香PYCON港 HONG KONG

AxonLabs

Chinese NLP with Open Source Tools in Python

Albert Au Yeung

Axon Labs Limited

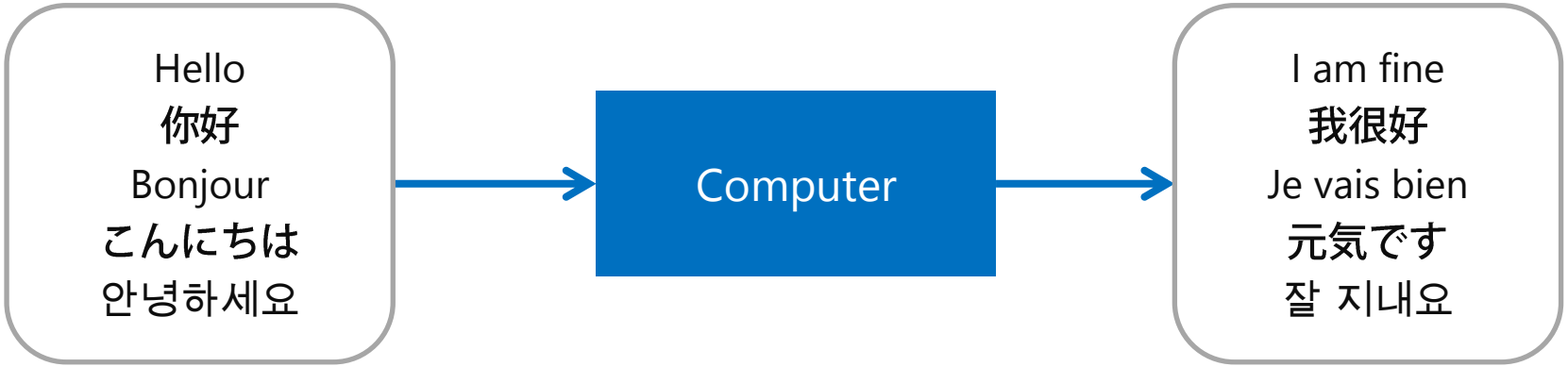
2015-11-07

About Me

- Co-founder of Axon Labs Limited (with Joey Cham)
- Compute science researcher (previously at NTT Labs, ASTRI, Huawei, on social network, machine learning, natural language processing)
- Part time lecturer at CUHK
- Android and Python developer

What is Natural Language Processing?

Making the computer understand natural language inputs and generate natural language outputs



Why NLP?


Google Translate
English
Spanish
hello
hola
Dictionary
hello
interjection:
1. ¡hola
2. ¡caramba
3. ¡aló
4. ¡diga
5. ¡oiga
Enter Conversation Mode
Alpha!
Google Translate
?
hola
hola como estas
hello how are you
Speak more English
I am fine thank you
Estoy bien gracias
Responder en español

Why NLP?

REUTERS

» Print

Could Twitter predict the stock market?

Thu, Feb 16 2012

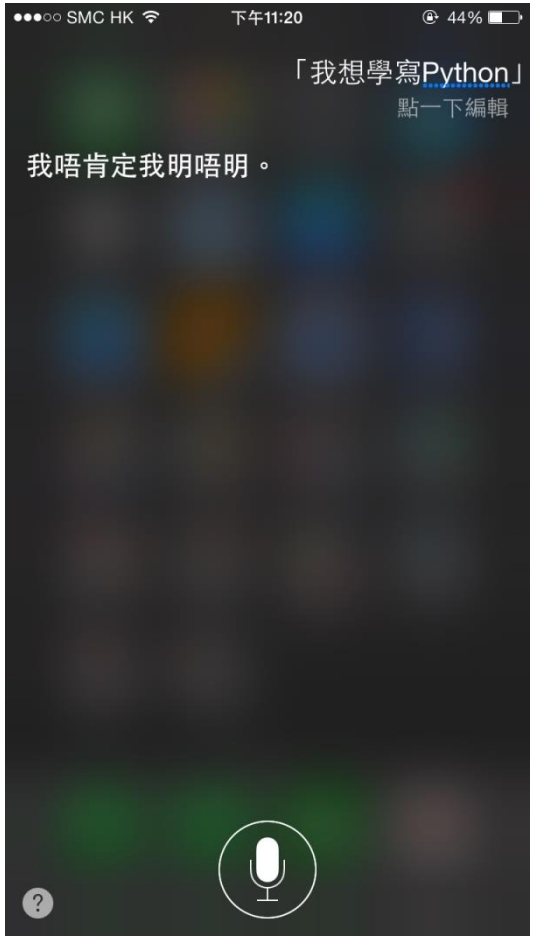
By Chris Taylor

NEW YORK (Reuters) - When Richard Peterson first started meeting with hedge funds about eight years ago to pitch using social media to predict market movement, investment managers looked at him as if he had just arrived from outer space.

Back then, what he was pitching them seemed pretty insane. Peterson, managing director of Santa Monica-based MarketPsych, said that social media can be mined for data about what people are thinking and feeling. And that, in turn, could translate into powerful investment ideas.

Why NLP?

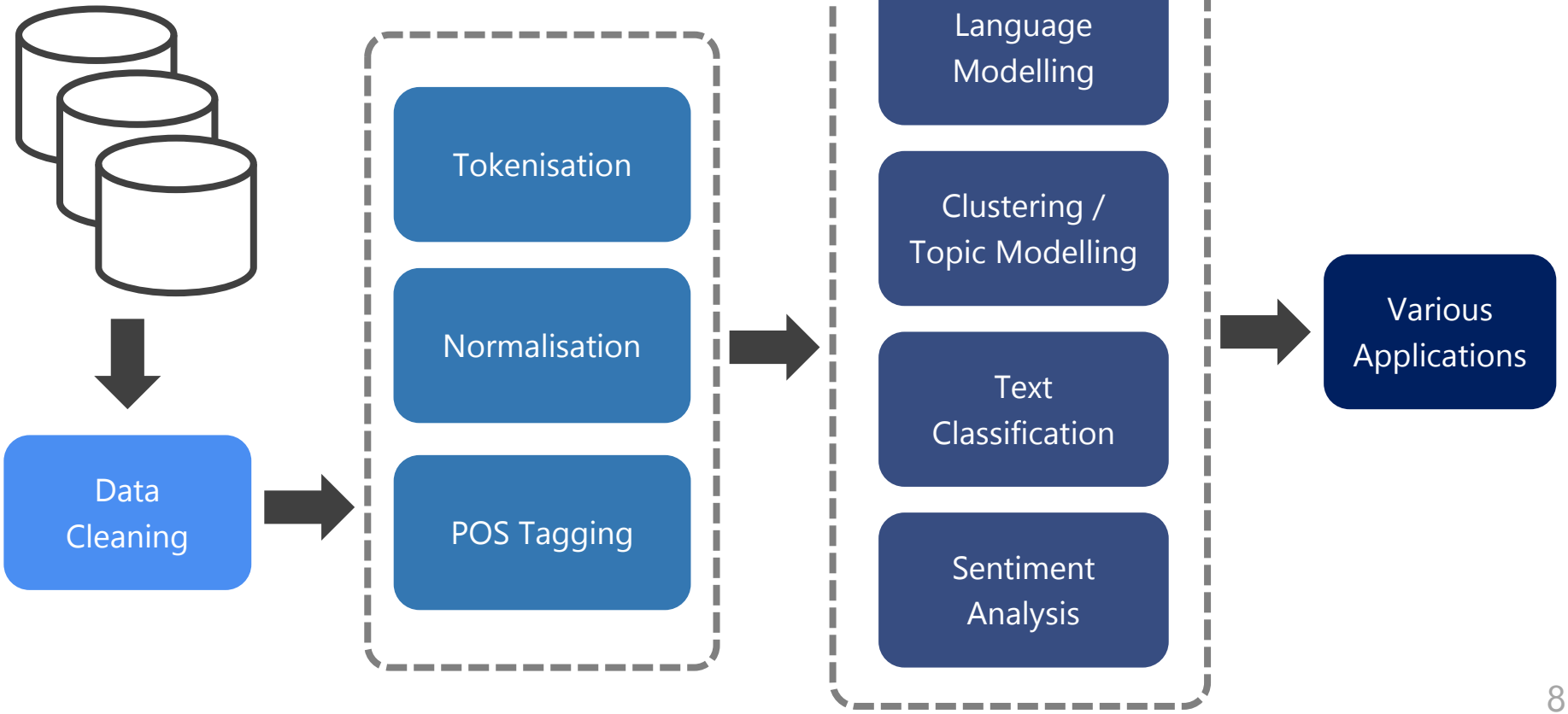

SMC HK
下午11:20
45%
「香港有咩好玩」
點一下編輯
我好想知喺主題公園入面係咪
真係可以搵番童真？
15 個結果
離島區附近
香港迪士尼樂園
大嶼山竹篙灣
13 公里
儿童游樂場(宏樂街)
元朗區
20 公里
美景游樂場
青康路和青衣路的交叉口附近
6.4 公里
馬鞍山游樂場
恆康街1號
12 公里


SMC HK
下午11:20
44%
「我想學寫Python」
點一下編輯
我唔肯定我明唔明。

Why NLP?

- Support searching and retrieval of documents
- Extract key sentences from documents
- Identify similar documents
- Identify major topics of in a group of documents
- Perform translation from one language to another
- Detect sentiment in user comments
- …

The NLP Pipeline



Chinese Language NLP

NLP in CJK (Chinese, Japanese, Korean) is more challenging because of the need to segment a sentence into words (tokens)

The discovery that neutrinos switch between different “flavours” has won the 2015 Nobel Prize in physics.

the, discovery, that, neutrinos, switch, between, different, flavours, has, won, the, 2015, nobel, prize, in, physics

2015年的諾貝爾物理學獎授予日本東京大學宇宙射線研究所所長梶田隆章和加拿大科學家阿瑟·麥克唐納。

?

Jieba (結巴)

AxonLabs

- https://github.com/fxsjy/jieba
- Chinese Segmentation module
- Support 3 different types of segmentation method (depending on your purpose)
- Allow the use of customised dictionary to improve sensitivity to new words
- MIT License

Jieba Example

```
>>> import jieba
>>>
>>> sentence = u“2015年的諾貝爾物理學獎授予日本東京大學宇宙射線研究所所長梶田隆章和加拿大科學家阿瑟·麥克唐納。”
>>>
>>> words = jieba.cut(sentence)
Building Trie..., from /usr/local/lib/python2.7/dist-packages/jieba/dict.txt
loading model from cache /tmp/jieba.cache
loading model cost  1.02380204201 seconds.
Trie has been built succesfully.
>>>
>>> print " / ".join(words)
2015 / 年 / 的 / 諾貝爾 / 物理 / 學獎 / 授予 / 日本 / 東京大學 / 宇宙 / 射線 / 研究所 / 所長 / 梶田隆章 / 和 / 加拿大 / 科學家 / 阿瑟 / · / 麥克 / 唐納 / 。
```

Jieba Cut Modes

AxonLabs

Jieba provides three different segmentation:

1. Accurate (default): jieba.cut(sentence, cut_all=False)
2. Full: jieba.cut(sentence, cut_all=True)
3. Search Engine Mode: jieba.cut_for_search(sentence)

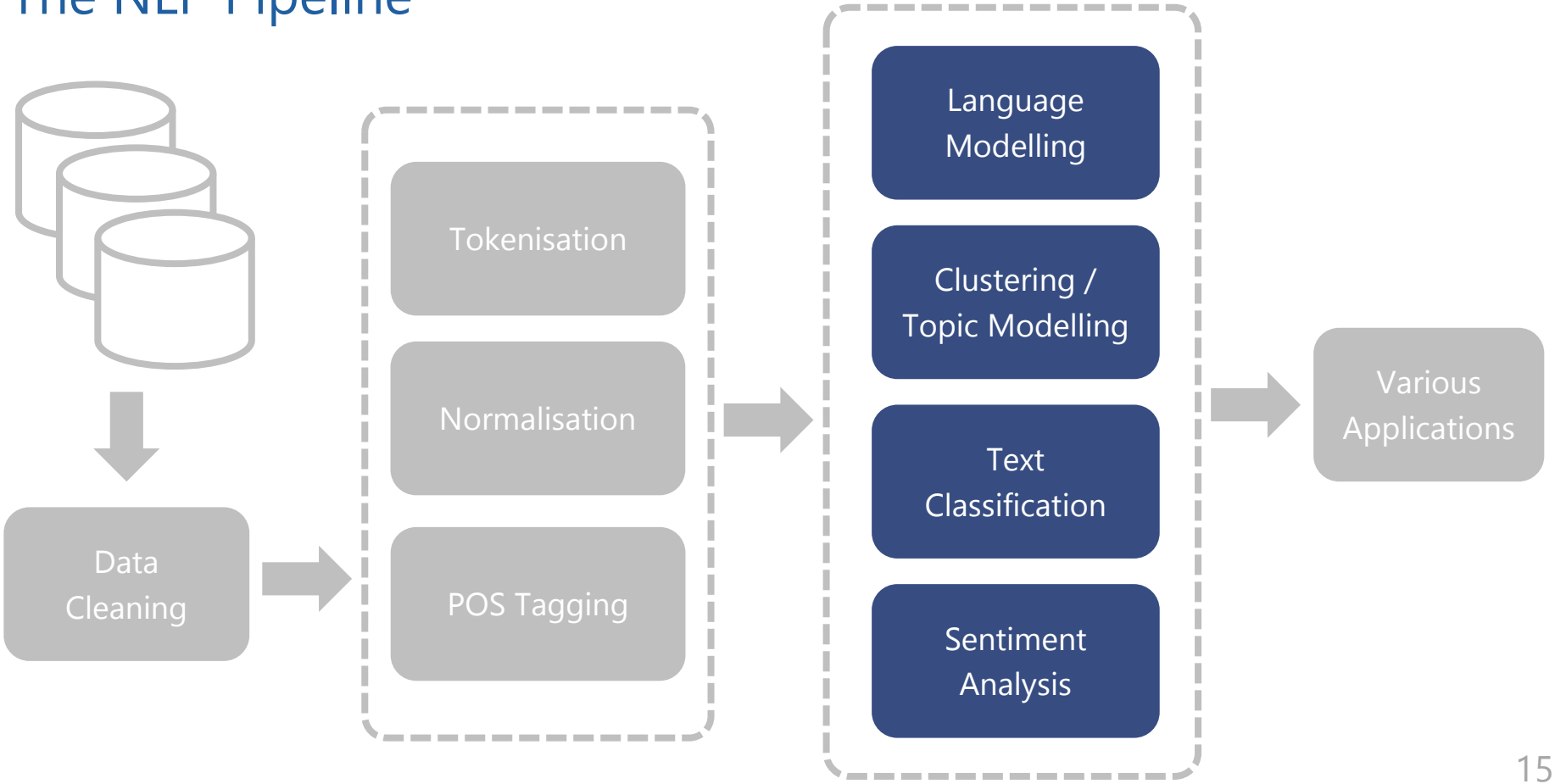
Search engine mode (cut_for_search) is particularly useful for supporting full text search of Chinese documents
(e.g. consider cut_for_search + MySQL Full Text Search)

Jieba Example

```
>>> print " / ".join(jieba.cut(sentence))
2015 / 年 / 的 / 諾貝爾 / 物理 / 學獎 / 授予 / 日本 / 東京大學 / 宇宙 / 射線 / 研究所
/ 所長 / 梶田隆章 / 和 / 加拿大 / 科學家 / 阿瑟 / · / 麥克 / 唐納 / 。
>>>
>>> print " / ".join(jieba.cut(sentence, cut_all=True))
2015 / 年 / 的 / 諾 / 貝 / 爾 / 物理 / 學 / 獎 / 授予 / 日本 / 東 / 京 / 大 / 學 /
宇宙 / 射 / 線 / 研究 / 研究所 / 所 / 長 / 梶 / 田 / 隆 / 章 / 和 / 加拿 / 加拿大 /
科 / 學 / 家 / 阿瑟 /  /  / 麥 / 克 / 唐 / 納 /  /
>>>
>>> print " / ".join(jieba.cut_for_search(sentence))
2015 / 年 / 的 / 諾貝爾 / 物理 / 學獎 / 授予 / 日本 / 東京大學 / 宇宙 / 射線 / 研究 /
研究所 / 所長 / 梶田隆章 / 和 / 加拿 / 加拿大 / 科學家 / 阿瑟 / · / 麥克 / 唐納 / 。
```

What next?

The NLP Pipeline



Example

AxonLabs

As an example, let's analyse the abstracts of the Chinese Wikipedia

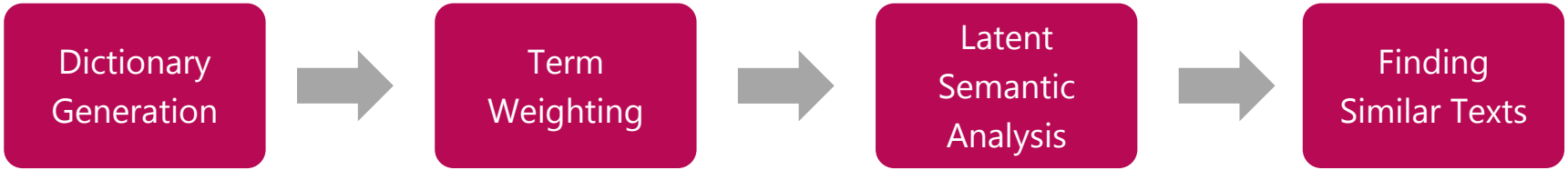
- Wikipedia XML dump available at:
  http://dumps.wikimedia.org/zhwiki/latest/
- The latest dump is ~ 1GB uncompressed

```
<feed>
<doc>
<title>Wikipedia：數學</title>
<url>https://zh.wikipedia.org/wiki/%E6%95%B0%E5%AD%A6</url>
<abstract>數學（Mathematics）是利用符號語言研究數量、結構、變化以及空間等概念的一門學科，從某種角度看屬於形式科學的一種。數學透過抽象化和邏輯推理的使用，由計數、計算、量度和對物體形狀及運動的觀察而產生。數學家們拓展這些概念，...
</abstract>
...
```

Gensim

- https://github.com/piskvorky/gensim/
- A Python library for topic modelling, document indexing and similarity computation, etc.
- Dependencies: numpy & scipy
- Allow multiprocessing or distributed processing
- GUN LGPL License

Example Tasks



Term Weighting

Not all words are of equal importance when doing NLP

- Some words appear very often (e.g. 你、我、的、他們、了)
- Even if they are not stop words, they might not be very useful (e.g. the term 法律 in a bunch of legal documents)
- We need a method to quantify the importance of each term in a document

  → term weighting with TF-IDF

Term Weighting with TF-IDF

AxonLabs

TF = Term Frequency

IDF = Inverse Document Frequency

For Details: https://en.wikipedia.org/wiki/Tf–idf

TF-IDF defines how important a term (word) X is to a document D

- X should appear often in D
- X should NOT appear often in all Ds

Term Weighting in Gensim

AxonLabs

abstracts.txt

歷史 簡稱 指人類 社會過 事件 行動 以及 這些 事件 行為 系統 記錄 詮釋 研究 ...
五代 十國 廣義 狹義 中國 歷史 一段 時期 唐朝 滅亡 開始 宋朝 ...
病毒 英語 一個 核酸 分子 蛋白 質構 非細胞 形態 寄生 生活 ...
楔形文字 底格里斯河 幼發 拉底 流域 古老 文字 這種 文字 ...
考古 源自 古希 臘文 古代 以及 學問 過去 人類 社會 研究 主要 ...
阿基米德 公元前 公元前 古希 臘哲學家 數學家 物理 學家 發明家 ...
稀有 氣體 惰性 氣體 惰性 氣體 稀有 氣體 稀有 氣體 惰性 ...
煉金術 中世紀 一種 化學 哲學 思想 始祖 當代化學 雛形 其目 ...
國內 生產 總值 縮寫 稱國內 生產 毛額 本地 生產 ...
二十四史 中國 古代 各朝 二十四部 史書 總稱 歷來 朝代 ...
...
...

Note: We only extract abstracts with more than 200 characters to filter away some short and trivial items (e.g. XXX 是 XXX 的一座城市). This results in about 12,000 lines in the above file.

Term Weighting in Gensim

```
import codecs
from gensim import corpora, models

# the segmented abstracts are stored in abstracts.txt, one per line
infile = codecs.open("abstracts.txt", "r", "utf-8")
documents = [ l.strip().split(" ") for l in infile ]
# documents is a list of list

dictionary = corpora.Dictionary(documents) # generate the dictionary
dictionary.filter_extremes(no_below=5) # ignore words appear less than 5 times

corpus = [ dictionary.doc2bow(d) for d in documents ] # transform our data
tfidf = models.TfidfModel(corpus) # build the TF-IDF model

# now we have a TF-IDF model of our corpus
```

Term Weighting in Gensim

```
>>> print corpus[0]
[(4, 1), (500, 1), (643, 1), (845, 1), (951, 1), (1101, 2), (1649, 1), (2517, 1),
(2805, 1), (2920, 1), (3296, 3), (4095, 1), (4370, 1), (4807, 1), (5014, 1),
(5524, 1), (6230, 1), (6689, 4), (7056, 1), (7137, 1), (7328, 1), (8219, 1),
(8642, 4), (8748, 1), (8948, 2), (9256, 1), (9689, 2), (9783, 1), ...]
>>>
>>>
>>> for id, cnt in corpus[0]:
...   print "(%s, %d) " % (dictionary[id], cnt),
...
(第二, 1) (同屬, 1) (含義, 1) (詞源, 1) (密切, 1) (簡稱, 2) (理解, 1) (參見,
1) (這些, 1) (年代, 1) (事件, 3) (相關, 1) (今人, 1) (依據, 1) (參考, 1)
(作為, 1) (計量, 1) (歷史學, 4) (詮釋, 1) (重要, 1) (系統, 1) (社會學, 1)
(歷史, 4) (行動, 1) (過去, 2) (人類, 1) (記錄, 2) (藝術, 1) (行事, 1) (以及,
1) (研究, 2) (史學, 1) (什麼, 1) (哲學, 1) (思考, 1) ...
```

Term Weighting in Gensim

```
>>> for id, score in tfidf[corpus[0]]:
...     print "(%s, %.2f) " % (dictionary[id], score),
...
(第二, 0.07)  (同屬, 0.13)  (含義, 0.13)  (詞源, 0.15)  (密切, 0.11)  (簡稱, 0.10)
(理解, 0.11)  (參見, 0.11)  (這些, 0.07)  (年代, 0.06)  (事件, 0.22)  (相關, 0.08)
(今人, 0.15)  (依據, 0.10)  (參考, 0.10)  (作為, 0.07)  (計量, 0.15)  (歷史學,
0.57)  (詮釋, 0.14)  (重要, 0.06)  (系統, 0.06)  (社會學, 0.12)  (歷史, 0.21)  (行
動, 0.09)  (過去, 0.20)  (人類, 0.08)  (記錄, 0.19)  (藝術, 0.08)  (行事, 0.15)
(以及, 0.05)  (研究, 0.12)  (史學, 0.15)  (什麼, 0.12)  (哲學, 0.10)  (思考, 0.13)
(指人類, 0.16)  (學科, 0.10)  (倫理, 0.15)  (提供, 0.07)  (行為, 0.09)  (未來,
0.10)  (一項, 0.09)  (新聞學, 0.16)  (成果, 0.12)  (考古, 0.11)  (隸屬, 0.10)
>>>
```

Finding Similar Documents

Our objective: give a document, find other documents that have similar content

Non-solution: check if two documents have common words

Are the following two documents similar?

電影、演員、 放映、時間 . . .	vs	戲院、售票、 票房、明星 . . .

Latent Semantic Analysis

AxonLabs

To overcome the problem mentioned above, one solution is to find out the relations between words, such that we know “電影” is related to “戲院”, and “演員” is related to “明星”.

LSA achieves this by discovering “latent topics” in the collection of documents using a mathematical tool called singular value decomposition.

Ref: http://nlp.stanford.edu/IR-book/pdf/18lsi.pdf

LSA in Gensim

```
...
dictionary = corpora.Dictionary(documents)
dictionary.filter_extremes(no_below=5)
corpus = [ dictionary.doc2bow(d) for d in documents ]
tfidf = models.TfidfModel(corpus)

# create a LSI model from the tf-idf transformed corpus with 200 topics
lsi = models.LsiModel(tfidf[corpus], id2word=dictionary, num_topics=200)
```

The number of topics should be significantly smaller than the number of words (e.g. 50 to 200), and depends on the size of the corpus.

LSA in Gensim

AxonLabs

After the model is computed, you can check the resultant topics:

```
>>> for i,j in lsi.show_topic(20): print j,
專輯 電視 節目 音樂 演唱 歌曲 唱片 發行 播出 歌手
>>>
>>> for i,j in lsi.show_topic(23): print j,
設計 電影 高速公路 公司 社會 建築 香港 選舉 屋苑 房屋
>>>
>>> for i,j in lsi.show_topic(50): print j,
系統 中國 公司 有限公司 巴士 軟體 語言 上海 文化 細胞
>>>
>>> for i,j in lsi.show_topic(95): print j,
設計 法國 蒙古 大利 廣場 分類 政治 加拿大 建築 國家
>>>
>>> for i,j in lsi.show_topic(177): print j,
條約 科學 病毒 治療 韓國 進行 認證 歐洲 世紀 出版
>>>
```

The top 10 words with the highest scores in that topic will be printed out

It is not uncommon that the top words do not form a coherent topic due to presence of noise or unclean data

LSA in Gensim

AxonLabs

With the trained LSI model, we can find similar documents very easily:

```
# Project the tf-idf corpus onto the LSI topics
corpus_lsi = lsi[corpus_tfidf]

# Generate an index of the LSI space
index = similarities.MatrixSimilarity(corpus_lsi)

# Query the index with a document transformed into the tf-idf space
# This returns a list of tuples in the form of (score, doc_id)
sims = index[tfidf[corpus[40]]]

# sims is a numpy array of float, length = number of documents
# array([ 0.08927711,  0.05128265,  0.07183448, ...,  0.03366148,
#         0.06356593,  0.06204636], dtype=float32)
```

LSA in Gensim

AxonLabs

```
# Sort the similarity values by their scores
sims = sorted(enumerate(sims), key=lambda item: -item[1])

# sims becomes:
# [(40, 1.0000001), (793, 0.89043325), (307, 0.79648018),
#  (6200, 0.7730481), (4888, 0.74860787), (2969, 0.69364899),
#  (1574, 0.68826246), (1893, 0.68282372), (10408, 0.68011516), ...

print " ".join(documents[40][:10])
# 電子學 作用 包括 有源 電子 元器件 例如 真空管 二極體 三極體

print " ".join(documents[307][:20])
# 半導體 器件 半導體 裝置 利用 半導體 材料 特殊 特性 完成

print " ".join(documents[6200][:20])
# 電子 電子 腳環 電子 監控 一種 具有 定時 自動 無線電
```

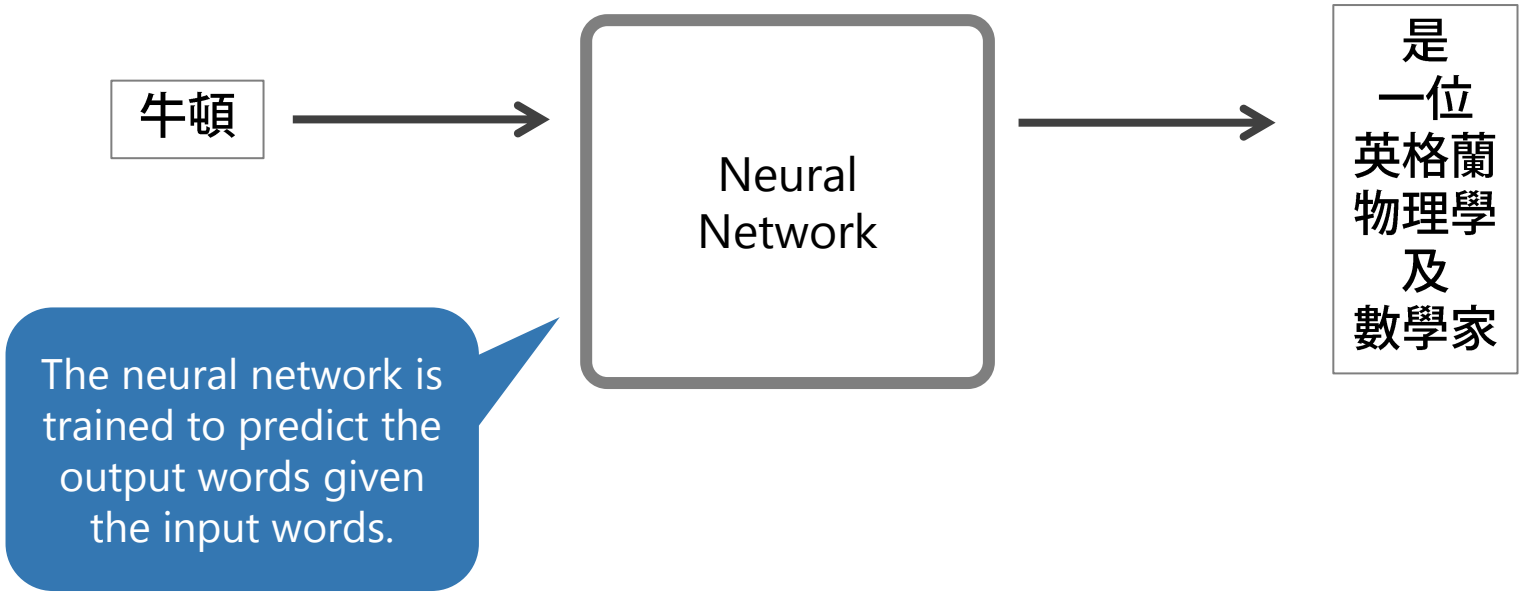
Word2Vec

Word2vec

- A new method for modelling words as a vector in a high dimensional space using deep learning (neural networks)
- In essence, this is a supervised machine learning to predict a word given its surrounding words (context)

Word2vec

Example: Given a sentence "牛頓 是 一位 英格蘭 物理學 及 數學家"



Word2vec in Gensim

```
import codecs
from gensim.models import Word2Vec

infname = “docs.txt”
infile = codecs.open(infname, “r”, “utf-8”)

lines = infile.readlines()
documents = []
for l in lines:
    words = l.strip().split(" ")
    documents.append(words)

model = Word2Vec(documents, size=200, window=5, min_count=5, workers=4)
```

How many words before and after to predict

The length of the word vectors (dimension)

Minimum frequency of a word

Word2vec in Gensim

```
>>> for w, score in  model.most_similar(u"數學")[:5]: print w,
醫學 科學 物理 特聘 哲學
>>>
>>> for w, score in  model.most_similar(u"奧運")[:5]: print w,
金牌 金牌得主 獎牌 跳水 銀牌
>>>
>>> for w, score in  model.most_similar([u"乾隆",u"康熙"])[:5]: print w,
五十三年 雲陽 雍正 貢生 元年
>>>
>>> print model.doesnt_match([u"法國",u"德國",u"數學",u"馬來西亞"])
數學
>>>
>>> print model.doesnt_match([u"足球",u"化學",u"數學",u"醫學"])
足球
>>>
```

Thank you & Happy NLP!

Slides and examples available at:
https://github.com/albertauyeung/pycon2015-chinese-nlp

albertauyeung@axon-labs.com
http://www.axon-labs.com/