Word Sense Disambiguation

Figure 1: Philip Resnik

Using Thesaurus

단어 중의성해소(WSD)에는 여러가지 방법이 있습니다. 첫 번째 다룰 방법은 어휘 분류 사전(thesaurus, 시소러스)을 활용한 방법 입니다. 기존에 구축 된 사전은 해당 단어에 대한 자세한 풀이와 의미, 동의어 등의 자세한 정보를 담고 있기 마련입니다. 이렇게 기존에 구축된 공개되어 있는 사전을 활용하여 중의성을 해소하고자 하는 방법 입니다.

WordNet

WordNet(워드넷)은 심리학 교수인 George Armitage Miller 교수가 지도하에 프린스턴 대학에서 1985년 부터 만들어진 프로그램 입니다. 처음에는 주로

기계번역(Machine Translation)을 돕기 위한 목적으로 만들어졌으며, 따라서 동의어 집합(Synset) 또는 상위어(Hypernym)나 하위어(Hyponym)에 대한 정보가 특히 잘 구축되어 있는 것이 장점 입니다. 단어에 대한 상위어와 하위어 정보를 구축하게 됨으로써, Directed Acyclic Graph(유방향 비순환 그래프)를 이루게 됩니다. (Tree구조가 아닌 이유는, 여러 상위어가 하나의 공통 하위어를 가질 수 있기 때문입니다.)

WordNet은 프로그램으로 제공되므로 다운로드 받아 설치할 수도 있고, 웹사이트에서 바로 이용 할 수도 있습니다. 또한, NLTK에 랩핑(wrapping)되어 포함되어 있어, import하여 사용 가능합니다. 아래는 WordNet 웹사이트에서 bank를 검색한 결과 입니다.

이처럼 WordNet은 단어 별 여러가지 가능한 의미를 미리 정의 하고, numbering 해 놓았습니다. 또한 각 의미별로 비슷한 의미를 갖는 동의어(Synonym)를 링크 해 놓아, Synset을 제공합니다. 이것은 단어 중의성 해소에 있어서 매우 좋은 labeled data가 될 수 있습니다. 만약 WordNet이 없다면, 각 단어별로 몇 개의 의미가 있는지조차 알 수가 없을 것입니다. 즉, WordNet 덕분에 우리는 이 데이터를 바탕으로 supervised learning(지도 학습)을 통해 단어 중의성 해소 문제를 풀 수 있습니다.

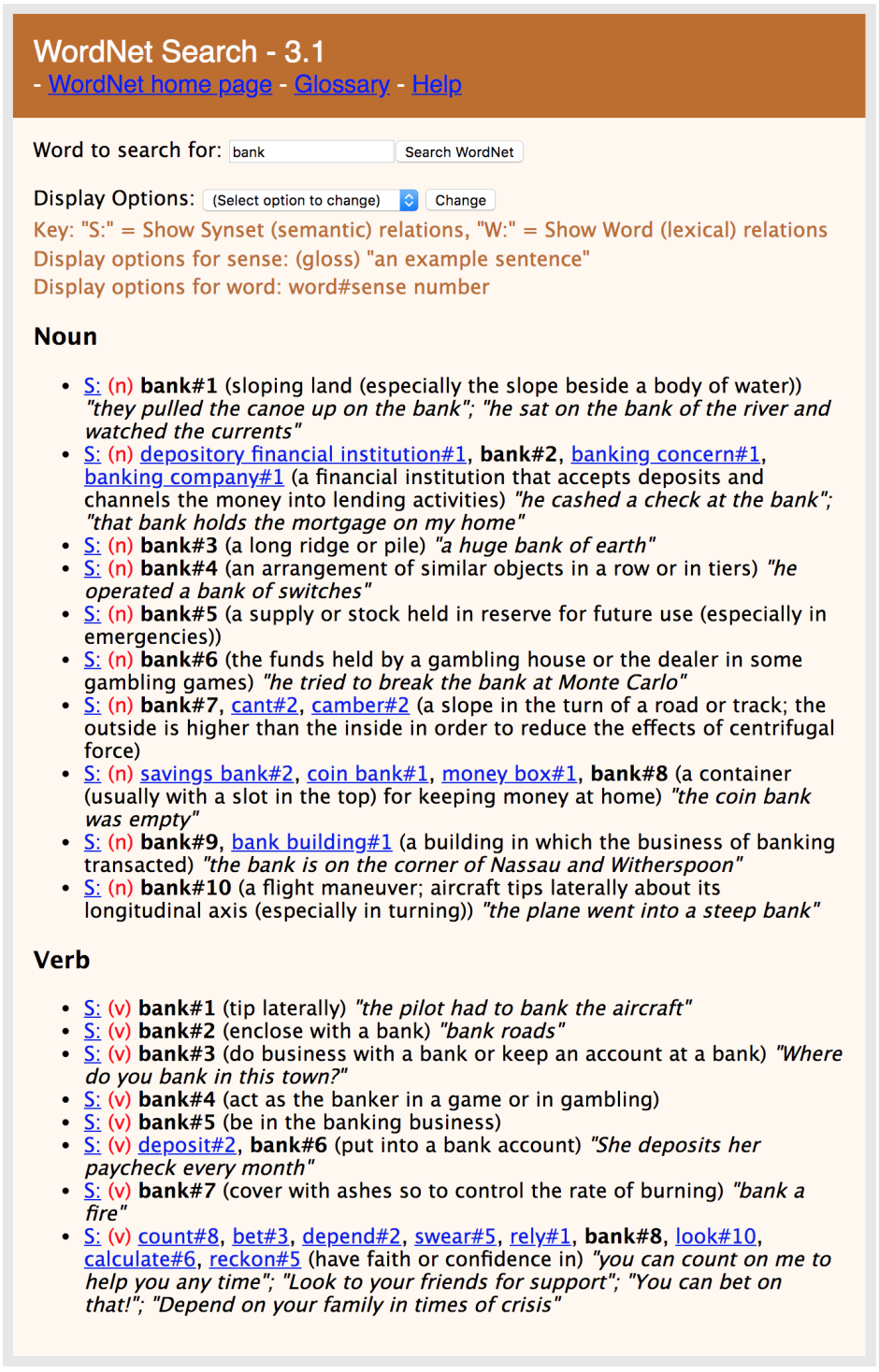
Lesk Algorithm

Lesk 알고리즘은 가장 간단한 사전 기반 중의성 해소 방법입니다. 주어진 문장에서 특정 단어에 대해서 의미를 명확히 하고자 할 때 사용 할 수 있습니다. 이를 위해서 Lesk 알고리즘은 간단한 가정을 하나 만듭니다. 문장 내에 같이 등장하는 단어(context)들은 공통 토픽을 공유한다는 것 입니다.

이 가정을 바탕으로 동작하는 Lesk 알고리즘의 개요는 다음과 같습니다. 먼저, 중의성을 해소하고자 하는 단어에 대해서 사전(주로 WordNet)의 의미별 설명과 주어진 문장 내에 등장한 단어의 사전에서 의미별 설명 사이의 유사도를 구합니다. 유사도를 구하는 방법은 여러가지가 있을테지만, 가장 간단한 방법으로는 겹치는 단어의 갯수를 구하는 것이 될 수 있습니다. 이후, 문장 내 단어들의 의미별 설명과 가장 유사도가 높은 (또는 겹치는 단어가 많은) 의미가 선택 됩니다.

예를 들어 NLTK의 WordNet에 'bass'를 검색해 보면 아래와 같습니다.

```
>>> from nltk.corpus import wordnet as wn
>>> for ss in wn.synsets('bass'):
...     print(ss, ss.definition())
...
```



WordNet Search - 3.1

- WordNet home page - Glossary - Help

Word to search for: bank Search WordNet

Display Options: (Select option to change) Change

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Display options for word: word#sense number

Noun

- S: (n) **bank#1** (sloping land (especially the slope beside a body of water)) *"they pulled the canoe up on the bank"; "he sat on the bank of the river and watched the currents"*
- S: (n) depository financial institution#1, **bank#2**, banking concern#1, banking company#1 (a financial institution that accepts deposits and channels the money into lending activities) *"he cashed a check at the bank"; "that bank holds the mortgage on my home"*
- S: (n) **bank#3** (a long ridge or pile) *"a huge bank of earth"*
- S: (n) **bank#4** (an arrangement of similar objects in a row or in tiers) *"he operated a bank of switches"*
- S: (n) **bank#5** (a supply or stock held in reserve for future use (especially in emergencies))
- S: (n) **bank#6** (the funds held by a gambling house or the dealer in some gambling games) *"he tried to break the bank at Monte Carlo"*
- S: (n) **bank#7**, cant#2, camber#2 (a slope in the turn of a road or track; the outside is higher than the inside in order to reduce the effects of centrifugal force)
- S: (n) savings bank#2, coin bank#1, money box#1, **bank#8** (a container (usually with a slot in the top) for keeping money at home) *"the coin bank was empty"*
- S: (n) **bank#9**, bank building#1 (a building in which the business of banking transacted) *"the bank is on the corner of Nassau and Witherspoon"*
- S: (n) **bank#10** (a flight maneuver; aircraft tips laterally about its longitudinal axis (especially in turning)) *"the plane went into a steep bank"*

Verb

- S: (v) **bank#1** (tip laterally) *"the pilot had to bank the aircraft"*
- S: (v) **bank#2** (enclose with a bank) *"bank roads"*
- S: (v) **bank#3** (do business with a bank or keep an account at a bank) *"Where do you bank in this town?"*
- S: (v) **bank#4** (act as the banker in a game or in gambling)
- S: (v) **bank#5** (be in the banking business)
- S: (v) deposit#2, **bank#6** (put into a bank account) *"She deposits her paycheck every month"*
- S: (v) **bank#7** (cover with ashes so to control the rate of burning) *"bank a fire"*
- S: (v) count#8, bet#3, depend#2, swear#5, rely#1, **bank#8**, look#10, calculate#6, reckon#5 (have faith or confidence in) *"you can count on me to help you any time"; "Look to your friends for support"; "You can bet on that!"; "Depend on your family in times of crisis"*

Figure 2: WordNet 웹사이트에서 단어 'bank'를 검색 한 결과

```
Synset('bass.n.01') the lowest part of the musical range
Synset('bass.n.02') the lowest part in polyphonic music
Synset('bass.n.03') an adult male singer with the lowest voice
Synset('sea_bass.n.01') the lean flesh of a saltwater fish of the family Serranida
Synset('freshwater_bass.n.01') any of various North American freshwater fish with
Synset('bass.n.06') the lowest adult male singing voice
Synset('bass.n.07') the member with the lowest range of a family of musical instru
Synset('bass.n.08') nontechnical name for any of numerous edible marine and freshw
Synset('bass.s.01') having or denoting a low vocal or instrumental range
```

Lesk 알고리즘 수행을 위하여 간단하게 랩핑(wrapping)하도록 하겠습니다.

```
def lesk(sentence, word):
    from nltk.wsd import lesk

    best_synset = lesk(sentence.split(), word)
    print(best_synset, best_synset.definition())
```

아래와 같이 주어진 문장에서는 'bass'는 물고기의 의미로 뽑히게 되었습니다.

```
>>> sentence = 'I went fishing last weekend and I got a bass and cooked it'
>>> word = 'bass'
>>> lesk(sentence, word)

Synset('sea_bass.n.01') the lean flesh of a saltwater fish of the family Serranida
```

또한, 아래의 문장애서는 음악에서의 역할을 의미하는 것으로 추정되었습니다.

```
>>> sentence = 'I love the music from the speaker which has strong beat and bass'
>>> word = 'bass'
>>> lesk(sentence, word)

Synset('bass.n.02') the lowest part in polyphonic music
```

여기까지 보면 Lesk 알고리즘은 비교적 잘 동작하는 것으로 보입니다. 하지만, 비교적 정확하게 예측해낸 위의 두 사례와 달리, 아래의 문장에서는 전혀 다른 의미로 예측하는 것을 볼 수 있습니다.

```
>>> sentence = 'I think the bass is more important than guitar'
>>> word = 'bass'
>>> lesk(sentence, word)
```

```
Synset('sea_bass.n.01') the lean flesh of a saltwater fish of the family Serranida
```

이처럼 Lesk 알고리즘은 명확한 장단점을 지니고 있습니다. WordNet과 같이 잘 분류된 사전이 있다면, 쉽고 빠르게 중의성해소를 해결할 수 있을 것 입니다. 또한 WordNet에서 이미 단어별로 몇개의 의미를 갖고 있는지 잘 정의 해 놓았기 때문에, 크게 고민 할 필요도 없습니다. 하지만 보다시피 사전의 단어 및 의미에 대한 설명에 크게 의존하게 되고, 설명이 부실하거나 주어진 문장에 큰 특징이 없을 경우 단어 중의성해소 능력은 크게 떨어지게 됩니다. 그리고 WordNet이 모든 언어에 대해서 존재하는 것이 아니기 때문에, 사전이 존재하지 않는 언어에 대해서는 Lesk 알고리즘 자체의 수행이 어려울 수도 있습니다.

한국어 WordNet

다행히 영어 WordNet과 같이 한국어를 위한 WordNet도 존재 합니다. 다만, 아직까지 표준이라고 할만큼 정해진 것은 없고 몇 개의 WordNet이 존재하고 있습니다. 아직까지 지속적으로 발전하고 있는 만큼, 작업에 따라 필요한 한국어 WordNet을 이용하면 좋습니다.

이름	기관	웹사이트
KorLex	부산대학교	http://korlex.pusan.ac.kr/
Korean WordNet(KWN)	KAIST	http://wordnet.kaist.ac.kr/

Using Feature Similarity

이번엔 다른 방식으로 단어 중의성 해소(WSD)에 접근 해보겠습니다. 자체적으로 단어에 대한 특성(feature)들을 모아 feature vector로 만들거나 유사도(similarity)를 계산하는 연산을 통해 단어의 중의성을 해소하는 방법입니다. 지금이야 어렵지않게 단어를 vector 형태로 embedding 할 수 있지만, 딥러닝 이전의 시대에는 쉽지 않은 일이었습니다. 다른 word embedding 방식은 추후 Word Embedding Vector 챕터에서 다루도록 하고, 중의성 해소를 위한 피쳐를 추출하고 유사도를 계산하는 방식을 살펴보겠습니다.

Based on Path in WordNet

```
from nltk.corpus import wordnet as wn

def get_hypernyms(synset):
    current_node = synset
    while True:
        print(current_node)
        hypernym = current_node.hypernyms()
        if len(hypernym) == 0:
            break
        current_node = hypernym[0]
```

위의 코드를 사용하면 WordNet에서 특정 단어의 최상위 부모 노드까지의 경로를 구할 수 있습니다. 아래와 같이 'policeman'은 'firefighter', 'sheriff'와 매우 비슷한 경로를 가짐을 알 수 있습니다. 'student'와도 매우 비슷하지만, 'mailman'과 좀 더 비슷함을 알 수 있습니다.

```
>>> get_hypernyms(wn.synsets('policeman')[0])
Synset('policeman.n.01')
Synset('lawman.n.01')
Synset('defender.n.01')
Synset('preserver.n.03')
Synset('person.n.01')
Synset('causal_agent.n.01')
Synset('physical_entity.n.01')
Synset('entity.n.01')
```

```
>>> get_hypernyms(wn.synsets('firefighter')[0])
Synset('fireman.n.04')
Synset('defender.n.01')
Synset('preserver.n.03')
Synset('person.n.01')
Synset('causal_agent.n.01')
Synset('physical_entity.n.01')
Synset('entity.n.01')
```

```
>>> get_hypernyms(wn.synsets('sheriff')[0])
Synset('sheriff.n.01')
```

```
Synset('lawman.n.01')
Synset('defender.n.01')
Synset('preserver.n.03')
Synset('person.n.01')
Synset('causal_agent.n.01')
Synset('physical_entity.n.01')
Synset('entity.n.01')
```

```
>>> get_hypernyms(wn.synsets('student')[0])
Synset('student.n.01')
Synset('enrollee.n.01')
Synset('person.n.01')
Synset('causal_agent.n.01')
Synset('physical_entity.n.01')
Synset('entity.n.01')
```

```
>>> get_hypernyms(wn.synsets('mailman')[0])
Synset('mailman.n.01')
Synset('deliveryman.n.01')
Synset('employee.n.01')
Synset('worker.n.01')
Synset('person.n.01')
Synset('causal_agent.n.01')
Synset('physical_entity.n.01')
Synset('entity.n.01')
```

Based on Co-Occurrence

http://www.let.rug.nl/nerbonne/teach/rema-stats-meth-seminar/presentations/Olango-Naive-Bayes-2009.pdf # Selectional Preference

Pseudo Word

A Simple, Similarity-based Model for Selectional Preferences

[Erk et al.2007]

[Erk et al.2012]