Assignment-Discussion
Vector Based WSD

Dhvanit, 200050035
Margav, 200050072
Sartaj, 200050128

26 November 2022

Problem Statement: Part-1: Extended Lesk

- Given a sequence of words, produce the synset IDs of only NOUNS
- Data SemCor (used nltk.corpus.semcor)
- First Technique to be used: Extended Lesk

First Baseline

- Most Frequent Sense (MFS)
  Calculated word-synset frequencies for the first 10000 entries of the SemCor dataset, for new words will return Synset as ‘None’ otherwise the MF-Synset

- Performance:
  (on 5000 unseen sentences)

Precision	: 47.2
Recall	: 52.92
Fscore	: 47.77
Accuracy	: 52.92

Second Baseline

- Wordnet First Sense (WFS)
  nltk.corpus.wordnet.synsets(word) returns a list of possible Synsets, we always chose the first element in the list

- Perfomance:
  (on same 5000 sentences)

Precision	: 70.59
Recall	: 76.03
Fscore	: 71.16
Accuracy	: 74.85

- WFS generalises better than MFS. This is not counterintuitive Since MFS gives accuracy ~81% on training dataset. But for unseen/general examples the wordnet seems to have some better hindsight than just frequency.

Performance report of Extended Lesk

• Precision :	67.69
• Recall :	61.95
• F1-score :	61.42
• Accuracy :	60.77

Part-2: Page Rank Based

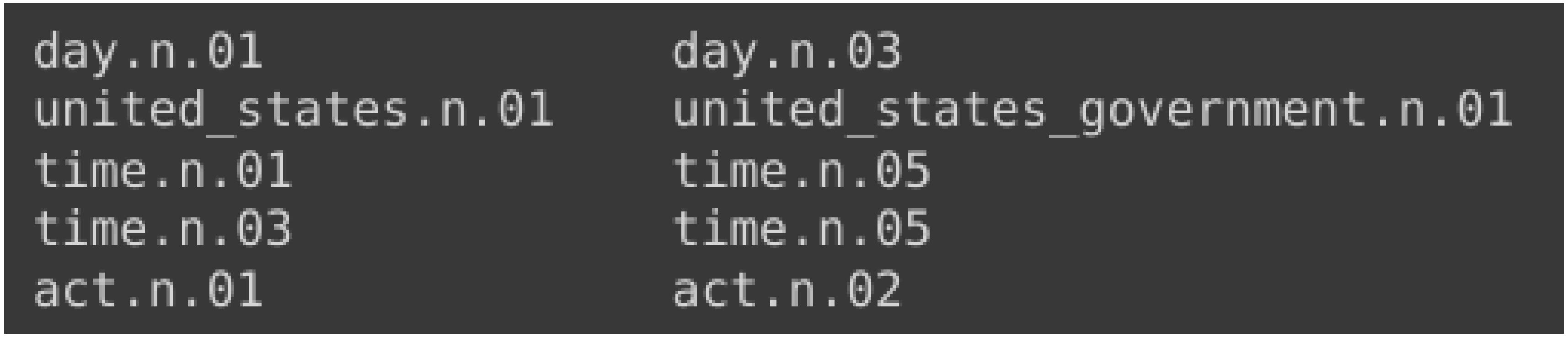
	number-common-words	word-vectors
• Precision :	66.18	65.73
• Recall :	47.18	43.49
• F1-score :	49.46	47.72
• Accuracy :	47.18	43.49

- Observation- significant difference between precision and Recall is because we only connect synsets of consecutive Words. So if two synsets that should reinforce each other are from consecutive words, their relation is identified but otherwise it is not. So precision is higher than recall (if we have a connection between two synsets, it is correct; but not all connections are identified and used). We can't solve this by connecting all synsets (tried) since it dilutes the pageranks and we end up taking argmax of low variance data.

Confusion Cases (for EL)

Following are the top 5 most confused synset pairs (not in order):

(first, second) where ‘second’ is predicted when ‘first’ is correct

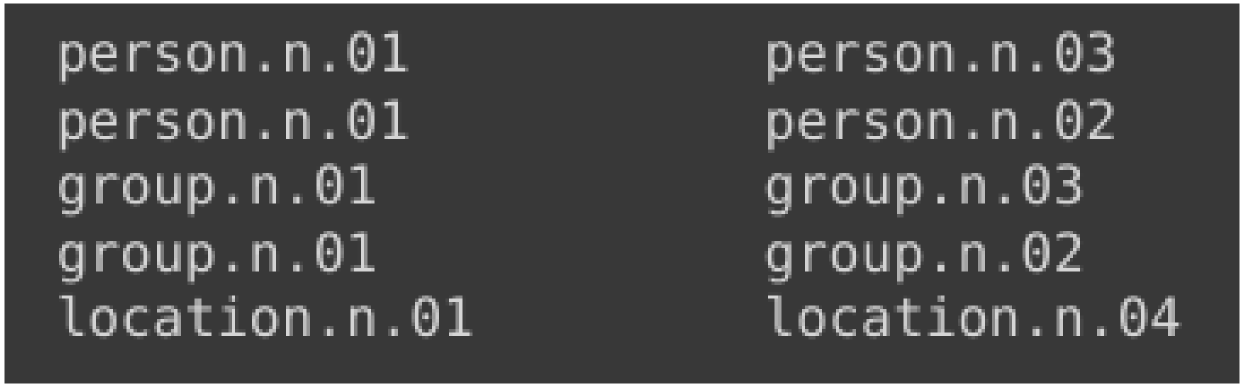


```
day.n.01                day.n.03
united_states.n.01      united_states_government.n.01
time.n.01               time.n.05
time.n.03               time.n.05
act.n.01                act.n.02
```

Confusion Cases (for PR)

Following are the top 5 most confused synset pairs (not in order):

(first, second) where ‘second’ is predicted when ‘first’ is correct



```
person.n.01           person.n.03
person.n.01           person.n.02
group.n.01            group.n.03
group.n.01            group.n.02
location.n.01         location.n.04
```

Interpretation of confusion (error analysis: EL)

- The definitions assigned to the two synsets of day (and Similarly to the three synsets of time) are technical definitions so it contains words that won’t usually be in the *context* of the word in sentence.

- The two definitions of united_states don’t have overlap with Each other but tend to have equal amount of overlap with The context since both contain terms that might be found in a conversation about USA and both have similar network of hypernyms/hyponyms
- The set of hyponyms for the respective synsets of ‘act’ and ‘time’ is quite large so it dilutes the main synset’s contribution

Interpretation of confusion (error analysis: PR)

- The definitions of person.n.01 and person.n.02 are short and Almost the same so it is not clearly distinguishable
- For a lot of these (synsets of person, group, location) the different meanings are subtle and have to be specifically defined. This increased length means they have a tendency to reinforce and get reinforced by synsets just because they Have a lot of words to match
- This is because we are matching individual synsets without Looking at their neighbours (hyper/hypo - nymns)

Data Processing Info

- Pretrained word-vectors from *word2vec-google-news-300* using gensim
  For unseen words, we choose vector as a normal distribution with
  Mean 0 and standard dev 1 (as in previous assignments)

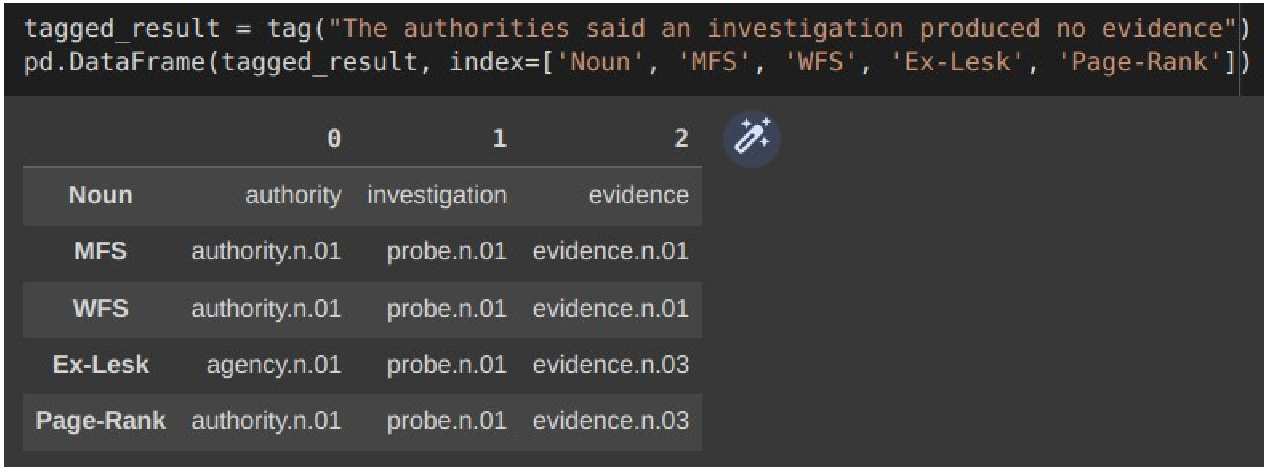
- For proceeding with our algorithms we require pos-tagged
  (so we can select the nouns) and synset-tagged words
  (so we can compare our prediction).

- Nltk’s SemCor reader provides tags for chunked data where
  It replaces identified multi-word-expressions with a single
  Lemma and provides the synset for that lemma. So we need
  To navigate the Tree() objects it returns when preprocessing.

- Demo sentence: we use nltk’s pos_tagger and lemmatizer

MUST: GUI

- Given a sentence as a string, we use nltk’s tokenizer, pos_tagger, and lemmatizer to produce a list of (word,pos) pairs.
- This list is subjected to all four algorithms to get synsets for only the nouns. We print the identified nouns and their Predicted synsets



```
tagged_result = tag("The authorities said an investigation produced no evidence")
pd.DataFrame(tagged_result, index=['Noun', 'MFS', 'WFS', 'Ex-Lesk', 'Page-Rank'])
```

	0	1	2
Noun	authority	investigation	evidence
MFS	authority.n.01	probe.n.01	evidence.n.01
WFS	authority.n.01	probe.n.01	evidence.n.01
Ex-Lesk	agency.n.01	probe.n.01	evidence.n.03
Page-Rank	authority.n.01	probe.n.01	evidence.n.03