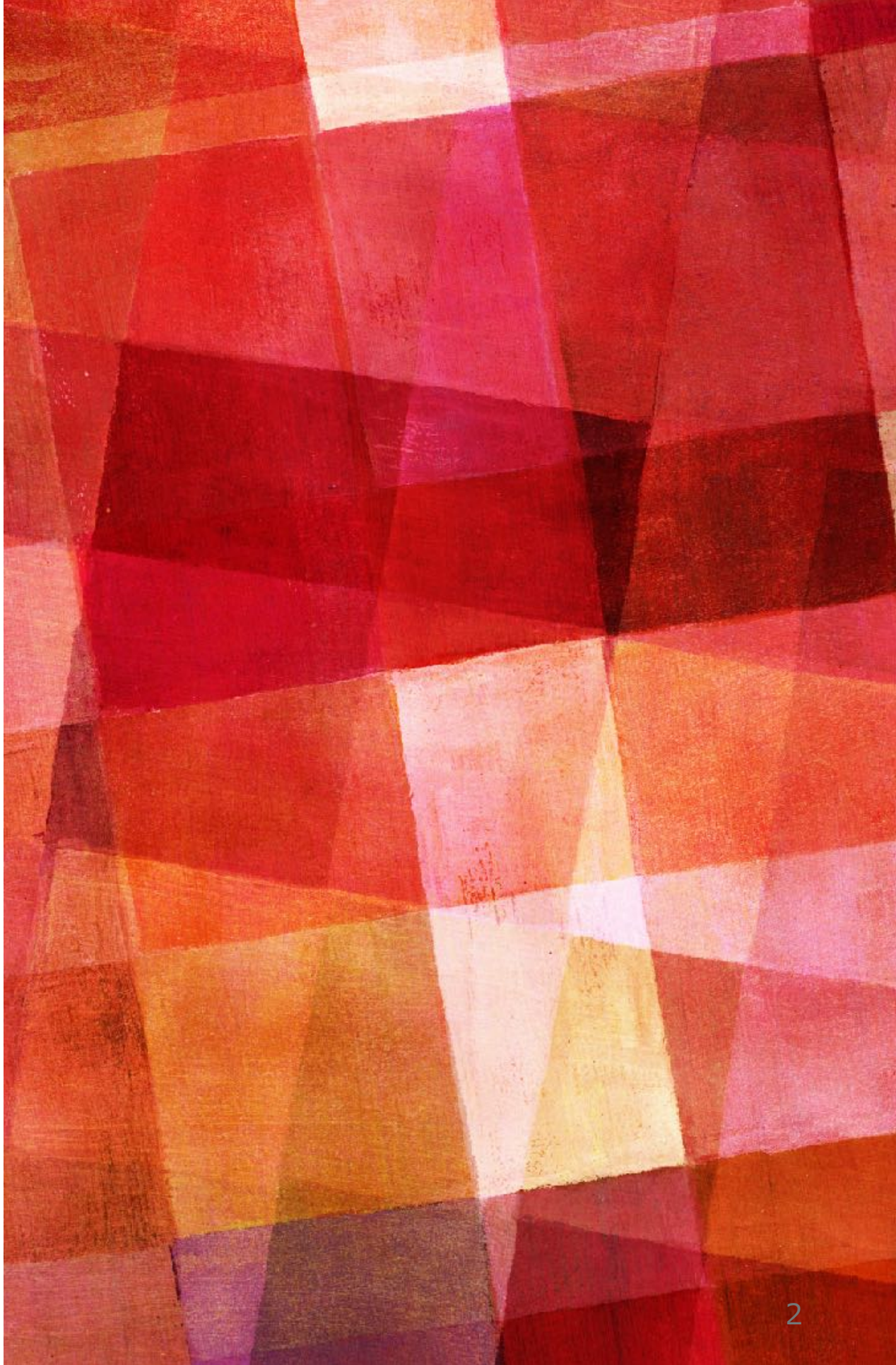
ポワンカレ空間における特徴ベクトル埋め込みの実装

日本大学文理学部情報科学科　谷聖一研究室
須永光葵

目次



1.はじめに

1-0 :導入

- 3年次履修科目：知識情報処理
- 人間が持っている経験や知識をコンピュータ内で蓄積・処理する方法論を学習する

  (2016年文理学部情報科学科,知識情報処理授業テーマより)
- 人間の話す言語(自然言語)を機械に理解させるための手段はすでに考えられている

1.はじめに

1-0 :導入

自然言語処理は処理の段階がある

形態素解析 → 構文解析 → 意味解析 → 文脈解析

今回は

語彙体系・単語理解

に着目する

機械に意味を理解させるには？

1.はじめに

1-1:”単語”の意味

文章に現れる単語はそれ自身のみでは意味を持たず、周辺に現れる単語によって意味が付加される(by John Rupert Firth)

ジョン ルパート ファース

- 私はPythonのコードを書いています
- 私はRubyのコードを書いています
- 私はプログラミングの授業でPythonを習いました

文脈から
PythonやRubyが似ていること

プログラミングやコードなどの周辺語からプログラミング言語であることが推察できる

1.はじめに

意味を理解させるための様々な手法の提案

- 単語の意味を分解して表現する方法(80年代)

  例.) 幸せ→感情(誰か(無表情→笑顔))

  →何を基準におけば十分なのかが曖昧

- 共起表現(90年代)

  その単語が使われた文脈に共起する単語で意味の曖昧さを解消

  - 共起する単語の意味は？？(大規模コーパスを使用)
  - 全単語を行列のラベルにし、共起数を要素とする

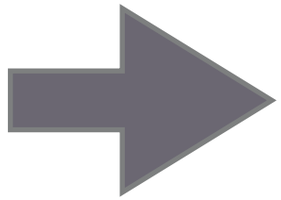
    →単語数により次元が上がりストレージを圧迫したり、巨大で疎な行列になり頑健性に乏しい

1.はじめに

1-2 :分散表現(特徴ベクトル埋め込み)

➤分散表現とは、単語を高次元の実数ベクトルで表現する技術

(岩波データサイエンス, https://sites.google.com/site/iwanamidatascience/vol2/word-embedding, 2018/01/24参照)



意味を数値化！

➤ ベクトルであれば加法・減法を定義したり、単語同士の“近さ・類似度”を数値として見ることができる

1 - 2 :分散表現(有名な例)

王様 - 男 + 女 =　?

1.はじめに

1-2 :分散表現(有名な例)

仮に、ベクトルの要素が4つで表現されているとする

王様(0.8 , 0.5 , 0.2 , 0.7)

男　(0.4 , 0.5 , 0.1 , 0.5)

女　(0.2 , 0.1 , 0.4 , 0.5)

女王(0.6 , 0.2 , 0.4 , 0.6)

このベクトルで

王様-男+女 を計算すると…?

(0.8-0.4+0.2 , 0.5-0.5+0.1 , 0.2-0.1+0.4 , 0.7-0.5+0.5)

=(0.6,0.1,0.5,0.7) ≒ 女王

１.はじめに

１-２ :分散表現

これで機械も意味を扱うことができそう！

しかし、日常会話の80~90%を理解するには英単語3000語(日本語では10,000語！)と言われている

(Glass Onion is Unbreakable ,http://103tommy.com/vocabulary,2018/01/24参照)

1 .はじめに

1 - 3 :word2vec

- 米googleの研究者が開発した自然言語処理の手法(2013年)
- 200次元ほどのベクトルで単語を表現(分散表現)
- Skip-gramとCBOWという学習モデルから構成
- 入力層・隠れ層(1層)・出力層のみからなるニューラルネットワーク
- 単純な共起回数ではなく周辺に現れる語を確率的に扱う
- 高速で新しい単語の追加も容易

(https://deeplearning4j.org/ja/word2vec#embed,2018/01/24参照)

1 .はじめに

1 - 4 :背景

2017 : 分散表現をポワンカレ空間に埋め込むことにより

word2vecで精度が向上したと主張

Poincaré Embeddings for Learning Hierarchical Representations

Maximilian Nickel
Facebook AI Research
maxn@fb.com

Douwe Kiela
Facebook AI Research
dkiela@fb.com

Abstract

Representation learning has become an invaluable approach for learning from symbolic data such as text and graphs. However, while complex symbolic datasets often exhibit a latent hierarchical structure, state-of-the-art methods typically learn embeddings in Euclidean vector spaces, which do not account for this property. For this purpose, we introduce a new approach for learning hierarchical representations of symbolic data by embedding them into hyperbolic space – or more precisely into an n-dimensional Poincaré ball. Due to the underlying hyperbolic geometry, this allows us to learn parsimonious representations of symbolic data by simultaneously capturing hierarchy and similarity. We introduce an efficient algorithm to learn the embeddings based on Riemannian optimization and show experimentally that Poincaré embeddings outperform Euclidean embeddings significantly on data with latent hierarchies, both in terms of representation capacity and in terms of generalization ability.

(Maximilian Nickel &Douwe Kiela.
Poincaré Embeddings for
Learning Hierarchical Representations. 2017)

1.はじめに

1-5 :動機(背景から動機への推移)

- 数学的に面白い空間で興味深い

この論文を理解したい

さらなる応用を考えたい

2.必要知識

word2vec(おさらい)

- 米googleの研究者が開発した自然言語処理の手法(2013年)
- 200次元ほどのベクトルで単語を表現(分散表現)
- Skip-gramとCBOWという学習モデルから構成
- 入力層・隠れ層(1層)・出力層のみからなるニューラルネットワーク
- 単純な共起回数ではなく周辺に現れる語を確率的に扱う
- 高速で新しい単語の追加も容易

(https://deeplearning4j.org/ja/word2vec#embed,2018/01/24参照)

２.必要知識

word2vec　(１)ツールについて

以下、

私は今日、プログラミングの授業でPythonとRubyのコードを書きました。

という例文で解説

この文章を単語で分かち書きにする

私 は 今日 プログラミング の 授業 Python と Ruby の コード を 書き ました

2.必要知識

word2vec　(1)ツールについて

前後何単語が関連するかを決める

例)”Python”に注目し、前後3単語が関連するとすれば関連語は

“プログラミング” “の” “授業” ”と” ”Ruby” “コード”

(コードでは “の” や ”と” などの頻出語は学習に含めない処理がなされている)

これらの関連語が出現する確率を最大化させるようなN次元の分散表現を求めることがword2vecの学習目的

2.必要知識

word2vec　(1)ツールについて

重複のない単語の集合にする

{私 今日 プログラミング の 授業 Python と Ruby コード を 書き ました}

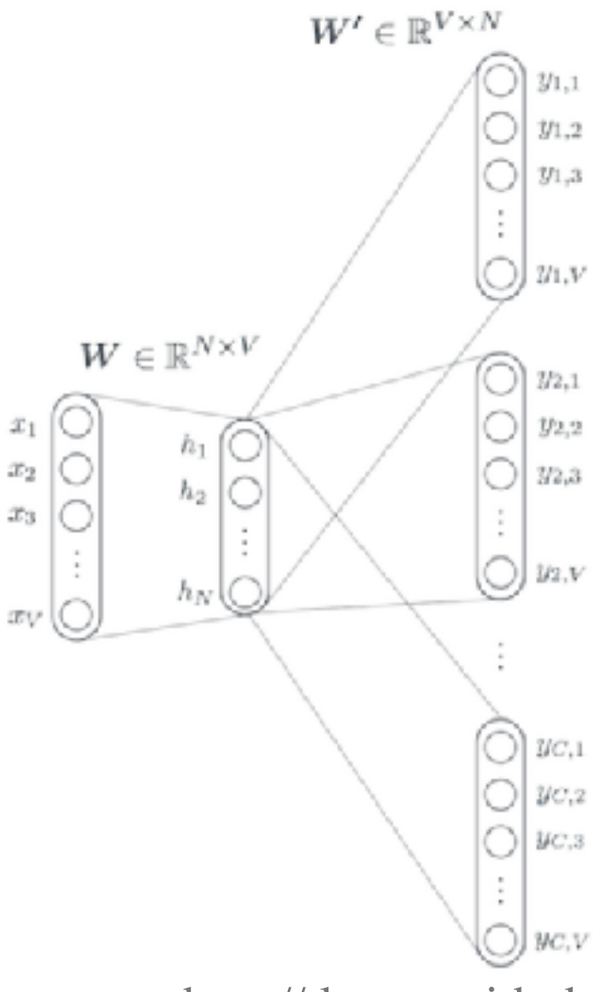
これが語彙(辞書)になる: V

(実際の V はテストデータの文章に出現する全単語を、重複をなく格納するため膨大になる)

2.必要知識

word2vec　(1)ツールについて

- ニューラルネットワークの構築



W　入力層から隠れ層への重み

W'　隠れ層から出力層への重み

C　コンテキスト数

注目語の前後にある単語数

N　単語ベクトルの次元

(Word2vecのニューラルネットワーク学習過程を理解する

http://tkengo.github.io/blog/2016/05/09/understand-how-to-learn-word2vec/ 2018/01/30参照)

2.必要知識

word2vec　(1)ツールについて

- ニューラルネットワークの構築

- 入力層

入力値:one-hot-vactor

次元　：V (ボキャブラリー数に依存)

2.必要知識

word2vec　(1)ツールについて

- ニューラルネットワークの構築

- 隠れ層

重み:全単語のベクトルを横に並べた行列

初期値:乱数

2.必要知識

word2vec　(1)ツールについて

- ニューラルネットワークの構築

- 出力層

重み:全単語のベクトルを転置し縦に並べた行列

活性化関数:softmax関数

出力:注目語の近くに関連語が現れる確率

2.必要知識

word2vec　(2)アルゴリズム(Skip-gram)について(要点)

① 関連する単語が出現する確率のモデル化

② 計算量の点におけるモデルの修正

③ 学習過程

2.必要知識

word2vec　(2)アルゴリズム(Skip-gram)について(要点)

① 関連する単語が出現する確率のモデル化

”Python”という単語に注目した時、”プログラミング”という単語が近くに現れる確率(softmax関数)

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

w_t :注目語(target)

w_a :周辺語(around)

w_v :一般単語(Vの元)

v :ベクトル

V :単語群・辞書(vocabulary)

２.必要知識

word2vec　（２）アルゴリズム(Skip-gram)について(要点)

① 関連する単語が出現する確率のモデル化

日常会話で使われる語だけで3000語以上

分母の計算に途方も無い時間がかかる

2.必要知識

word2vec　(2)アルゴリズム(Skip-gram)について(要点)

② 計算量の点におけるモデルの修正

- ネガティブサンプリング

本来”Python”から予想される単語は何か？

という質問に答えるためのモデルだが、

“Python”から予想される単語は”XX”か？

という質問にすり替える

2.必要知識

word2vec　(2)アルゴリズム(Skip-gram)について(要点)

② 計算量の点におけるモデルの修正

- ネガティブサンプリング

”XX”の部分に”プログラミング”や”Ruby”が入れば正解！

しかしそれだけだと常に真になってしまうので偽のサンプルも必要

ある確率分布に従い、偽サンプルを5~20個程度選び学習に混ぜる

(https://qiita.com/n_kats_/items/2691c08639468e30abdd 2018/01/30参照)

2.必要知識

word2vec　(2)アルゴリズム(Skip-gram)について(要点)

② 計算量の点におけるモデルの修正

- 階層的ソフトマックス
- 全単語を木構造に埋め込む
- ある単語の予測確率は、rootから単語に至るpath上の分岐点に対する確率の総乗

大規模データから単語の意味表現学習-word2vec, ボレガラダヌシカ

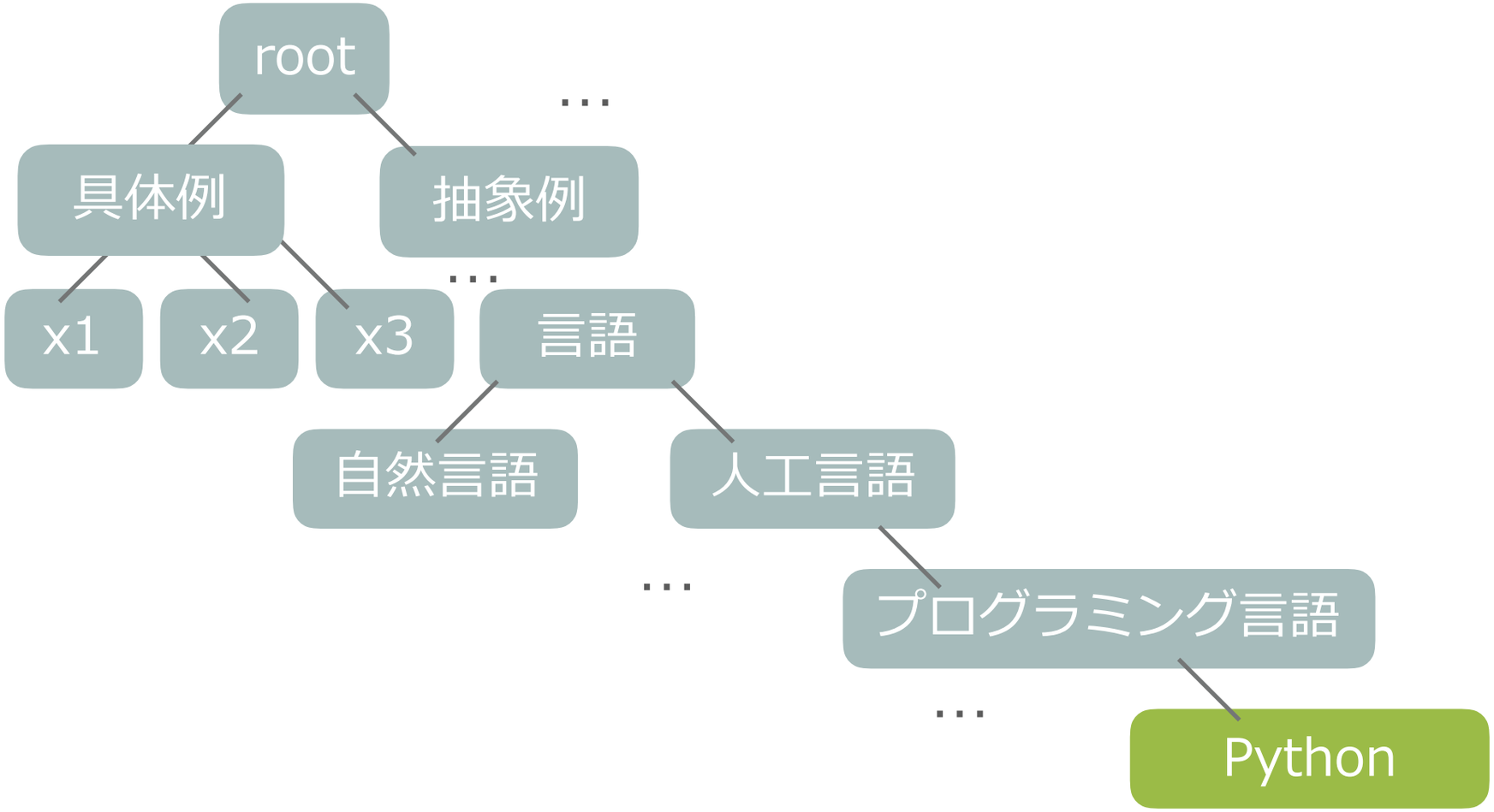
https://www.slideshare.net/adtech-sat/word2vec20140413

2.必要知識

word2vec　(2)アルゴリズム(Skip-gram)について(要点)

② 計算量の点におけるモデルの修正

- 階層的ソフトマックス



1単語に200次元はやっぱりコストが高い

3.先行研究

ポワンカレ埋め込みに関する論文要点

- 従来、word2vecではユークリッド空間に特徴ベクトルを埋め込んでいたが、ポワンカレ空間に埋め込んだ
- 低次元でより精度が高いという検証結果が得られた

(Maximilian Nickel &Douwe Kiela.
Poincaré Embeddings for Learning Hierarchical Representations. 2017)

3 .先行研究

Table 1: Experimental results on the transitive closure of the WORDNET noun hierarchy. Highlighted cells indicate the best Euclidean embeddings as well as the Poincaré embeddings which acheive equal or better results. Bold numbers indicate absolute best results.

			Dimensionality					
			5	10	20	50	100	200
WORDNET Reconstruction	**Euclidean**	Rank	3542.3	2286.9	1685.9	1281.7	1187.3	1157.3
		MAP	0.024	0.059	0.087	0.140	0.162	0.168
	Translational	Rank	205.9	179.4	95.3	92.8	92.7	91.0
		MAP	0.517	0.503	0.563	0.566	0.562	0.565
	Poincaré	Rank	4.9	4.02	3.84	3.98	3.9	**3.83**
		MAP	0.823	0.851	0.855	0.86	0.857	**0.87**
WORDNET Link Pred.	**Euclidean**	Rank	3311.1	2199.5	952.3	351.4	190.7	81.5
		MAP	0.024	0.059	0.176	0.286	0.428	0.490
	Translational	Rank	65.7	56.6	52.1	47.2	43.2	40.4
		MAP	0.545	0.554	0.554	0.56	0.562	0.559
	Poincaré	Rank	5.7	**4.3**	4.9	4.6	4.6	4.6
		MAP	0.825	0.852	0.861	**0.863**	0.856	0.855

(Maximilian Nickel &Douwe Kiela.

Poincaré Embeddings for Learning Hierarchical Representations. 2017)

3.先行研究

ポワンカレ空間(ポワンカレ円板)

- 非ユークリッド幾何の分野で考えられる双曲幾何のモデル

(非ユークリッド幾何学：三角形の和=180°が成り立たない幾何学のこと)

- ユークリッド空間を基準に考えれば”歪んだ空間”

図:Maurits Cornelis Eschers作,「円の極限III」,1959.
(http://www.mcescher.com/ 2018/01/26)

3.先行研究

ポワンカレ空間(ポワンカレ円板)

ユークリッド空間に存在する2点間の距離は直線

→歪んだ空間における２点の直線の定義は？

測地線

イメージ

飛行機の航路

球面も非ユークリッド幾何のモデルの１つ

3.先行研究

ポワンカレ空間(ポワンカレ円板)

➤ 定義

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

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

のとき

$d : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ は $\mathcal{D}$ の距離であり $(\mathcal{D}, d)$ は距離空間

単位円板の中に縮尺 $\dfrac{2}{1 - ||z||^2}$ で距離が入る

($||\cdot||$は通常のユークリッドノルム)

3.先行研究

ポワンカレ空間(ポワンカレ円板)

より使いやすくするため制限をする

$u, v \in \mathbb{R}^n$ がユークリッドノルムを備え、そのノルムが1より小さいとき

ポワンカレ空間における2点の距離 $d(u, v)$ は

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

ここで $\delta(u, v)$ は

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

3.先行研究

ポワンカレ空間(ポワンカレ円板)

- ポワンカレ空間の良い性質
  - 円板の淵へ行くほど空間が広がっている

    (空間の広がり方もわかっている)
  - 2点間の距離が明示的に計算できる
  - 木構造を自然な形で埋め込める

    (経験的に双曲空間一般でそうなることが知られている)

４.本演習について

先行論文では、2つの単語ベクトルの内積の部分を
ポワンカレ空間における距離で置き換えている

(単純に距離を使うのではなくその負値をとり、パラメータなどで補正している)

４.本演習について

- 獲得したベクトル

```
サンデーサイレンス -0.482454 5.618079 0.903902 -0.347189 3.479935
Halo -1.443560 6.841554 0.401120 0.384890 2.936371
WishingWell -1.226594 7.713564 0.680608 0.137879 3.726204
Mr.Prospector 2.127639 -3.753982 -0.796426 5.584993 -0.135832
Roberto 7.711109 6.218075 0.017276 -1.522113 -4.051614
ノーザンテースト -1.855956 7.575524 3.844712 3.029655 5.509552
トニービン -4.913055 5.647399 0.548582 4.706268 8.140084
Kingmambo -5.185405 -1.098755 -2.831470 9.669098 3.989560
Nureyev -1.070739 2.030840 -7.570820 -0.077712 2.873229
```

5.今後の課題

獲得したベクトルで親子関係を(母馬+父馬 = 子)になるか数頭検証したがあまり精度は良くなかった

- コードの見直し(パラメータの変更等)

実際に精度が出る実装ができたら

- 言語だけではなく他の学習への応用を考える