SDML HW1 Task2

組員：
r07922035 李漪莛
r07922119 吳達懿
b04902113 陳柏叡

Preprocessing

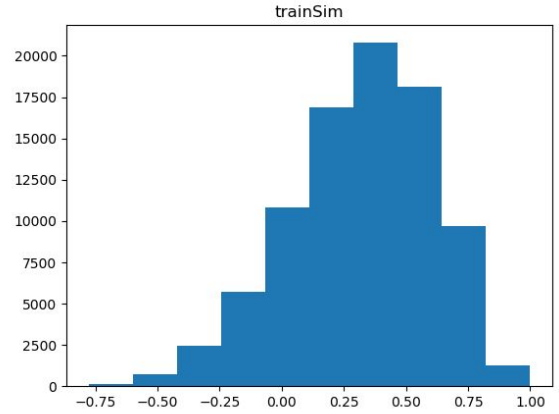
去掉標點符號, latex, stopwords, stem (ing, ed 結尾), title, abstract 等字。

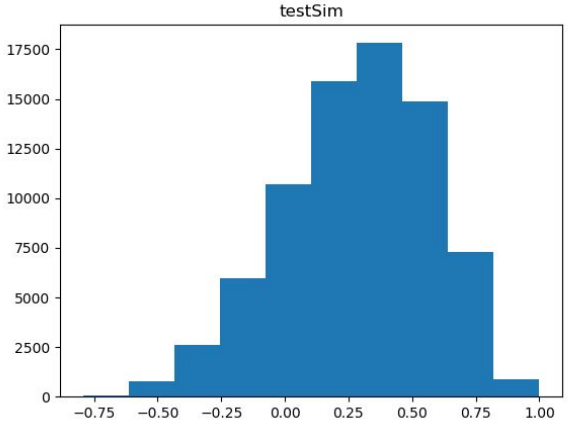
Embedding

嘗試方法: Doc2vec , Doc2vec with pretrained word2vec, GCN, naïve based, word2vec, fasttext

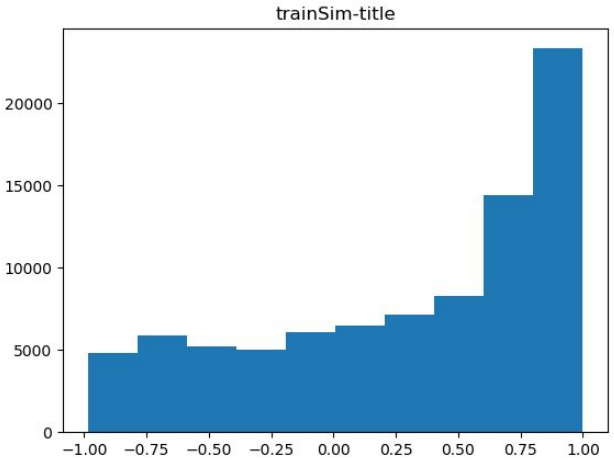
1. Doc2vec

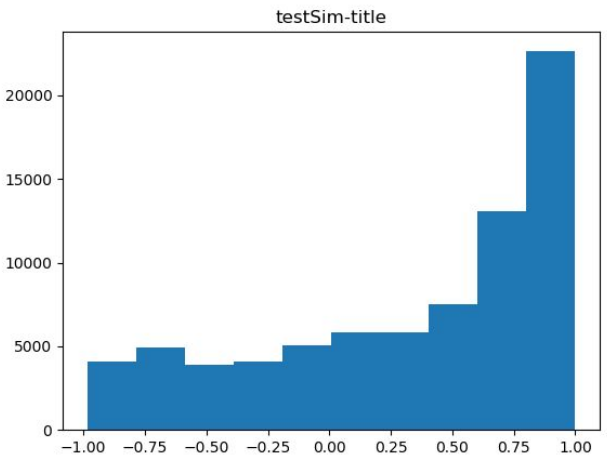
有分成用這個 task 的文本 train好的model， 和網路上pretrained的model。
此次實驗中， 以Doc2vec做300維度的embedding所得到的training， test similarity。 testSim的中位數是0.30, 而trainSim的中位數是0.34。但其分佈十分的相似。





由於其分佈過於相似， 於是分別針對title, abstract做doc2vec。發現title的部分， testSim中位數反而比trainSim高。而abstract則就跟全部接再一起丟相類似。因此可以判斷出， title的資訊與是否相連較不相關， 從後面的naïve base也可以得到這個結論。





2. GCN base:
我們利用word2vec 以及 doc2vec所訓練出來的embedding當作gcn的feature，並且將文章用關鍵字的方式分成五類，分別是代表五種物理不同的領域，並將此當作label去training GCN。在GCN下training準確率約為60%。並用lgboost作為clf. 第二種方法仍是用doc2vec的embedding當作feature。但是利用其連結到的點的embedding的平均當作label。

3. Naïve base:
直接判斷title中，最長共同字符。在training data中，最長共同單字中位數5個字符，在testing data中，中位數也落在5的位置。

4. Word2vec:
合併所有文本，訓練 word2vec model，每個字 300 維，訓練好後，將 title+abstract 或 title 或 abstract 分別丟入 model，取每個字轉成向量後的平均，當做這篇文章的 embedding。

5. Fasttext:
利用網路上提供的 pretrained model，訓練在 wikipedia 和 news data 上，共有 1M 個字的 word vectors，實際轉換後，發現約有 80% 的字能在 pretrained model 中找到。將 title+abstract 或 title 或 abstract 分別丟入 model，取每個字轉成向量後的平均，當做這篇文章的 embedding。

Other self-defined features

1. 兩個 embedding 的 cosine similarity, enclidean distance, correlation
2. 兩個 embedding 向量 element-wise 的乘積（同樣是300維）
3. 兩個 embedding 向量的差

上傳後 public score 介在 0.501~0.503 之間，和原先差距不大。

Retrofitting

利用 retrofitting 和 train.txt 中的 link，調整文本的 embedding，測試過 alpha:beta = 1:1或1:0.1。
alpha: （調整後的向量 - 原向量）的平方和形成的loss的係數
beta: （鄰居的向量 - 調整後的向量）的平方和形成的loss的係數

Negative sampling

Random walk 2 steps in the graph (train.txt). If there is no link between source and destination, then add this pair into negative sample.

Classifier

1. Similarity: 在這個 task 中通常效果不好
2. NN:
   2 layers, each layer with 512 units, ReLU, and dropout(0.5).
   Use median as threshold
3. Lgboost

Results

- 沒有 retrofitting，各種 embedding 和對應的 classifier 的結果比較

	Title和abstract一起embedding (dim = 300)	Title和abstract分開embedding再concat起來(dim = 600)	Similarity (Median)	Lgboost	NN
Doc2vec	V		0.49793	0.50289	0.50193
Doc2vec Pretrained	V		0.49720	-	-
Naive	V		0.5009	-	-
GCN	V		-	0.50219	-
Word2vec	V		0.50339	-	0.49484
Word2vec		V	-	-	0.49980
Fasttext	V		0.49871	-	0.50120
Fasttext		V	0.50131	-	-

- 幾乎所有情況結果都在0.5附近，而且差距都不大，甚至最極端的情況“0.49484“，將01相反後，可以得到最好的正確率，讓我們很難判斷哪一種組合的結果較好。
- retrofitting 的比較
  當 alpha:beta = 1:0時表示沒有retrofitting
  都是使用title和abstract一起embedding成300維的做法

	alpha:beta	NN
Word2vec	1:1	0.49765
Word2vec	1:0.1	0.49489
Word2vec	1:0	0.49484
fasttext	1:0.1	0.50295
fasttext	1:0	0.50120
doc2vec	1:0.1	0.49556
doc2vec	1:0	0.50193

- 發現在大部分的情況下，有retrofitting會有稍好的結果，但是doc2vec則沒有，而且其實就算也有進步也進步不大，差距是在random seed的誤差範圍內，因此很難判斷最佳的retrofitting hyperparameters的係數應該要是多少，這是這個task困難的點。

Reference

[1] word2vec: https://radimrehurek.com/gensim/models/word2vec.html
[2] fasttext: https://fasttext.cc/docs/en/english-vectors.html
[3] doc2vec: https://radimrehurek.com/gensim/models/doc2vec.html