多媒體程式設計
文字資料處理

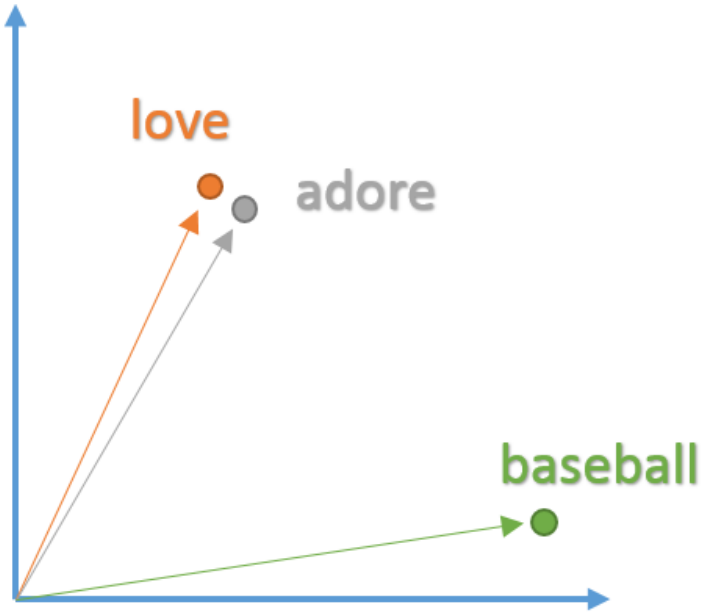
Instructor: 馬豪尚

自然語言處理
文字相似度（Word Similarity）

- 自然語言處理中，我們會透過將單詞乃至文本量化成在向量空間中的向量
  - Bag of word
  - Bag of N-gram
  - One-hot encoding
- 文字向量之間的「距離」即可用來表示文字相似度

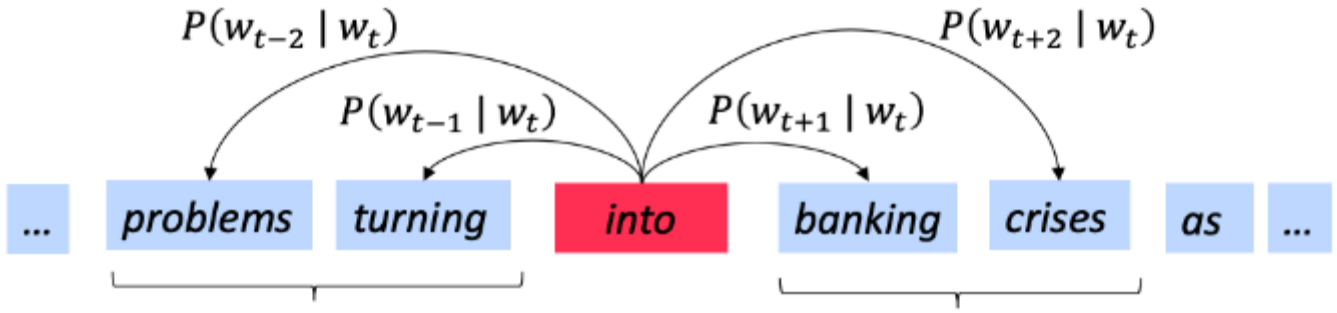
詞嵌入Word Embedding

- 語意相似度資訊
  - 「物以類聚」是我們耳熟能詳的一句諺語
  - 「分布假說」是在語言學的脈絡裡，語言學家認為在相同上下文中一起出現的兩個單詞會有相似的意義
- 詞嵌入的優點
  - 維度縮減（ diemension reduction ）
  - 上下文相似性（ context similarity ）



Word2vec

- 詞向量的目的
  - 相似的詞要聚集在一起
  - 向量的方向對應相對關係
- 運用「分布假說」來設計，以機率來預測在一個句子中某一個位置可能會出現的詞



Gensim套件

- Gensim套件
  - 使用unsupervised machine learning處理原始的、非結構化的文本（text），藉由統計training documents在語料庫中字與字之間組合或共同出現(co-occurrence)的模式
  - 安裝:
    - pip install --upgrade genism

Gensim核心概念

- Document(文章)
  - 一群文字。可以是 140 個字的簡短推文、單個段落（即期刊文章摘要）、新聞文章或書籍中的任何內容。
- Corpus(語料庫)它在Gensim中扮演兩個角色
  - 當做 model 的輸入。Model 藉由 training corpus 來初始化模型內部的參數。
  - 可以被處理後並組織成document。當一個關鍵字提取模型被訓練後，可以從新的 document (在training corpus中未看到的 document) 中提取 新的關鍵字。

Gensim核心概念

- Vector為具有數學意義的 document
  - 帶有 features 的 vector 正好可以滿足這個需求，而一個 single feature 可以被認為是一個 question-answer 對，questions 通常以整數的方式標示，answer 只被允許是浮點數 (single floating point number)。
    - Document 中單詞splonge出現幾次? 0次 (bag of words詞頻統計)
      - (1, 0.0)
    - Document 中由多少段落組成? 2個
      - (2, 2.0)
    - Document 使用了多少種字體？ 5種
      - (3, 5.0)

Gensim核心概念

- Model
  - 一種將 vector 從一種表示轉換為另一種表示的演算法
  - 透過訓練的過程學習不同的特徵來達到
  - 一旦你創建了 model，你就可以用它做各種很酷的事情。 例如，透過 Tf-Idf model轉換整個語料庫並對其進行索引，為相似性查詢做準備

詞嵌入向量

- 常見的word embedding模型
  - Word2vec
    - 是用來產生詞向量的相關模型。用來訓練以重新建構語言學之詞文本，輸入以詞為單位，並且需猜測相鄰位置的輸入詞
  - GloVe (2014)
    - 預測字跟字同時出現的次數（co-occurrence count）來訓練。也是滿早期的model，在小一點的 dataset 也能有效訓練。
- 下載預先訓練好的詞嵌入向量(Glove)
  - https://github.com/stanfordnlp/GloVe

Gensim 載入預訓練好的模型

- 載入genism 的word2vec模組
  - from gensim.models import word2vec
- 把 GloVe word vectors 儲存格式轉為Gensim可以用的 word2vec 格式
  - 指定glove文件輸入位置
    - glove_file = './data/glove.6B.100d.txt'
  - 指定word2vec格式輸出文件位置
    - word2vec_glove_file = './data/glove.6B.100d.word2vec.txt'
  - 用gensim提供的函數轉換格式
    - glove2word2vec(glove_file, word2vec_glove_file)

Gensim 載入預訓練好的模型

- 載入模型
  - model = KeyedVectors.load_word2vec_format(word2vec_glove_file)
- 使用模型
  - model.函數名稱()

Gensim 使用載入的模型

- 查詢指定詞的詞嵌入向量
  - wv = model.wv[word]
- 找到與指定詞top-K最相似的詞和 similarity score
  - model.most_similar(positive[word])
- 計算詞跟詞的相對關係，依照給定的一組兩個詞的關係，找出跟指定詞有相對類似關係的詞
  - model.most_similar(positive=[y_1, x_2], negative=[x_1])
    - 如果美國相對於漢堡，那加拿大相對於什麼?
    - 日本相對於日文，等於法國相對於什麼?

us-ham=can-?
us-ham-can = -?
-us +ham +can = ?
han+can -us = ?

練習1

- 用gensim載入glove預先訓練好的word embedding
- 讀取bbcnews文件，做好英文斷詞之後，建立一個詞典包含文件內的詞 → {word: word embedding}
  - 在glove裡取不到的詞就忽略

使用Gensim自己訓練Word2vec

- 訓練模型
  - model = word2vec.Word2Vec(參數)
- 設定模型參數
  - train_data, (訓練資料)
  - min_count=1, (詞頻少於 min_count 的詞不會參與訓練)
  - vector_size=100, (詞嵌入向量的維度)
  - workers=8, (訓練並行的數量)
  - epochs=10, (訓練的迭代次數)
  - window=10, (選取周圍詞的數量)
  - sg=0, (0或1兩種，0是CBOW，1是Skip-Gram)
  - seed=546, (亂數種子，模型隨機選取起始點)
  - batch_words=1000, (每次給予多少詞來訓練)

已訓練模型儲存和調用

- 模型儲存
  - model.save(“model_name")
- 模型讀取
  - from gensim import models
  - model = models.Word2Vec.load(" model_name ")
- 模型接續訓練新詞
  - model.train([["hello", "world"]], total_examples=1, epochs=1)
    - 第一個參數為要訓練的句子
    - total_examples為要訓練的句子數量
    - Epochs為迭代次數

使用已訓練模型

- 查詢指定詞的詞嵌入向量
  - wv = model.wv[word]
- 找到與指定詞top-K最相似的詞和 similarity score
  - model.wv.most_similar(positive[word])
- 計算詞跟詞的相對關係，依照給定的一組兩個詞的關係，找出跟指定詞有相對類似關係的詞
  - model.wv.most_similar(positive=[y_1, x_2], negative=[x_1])

使用已訓練模型

- 計算兩個字詞的相似度
  - model.wv.similarity(word1, word2)

練習2

- 用gensim的word2vec模型訓練一個中文的模型
- 訓練模型
  - 使用作業一中你取出的兩千篇維基文件的內容來訓練
  - 讀取文章內容後做斷詞和清理，將所有文章的內容變成一堆句子，輸入模型進行訓練
- 儲存模型
- 載入儲存的模型
  - 分析詞與詞的關係