
2019
KGC: Knowledge Graph Reasoning Challenge
ナレッジグラフ推論チャレンジ
株式会社サキヨミAIラボ

はじめに

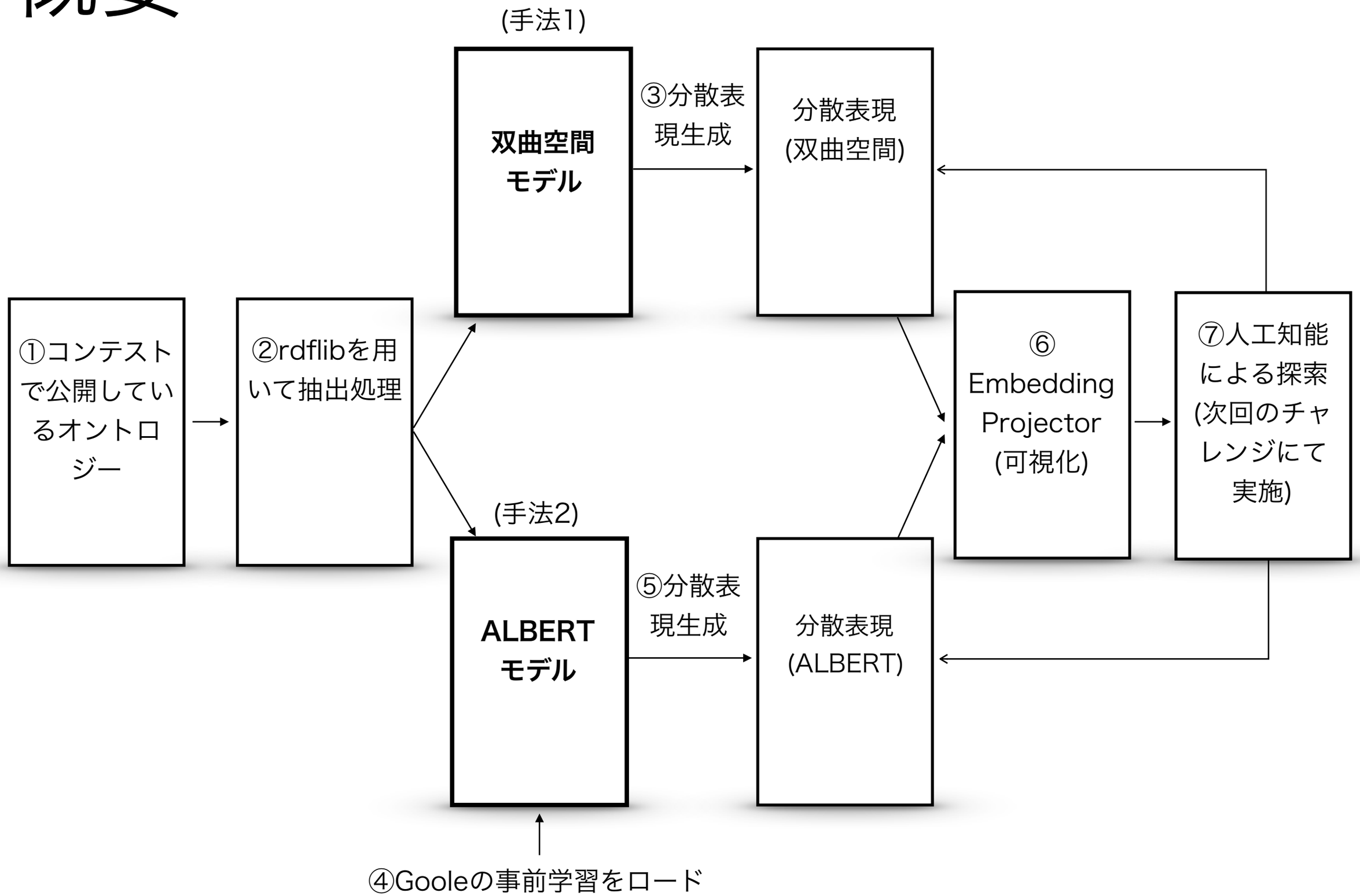
ナレッジグラフ

人間の持つ知識を人工知能に伝えるための重要な手段

人工知能推論

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。

概要

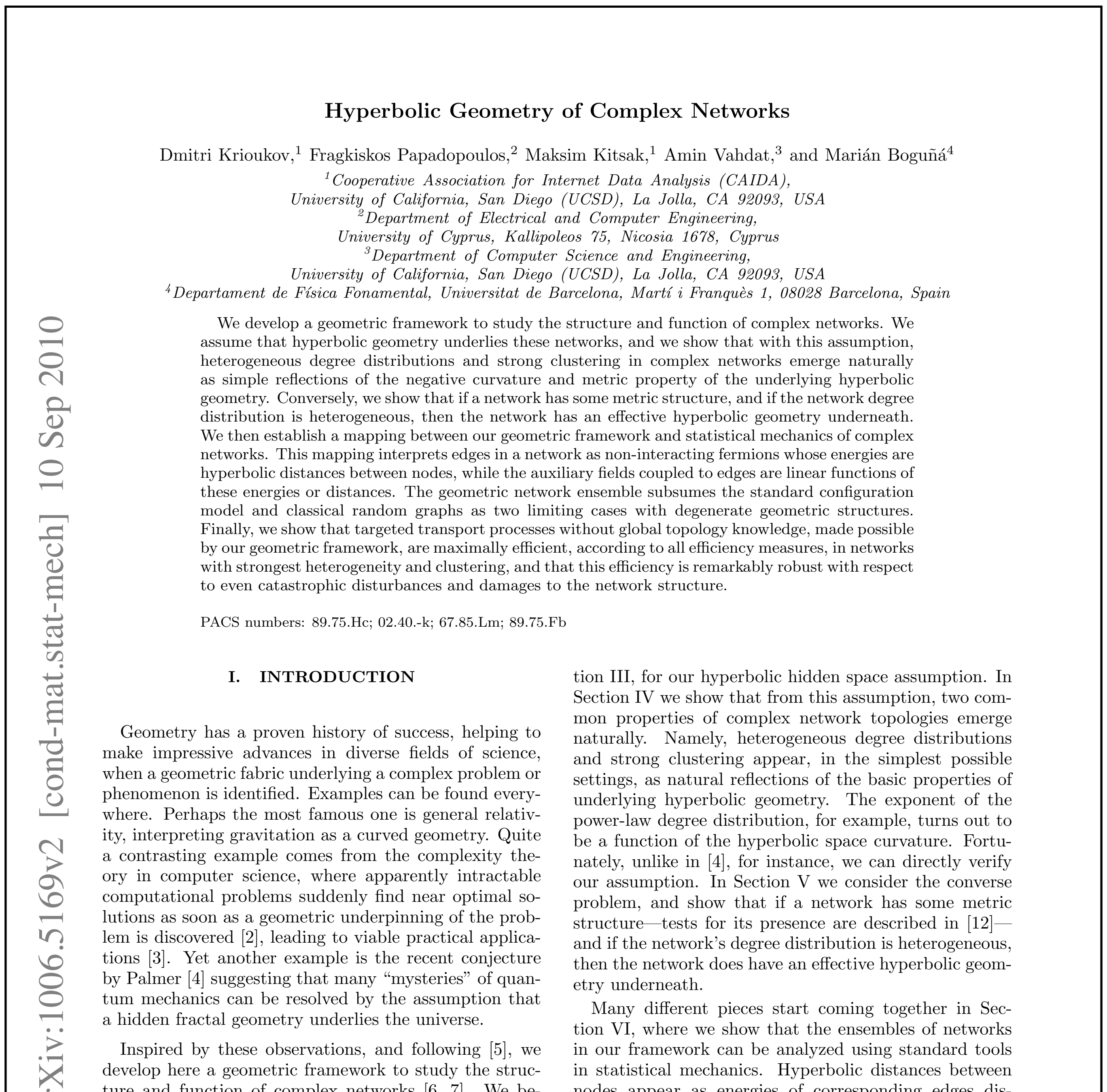

(手法1)
双曲空間
モデル
③分散表
現生成
分散表現
(双曲空間)
①コンテスト
で公開してい
るオントロ
ジー
②rdflibを用
いて抽出処理
⑥
Embedding
Projector
(可視化)
⑦人工知能
による探索
(次回のチャ
レンジにて
実施)
(手法2)
ALBERT
モデル
⑤分散表
現生成
分散表現
(ALBERT)
④Gooleの事前学習をロード

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」[1,5]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。

Hyperbolic Geometry of Complex Networks

Krioukov, Dmitri et al., "Hyperbolic geometry of complex networks", Physical Review 2010



Hyperbolic Geometry of Complex Networks

Dmitri Krioukov,[1] Fragkiskos Papadopoulos,[2] Maksim Kitsak,[1] Amin Vahdat,[3] and Marián Boguñá[4]
[1]*Cooperative Association for Internet Data Analysis (CAIDA), University of California, San Diego (UCSD), La Jolla, CA 92093, USA*
[2]*Department of Electrical and Computer Engineering, University of Cyprus, Kallipoleos 75, Nicosia 1678, Cyprus*
[3]*Department of Computer Science and Engineering, University of California, San Diego (UCSD), La Jolla, CA 92093, USA*
[4]*Departament de Física Fonamental, Universitat de Barcelona, Martí i Franquès 1, 08028 Barcelona, Spain*

We develop a geometric framework to study the structure and function of complex networks. We assume that hyperbolic geometry underlies these networks, and we show that with this assumption, heterogeneous degree distributions and strong clustering in complex networks emerge naturally as simple reflections of the negative curvature and metric property of the underlying hyperbolic geometry. Conversely, we show that if a network has some metric structure, and if the network degree distribution is heterogeneous, then the network has an effective hyperbolic geometry underneath. We then establish a mapping between our geometric framework and statistical mechanics of complex networks. This mapping interprets edges in a network as non-interacting fermions whose energies are hyperbolic distances between nodes, while the auxiliary fields coupled to edges are linear functions of these energies or distances. The geometric network ensemble subsumes the standard configuration model and classical random graphs as two limiting cases with degenerate geometric structures. Finally, we show that targeted transport processes without global topology knowledge, made possible by our geometric framework, are maximally efficient, according to all efficiency measures, in networks with strongest heterogeneity and clustering, and that this efficiency is remarkably robust with respect to even catastrophic disturbances and damages to the network structure.

PACS numbers: 89.75.Hc; 02.40.-k; 67.85.Lm; 89.75.Fb

I. INTRODUCTION

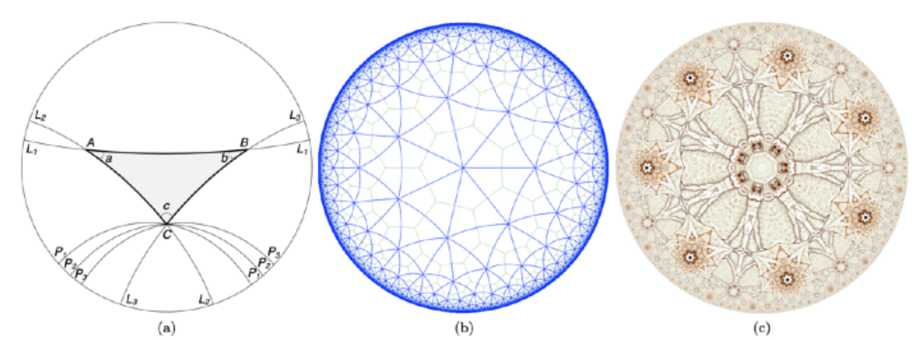
Geometry has a proven history of success, helping to make impressive advances in diverse fields of science, when a geometric fabric underlying a complex problem or phenomenon is identified. Examples can be found everywhere. Perhaps the most famous one is general relativity, interpreting gravitation as a curved geometry. Quite a contrasting example comes from the complexity theory in computer science, where apparently intractable computational problems suddenly find near optimal solutions as soon as a geometric underpinning of the problem is discovered [2], leading to viable practical applications [3]. Yet another example is the recent conjecture by Palmer [4] suggesting that many "mysteries" of quantum mechanics can be resolved by the assumption that a hidden fractal geometry underlies the universe.

Inspired by these observations, and following [5], we develop here a geometric framework to study the structure and function of complex networks [6, 7]. We be-

tion III, for our hyperbolic hidden space assumption. In Section IV we show that from this assumption, two common properties of complex network topologies emerge naturally. Namely, heterogeneous degree distributions and strong clustering appear, in the simplest possible settings, as natural reflections of the basic properties of underlying hyperbolic geometry. The exponent of the power-law degree distribution, for example, turns out to be a function of the hyperbolic space curvature. Fortunately, unlike in [4], for instance, we can directly verify our assumption. In Section V we consider the converse problem, and show that if a network has some metric structure—tests for its presence are described in [12]—and if the network's degree distribution is heterogeneous, then the network does have an effective hyperbolic geometry underneath.

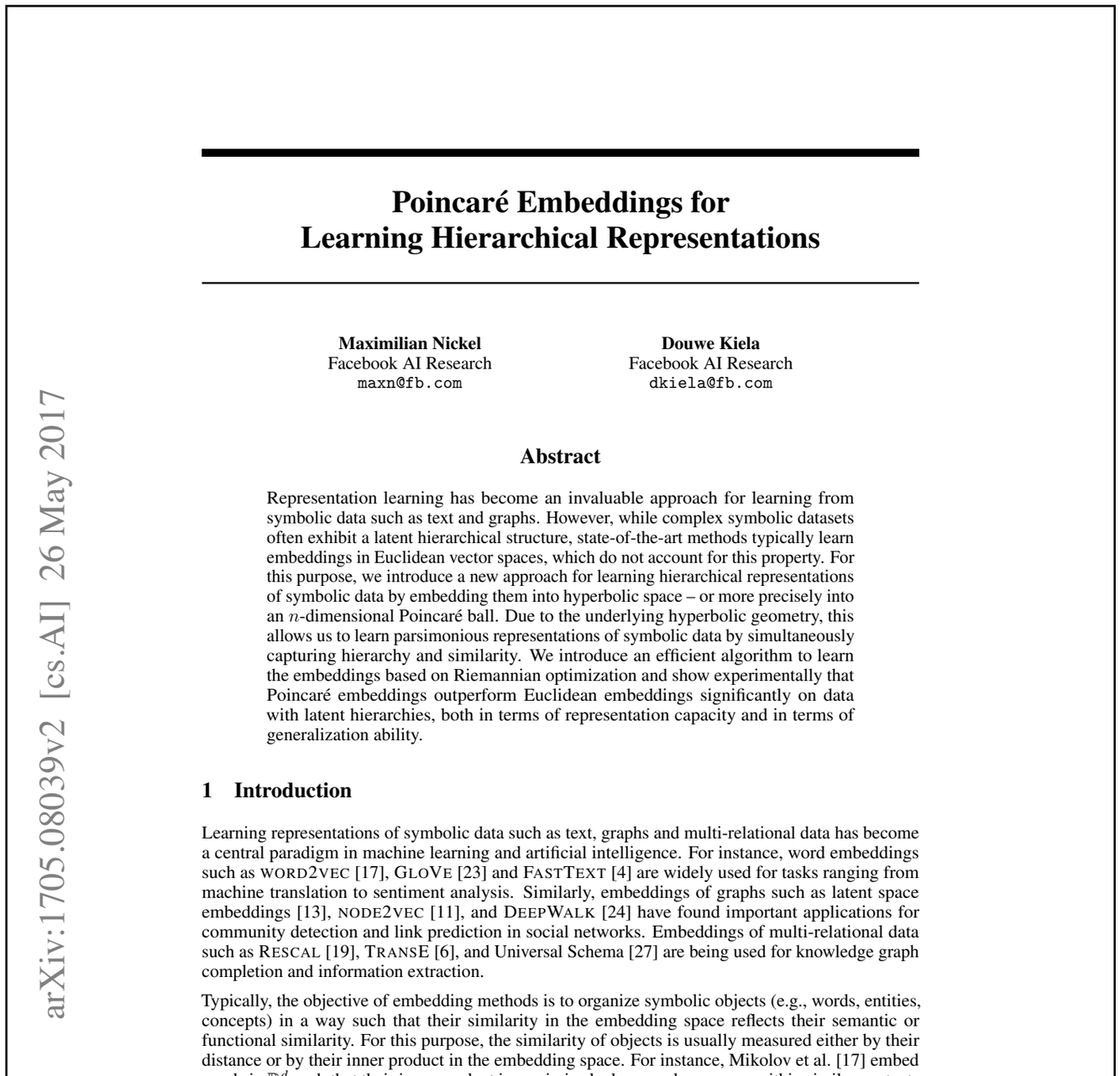
Many different pieces start coming together in Section VI, where we show that the ensembles of networks in our framework can be analyzed using standard tools in statistical mechanics. Hyperbolic distances between nodes appear as energies of corresponding edges dis-

- 双曲空間には木構造を自然な形で埋め込むことができるという特殊な性質が知られており、木構造を構成するノード間の距離を保つように、適当な次元の双極空間へ埋め込むことができる



Poincare Embeddings

Maximilian Nickel, Douwe Kiela, "Poincaré Embeddings for Learning Hierarchical Representations", arXiv:1705.08039v2 (2017)



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.

1 Introduction

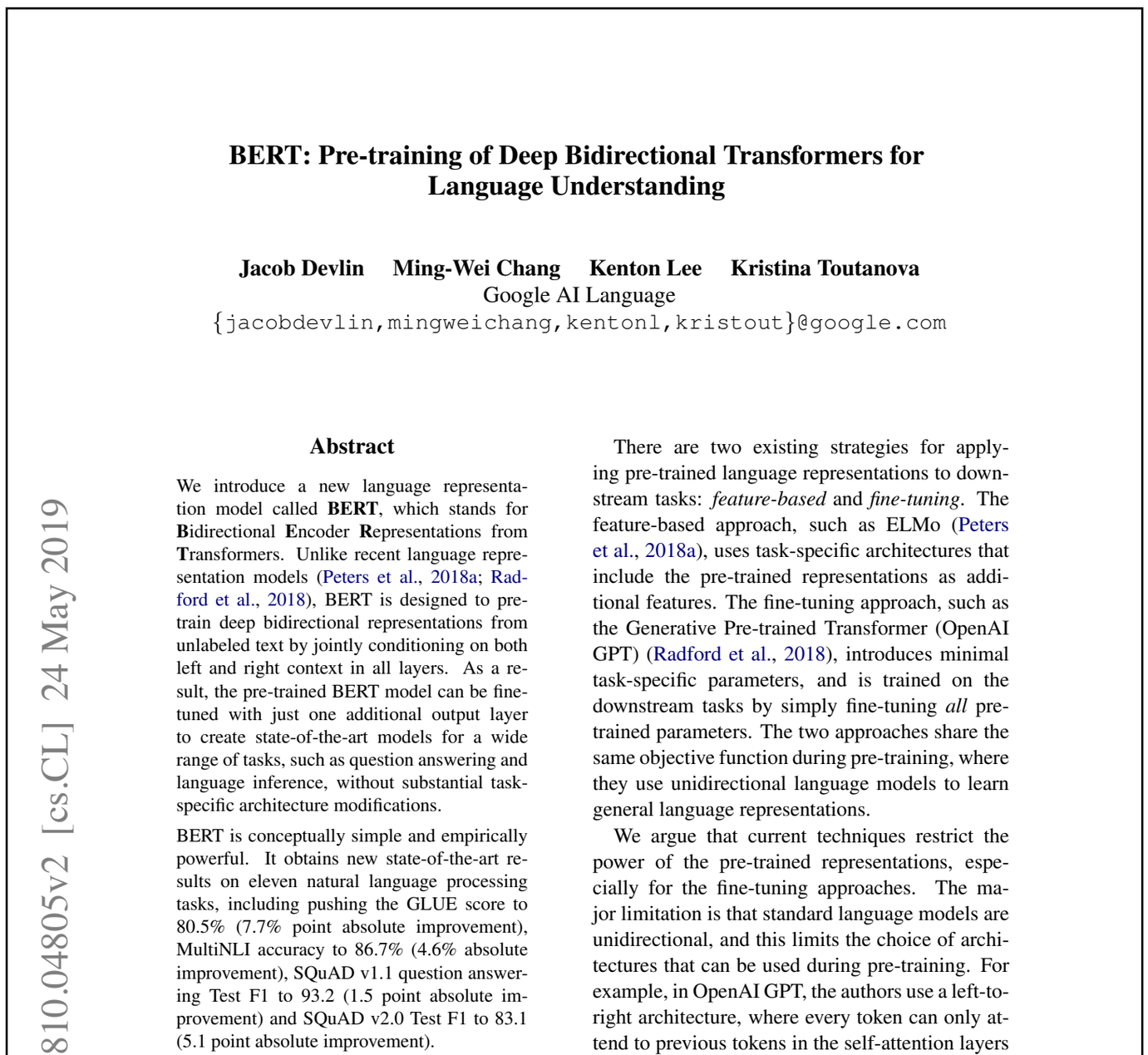
Learning representations of symbolic data such as text, graphs and multi-relational data has become a central paradigm in machine learning and artificial intelligence. For instance, word embeddings such as WORD2VEC [17], GLOVE [23] and FASTTEXT [4] are widely used for tasks ranging from machine translation to sentiment analysis. Similarly, embeddings of graphs such as latent space embeddings [13], NODE2VEC [11], and DEEPWALK [24] have found important applications for community detection and link prediction in social networks. Embeddings of multi-relational data such as RESCAL [19], TRANSE [6], and Universal Schema [27] are being used for knowledge graph completion and information extraction.

Typically, the objective of embedding methods is to organize symbolic objects (e.g., words, entities, concepts) in a way such that their similarity in the embedding space reflects their semantic or functional similarity. For this purpose, the similarity of objects is usually measured either by their distance or by their inner product in the embedding space. For instance, Mikolov et al. [17] embed

- **ユークリッド空間への埋込に比べて、空間を指数関数的に効率よく利用できる**

BERT

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", arXiv:1810.04805v2 (2019)



BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin **Ming-Wei Chang** **Kenton Lee** **Kristina Toutanova**
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

Abstract

We introduce a new language representation model called **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

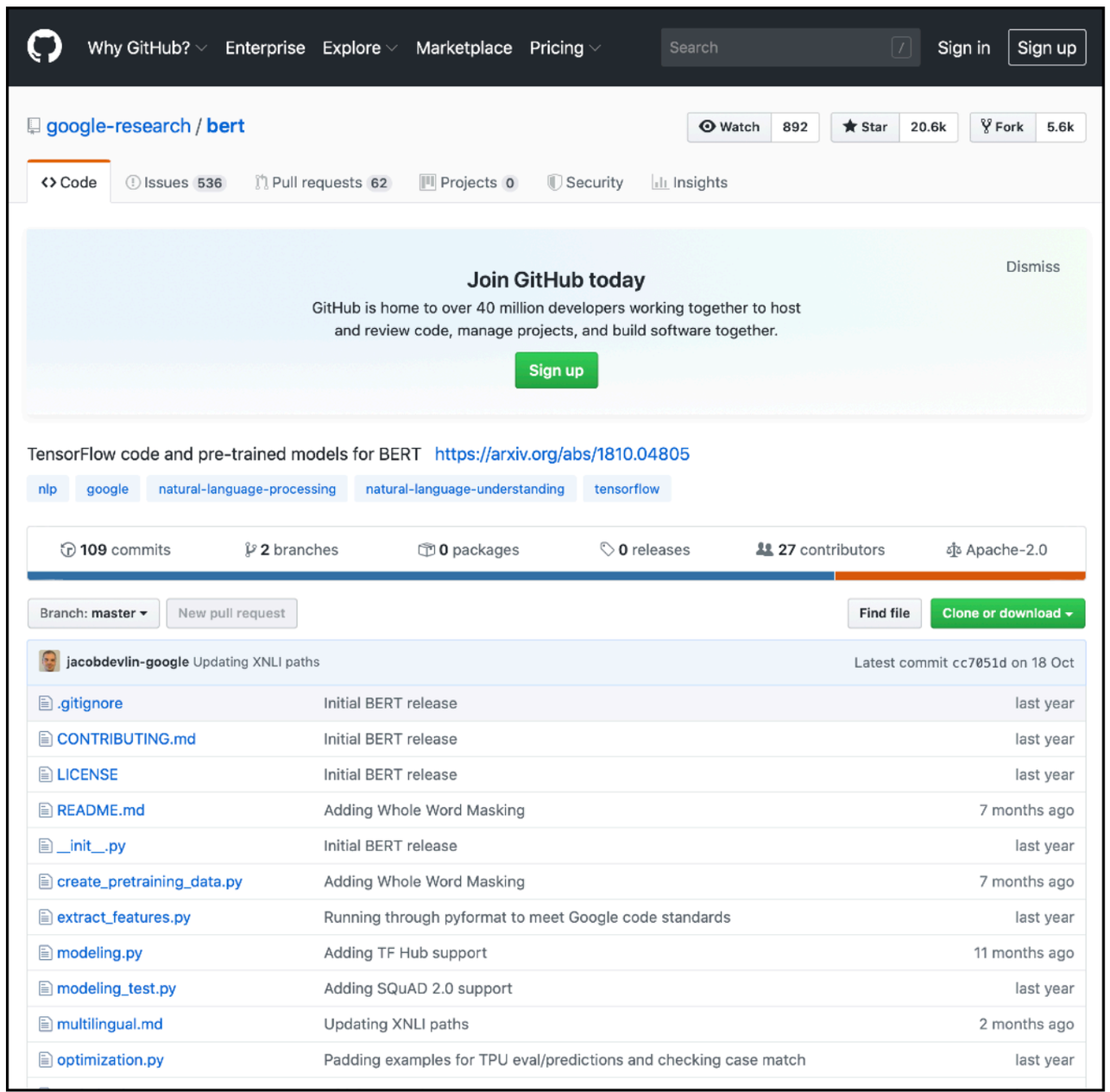
BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering Test F1 to 93.2 (1.5 point absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 point absolute improvement).

There are two existing strategies for applying pre-trained language representations to downstream tasks: *feature-based* and *fine-tuning*. The feature-based approach, such as ELMo (Peters et al., 2018a), uses task-specific architectures that include the pre-trained representations as additional features. The fine-tuning approach, such as the Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018), introduces minimal task-specific parameters, and is trained on the downstream tasks by simply fine-tuning *all* pre-trained parameters. The two approaches share the same objective function during pre-training, where they use unidirectional language models to learn general language representations.

We argue that current techniques restrict the power of the pre-trained representations, especially for the fine-tuning approaches. The major limitation is that standard language models are unidirectional, and this limits the choice of architectures that can be used during pre-training. For example, in OpenAI GPT, the authors use a left-to-right architecture, where every token can only attend to previous tokens in the self-attention layers

- **BERTとは「Bidirectional Encoder Representations from Transformers（Transformerによる双方向のエンコード表現）」を指し、2018年10月11日にGoogleが発表した自然言語処理モデル**

- **WikipediaやBooksCorpusなどから得た大量の文章データを学習モデルが事前学習し、文章理解や感情分析などの様々なタスクに応用できる**



- Google による事前学習モデルを利用可能

Pre-trained models

We are releasing the `BERT-Base` and `BERT-Large` models from the paper. `Uncased` means that the text has been lowercased before WordPiece tokenization, e.g., `John Smith` becomes `john smith`. The `Uncased` model also strips out any accent markers. `Cased` means that the true case and accent markers are preserved. Typically, the `Uncased` model is better unless you know that case information is important for your task (e.g., Named Entity Recognition or Part-of-Speech tagging).

These models are all released under the same license as the source code (Apache 2.0).

For information about the Multilingual and Chinese model, see the Multilingual README.

When using a cased model, make sure to pass `--do_lower=False` to the training scripts. (Or pass `do_lower_case=False` directly to `FullTokenizer` if you're using your own script.)

The links to the models are here (right-click, 'Save link as...' on the name):

- **`BERT-Large, Uncased (Whole Word Masking)`** : 24-layer, 1024-hidden, 16-heads, 340M parameters
- **`BERT-Large, Cased (Whole Word Masking)`** : 24-layer, 1024-hidden, 16-heads, 340M parameters
- **`BERT-Base, Uncased`** : 12-layer, 768-hidden, 12-heads, 110M parameters
- **`BERT-Large, Uncased`** : 24-layer, 1024-hidden, 16-heads, 340M parameters
- **`BERT-Base, Cased`** : 12-layer, 768-hidden, 12-heads , 110M parameters
- **`BERT-Large, Cased`** : 24-layer, 1024-hidden, 16-heads, 340M parameters
- **`BERT-Base, Multilingual Cased (New, recommended)`** : 104 languages, 12-layer, 768-hidden, 12-heads, 110M parameters
- **`BERT-Base, Multilingual Uncased (Orig, not recommended)`** **(Not recommended, use `Multilingual Cased` instead)**: 102 languages, 12-layer, 768-hidden, 12-heads, 110M parameters
- **`BERT-Base, Chinese`** : Chinese Simplified and Traditional, 12-layer, 768-hidden, 12-heads, 110M parameters

ALBERT

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)

ALBERT: A LITE BERT FOR SELF-SUPERVISED LEARNING OF LANGUAGE REPRESENTATIONS

Zhenzhong Lan[1] **Mingda Chen**[2*] **Sebastian Goodman**[1] **Kevin Gimpel**[2]
Piyush Sharma[1] **Radu Soricut**[1]
[1]Google Research [2]Toyota Technological Institute at Chicago
{lanzhzh, seabass, piyushsharma, rsoricut}@google.com
{mchen, kgimpel}@ttic.edu

ABSTRACT

Increasing model size when pretraining natural language representations often results in improved performance on downstream tasks. However, at some point further model increases become harder due to GPU/TPU memory limitations, longer training times, and unexpected model degradation. To address these problems, we present two parameter-reduction techniques to lower memory consumption and increase the training speed of BERT (Devlin et al., 2019). Comprehensive empirical evidence shows that our proposed methods lead to models that scale much better compared to the original BERT. We also use a self-supervised loss that focuses on modeling inter-sentence coherence, and show it consistently helps downstream tasks with multi-sentence inputs. As a result, our best model establishes new state-of-the-art results on the GLUE, RACE, and SQuAD benchmarks while having fewer parameters compared to BERT-large. The code and the pretrained models are available at https://github.com/google-research/google-research/tree/master/albert.

1 INTRODUCTION

Full network pre-training (Dai & Le, 2015; Radford et al., 2018; Devlin et al., 2019; Howard & Ruder, 2018) has led to a series of breakthroughs in language representation learning. Many non-trivial NLP tasks, including those that have limited training data, have greatly benefited from these pre-trained models. One of the most compelling signs of these breakthroughs is the evolution of machine performance on a reading comprehension task designed for middle and high-school English exams in China, the RACE test (Lai et al., 2017): the paper that originally describes the task and formulates the modeling challenge reports then state-of-the-art machine accuracy at 44.1%; the latest

- **2019年9月に、BERTを軽量化し高速化を行った「ALBERT」がGoogleによって公開された**
- **大幅なパラメータ削減による速度と性能を向上させている**

提案手法

手法1(双曲空間埋込モデル)

データのもつ階層構造、木構造、Directed Acyclic Graph (DAG)を自動的に抽出できる性質により、ナレッジグラフのグラフ構造を意味的な距離にて表現できるようになることを期待

処理手順

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(gensimのPoincare Embeddings を利用)

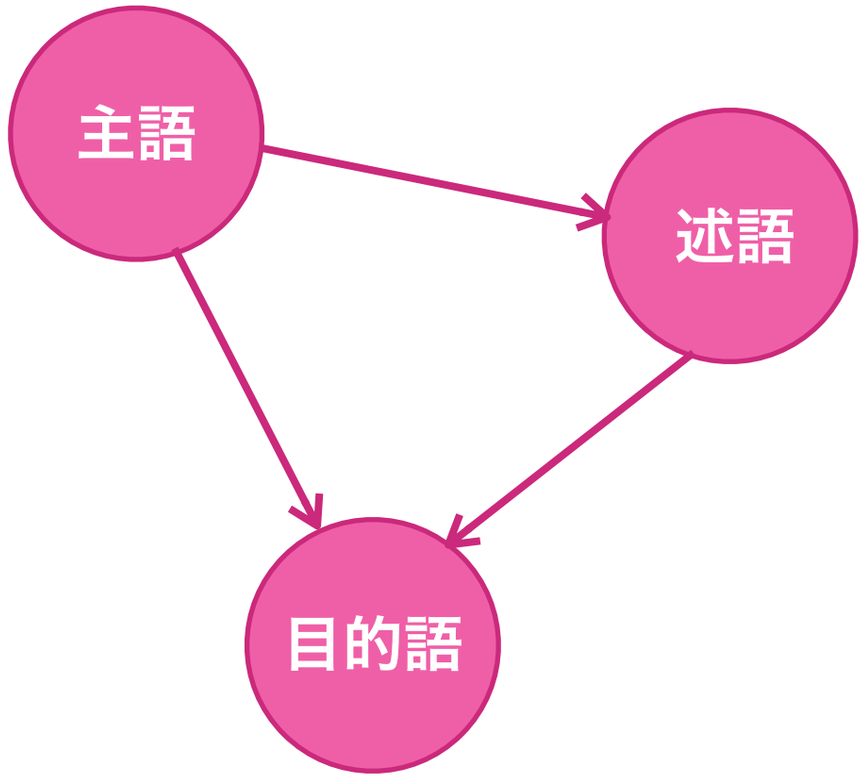
3. 分散ベクトルファイルを作成する（タブ区切り形式）

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

上位・下位関係のペアで表現

<u>入力形式:</u>

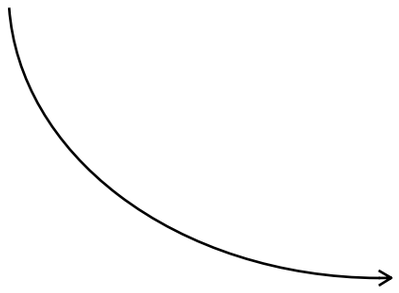
- 主語,述語
- 述語,目的語
- 主語,目的語



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(ALBERTモデル)

Googleによる事前学習モデルによる分散表現を用いることにより、意味を重視した分散表現を期待

事前学習データ

https://github.com/google-research/ALBERT

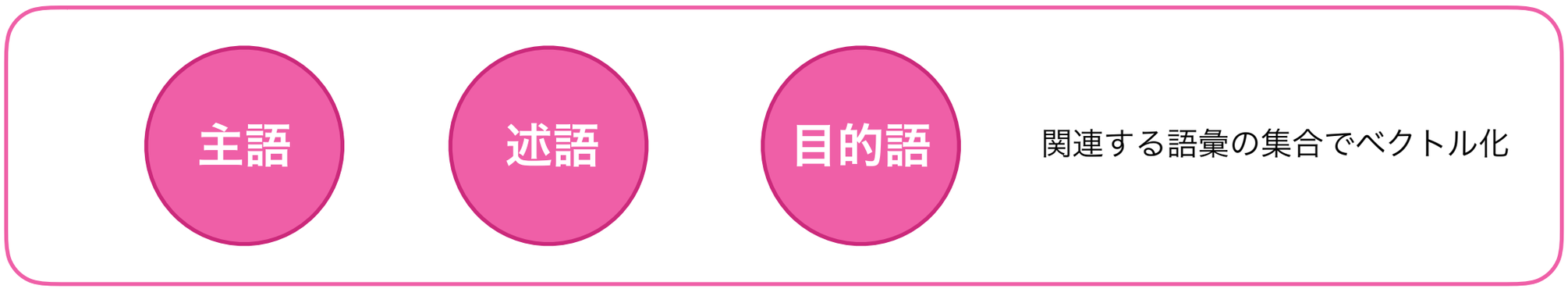
処理手順

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

トリプルを関連性の高い集合として入力

<u>入力形式:</u>

[CLS]主語[SEP]述語[SEP]目的語[SEP]



変換例

変換前）

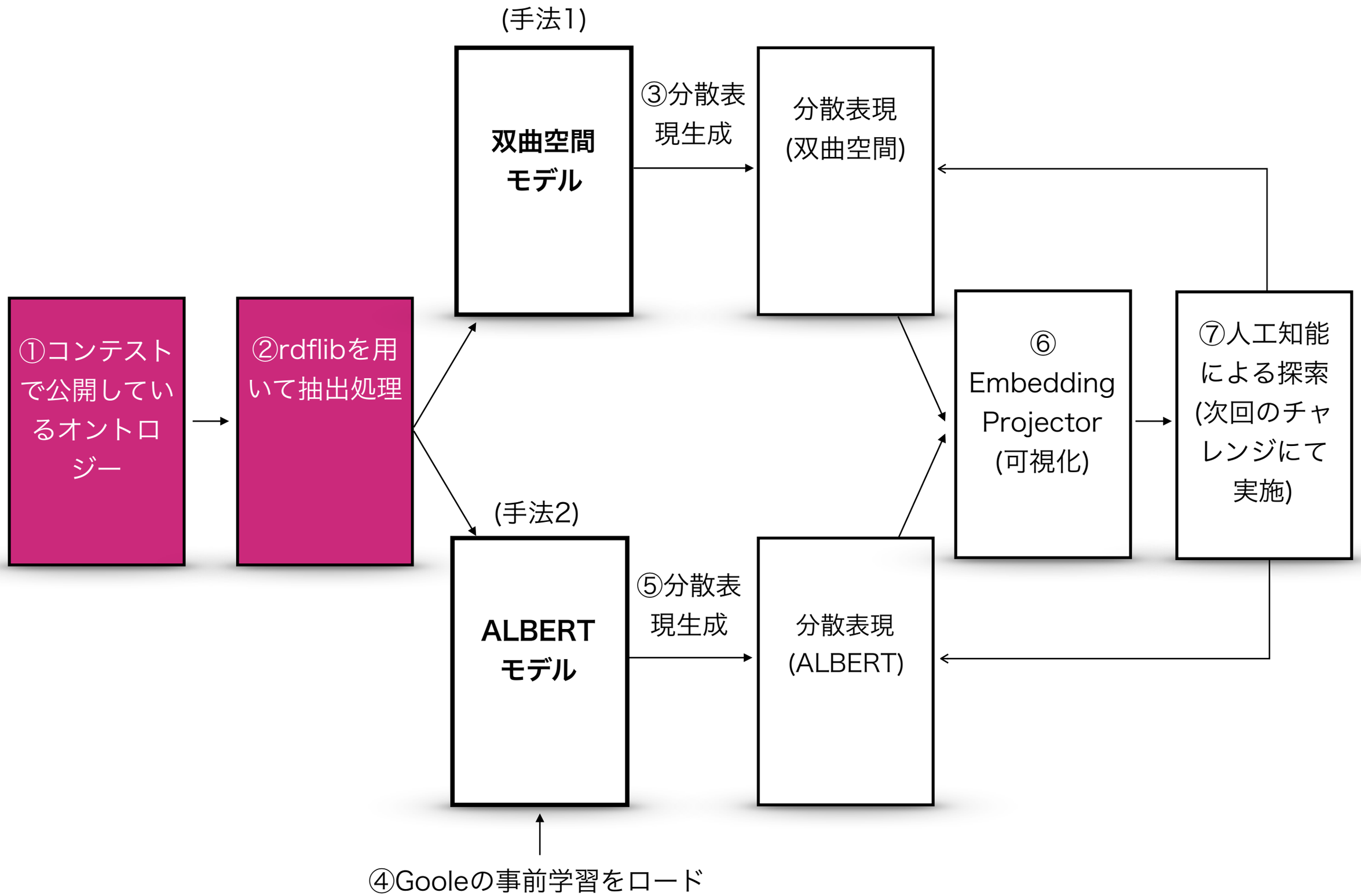
・ <http://kgc.knowledge-graph.jp/data/ACaseOfIdentity/001>
rdf:type kgc:Situation ;
kgc:source "ホームズは椅子から立った"@ja ;
kgc:source "Holmes stood out of a chair"@en ;
kgc:hasPredicate <http://kgc.knowledge-graph.jp/data/predicate/stand> ;
kgc:subject <http://kgc.knowledge-graph.jp/data/ACaseOfIdentity/Holmes> ;
kgc:from <http://kgc.knowledge-graph.jp/data/ACaseOfIdentity/Chair> ;
kgc:time "1891-09-01T10:00:00"^^xsd:dateTime .

変換後）

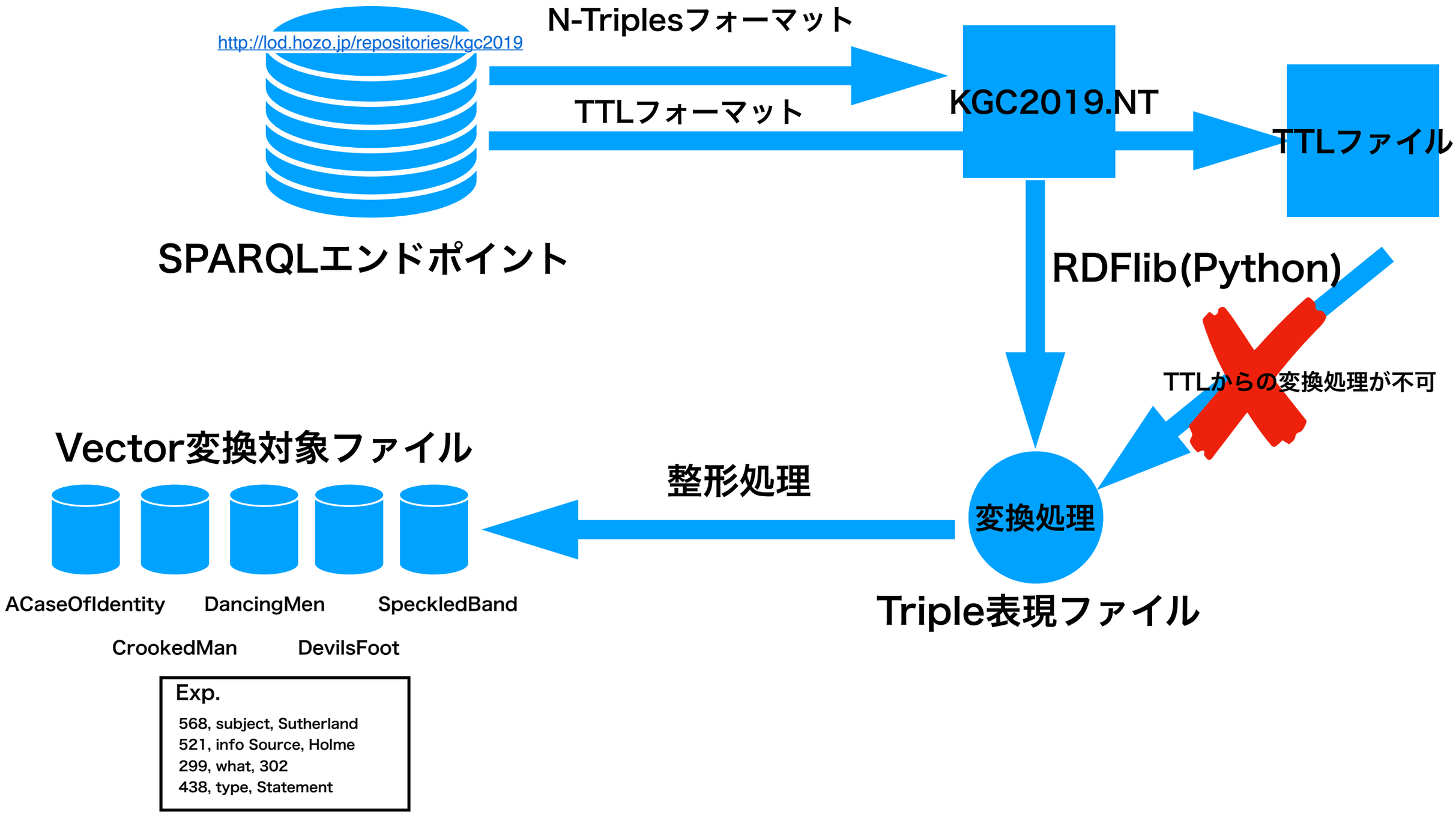
- [CLS]001[SEP]has Predicate[SEP]stand[SEP]
- [CLS]001[SEP]time[SEP]1891-09-01 10:00:00[SEP]
- [CLS]001[SEP]subject[SEP]Holmes[SEP]
- [CLS]001[SEP]when[SEP]1891-09-01T10[SEP]
- [CLS]001[SEP]source[SEP]Holmes stood out of a chair[SEP]
- [CLS]001[SEP]from[SEP]Chair[SEP]
- [CLS]001[SEP]type[SEP]Situation[SEP]

使用したツールと処理内容

ナレッジグラフデータの抽出


(手法1)
双曲空間
モデル
③分散表
現生成
分散表現
(双曲空間)
①コンテスト
で公開してい
るオントロ
ジー
②rdflibを用
いて抽出処理
⑥
Embedding
Projector
(可視化)
⑦人工知能
による探索
(次回のチャ
レンジにて
実施)
(手法2)
ALBERT
モデル
⑤分散表
現生成
分散表現
(ALBERT)
④Gooleの事前学習をロード

ナレッジグラフデータの抽出



ナレッジグラフデータの抽出

#1. SPARQLエンドポイント

https://github.com/KnowledgeGraphJapan/Challenge/tree/master/rdf/2019

推論チャレンジ2019用ナレッジグラフ

ナレッジグラフ推論チャレンジ2019用のナレッジグラフを公開するレポジトリです.

2019/08/26 バージョン

一部，修正を入れる可能性がありますが，ほぼ正式版のバージョンとなります.
※「まだらのひも」については，ナレッジグラフ推論チャレンジ2018のバージョンのものをそのまま公開していますが，他の小説と合わせて修正をする予定があります. ※「悪魔の足」については，一部，ナレッジグラフを修正作業中です.

修正リクエストはGitHubの他、こちらのフォームからも受け付けております。

SPARQLエンドポイント http://lod.hozo.jp/repositories/kgc2019

小説	グラフIRI
まだらのひも	<http://kgc.knowledge-graph.jp/data/SpeckledBand>
踊る人形	<http://kgc.knowledge-graph.jp/data/DancingMen>
花婿失踪事件（同一事件）	<http://kgc.knowledge-graph.jp/data/ACaseOfIdentity>
悪魔の足	<http://kgc.knowledge-graph.jp/data/DevilsFoot>
背中の曲がった男（曲がれる者）	<http://kgc.knowledge-graph.jp/data/CrookedMan>

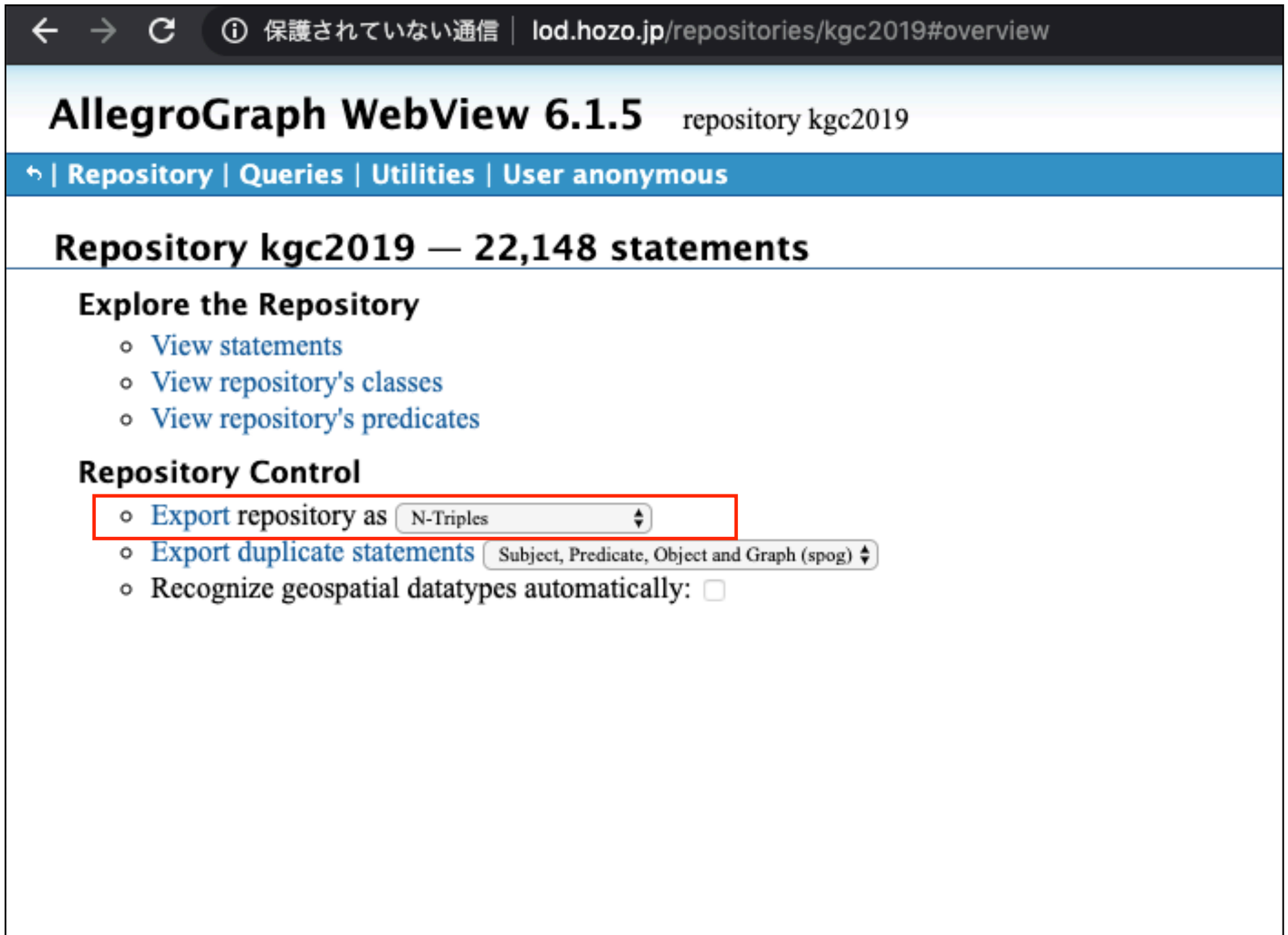
サンプルSPARQLクエリ

下記に，このナレッジグラフを対象としたSPARQLクエリ例をまとめています.

ナレッジグラフデータの抽出

#1.SPARQLエンドポイント(ダウンロードサイト)

http://lod.hozo.jp/repositories/kgc2019#overview



ナレッジグラフデータの抽出

#2. RDFlibを利用(Python)

https://pypi.org/project/rdflib/#description
https://github.com/RDFLib/rdflib

rdflib 4.2.2

Latest version

```
pip install rdflib
```

Last released: Jan 30, 2017

RDFLib is a Python library for working with RDF, a simple yet powerful language for representing information.

Navigation

- Project description
- Release history
- Download files

Project links

- Homepage

Statistics

Project description

RDFLib is a Python library for working with RDF, a simple yet powerful language for representing information.

The library contains parsers and serializers for RDF/XML, N3, NTriples, Turtle, TriX, RDFa and Microdata . The library presents a Graph interface which can be backed by any one of a number of Store implementations. The core rdflib includes store implementations for in memory storage, persistent storage on top of the Berkeley DB, and a wrapper for remote SPARQL endpoints.

A SPARQL 1.1 engine is also included.

If you have recently reported a bug marked as fixed, or have a craving for the very latest, you may want the development version instead:

> *easy_install https://github.com/RDFLib/rdflib/tarball/master*

Read the docs at:

ナレッジグラフデータの抽出

#2. RDFlib(Sample code)

jupyter knowledge_present Last Checkpoint: 2019/11/04 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
kgc = rdflib.Namespace("http://kgc.knowledge-graph.jp/ontology/kgc.owl#")

g = rdflib.Graph()
g.parse('kgc2019.nt', format='nt')

# Iterate over triples in store and print them out.
print("--- printing raw triples ---")

for s, p, o in g.triples((None, None, None)):
    if isinstance(o, rdflib.term.Literal):
        if o.language == 'ja':
            continue
    (story,seenNo) = parse_subject(s)

    if my_index(booklist, story) is False:
        continue

    index = my_index(booklist, story)
    if os.path.exists(filelist[index]):
        with open(filelist[index], 'a') as f:
            f.write(seenNo + "," + parse_predicate(p) + "," + parse_object(o) + "\n")
```

ナレッジグラフデータの抽出

#3.Triples表現ファイルの生成(ストーリ毎)

フォーマット：主語 + 述語 + 目的語

主語	述語	目的語
52	to	bedroom_of_Roylott
394	when	393
275	hasPredicate	smell
78	type	Situation
179	type	Situation
295	source	Holmes and Watson do not sleep.
Exist	type	Object
329	hasPredicate	see
227	hasPredicate	have
bedroom_of_Julia	type	Place
217	hasPredicate	notWork
199	where	chest
286	source	Julia could not move the bed.
369	type	Situation
80	what	Roylott
sname	label	sname
345	subject	safe
notSee	type	Action
knee_of_Roylott	type	Object

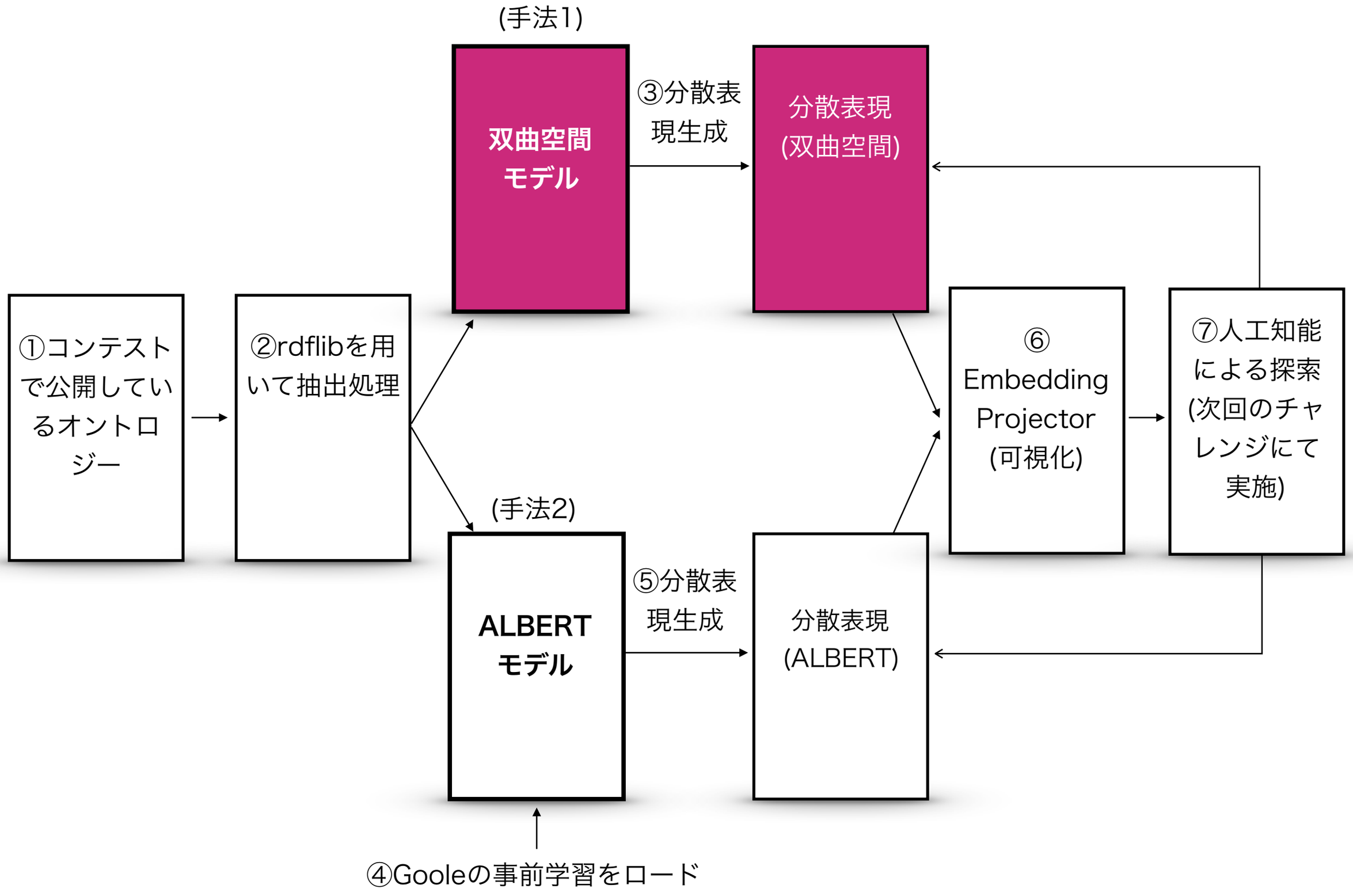
ナレッジグラフデータの抽出

#4.Triples表現ファイルの整形(ストーリ毎)

1. 述語の単語分離
2. 記号の削除(空白, アンダースコア, Time記号 etc)
3. 無関係記述の排除(Meta)

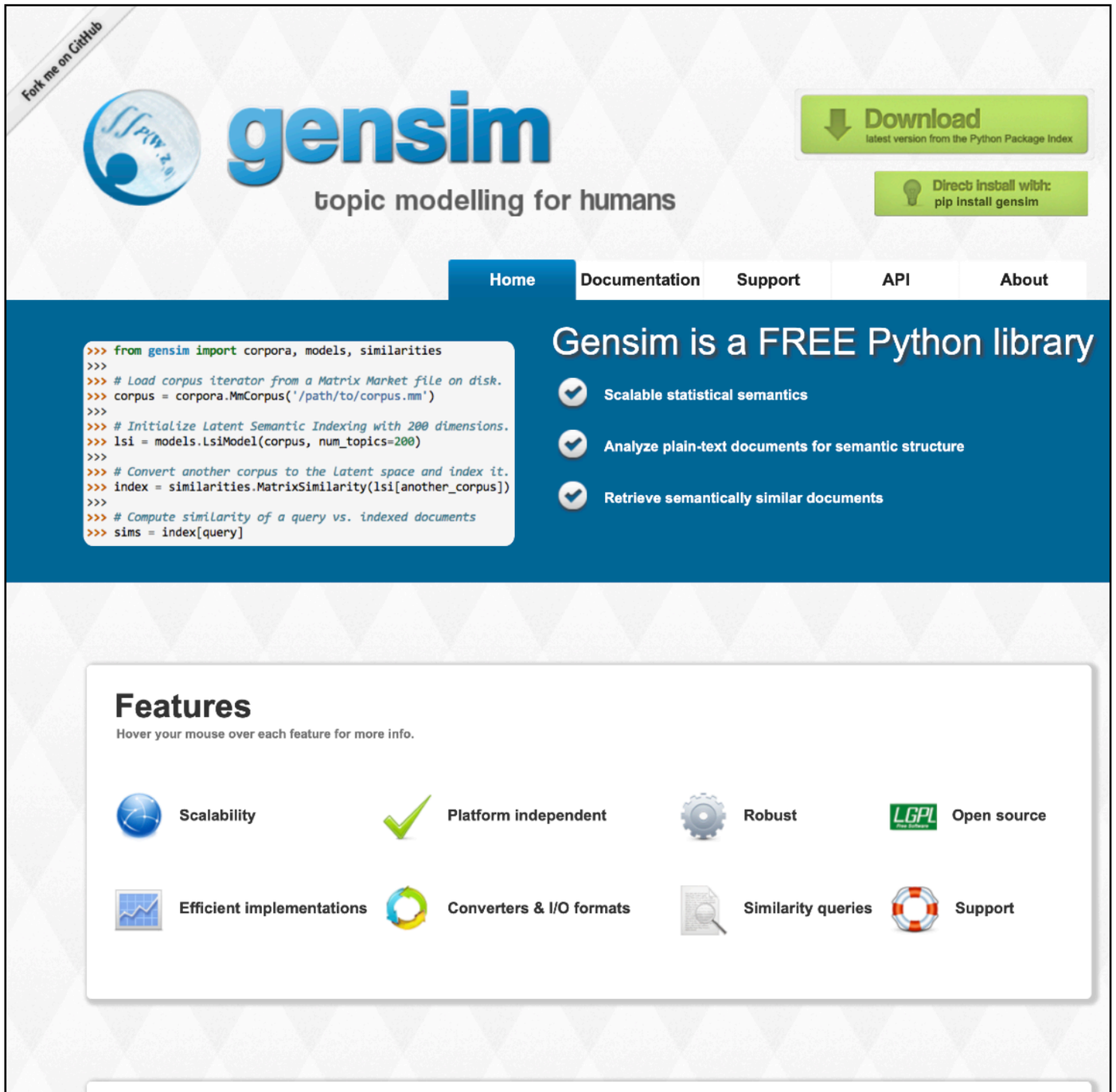
主語	述語	目的語
52	to	bedroom of Roylott
394	when	393
275	has Predicate	smell
78	type	Situation
179	type	Situation
295	source	Holmes and Watson do not sleep.
Exist	type	Object
329	has Predicate	see
227	has Predicate	have
bedroom_of_Julia	type	Place
217	has Predicate	not Work
199	where	chest
286	source	Julia could not move the bed.
369	type	Situation
80	what	Roylott
sname	label	sname
345	subject	safe

手法1:双曲空間への埋込


(手法1)
双曲空間
モデル
③分散表
現生成
分散表現
(双曲空間)
①コンテスト
で公開してい
るオントロ
ジー
②rdflibを用
いて抽出処理
⑥
Embedding
Projector
(可視化)
⑦人工知能
による探索
(次回のチャ
レンジにて
実施)
(手法2)
ALBERT
モデル
⑤分散表
現生成
分散表現
(ALBERT)
④Gooleの事前学習をロード

gensim

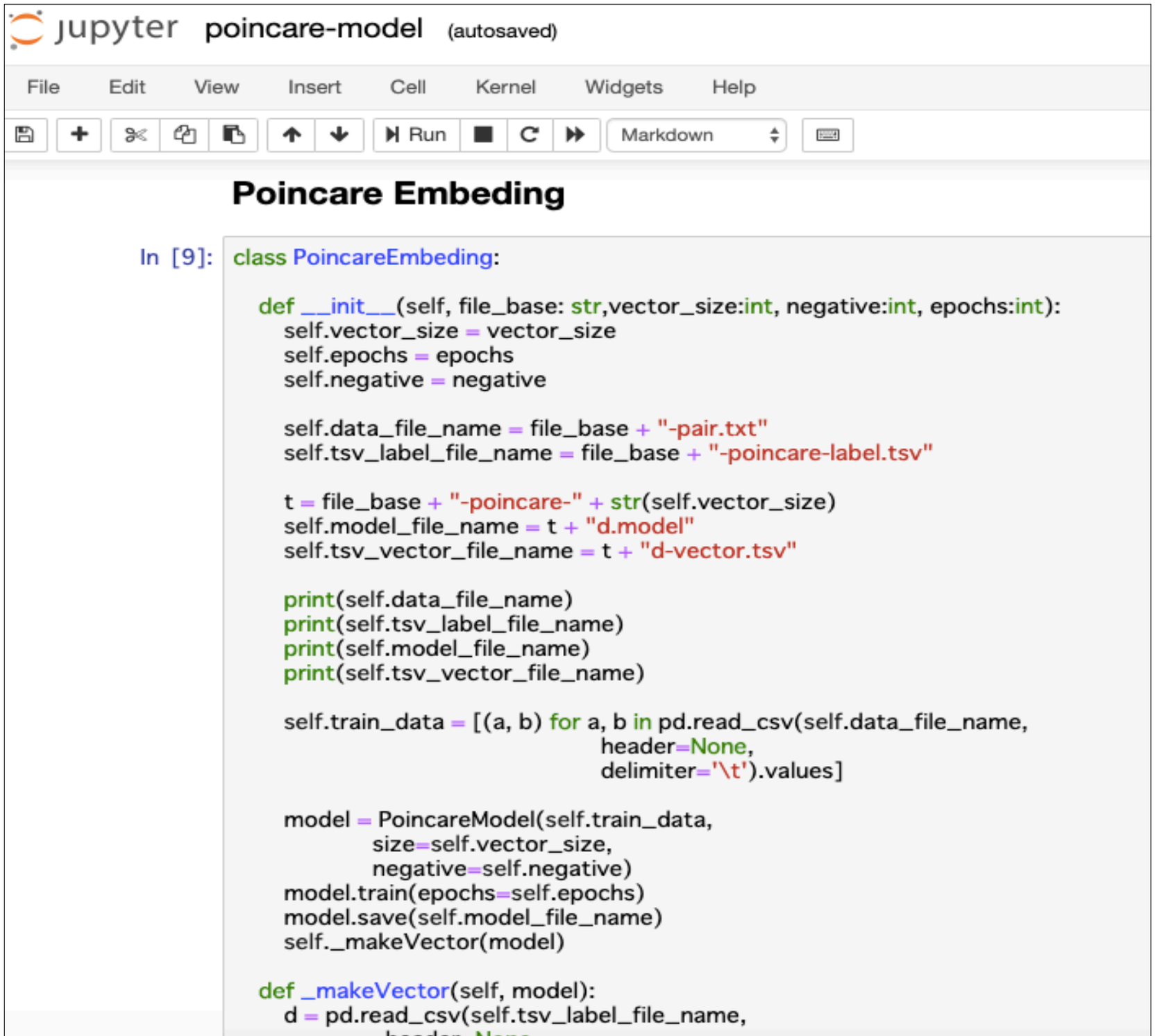
https://radimrehurek.com/gensim/



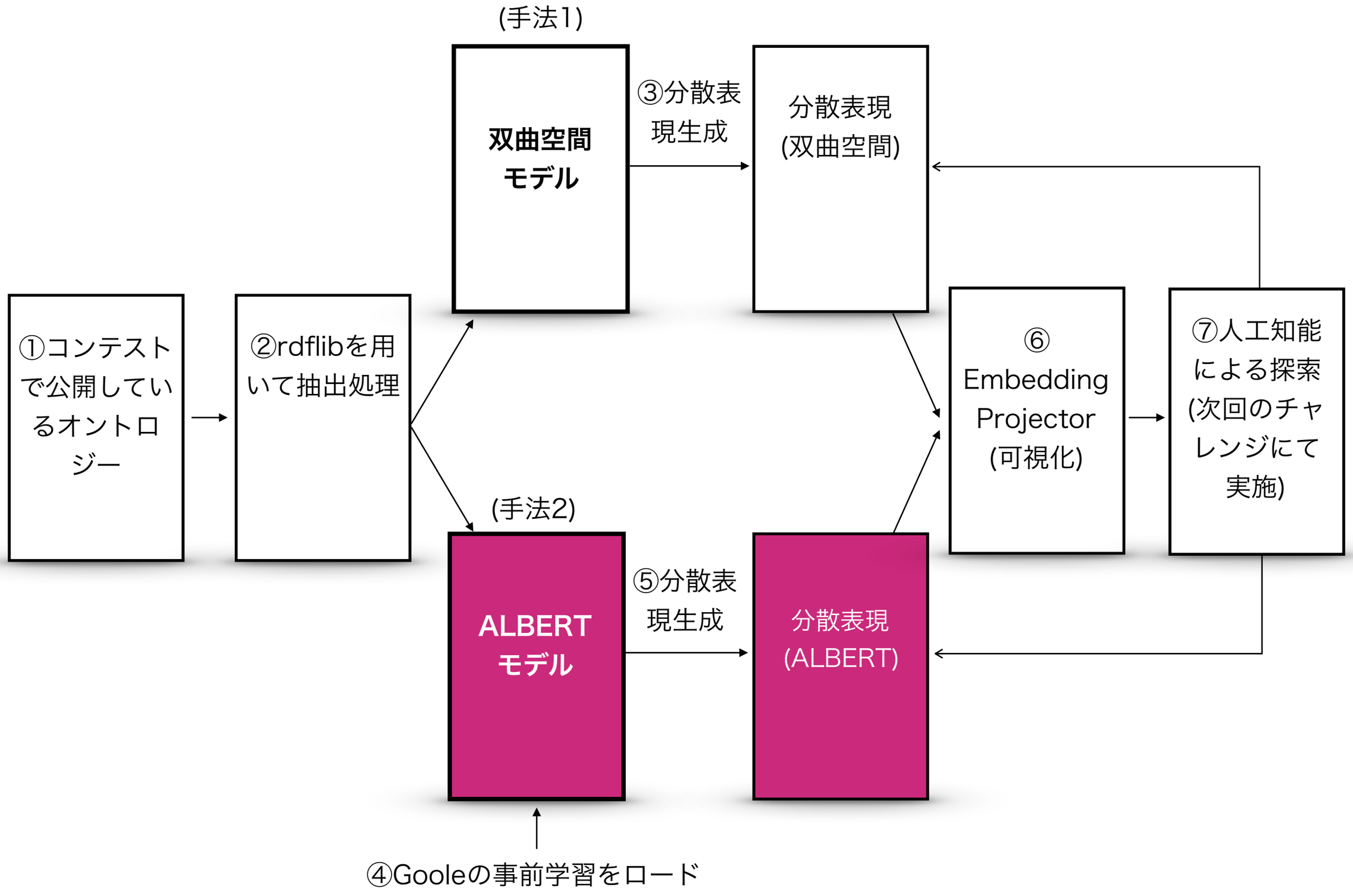
- 様々なトピックモデルを実装したPythonライブラリ
- 今回は、Poincare Model を使用した

Gensim

https://github.com/sakiyomi-ai/sherlock2019/blob/master/books/poincare-model.ipynb

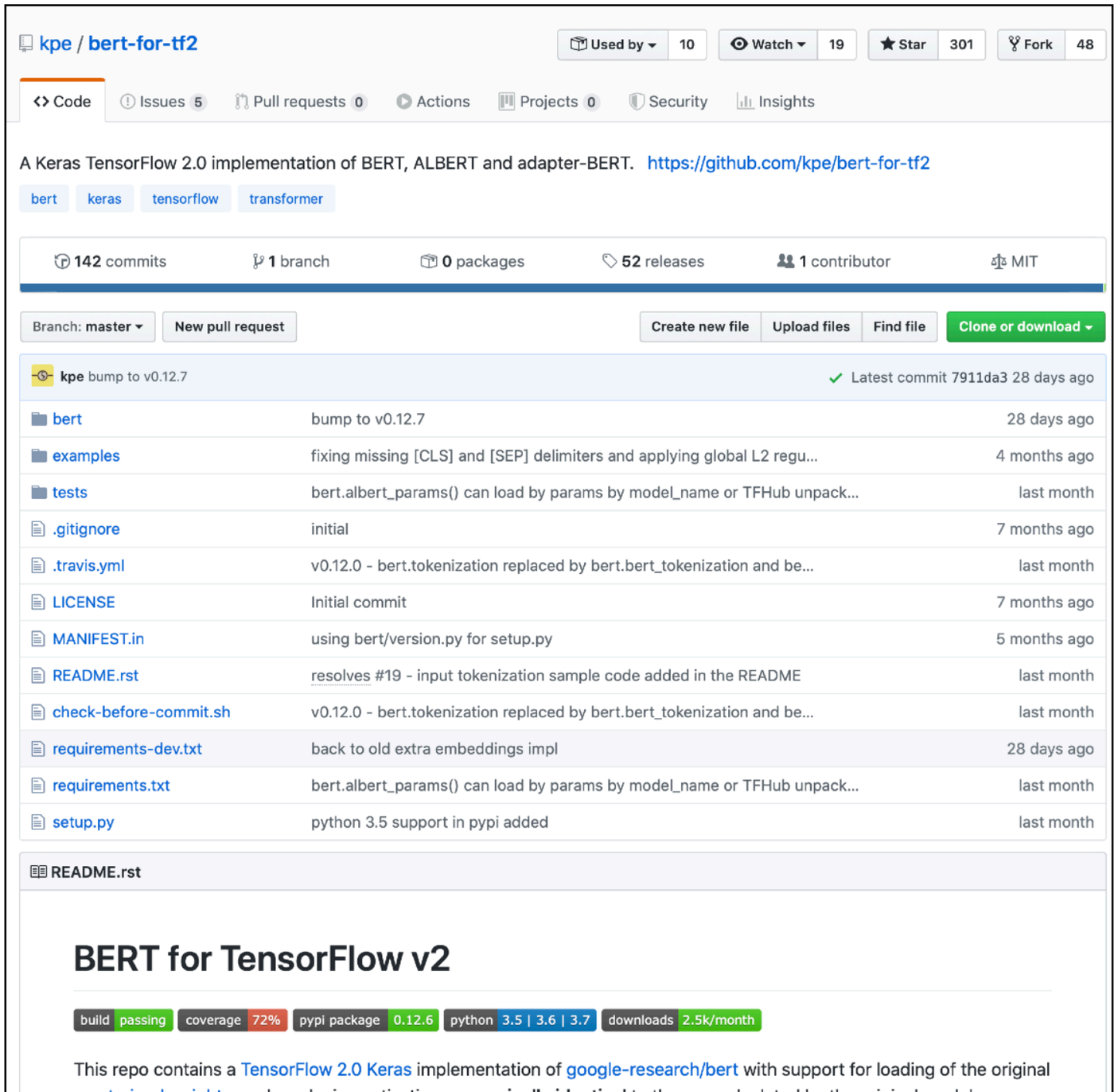


手法2:ALBERTによる分散表現


(手法1)
双曲空間
モデル
③分散表
現生成
分散表現
(双曲空間)
①コンテスト
で公開してい
るオントロ
ジー
②rdflibを用
いて抽出処理
⑥
Embedding
Projector
(可視化)
⑦人工知能
による探索
(次回のチャ
レンジにて
実施)
(手法2)
ALBERT
モデル
⑤分散表
現生成
分散表現
(ALBERT)
④Gooleの事前学習をロード

BERT for TensorFlow v2

https://github.com/kpe/bert-for-tf2



- Deep Learning向けライブラリの一つであるKerasから利用可能なBERT Model Layer
- BERTに加え、ALBERTやadapter-BERTも利用可能
- Googleの事前学習データを利用(https://github.com/google-research/ALBERT)

Bert for Tensorflow v2

https://github.com/sakiyomi-ai/sherlock2019/blob/master/books/bert-model-A.ipynb

jupyter bert-model-A_20191123 Last Checkpoint: 2019/11/24 (autosaved)

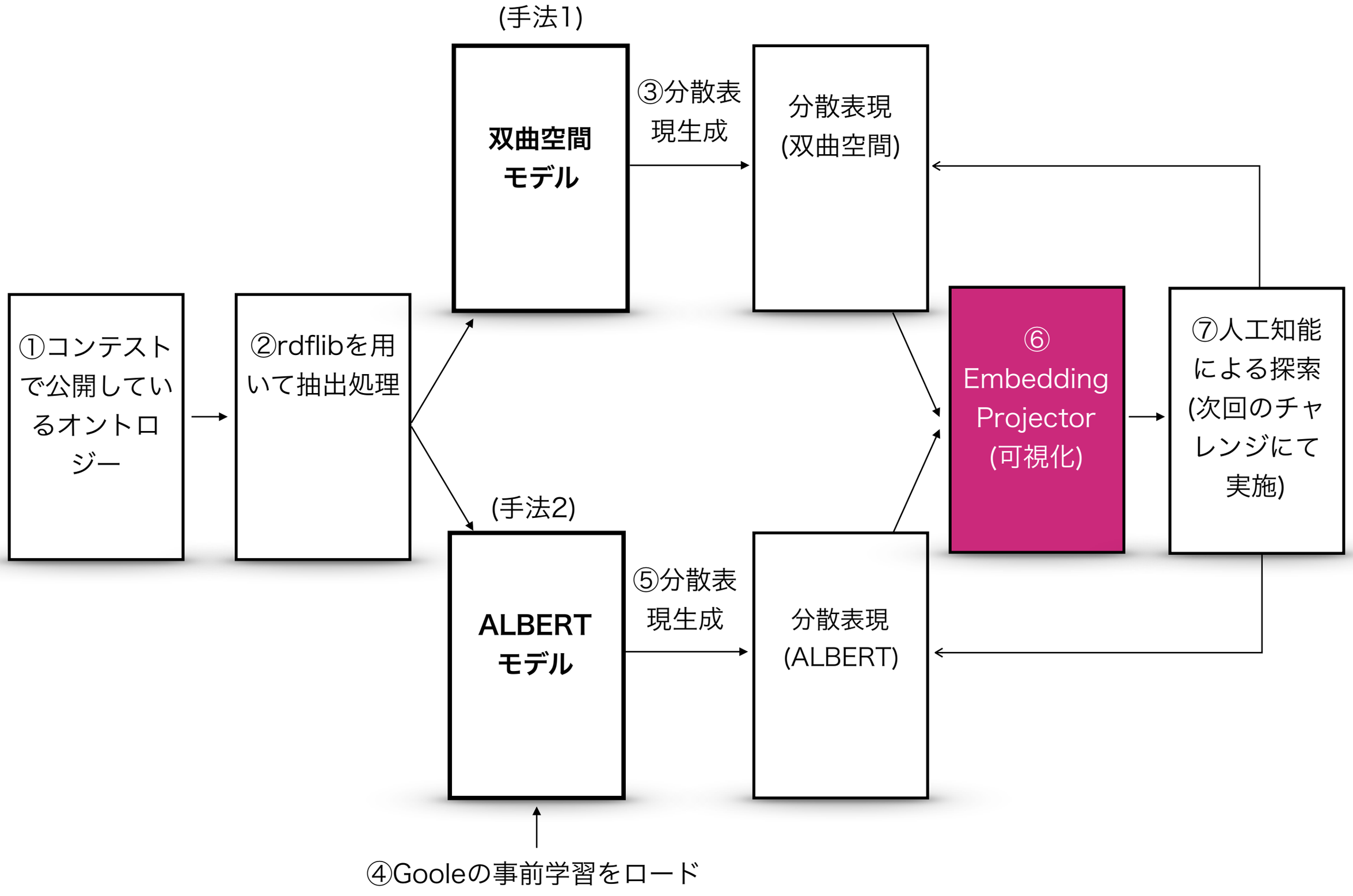
File Edit View Insert Cell Kernel Widgets Help

Run Markdown

In [ ]:

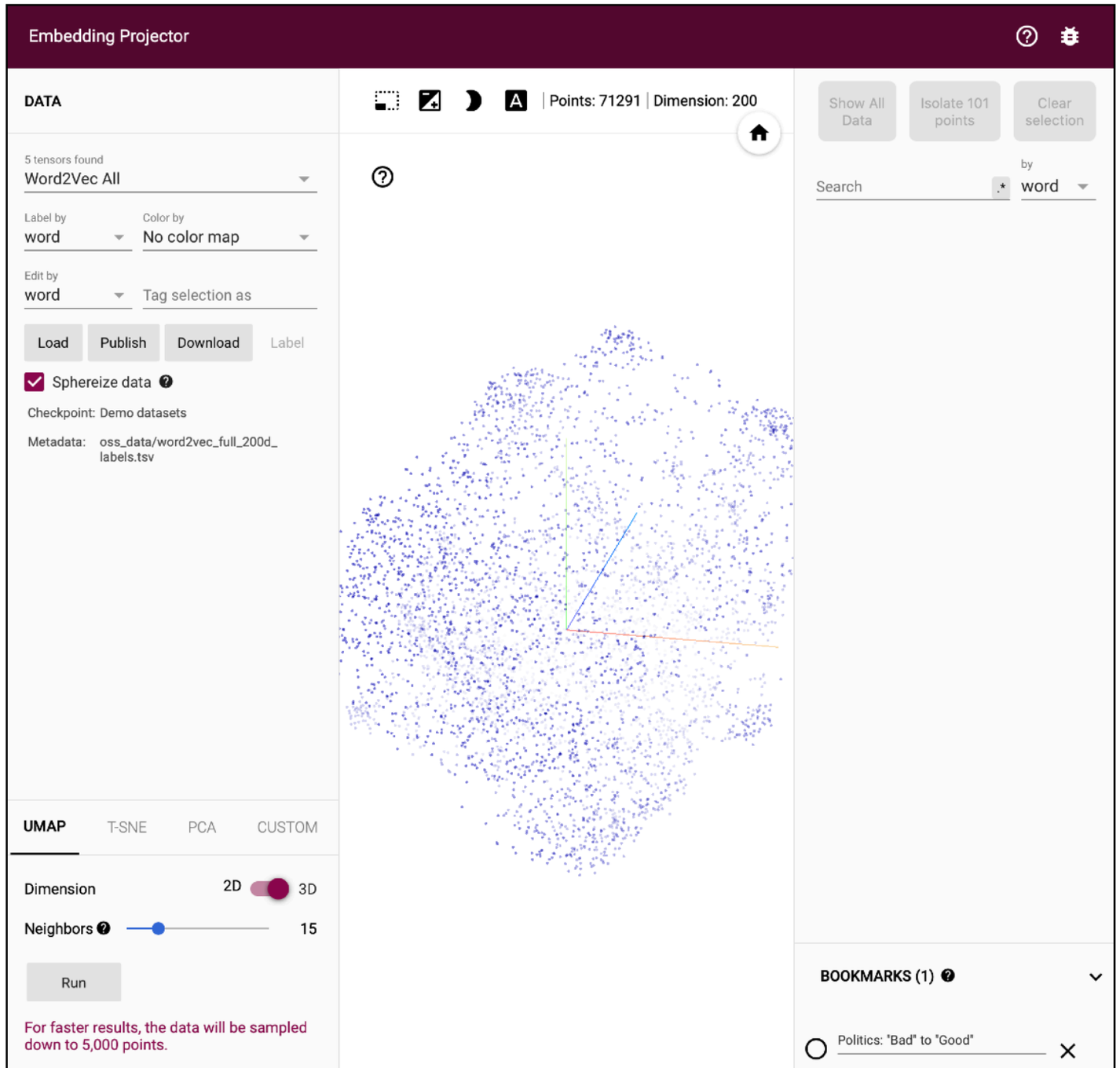
```
pattern = 'p2'
for story_name in story_names:
    model_file_name = story_name + "-" + model_name + "-" + pattern + ".npz"
    data_file_name = story_name + ".txt"

    print("================> " + story_name)

    spm_model = os.path.join(bert_ckpt_dir, "assets", "30k-clean.model")
    sp = spm.SentencePieceProcessor()
    sp.load(spm_model)
    do_lower_case = True

    modelData = SherlockModelData(sp=sp,data_file_name=data_file_name, pattern=pattern,lower=do_lower_case)
    print(modelData.train)

    model_params = bert.albert_params(model_name)
    l_bert = bert.BertModelLayer.from_params(model_params, name="albert")
    l_input_ids = keras.layers.Input(shape=(modelData.max_seq_len,), dtype='int32')
    output = l_bert(l_input_ids)
    model = keras.Model(inputs=l_input_ids, outputs=output)
    model.build(input_shape=(None, modelData.max_seq_len))
    bert.load_albert_weights(l_bert, bert_ckpt_dir)
    model.summary()

    vector = model.predict(modelData.train)
    np.savez(model_file_name,vector,modelData.label)
```

既存ツールによる可視化


(手法1)
双曲空間
モデル
③分散表
現生成
分散表現
(双曲空間)
①コンテスト
で公開してい
るオントロ
ジー
②rdflibを用
いて抽出処理
⑥
Embedding
Projector
(可視化)
⑦人工知能
による探索
(次回のチャ
レンジにて
実施)
(手法2)
ALBERT
モデル
⑤分散表
現生成
分散表現
(ALBERT)
④Gooleの事前学習をロード

Embedding projector

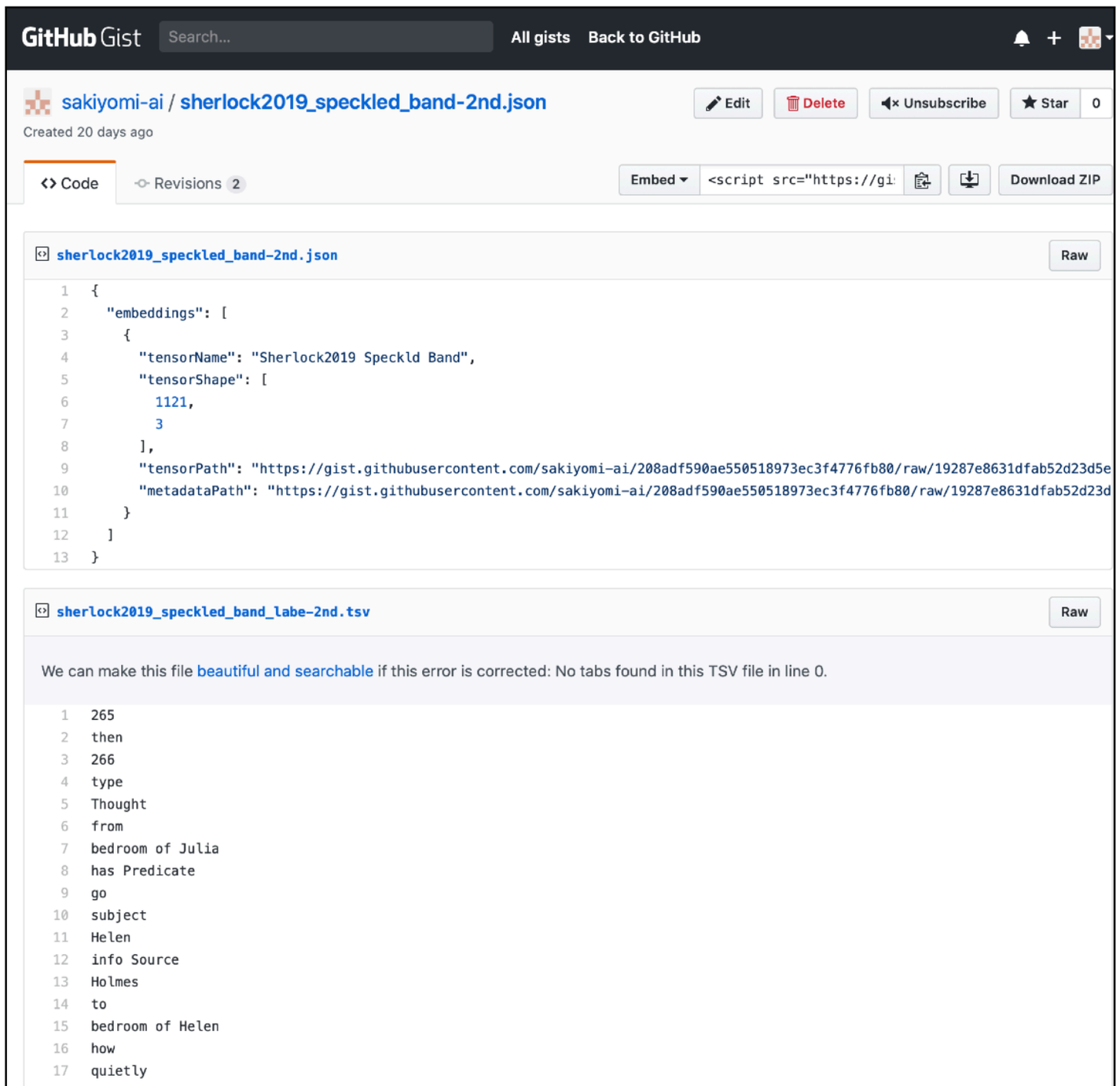
https://projector.tensorflow.org



- タブ区切り形式のベクトルデータを次元削減して2Dまたは3Dにて表示する
- 距離計算は、コサイン類似度とユークリッド距離を利用可能
- 「Publish」機能で、ベクトルデータ、メタデータを個別にアップロードする操作無しに公開可能(CORSに注意)

GitHub Gist

https://gist.github.com



- Embedding Projectorを利用する際、Gistからデータを配信すると、CORS(Cross-Origin Resource Sharing)の問題は発生しない

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

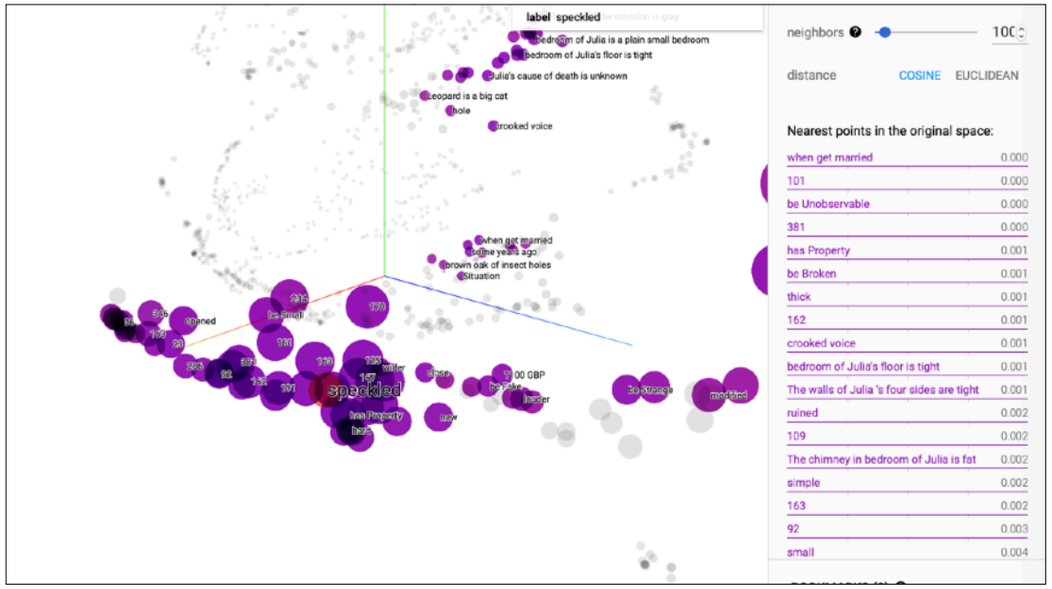
DEVILS FOOT
https://projector.tensorflow.org/?config=https://gist.githubusercontent.com/sakiyomi-ai/832364b703f580b53adbede8b255dcb6/raw/96bfb96554a71a7367641de0d6423da1cebe7229/sherlock2019_devils_foot-2nd.json

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

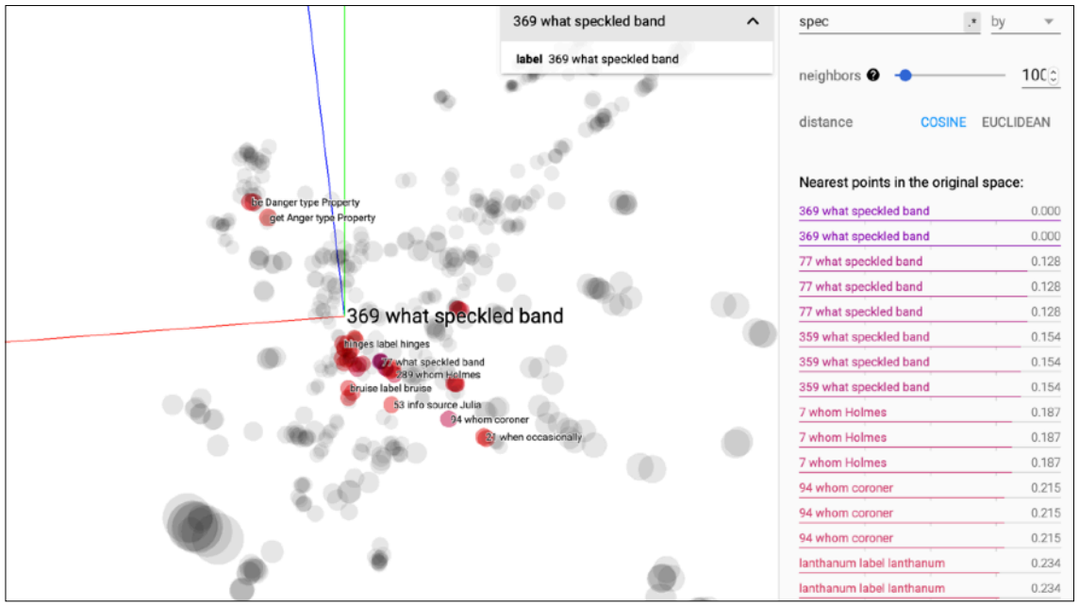
既存ツールによる可視化

まだらの紐

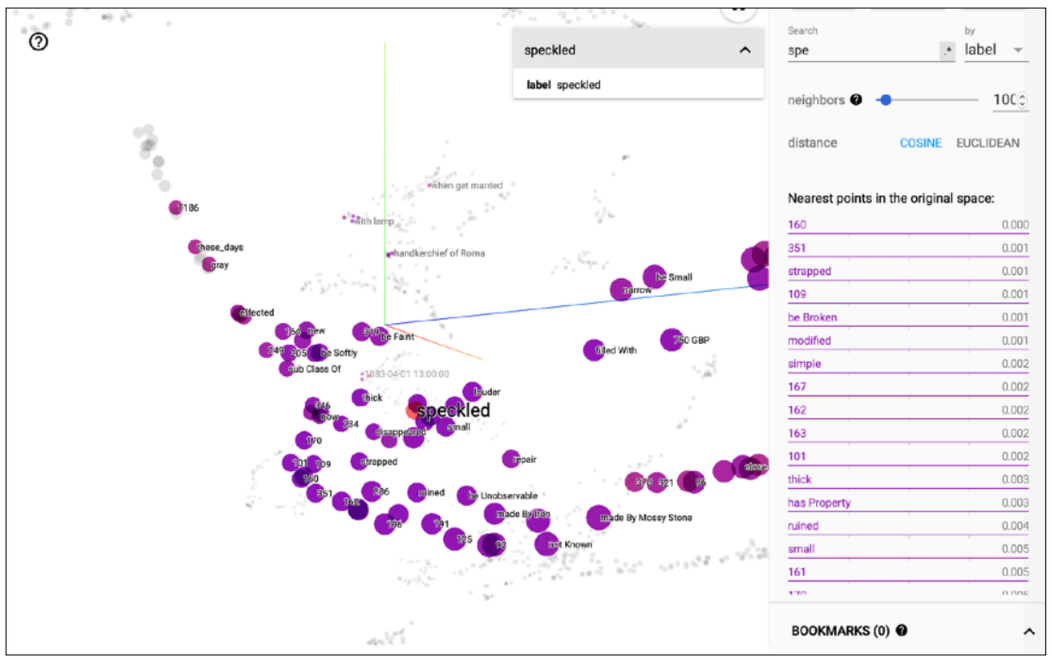
双曲空間版(source含めて学習)



ALBERT版(source含めて学習)

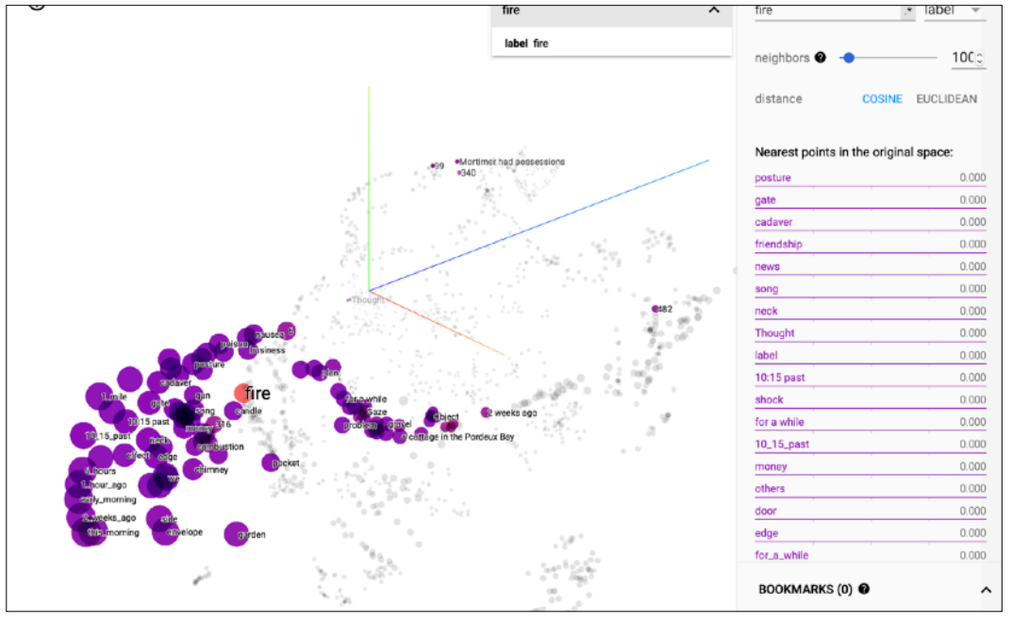


双曲空間版(sourceを含めず学習)

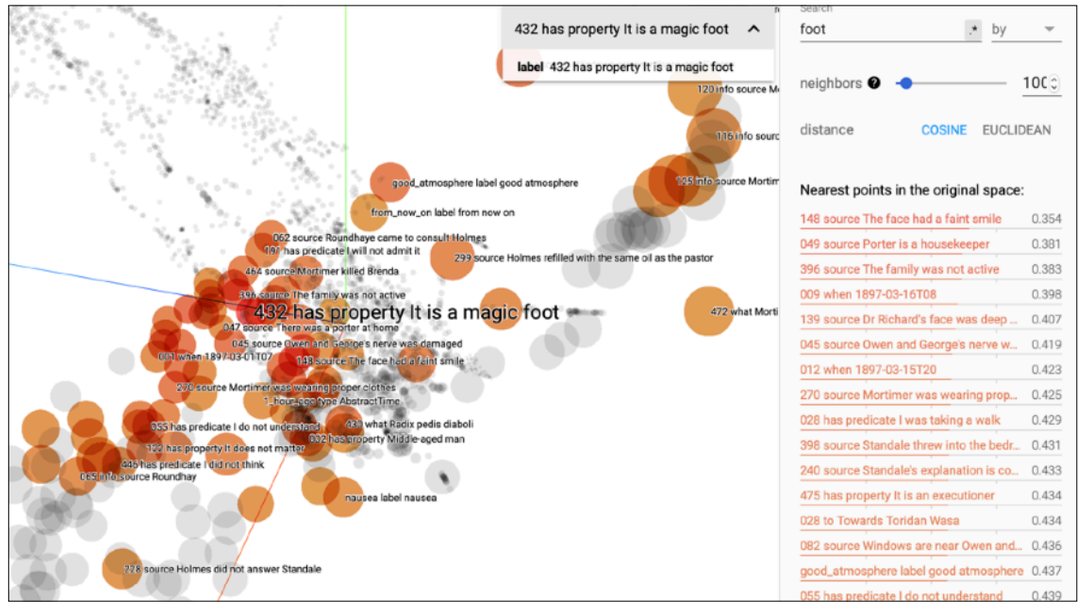


悪魔の足

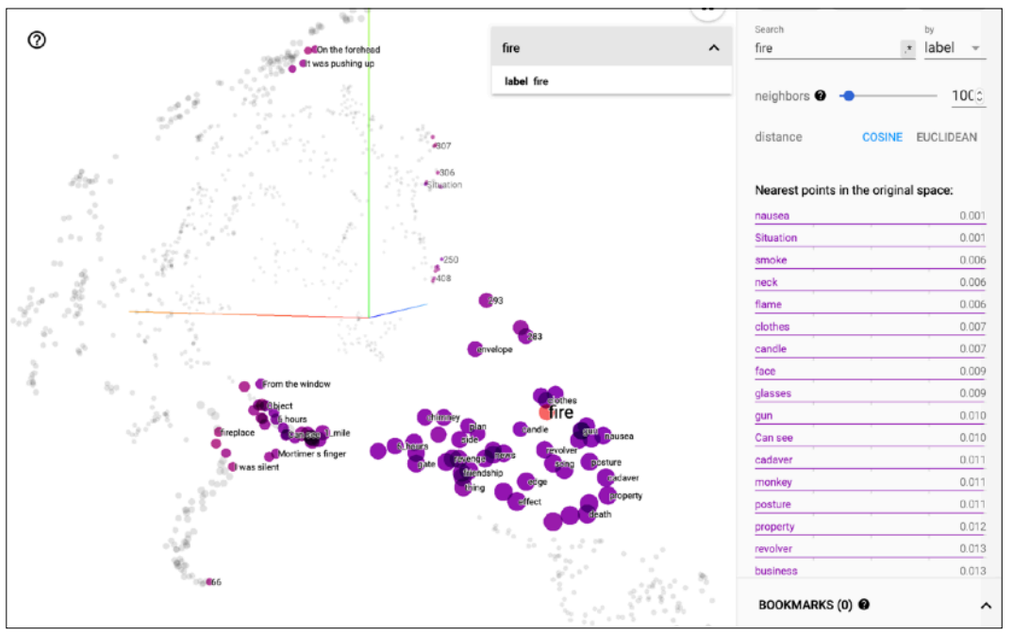
双曲空間版(source含めて学習)



ALBERT版(source含めて学習)

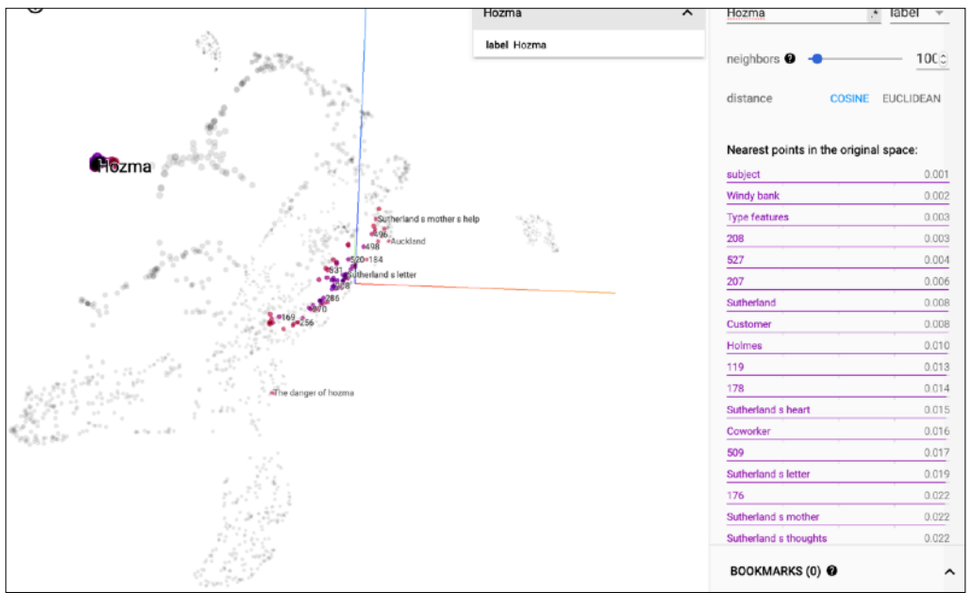


双曲空間版(sourceを含めず学習)

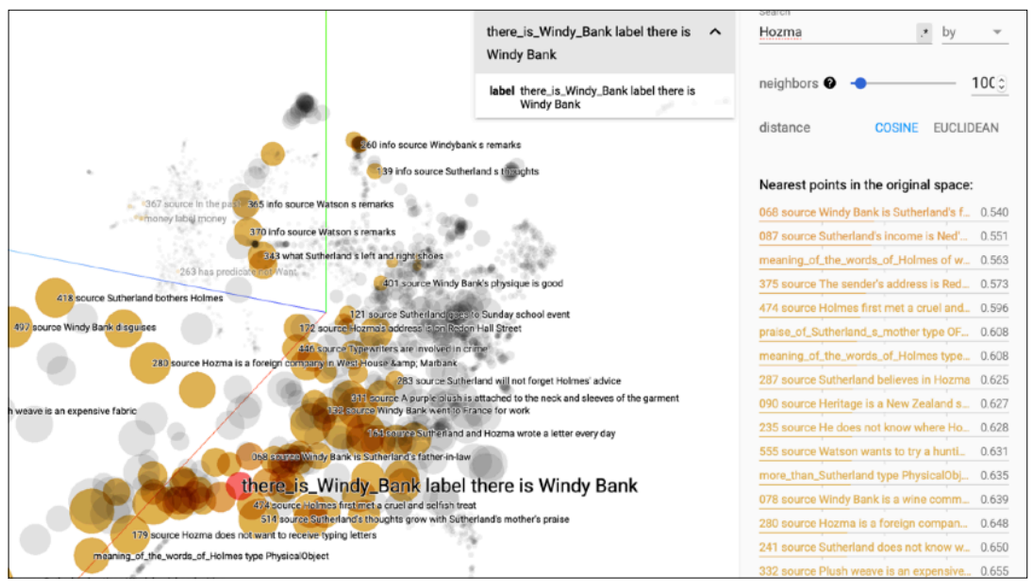


同一事件

双曲空間版(source含めて学習)



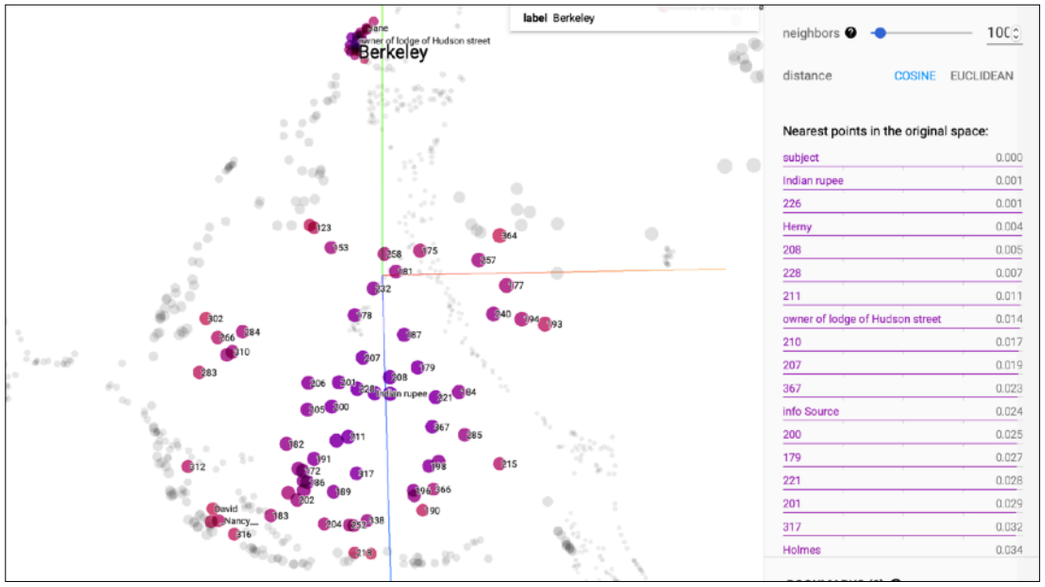
ALBERT版(source含めて学習)



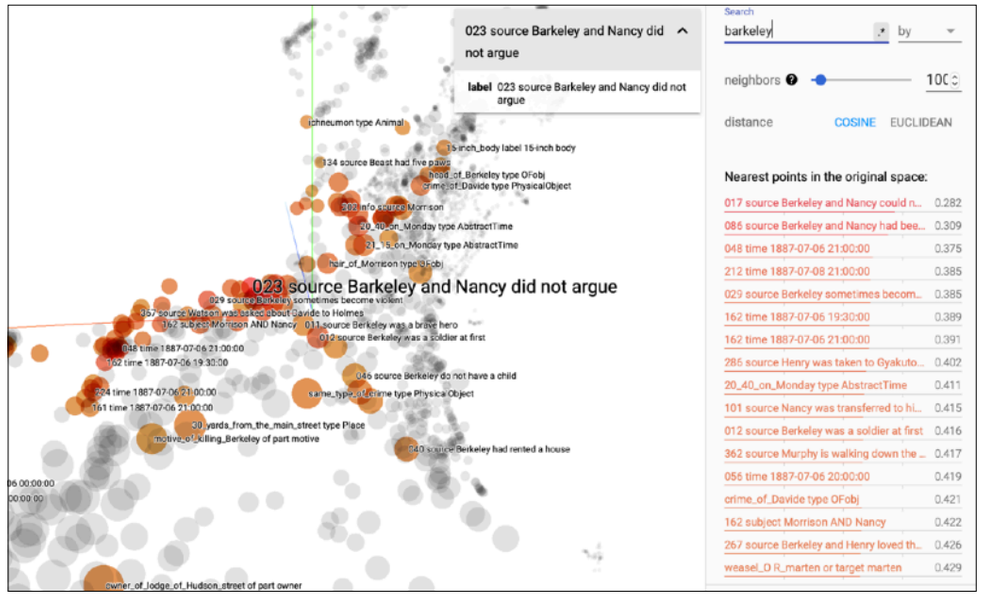
双曲空間版(sourceを含めず学習)

背中の曲がった男

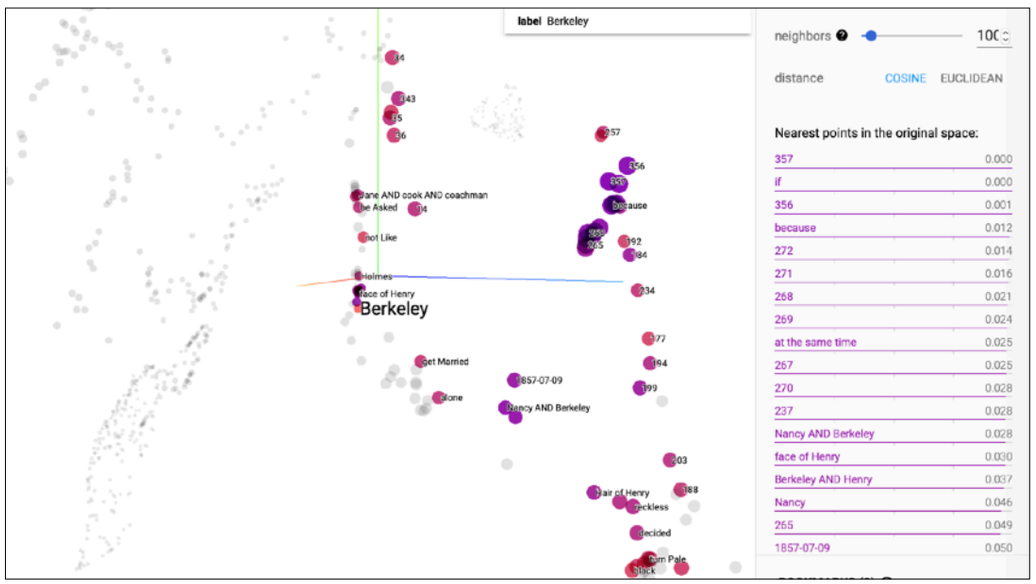
双曲空間版(source含めて学習)



ALBERT版(source含めて学習)

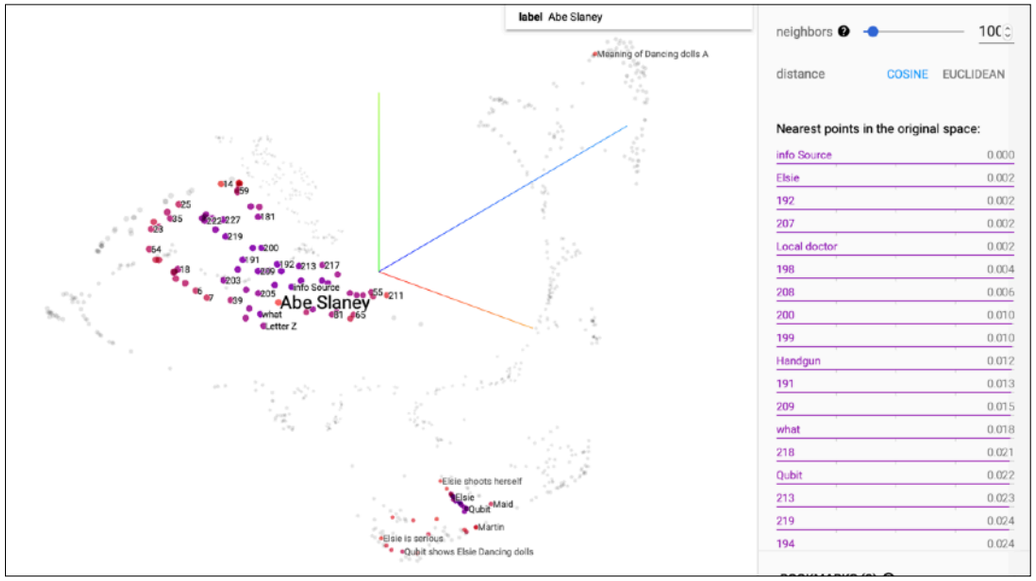


双曲空間版(sourceを含めず学習)

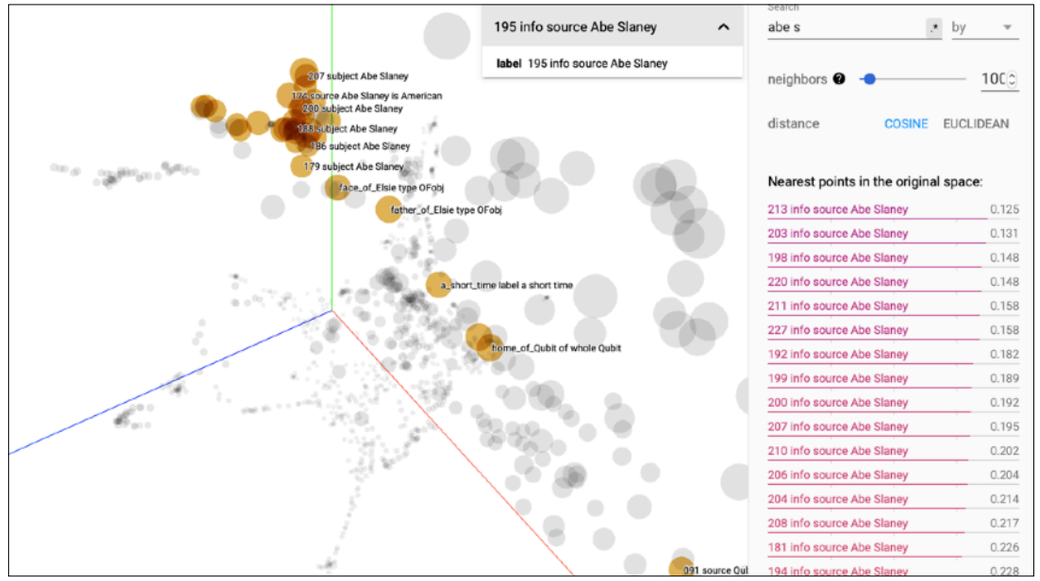


踊る人形

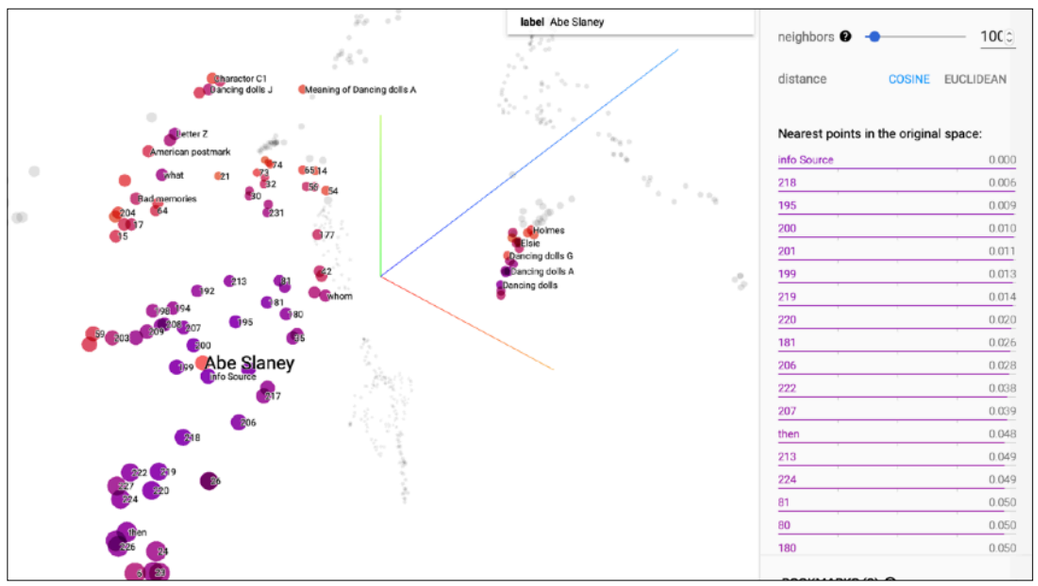
双曲空間版(source含めて学習)



ALBERT版(source含めて学習)



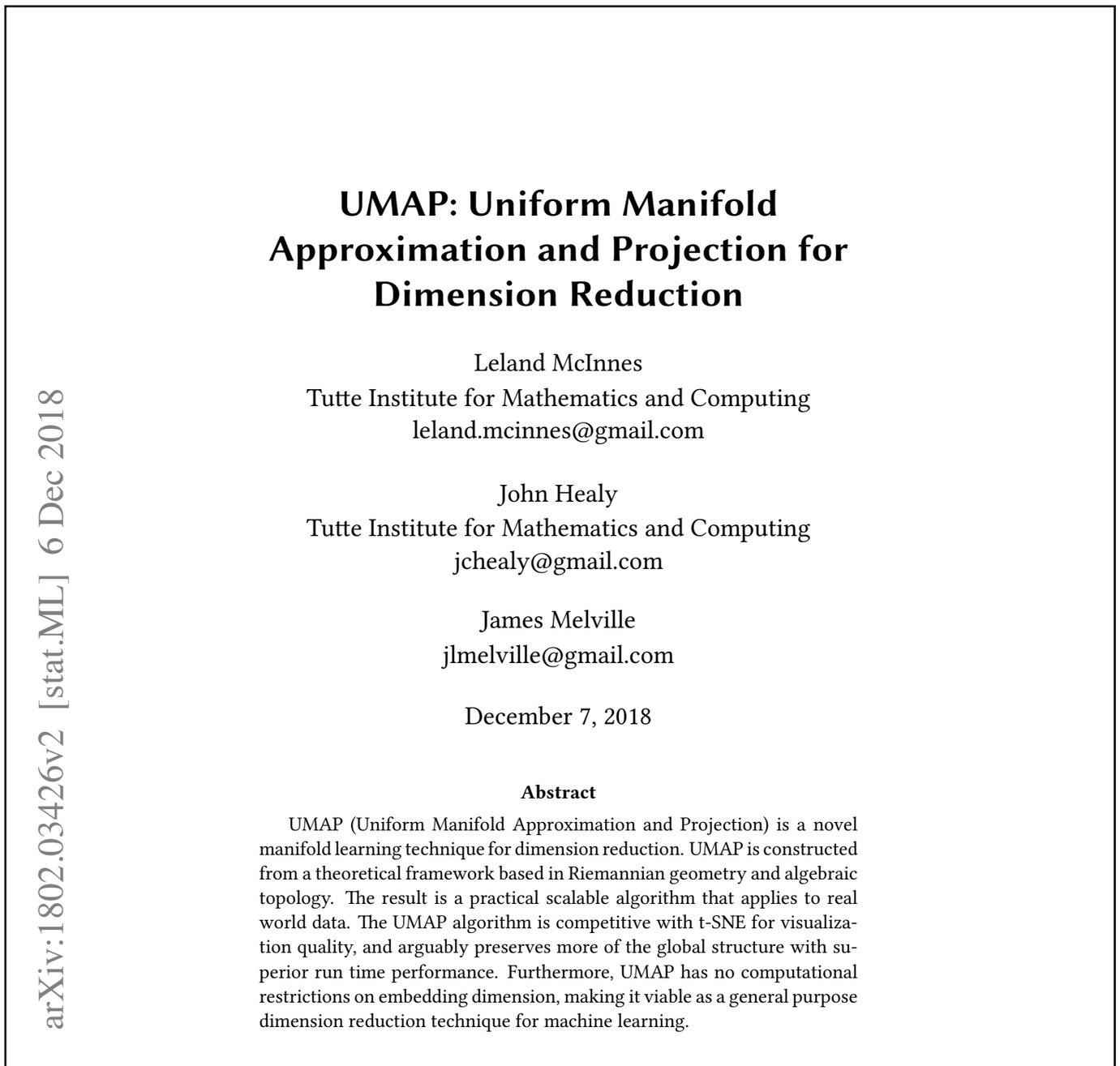
双曲空間版(sourceを含めず学習)



オリジナルツールによる可視化

UMAP

Leland McInnes, John Healy, James Melville, "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction", arXiv:1802.03426v2



UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction

Leland McInnes
Tutte Institute for Mathematics and Computing
leland.mcinnes@gmail.com

John Healy
Tutte Institute for Mathematics and Computing
jchealy@gmail.com

James Melville
jlmelville@gmail.com

December 7, 2018

Abstract

UMAP (Uniform Manifold Approximation and Projection) is a novel manifold learning technique for dimension reduction. UMAP is constructed from a theoretical framework based in Riemannian geometry and algebraic topology. The result is a practical scalable algorithm that applies to real world data. The UMAP algorithm is competitive with t-SNE for visualization quality, and arguably preserves more of the global structure with superior run time performance. Furthermore, UMAP has no computational restrictions on embedding dimension, making it viable as a general purpose dimension reduction technique for machine learning.

- **Uniform Manifold Approximation and Projection (UMAP) は t-SNE と同様に, 与えられた点列について次元削減 (次元圧縮) を行って 2, 3 次元ユークリッド空間に写すことで, 可視化を行ってくれる**

- 双曲空間への埋込結果の可視化は、”Poincare Ball”を用いるべきかもしれないが、UMAPへ双曲線距離計算を指定することにした

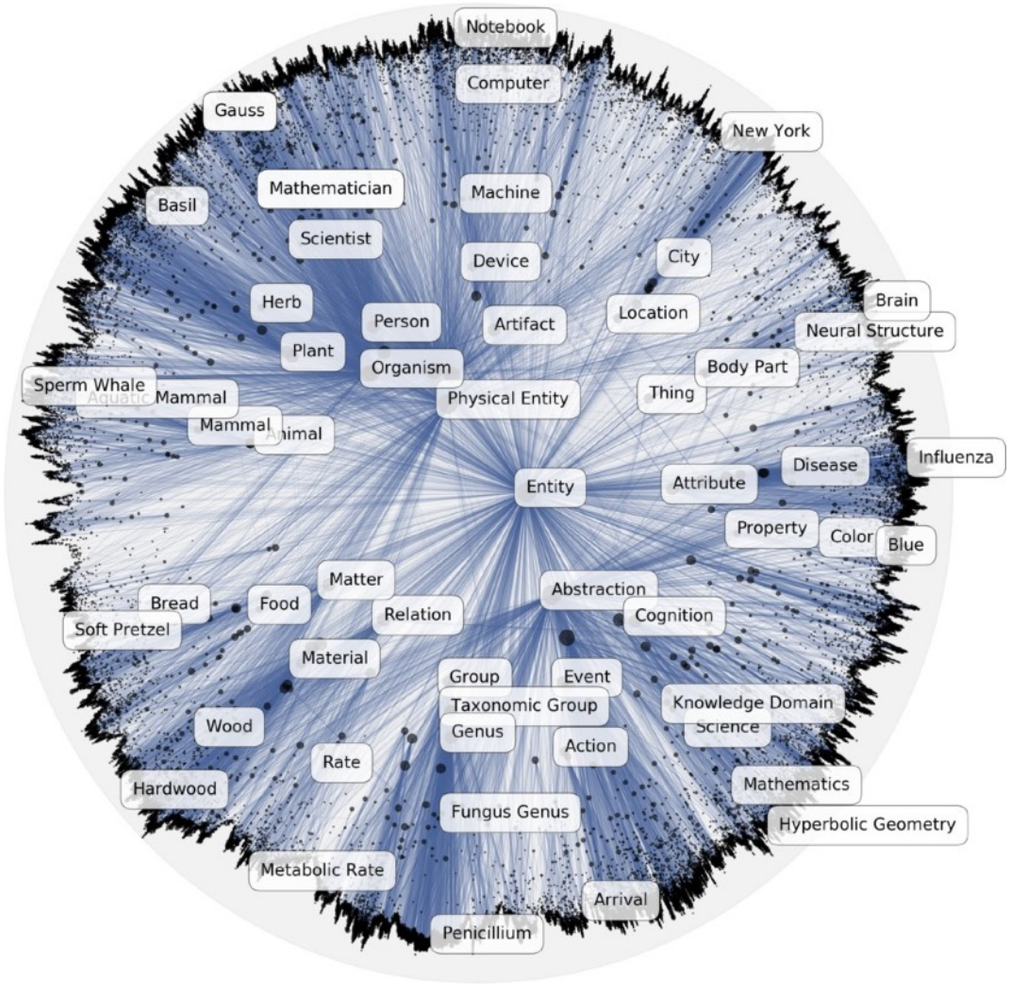


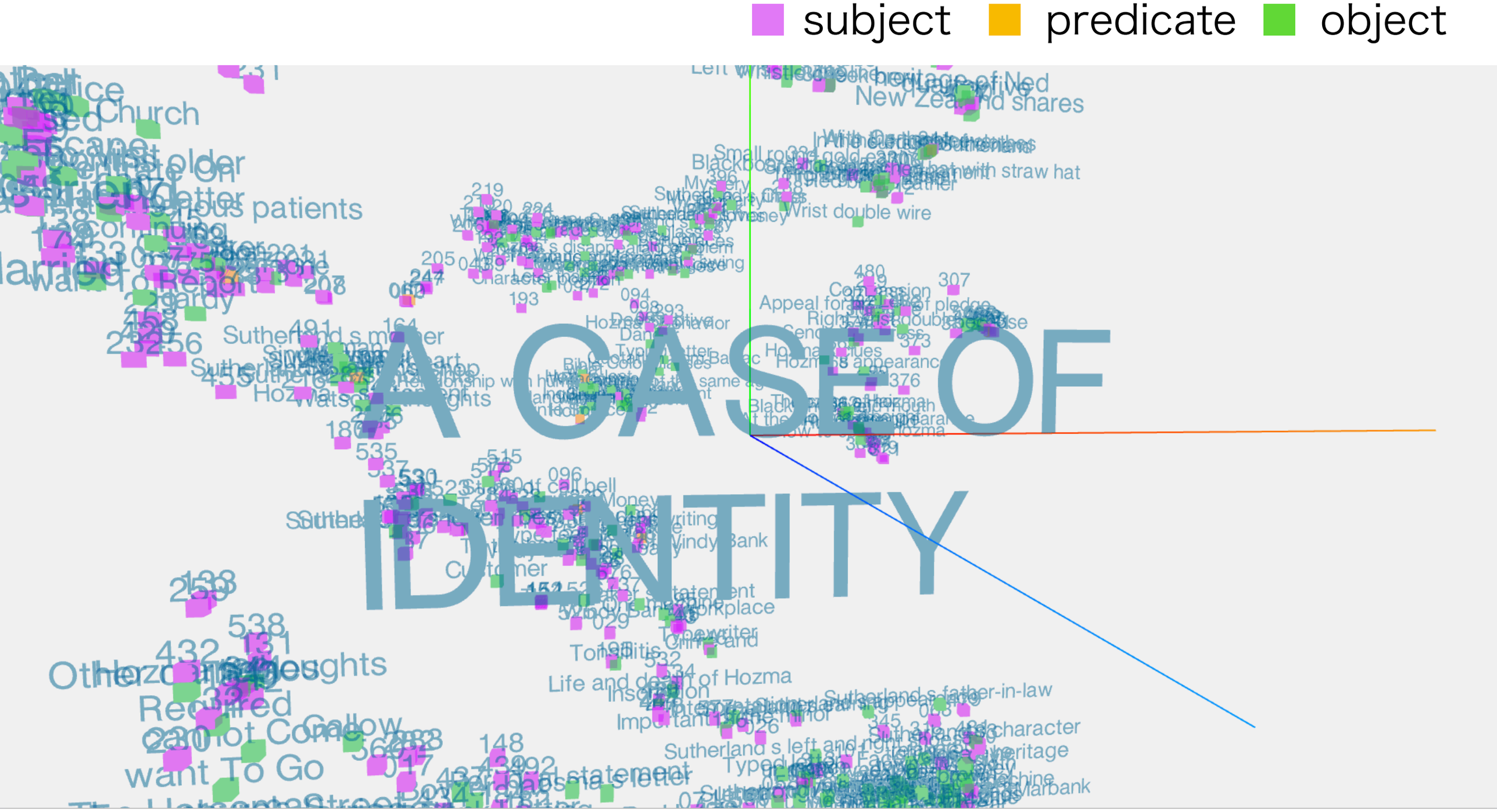
図 Facebook ResearchによるPoincare Embedding

双曲線距離計算

UMAP-JSへ指定して可視化を実施

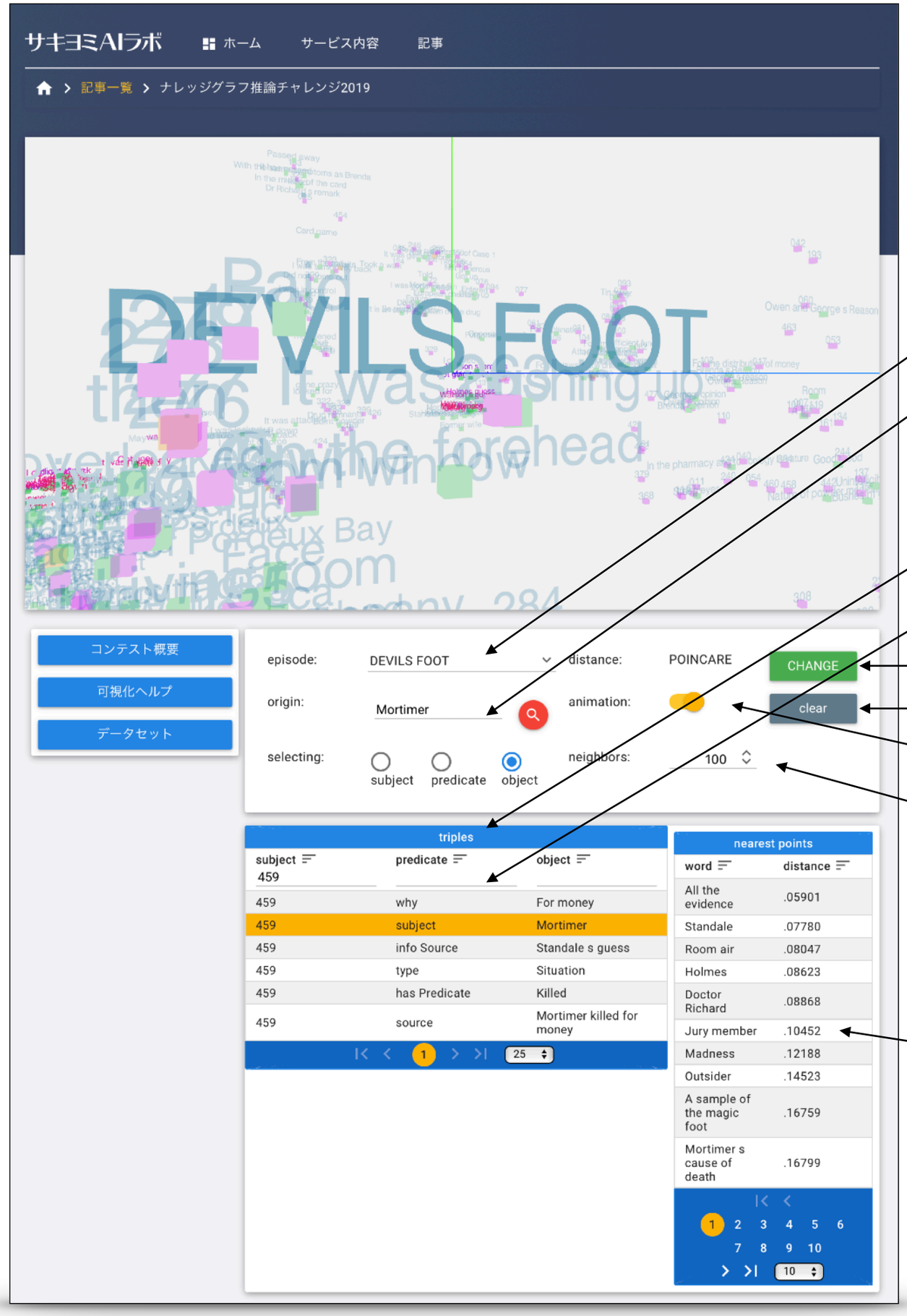
測地線の長さ　$$d(x,y) = arccosh\left(1 + 2\frac{\|x-y\|^2}{(1-\|x\|^2)(1-\|y\|^2)}\right)$$

```
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)))
    );
}
```



【マウス操作】

- オービット：ドラッグ
- ズーム: ホイール
- パン: 右ボタンドラッグ



・エピソード 選択
・探索ワード

・triplesソース
・絞込みフィルタ

・エピソード切替
・探索ワードクリア

・アニメーション

・表示近傍数

・originからの
近傍ワードと
距離の一覧

まだらの紐 (1/2)

ヘレンを殺したのは誰か?

Roylott

nearest points	
word	distance
reward of request	.25736
wall of right building of mansion of Roylott	.27004
wall of bedroom of Helen	.27963
floor of bedroom of Julia	.28698
window of bedroom of Julia	.29394
smell of Indian cigarettes	.31966
intense fear	.32595
safe	.33845
four walls of bedroom of Julia	.34016
sister of mother of Helen	.34734

Julia

nearest points	
word	distance
Helen	.10816
dog whip	.16756
small dish	.20711
snake	.21618
milk	.21772
rope of bell	.27361
price of asset of mother of Helen	.28208
Roma	.28574
Percy Armitage	.29791
whistle	.31971

Helen

nearest points	
word	distance
Julia	.10816
snake	.18992
milk	.19176
small dish	.20727
dog whip	.22099
Roma	.22108
rope of bell	.22134
Percy Armitage	.29045
subject	.30353
price of asset of mother of Helen	.31329

まだらの紐 (2/2)

ヘレンを殺したのは誰か?

small

nearest points	
word	distance
simple	.35107
be Broken	.36832
louder	.49731
speckled	.57476
modified	.63165
has Property	.65547
filled With	.67225
made By Iron	.72004
hotel	.84027
ruined	.94607

speckled

nearest points	
word	distance
be Broken	.31000
simple	.33874
modified	.35284
has Property	.37188
louder	.53517
small	.57476
filled With	.71595
thick	.73427
disappeared	.78700
strapped	.80573

踊る人形

暗号を解け

Abe Slaney

nearest points

word	distance
info Source	.12065
Letter Z	.93625
what	1.04966
Elsie	1.07054
Past of Elsie	1.07756
Dancing dolls	1.10261
Dancing dolls A	1.11398
Horse boy	1.11690
Dancing dolls J	1.13066
Handgun	1.15084

Dancing dolls

nearest points

word	distance
Dancing dolls E	.25870
Handgun	.27188
Dancing dolls J	.36389
Dancing dolls Y	.36607
Statement	.37269
Dancing dolls A	.38515
Dancing dolls B	.39061
subject	.46159
Horse boy	.49150
Dancing dolls G	.49915
Signs of evil	.50864
Watson	.52422
close	.52640
Sentence I	.55867
Qubit	.57287
Character C2	.57853

filter: dolls

triples

subject	predicate subject	object dolls
167	subject	Dancing dolls Y
151	subject	Dancing dolls F
047	subject	Dancing dolls B
149	subject	Dancing dolls
162	subject	Dancing dolls X
161	subject	Dancing dolls X
153	subject	Dancing dolls A
061	subject	Dancing dolls D
167a	subject	Dancing dolls Y
051	subject	Dancing dolls C
005	subject	dolls
033	subject	Dancing dolls
050	subject	Dancing dolls C
168	subject	Dancing dolls Y
001	subject	Dancing dolls A
158	subject	Dancing dolls D
156	subject	Dancing dolls D
62a	subject	Dancing dolls C
147	subject	Dancing dolls
160	subject	Dancing dolls X
008	subject	Dancing dolls A
062	subject	Dancing dolls D
165	subject	Dancing dolls X
072	subject	Dancing dolls E
166	subject	Dancing dolls X

1 2 25

背中の曲がった男

バークリはなぜ死んだのか?

Berkeely

nearest points	
word	distance
face of Henry	.14022
Nancy	.14384
Jane	.25368
Murphy	.32334
Henry	.33146
Nancy AND Henry	.33842
Berkeley AND Nancy	.34981
subject	.35102
Berkeley AND Morrison	.39323
man	.41347

apoplexia

nearest points	
word	distance
fear	.66807
wrinkle	.85428
atonement	.95891
guilt	1.10981
why	1.21190
furnace lattice	1.22176
mongoose	1.36803
suffer	1.39367
help	1.41637
overlook	1.44758

Holmes

nearest points	
word	distance
info Source	.39258
Morrison	.52686
Henry	.59184
subject	.65660
Nancy AND Henry	.68192
Jane	.70450
Murphy	.71153
Teddy	.71823
Berkeley AND Morrison	.75273
Nancy	.86157

悪魔の足

各人物を殺したのは誰か?

Mortimer

nearest points	
word	distance
All the evidence	.05901
Standale	.07780
Room air	.08047
Holmes	.08623
Doctor Richard	.08868
Jury member	.10452
Madness	.12188
Outsider	.14523
A sample of the magic foot	.16759
Mortimer s cause of death	.16799

Room air

nearest points	
word	distance
Mortimer	.08047
Holmes	.09994
A sample of the magic foot	.10364
Jury member	.11390
Madness	.11433
Doctor Richard	.12365
All the evidence	.12985
Drug	.13651
Outsider	.15377
Standale	.15673

Standale

nearest points	
word	distance
All the evidence	.06823
Mortimer	.07780
Doctor Richard	.09623
something	.11419
Holmes	.11901
Madness	.15514
Jury member	.15637
Room air	.15673
Outsider	.15739
Porter	.16723

同一事件

花婿はなぜ消えたか?

Hozma

nearest points	
word	distance
All the evidence	.05901
Standale	.07780
Room air	.08047
Holmes	.08623
Doctor Richard	.08868
Jury member	.10452
Madness	.12188
Outsider	.14523
A sample of the magic foot	.16759
Mortimer s cause of death	.16799

Room air

nearest points	
word	distance
Mortimer	.08047
Holmes	.09994
A sample of the magic foot	.10364
Jury member	.11390
Madness	.11433
Doctor Richard	.12365
All the evidence	.12985
Drug	.13651
Outsider	.15377
Standale	.15673

Standale

nearest points	
word	distance
All the evidence	.06823
Mortimer	.07780
Doctor Richard	.09623
something	.11419
Holmes	.11901
Madness	.15514
Jury member	.15637
Room air	.15673
Outsider	.15739
Porter	.16723

https://sakiyomi.ai/service/sherlock/2019/console.xhtml

結 論

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。

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。