180 °」を満たさない．身近な例としては球面を考えると良い．球面は双曲幾何のモデルではないが，球面上の三角形は内角の和が 180 °よりも大きくなる．飛行機の航路を地球儀に弧を描くように引いたことのある人が多いと思うが，これは測地線と呼ばれ，ユークリッドで考える直線に相当する．

数学的な定義は，単位円板 $\mathcal{D} = \{z \in \mathbb{C} : |z| < 1\}$ の元 u, v に対し

$$d(u,v) = \log \frac{1 + \frac{|v-u|}{|1-\overline{u}v|}}{1 - \frac{|v-u|}{|1-\overline{u}v|}}$$

とすれば $d : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ は $\mathcal{D}$ の距離であり，$(\mathcal{D}, d)$ は距離空間である．単位円というある意味有限の空間に無限の空間を埋め込んでいるようなものなので，同じ距離の測り方では不可能だということがわかるであろう．エッシャーが描いた円の極限という絵はまさしくポワンカレ円板をユークリッド平面に描こうとしたものである．

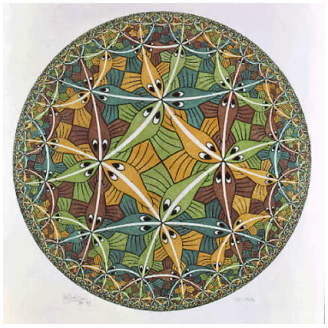

図 2: 円の極限　http://www.mcescher.com/

実際，縮尺 $\frac{2}{1-\|z\|^2}$ で距離が入るため周縁部に向かって爆発的に空間が広がっているならば，平面に描こうとすれば密に描かなくてはならない．また，厳密な定義では複素数を含んでいるが，実数のみを扱いたいので少し制限をかける．

$u, v \in \mathbb{R}^n$ がユークリッドノルムを備えそのノルムが 1 よりも小さい時，ポワンカレ空間における 2 点間の距離 $d(u,v)$ は

$$d(u,v) = arcosh(1 + \delta(u,v))$$

ここで $\delta(u,v)$ は

$$\delta(u,v) = 2\frac{\|u-v\|^2}{(1-\|u\|^2)(1-\|v\|^2)}$$

である．

4.2　ポワンカレ空間に埋め込む利点

ここまでの数学的定義から見えてくる利点が 2 つある．

- 測地線の計算が明示的にでき，等長変換によりユークリッド空間との対応もわかる
- ユークリッド空間よりも空間効率が良い

また，経験的に双曲空間には木構造が自然に埋め込めることが知られている ([7])．上位 (root に近いノード) が抽象語，下位が (葉ノード) 具体語であると考えれば，周縁部が広がっている方が確かに効率よく埋め込めそうである．これらの理由からポワンカレ空間に埋め込むのは良いアプローチであると思える．

4.3 既存関数との違い

ではどのようにこの空間に埋め込むのか．ユークリッドと同様共起単語の内積で定義するのが自然な考えであるが，先行研究では 2 点間の距離を用いている．(定義で距離しか扱わなかったのはこのためである．) もっと言えば，距離の性質からマイナスの値が取れないためマイナスを取り，パラメータで値を補正している．勉強していく中で双曲幾何における三角関数などのトピックが見つからなかったため，ユークリッド内積と対応したものを定義できないのではないかと考えられる．以上が本演習における実装に至るまでの数学的知識の要点である．

5 今後の課題

今回，実装したものの思うような結果は得られなかった．今後の課題として，コードを改善し精度の向上を目指す．それが実現した場合には，他の学習への応用も考えたい．

参考文献

[1] Maximilian Nickel and Douwe Kiela (2017)，Poincaré Embeddings for Learning Hierarchical Representations，https://arxiv.org/abs/1705.08039．

[2] MeCab: Yet Another Part-of-Speech and Morphological Analyzer，http://taku910.github.io/mecab/

[3] John Rupert Firth，https://en.wikipedia.org/wiki/John_Rupert_Firth

[4] Word2Vec，https://code.google.com/archive/p/word2vec/

[5] 自然言語処理の歴史的変遷，http://ocw.u-tokyo.ac.jp/lecture_files/is_01/12/notes/ja/langinfo12.pdf

[6] 共起，https://www.weblio.jp/content/共起

[7] 異空間への埋め込み!PoincareEmbeddings が拓く表現学習の新展開 http://tech-blog.abeja.asia/entry/poincare-embeddings