

第2回ナレッジグラフ推論チャレンジ2019 応募シート

1. 応募者に関する情報

氏名

伊鍋 貴宏

石 暁沂

所属

株式会社サキヨミAIラボ

メールアドレス

takahiro.inabe@sakiyomi.ai

版

2019.12.8

2. 推論・推理過程の説明

はじめに

人工知能による「推論」を実現する上で、ナレッジグラフは人間の持つ知識を人工知能に伝えるための重要な手段と考える。この知識に対する探索は「推論」実現における重要技術の一つである。我々は、グラフ構造で表現された知識を再整理することにより、人工知能が自ら生成(考案)した探索・推論の実現を目指す。この再整理した情報に対する探索行動を、「説明性」「解釈性」により評価する。今回のチャレンジでは、ナレッジ推論チャレンジサイトで公開されたRDFのトリプルのみを使用する方針とした。

提案手法

近年、機械学習による自然言語では「双曲空間への埋め込み」[1]や「BERT」[2]、「ALBERT」[3]による分散表現(ベクトル化)が注目を集めている。「双曲空間への埋め込み」では木構造データを低次元で精度よく埋め込めると示された。

「ALBERT」は汎用自然言語処理として複数の言語処理タスクにおいて性能向上が報告されている。我々は、ナレッジグラフの構造を反映させる整理を期待した「双曲空間への埋め込み」と言語の持つ意味による整理を期待した「ALBERT」による分散表現を比較して「説明性」「解釈性」のある探索行動に向けたナレッジグラフの再整理結果を観察する。

手法1:

「双曲空間への埋め込み」を用いて分散表現を作成する。通常、RDFリソース[5]の関係を主語(subject)、述語(predicate)、目的語(object)という3つの要素で表現する。この表現を「主語-述語」、「述語-目的語」、「主語-目的語」のペアによる表現へ変換することにより双曲空間への埋め込みを実現する。得られた分散表現は「Embedding Projector」[6]というツールで可視化することによって観察可能な状態にする。

入力形式:

- ・ 主語,述語
- ・ 述語,目的語
- ・ 主語,目的語

入力データの変換例:

変換前)

- ・ `<http://kgc.knowledge-graph.jp/data/ACaseOfIdentity/001>`
 `rdf:type kgc:Situation ;`
 `kgc:source "ホームズは椅子から立った"@ja ;`
 `kgc:source "Holmes stood out of a chair"@en ;`
 `kgc:hasPredicate <http://kgc.knowledge-graph.jp/data/predicate/stand> ;`

```
kgc:subject <http://kgc.knowledge-graph.jp/data/ACaseOfIdentity/Holmes> ;
kgc:from <http://kgc.knowledge-graph.jp/data/ACaseOfIdentity/Chair> ;
kgc:time "1891-09-01T10:00:00"^^xsd:dateTime .
```

変換後)

- ・ 001,has
- ・ has,Predicate,stand
- ・ 001,Predicate,stand
- ・ 001,time
- ・ time,1891-09-01 10:00:00
- ・ 001,1891-09-01 10:00:00
- ・ 001,subject
- ・ subject,Holmes
- ・ 001,Holmes
- ・ 001,when
- ・ when,1891-09-01 10:00:00
- ・ 001,1891-09-01 10:00:00
- ・ 001,source
- ・ source,Holmes stood out of a chair
- ・ 001,Holmes stood out of a chair
- ・ 001,from
- ・ from,Chair
- ・ 001,Chair
- ・ 001,type
- ・ type,Situation
- ・ 001,Situation

手法2:

「BERT」を発展させた「ALBERT」を用いて分散表現を作成する。この時Googleにより公開された事前トレーニング学習データを用いる。RDFのトリプルのみを使用して再整理するという方針に反しているが、「ALBERT」の分散表現能力を利用するために妥協した。入力に用いたテキストは、RDFのトリプルを「ALBERT」の表現形式に合わせて変換した形式とする。そして得られた分散表現を「手法1」と同様に「Embedding Projector」というツールを用いて可視化する。

入力形式:

[CLS]主語[SEP]述語[SEP]目的語[SEP]

入力データの変換例:

変換前)

- ・ <http://kgc.knowledge-graph.jp/data/ACaseOfIdentity/001>
rdf:type kgc:Situation ;

```

kgc:source "ホームズは椅子から立った"@ja ;
kgc:source "Holmes stood out of a chair"@en ;
kgc:hasPredicate <http://kgc.knowledge-graph.jp/data/predicate/stand> ;
kgc:subject <http://kgc.knowledge-graph.jp/data/ACaseOfIdentity/Holmes> ;
kgc:from <http://kgc.knowledge-graph.jp/data/ACaseOfIdentity/Chair> ;
kgc:time "1891-09-01T10:00:00"^^xsd:dateTime .

```

変換後)

- ・ [CLS]001[SEP]has Predicate[SEP]stand[SEP]
- ・ [CLS]001[SEP]time[SEP]1891-09-01 10:00:00[SEP]
- ・ [CLS]001[SEP]subject[SEP]Holmes[SEP]
- ・ [CLS]001[SEP]when[SEP]1891-09-01T10[SEP]
- ・ [CLS]001[SEP]source[SEP]Holmes stood out of a chair[SEP]
- ・ [CLS]001[SEP]from[SEP]Chair[SEP]
- ・ [CLS]001[SEP]type[SEP]Situation[SEP]

処理内容

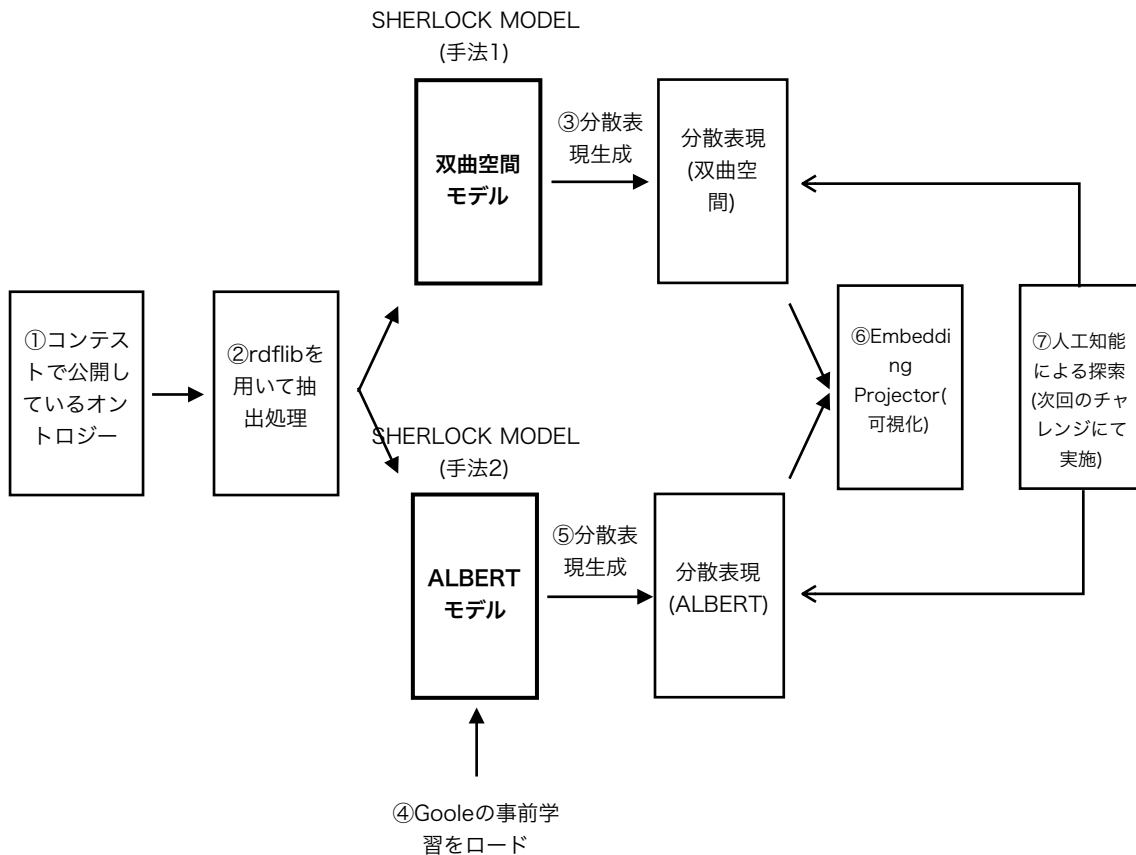


Fig. 1) 処理概要図

共通事前処理

1. ナレッジグラフ推論チャレンジサイトのSPARQLエンドポイントから、N-Triples形式でエクスポートする(例:kgc2019.nt)
2. N-Triples形式のデータを、主語, 述語, 目的語 のカンマ区切りデータへ変換する
3. 同時に以下の処理を実施する
4. 述語をノーマルな英単語へ変換
5. 述語がtimeの場合日付と時刻を分離

手法1の処理

1. 共通事前処理で作成した物語毎のデータを、"主語-述語", "述語-目的語", "主語-目的語"の形式へ変換する
2. 変換したデータを物語毎に、「双曲空間への埋め込み」を実施する
3. 分散ベクトルファイルを作成する（タブ区切り形式）
4. Embedding Projectorで作成した分散ベクトルとラベル情報をロードして可視化する

手法2の処理

1. 共通事前処理で作成した物語毎のデータを、"[CLS]主語[SEP]述語[SEP]目的語[SEP]" の形式へ変換する
2. Googleが公開している事前学習データ(xx-largeモデル)をロードする
3. 変換したデータを物語毎に、「ALBERT」で分散表現を作成する
4. 分散ベクトルファイルを作成する（タブ区切り形式）
5. Embedding Projectorで作成した分散ベクトルとラベル情報をロードして可視化する

未実施

1. ナレッジグラフ推論チャレンジタスク用の可視化ツールを開発
2. 個別タスクを実行するベクトル探索+推論を実施(次回チャレンジ)

結果

分散表現をEmbedding ProjectorへロードしてUMAP[4]により可視化した。この可視化結果は、Embedding ProjectorのPublish機能を用いて公開した為、自由に閲覧可能である。さらに、分散表現されたナレッジグラフを操作している様子の動画も同時に公開した。

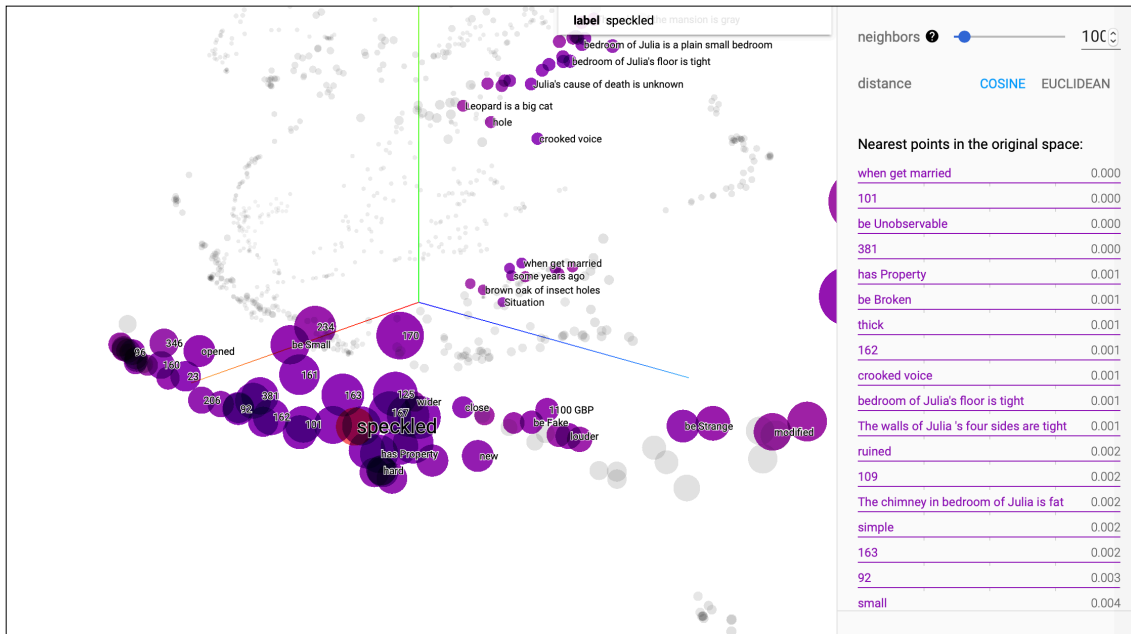
今回、人工知能による探索行動を、「説明性」「解釈性」により評価するに至らなかったが、「可視化ツール」の基礎部分を構築できた。可視化ツール上で動作する人工知能による推論構築とその評価については次回のチャレンジに向けて準備することにした。

次頁にはEmbedding Projectorによる可視化の様子を掲載し、最後に「可視化ツール」の様子を掲載した。

まだらの紐

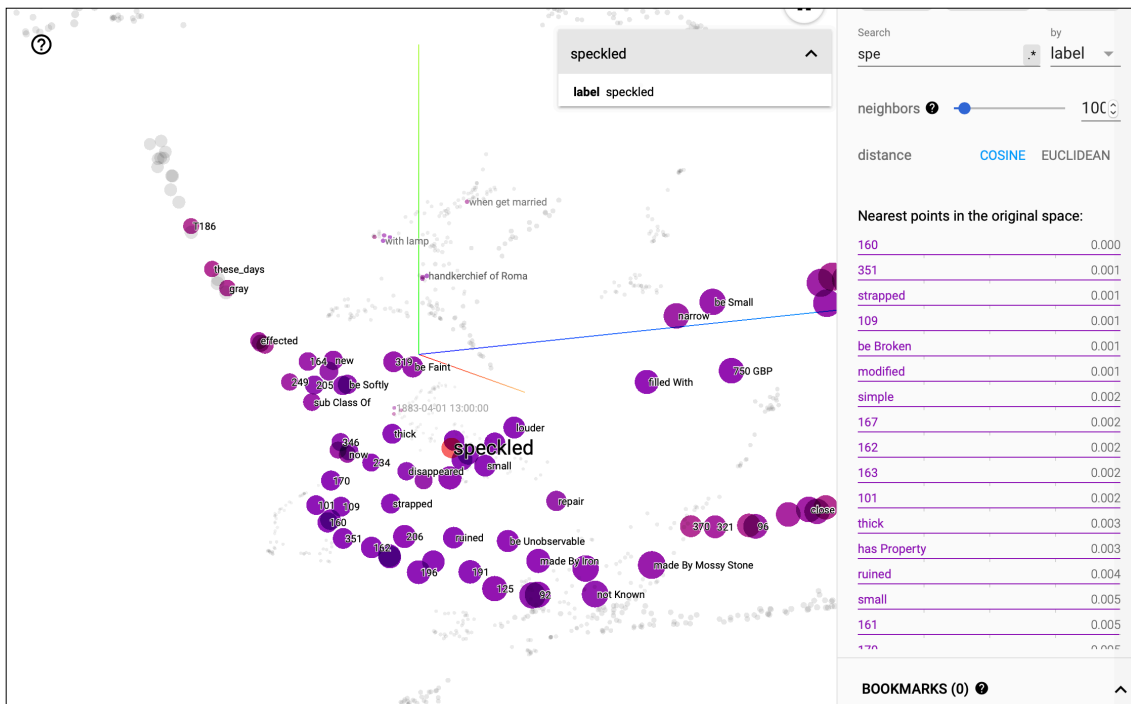
ヘレンを殺したのは誰か？（犯人+説明）

双曲空間版(SOURCE含めて学習)



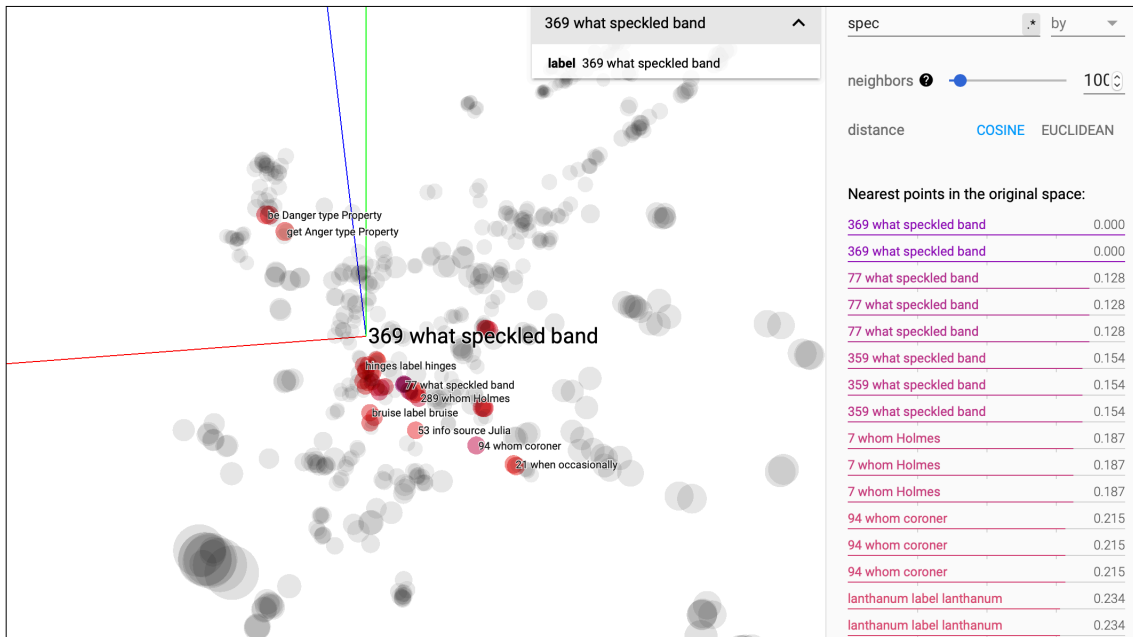
「speckled band」の周辺には、ロイロットに関連する重要な要素が集まっている。

双曲空間版(SOURCEを含めず学習)



sourceを含めたものとは様子が異なっている。

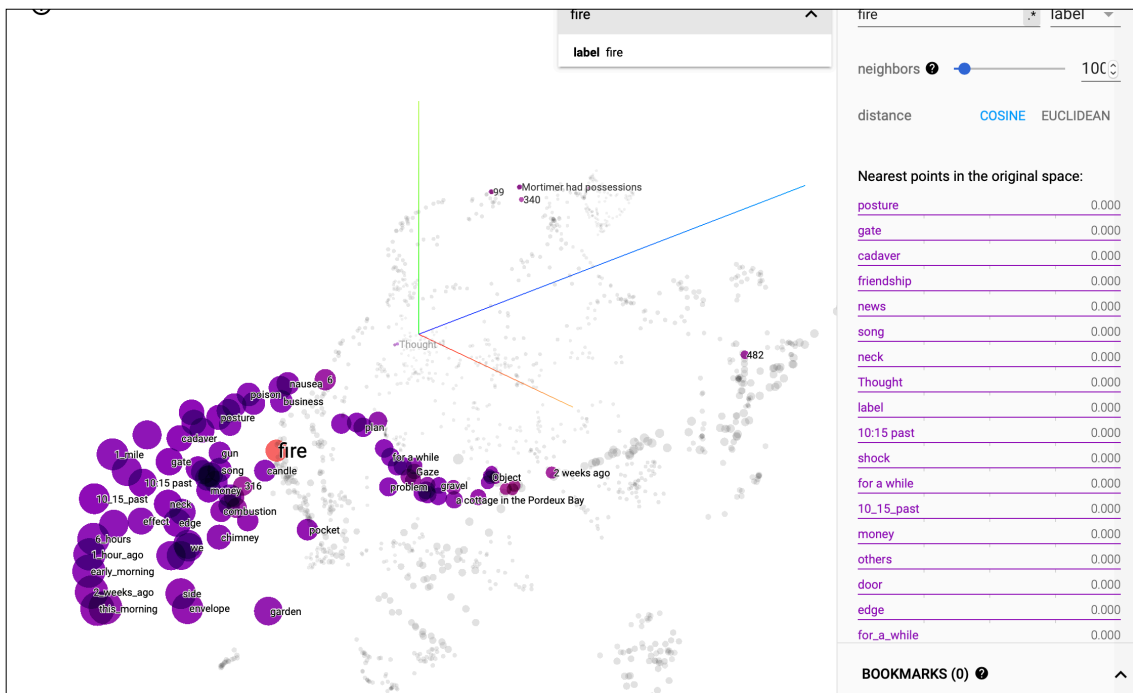
ALBERT版(SOURCE含めて学習)



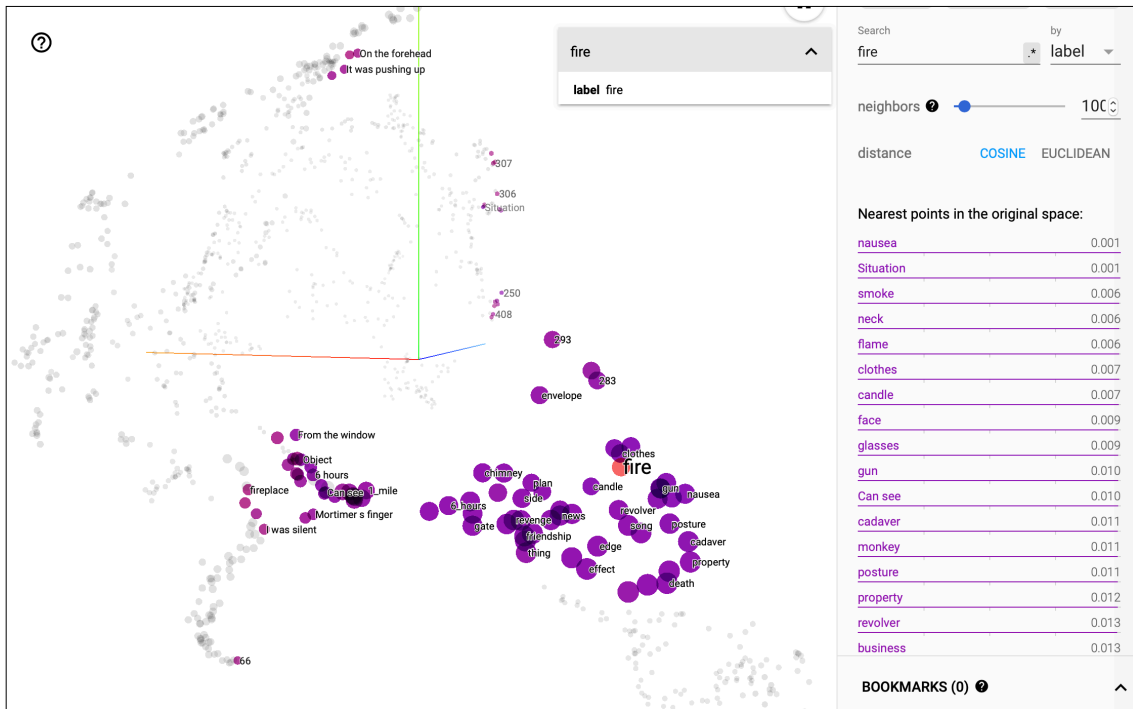
悪魔の足

各人物を殺したのは誰か？（犯人+説明）

双曲空間版(SOURCE含めて学習)

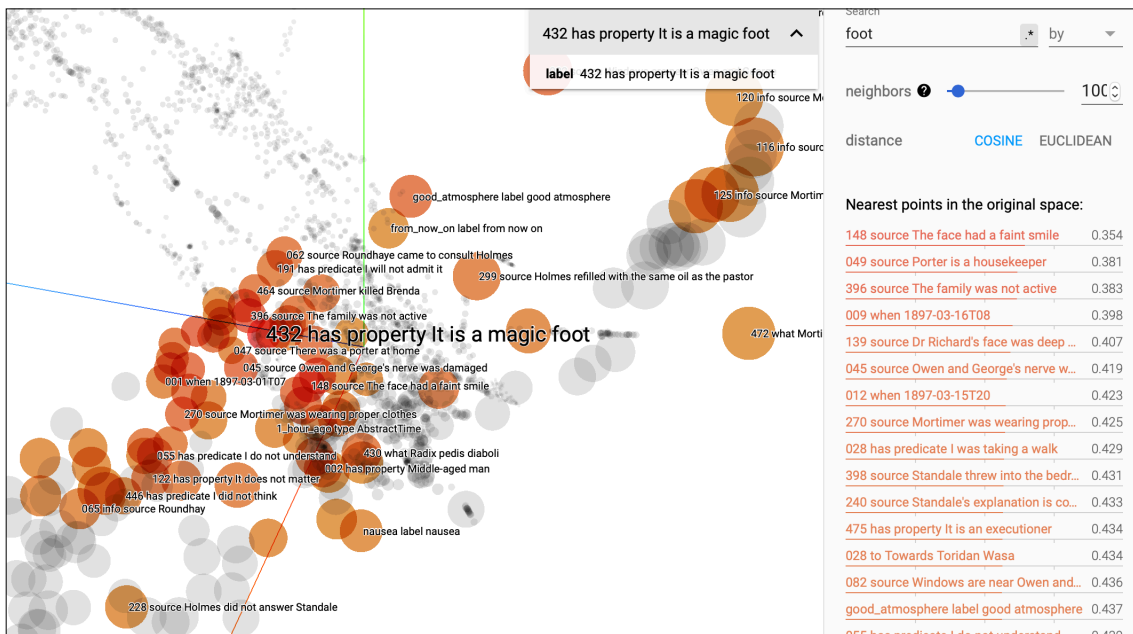


双曲空間版(SOURCEを含めず学習)



近傍を探索することにより、犯人にたどり着くことが期待される。

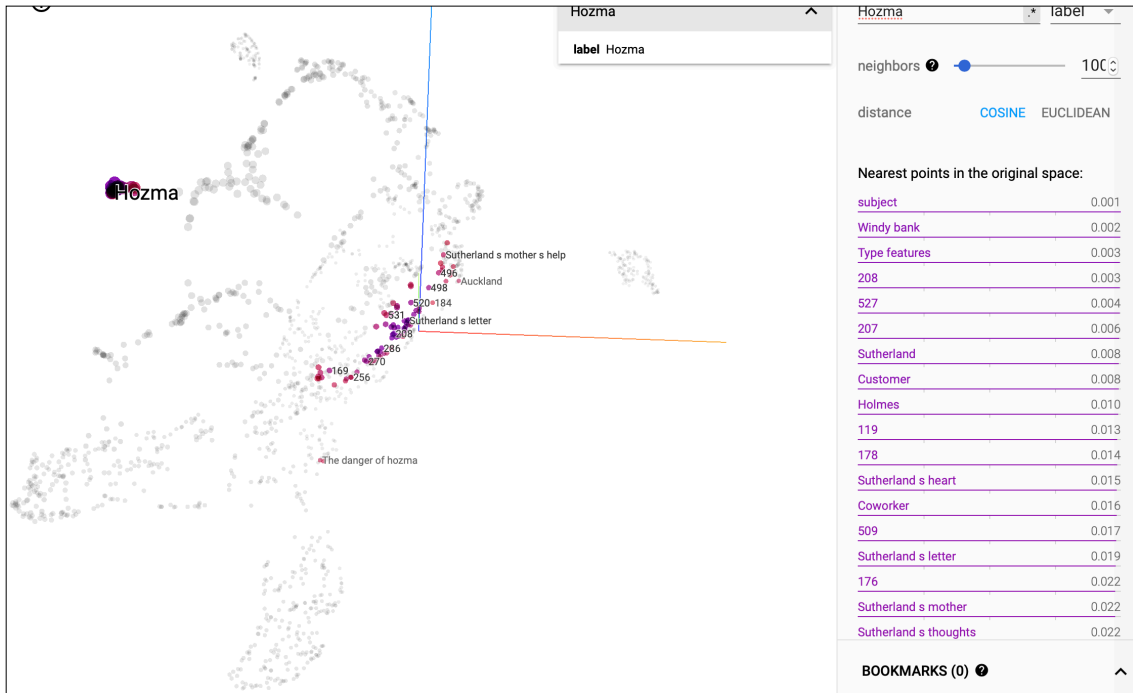
ALBERT版(SOURCE含めて学習)



同一事件

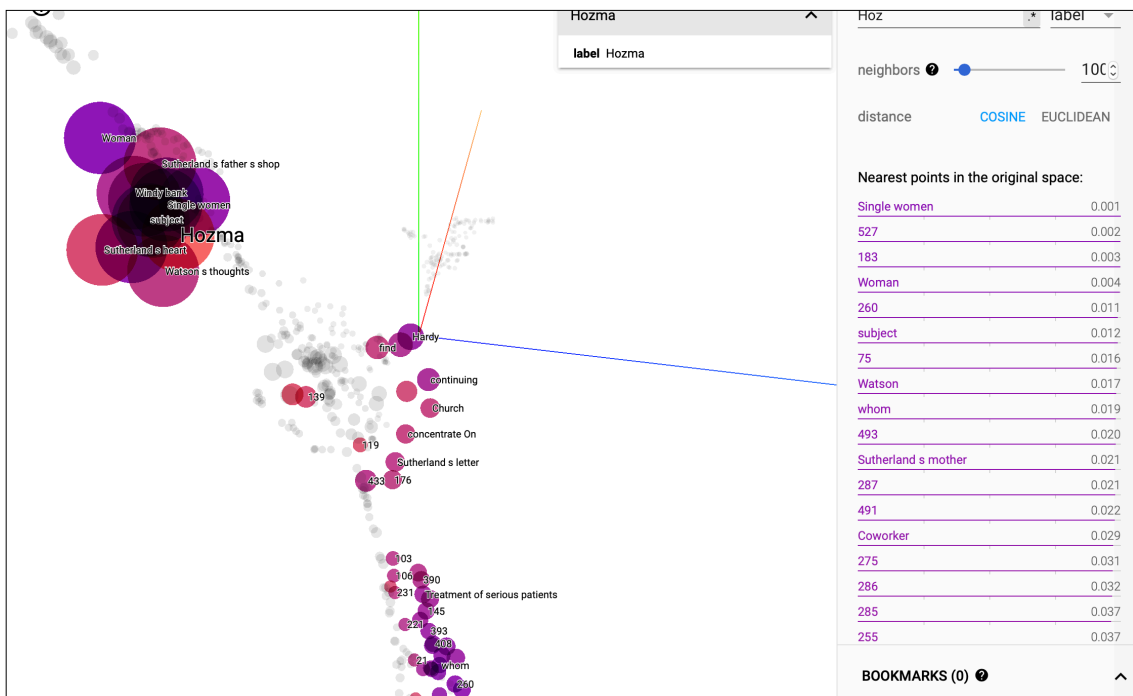
花婿はなぜ消えたか？（説明）

双曲空間版(SOURCE含めて学習)

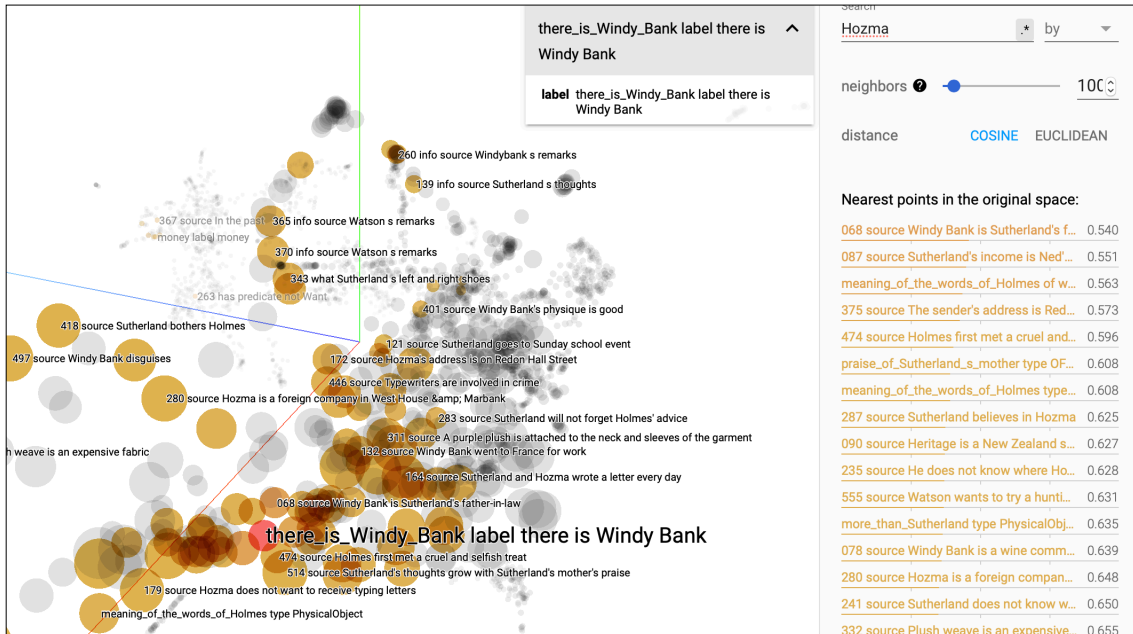


「Hozma」を選択すると「Windy bank」の距離が近いことが観察された。

双曲空間版(SOURCEを含めず学習)



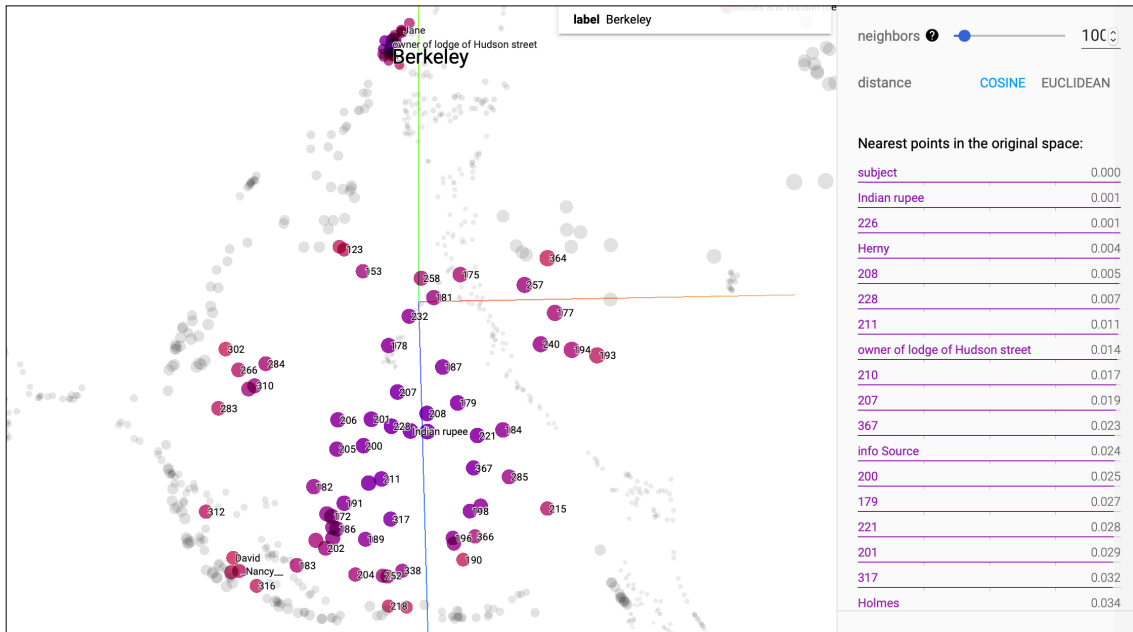
ALBERT版(SOURCE含めて学習)



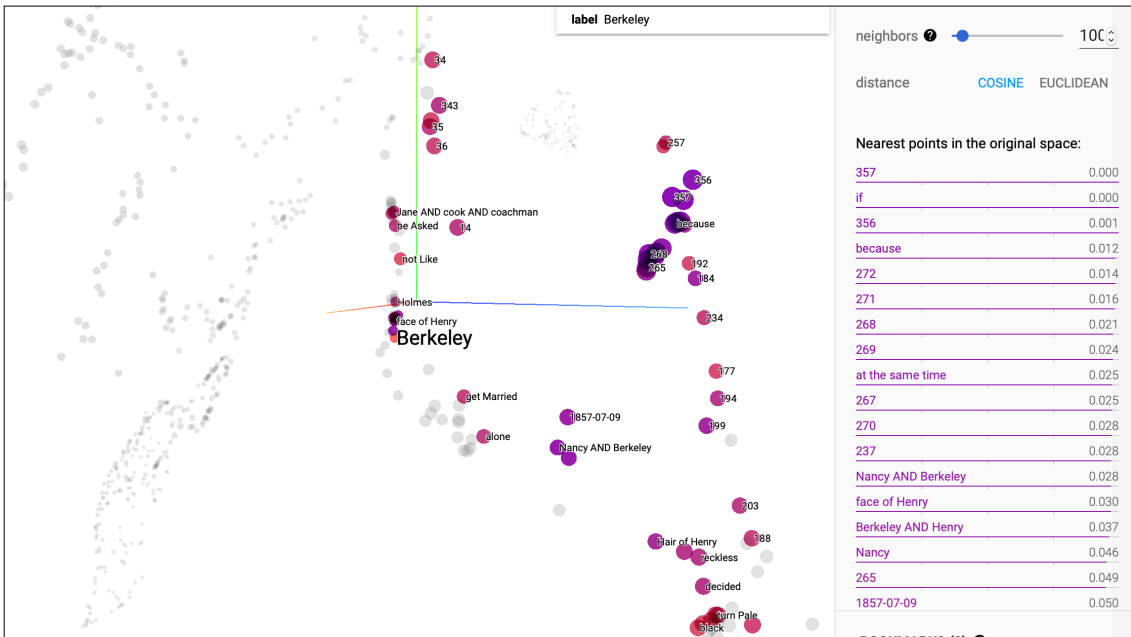
背中の曲がった男

バークリはなぜ死んだのか？（説明）

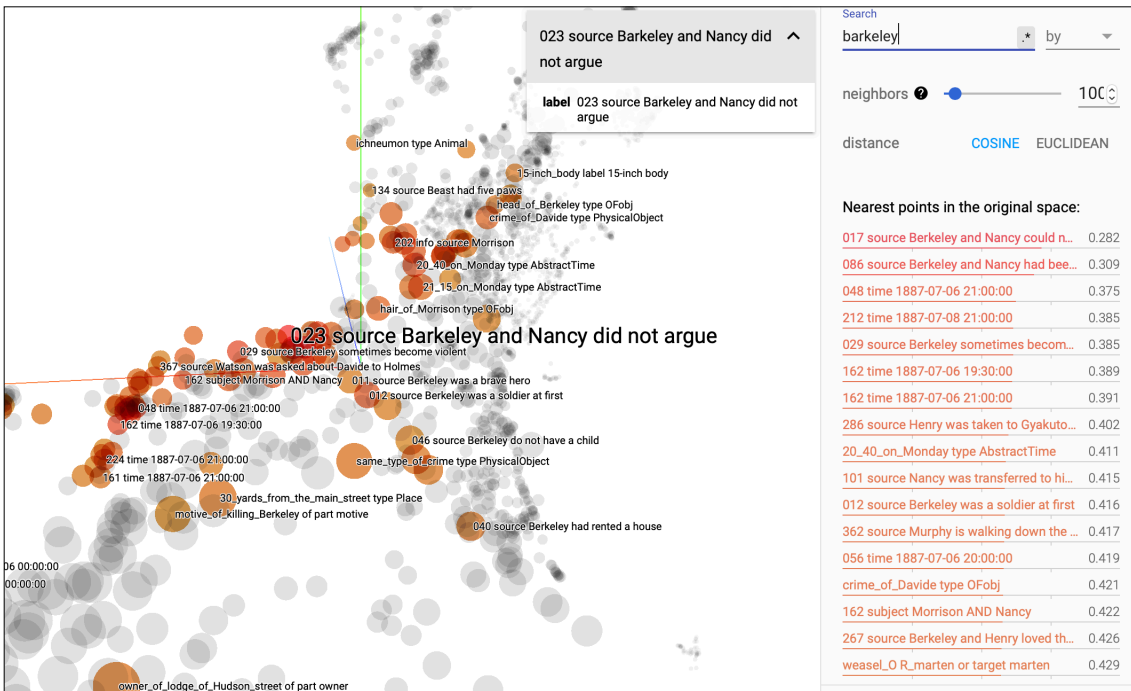
双曲空間版(SOURCE含めて学習)



双曲空間版(SOURCEを含めず学習)



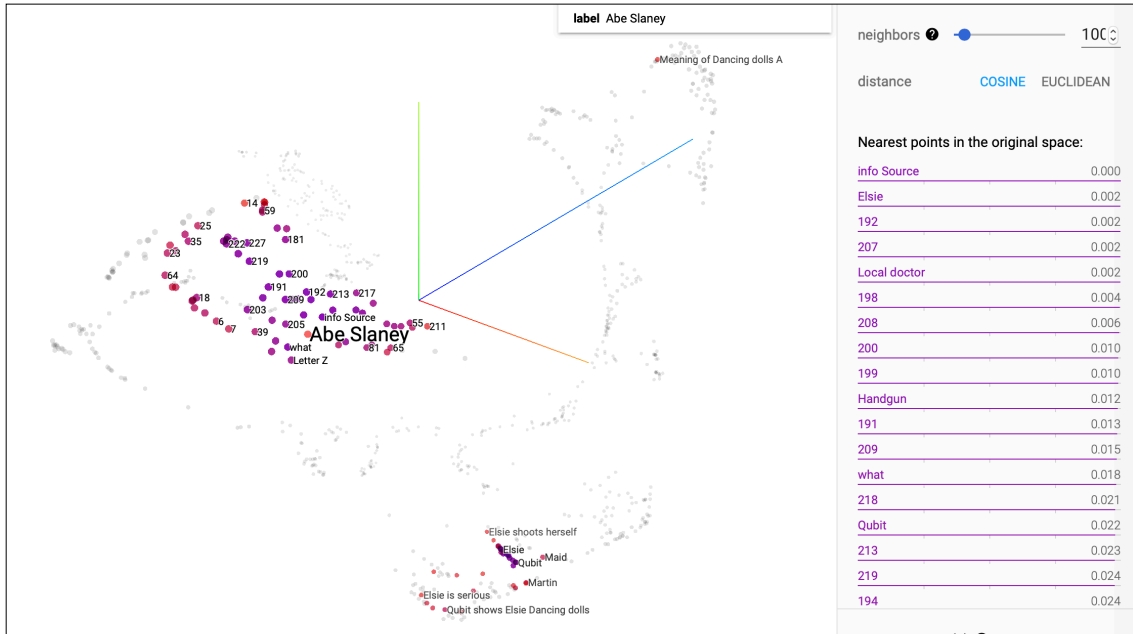
ALBERT版(SOURCE含めて学習)



踊る人形

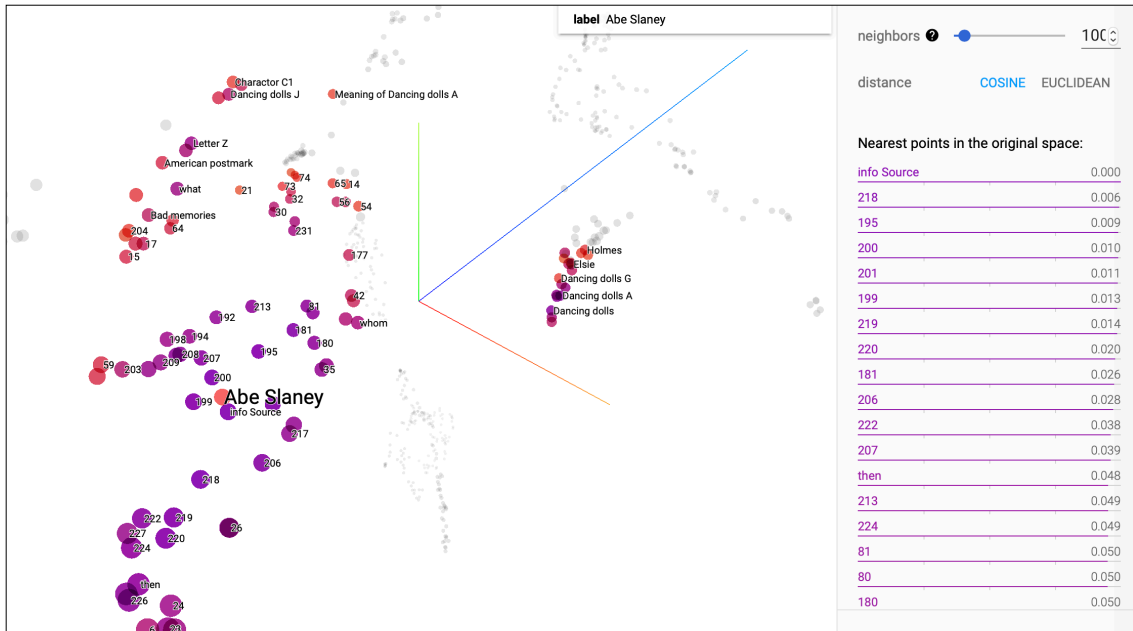
暗号を解け (暗号の解読)

双曲空間版(SOURCE含めて学習)

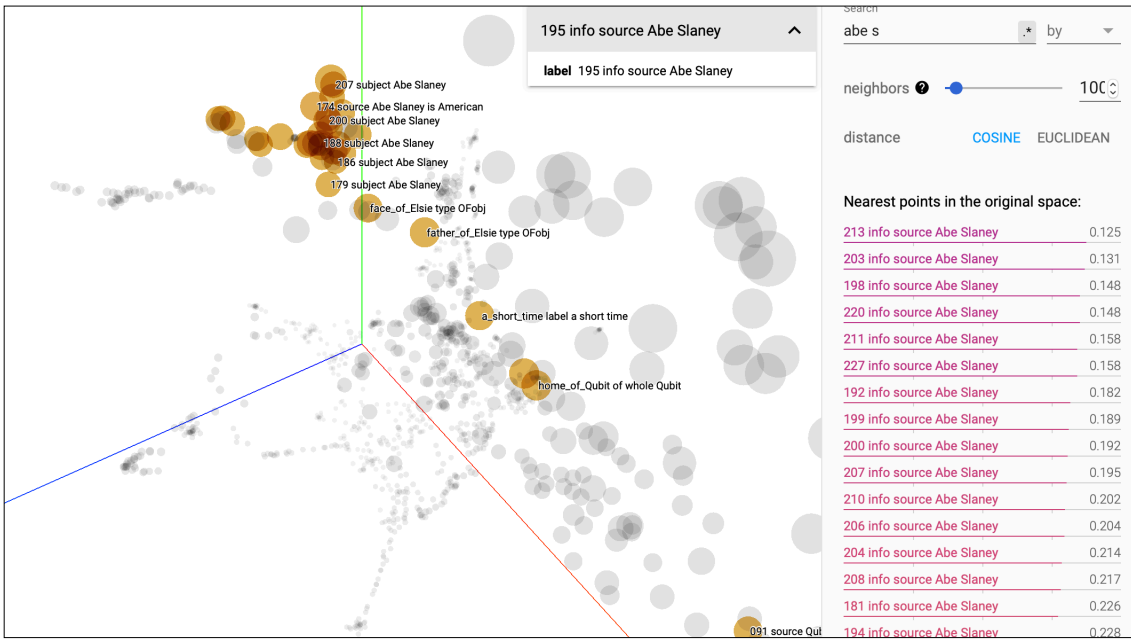


人形の形状から推理することは、本手法では困難であると考える。

双曲空間版(SOURCEを含めず学習)



ALBERT版(SOURCE含めて学習)



双曲空間埋め込み版(SOURCE含めて学習) 分散表現可視化

A CASE OF IDENTITY

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/c5b6306c61f8297ca6230f7945a851a8/raw/4d559f3902e8d2f883c0db89591048e78fd78d62/sherlock2019_a_case_of_identity.json

CROOKED MAN

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/dd2fd9eb3570ff36a0367f9d269f213f/raw/b6bd906eb79a227822b021e0d44c4c998eb267a3/sherlock2019_crooked_man.json

DANCING MEN

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/02650721fe2c0e94ea12438384a7e558/raw/b3171d2879a7e754c2c32be8d67fa6b1558d0f03/sherlock2019_dancing_men.json

DEVILS FOOT

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/f76f0bd06b3410223ae6540fa51626d2/raw/b8c9fbc59f3b0ae4404c6fc393966bb8381b61235/sherlock2019_devils_foot.json

SPECKLED BAND

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/4395b85150ef6b6572655d5a371675e2/raw/f36eab82fdc89ee5505cc50ffd6607385a77bde0/sherlock2019_speckled_band.json

双曲空間埋め込み版(SOURCE除く) 分散表現可視化

A CASE OF IDENTITY

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/28a7e3e18fcb74fe3b302378d761262d/raw/9e2d87f01e1bc7f125d49907176c3fa402decf1e/sherlock2019_a_case_of_identity-2nd.json

CROOKED MAN

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/c1dc6a574be2a817569647770fbcbb03/raw/efc9f86cd342763a3e083ce2e906fd9fe0b0cad0/sherlock2019_crooked_man-2nd.json

DANCING MEN

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/14c6d0b11b7cab0441d54be8818b3062/raw/5f1ebaf21cc789d8171b4dcb775edfd1a128c444/sherlock2019_dancing_men-2nd.json

DEVILS FOOT

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/832364b703f580b53adbede8b255dcb6/raw/96bfb96554a71a7367641de0d6423da1cebe7229/sherlock2019_devils_foot-2nd.json

SPECKLED BAND

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/208adf590ae550518973ec3f4776fb80/raw/f4d6e7b2636fa8e2e650b673b8c118c47b8babdb/sherlock2019_speckled_band-2nd.json

ALBERT版 分散表現可視化

ALBERT版データは、GitHubサイズ制限からアップロードできなかった。GitHub(<https://github.com/sakiyomi-ai/sherlock2019>)からダウンロードしたプログラムにて分散表現とラベルを生成後、Embedding Projectorへブラウザから個別に読み込ませることにより可視化できる。

分散表現を観察している様子の動画

<https://github.com/sakiyomi-ai/sherlock2019/tree/master/movie/poincare/search>

結論

手法1の分散表現は、ナレッジグラフの構造を反映しているように観察された。手法2の分散表現は、自然言語の文により整理されているように観察された。我々は、ナレッジグラフにより表現された知識の活用という観点から手法1に対する探索技術の研究を進めることが有望だと考えるが、定量的な比較は今後の課題として残っている。

時間の制約上、今回は「ナレッジグラフ構造を反映した分散表現に対する探索行動を、「説明性」「解釈性」により評価する」までの研究に至らなかった。しかしながら、探索・推論過程を可視化ツールの構築が進行中である。次回のナレッジグラフ推論チャレンジでは、「双曲空間の埋め込み」により得られた分散表現に対する「探索・推論過程の可視化ツール」及び「人工知能による探索行動の評価」を発表する計画を立てた。

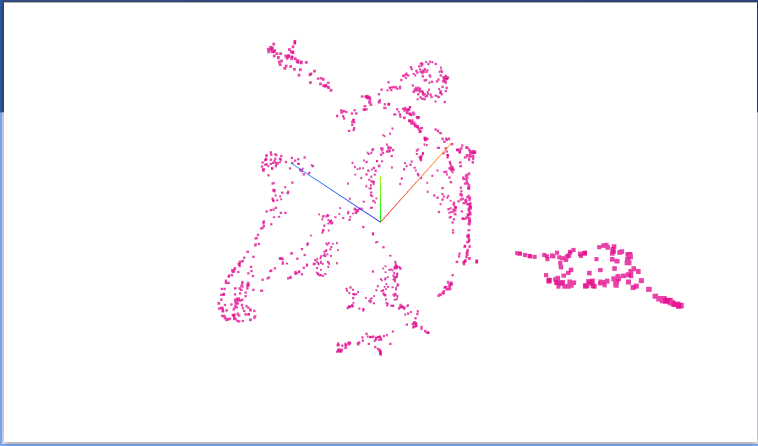
開発した可視化ツール(参考)

本ツールでは、UMAPで次元削減する際の距離を双曲線関数にて計算している。

```
function poincare(x, y) {  
  var euclidean_dists = 0.0;  
  var normX = 0.0;  
  var normY = 0.0;  
  for (var i = 0; i < x.length; i++) {  
    euclidean_dists += Math.pow((x[i] - y[i]), 2);  
    normX += Math.pow(x[i], 2);  
    normY += Math.pow(y[i], 2);  
  }  
  
  return Math.acosh(  
    1 + 2 * (euclidean_dists/((1 - normX) * (1 - normY)))  
  );  
}
```

サキヨミAIラボ

ホーム > シャーロック・ホームズ / 詳細



データセット

エージェントモデル

ログ

episode: A CASE OF IDENTITY

subject	predicate	object
001	from	Chair
001	has Predicate	stand
001	source	Holmes stood out of a chair
001	subject	Holmes
001	time	1891-09-01 10:00:00
001	type	Situation
001	when	1891-09-01 10
002	has Predicate	stand
002	source	Holmes stood between the windowed seams
002	type	Situation

サキヨミAIラボ
人工知能で幸せな社会を創る

© All Rights Reserved

参考文献

- [1] Maximilian Nickel, Douwe Kiela, "Poincaré Embeddings for Learning Hierarchical Representations", arXiv:1705.08039v2 (2017)
- [2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", arXiv:1810.04805v2 (2019)
- [3] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, Radu Soricut, "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations", arXiv:1909.11942(2019)
- [4] Leland McInnes, John Healy, James Melville, "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction", arXiv:1802.03426v2

参考サイト

- [5] <https://www.w3.org/TR/turtle/>
"Terse RDF Triple Language", W3C Recommendation 25 February 2014
- [6] <https://projector.tensorflow.org>

パフォーマンス情報

参考

- ・ 実行環境: MacBook Pro
- ・ メインメモリ: 16G

参考情報

ホームページ

<https://sakiyomi.ai>

論文

- ・ 2019年度 人工知能学会全国大会(第33回) [2C1-J-12-02] Decision Making for Model Based Design by Reinforcement Learning (酒井竜英 (長城汽車股份) ・伊鍋貴弘 (サキヨミAIラボ))
- ・ 自動車技術会 [20195422] 車両性能とパッケージを設計するNeural Networkの開発 (酒井竜英 (長城汽車股份) ・伊鍋貴宏 (サキヨミAIラボ))

3. 実行プログラム

今回作成したプログラム及びプログラムが生成したデータは、GitHubにて公開した。

ダウンロード方法

プログラム、データ

<https://github.com/sakiyomi-ai/sherlock2019>

実行方法

STEP1. 準備

STEP1-1. dockerイメージ作成

GitHubからダウンロードしたDockerfileと同じディレクトリへ移動後、Dockerイメージをビルドする。

コマンド例)

```
docker build -t tensor-keras .
```

STEP1-2. dockerコンテナ起動

GitHubからダウンロードしたjupyterノートブックと同じディレクトリへ移動後、Dockerコンテナを起動する。

コマンド例)

```
docker run -it --rm -v $(pwd):/tf/notebooks -p 8888:8888 tensor-keras
```

STEP1-3. jupyter notebookへアクセス

ターミナルに表示されたURLをコピーしてブラウザでアクセスする。

例) <http://>

127.0.0.1:8888/?token=0b80550899c16ce9112d608daf8e00a66afc2d391e8e8200

STEP2. PythonでRDFを物語毎に抽出する

ノートブック

- ・ pre-process.ipynb

入力ファイル

- ・ kgc2019.nt

出力ファイル

- ACaseOfIdentity.txt
- CrookedMan.txt
- DancingMen.txt
- DevilsFoot.txt
- SpeckledBand.txt

STEP3. 「述語をノーマルな英単語へ変換」、 「述語がtimeの場合日付と時刻を分離」 処理をする

ノートブック

- convert_sentence.ipynb

入力ファイル

- ACaseOfIdentity.txt
- CrookedMan.txt
- DancingMen.txt
- DevilsFoot.txt
- SpeckledBand.txt

中間ファイル

- ACaseOfIdentity_formatted.txt
- CrookedMan_formatted.txt
- DancingMen_formatted.txt
- DevilsFoot_formatted.txt
- SpeckledBand_formatted.txt

出力ファイル

- ACaseOfIdentity_formatted_S2.txt
- CrookedMan_formatted_S2.txt
- DancingMen_formatted_S2.txt
- DevilsFoot_formatted_S2.txt
- SpeckledBand_formatted_S2.txt
- ACaseOfIdentity-error.txt
- CrookedMan-error.txt
- DancingMen-error.txt
- DevilsFoot-error.txt
- SpeckledBand-error.txt

STEP 4. 双曲空間への畳み込みによる分散表現を作成する

STEP 4.1. 双曲空間への畳み込み用データを作成

ノートブック

- triplet2pair.ipynb

入力ファイル

- ACaseOfIdentity_formatted_S2.txt
- CrookedMan_formatted_S2.txt

- DancingMen_formatted_S2.txt
- DevilsFoot_formatted_S2.txt
- SpeckledBand_formatted_S2.txt

出力ファイル

- ACaseOfIdentity_formatted_S2-pair.txt
- CrookedMan_formatted_S2-pair.txt
- DancingMen_formatted_S2-pair.txt
- DevilsFoot_formatted_S2-pair.txt
- SpeckledBand_formatted_S2-pair.txt
- ACaseOfIdentity_formatted_S2-poincare-label.tsv
- CrookedMan_formatted_S2-poincare-label.tsv
- DancingMen_formatted_S2-poincare-label.tsv
- DevilsFoot_formatted_S2-poincare-label.tsv
- SpeckledBand_formatted_S2-poincare-label.tsv

STEP 4. 2. 双曲空間への畳み込みで分散表現を作成

ノートブック

- poincare-model.ipynb

入力ファイル

- ACaseOfIdentity_formatted_S2-pair.txt
- CrookedMan_formatted_S2-pair.txt
- DancingMen_formatted_S2-pair.txt
- DevilsFoot_formatted_S2-pair.txt
- SpeckledBand_formatted_S2-pair.txt
- ACaseOfIdentity_formatted_S2-poincare-label.tsv
- CrookedMan_formatted_S2-poincare-label.tsv
- DancingMen_formatted_S2-poincare-label.tsv
- DevilsFoot_formatted_S2-poincare-label.tsv
- SpeckledBand_formatted_S2-poincare-label.tsv

出力ファイル

- ACaseOfIdentity_formatted_S2-poincare-3d-vector.tsv
- CrookedMan_formatted_S2-poincare-3d-vector.tsv
- DancingMen_formatted_S2-poincare-3d-vector.tsv
- DevilsFoot_formatted_S2-poincare-3d-vector.tsv
- SpeckledBand_formatted_S2-poincare-3d-vector.tsv
- ACaseOfIdentity_formatted_S2-poincare-3d.model
- CrookedMan_formatted_S2-poincare-3d.model
- DancingMen_formatted_S2-poincare-3d.model
- DevilsFoot_formatted_S2-poincare-3d.model
- SpeckledBand_formatted_S2-poincare-3d.model

STEP 4. 3. Embedding Projectorへ読み込ませる

<https://projector.tensorflow.org>

STEP5.. ALBERTモデルで分散表現を作成

STEP5. 5. 1. ALBERTモデル用データを作成する

ノートブック

- bert-model-A.ipynb

入力ファイル

- ACaseOfIdentity_formatted_S2.txt
- CrookedMan_formatted_S2.txt
- DancingMen_formatted_S2.txt
- DevilsFoot_formatted_S2.txt
- SpeckledBand_formatted_S2.txt

出力ファイル

- ACaseOfIdentity_formatted_S2-albert_xxlarge-p2.npz
- CrookedMan_formatted_S2-albert_xxlarge-p2.npz
- DancingMen_formatted_S2-albert_xxlarge-p2.npz
- DevilsFoot_formatted_S2-albert_xxlarge-p2.npz
- SpeckledBand_formatted_S2-albert_xxlarge-p2.npz

STEP5. 2. ALBERTの分散表現をタブ区切りファイルへ変換する

ノートブック

- TSV_file_convert.ipynb

入力ファイル

- ACaseOfIdentity_formatted_S2-albert_xxlarge-p2.npz
- CrookedMan_formatted_S2-albert_xxlarge-p2.npz
- DancingMen_formatted_S2-albert_xxlarge-p2.npz
- DevilsFoot_formatted_S2-albert_xxlarge-p2.npz
- SpeckledBand_formatted_S2-albert_xxlarge-p2.npz

出力ファイル

- ACaseOfIdentity_formatted_S2-albert_xxlarge-p2-vector.tsv
- CrookedMan_formatted_S2-albert_xxlarge-p2-vector.tsv
- DancingMen_formatted_S2-albert_xxlarge-p2-vector.tsv
- DevilsFoot_formatted_S2-albert_xxlarge-p2-vector.tsv
- SpeckledBand_formatted_S2-albert_xxlarge-p2-vector.tsv
- ACaseOfIdentity_formatted_S2-albert_xxlarge-p2-label.tsv
- CrookedMan_formatted_S2-albert_xxlarge-p2-label.tsv
- DancingMen_formatted_S2-albert_xxlarge-p2-label.tsv
- DevilsFoot_formatted_S2-albert_xxlarge-p2-label.tsv
- SpeckledBand_formatted_S2-albert_xxlarge-p2-label.tsv

STEP5. 3. Embedding Projectorへ読み込ませる

<https://projector.tensorflow.org>

既存ツール

- ・ docker desktop community 19.03.5 (dockerコンテナが動作する環境であれば実行可能)
- ・ Embedding Projector (<https://projector.tensorflow.org>)
- ・ jupyter (<https://jupyter.org>)
- ・ tensorflow
- ・ keras
- ・ rdflib
- ・ sentencepiece
- ・ pandas
- ・ bert-for-tf2
- ・ tqdm
- ・ gensim
- ・ plotly

4. 資料の共有について

応募フォーム

公開の可否：

- 公開してよい
- 非公開とする

公開形式：

- ナレッジグラフ推論チャレンジのサイトで公開
- 独自のサイトで公開してリンクを希望

応募したプログラム、データ等

公開の可否：

- 公開してよい
- 非公開とする

公開形式：

- ナレッジグラフ推論チャレンジのサイトで公開
- 独自のサイトで公開してリンクを希望

5. 応募資料提出後の作業

sourceを含まない分散表現版を作成

双曲空間埋め込み版(source除く) 分散表現可視化

A CASE OF IDENTITY

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/28a7e3e18fcb74fe3b302378d761262d/raw/9e2d87f01e1bc7f125d49907176c3fa402decf1e/sherlock2019_a_case_of_identity-2nd.json

CROOKED MAN

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/c1dc6a574be2a817569647770fbcbb03/raw/efc9f86cd342763a3e083ce2e906fd9fe0b0cad0/sherlock2019_crooked_man-2nd.json

DANCING MEN

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/14c6d0b11b7cab0441d54be8818b3062/raw/5f1ebaf21cc789d8171b4dcb775edfd1a128c444/sherlock2019_dancing_men-2nd.json

DEVILS FOOT

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/832364b703f580b53adbede8b255dcb6/raw/96bfb96554a71a7367641de0d6423da1cebe7229/sherlock2019_devils_foot-2nd.json

SPECKLED BAND

https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/208adf590ae550518973ec3f4776fb80/raw/f4d6e7b2636fa8e2e650b673b8c118c47b8babdb/sherlock2019_speckled_band-2nd.json