Ling575 Summarization System

D5: Final System

Team ✨consonants✨

Rachel Hantz, Yi-Chien Lin, Yian Wang, Chenxi Li, Tashi Tsering

Overview

System Architecture

Content Realization

- Name Resolution
- Results & Error Analysis

Best System on Devtest

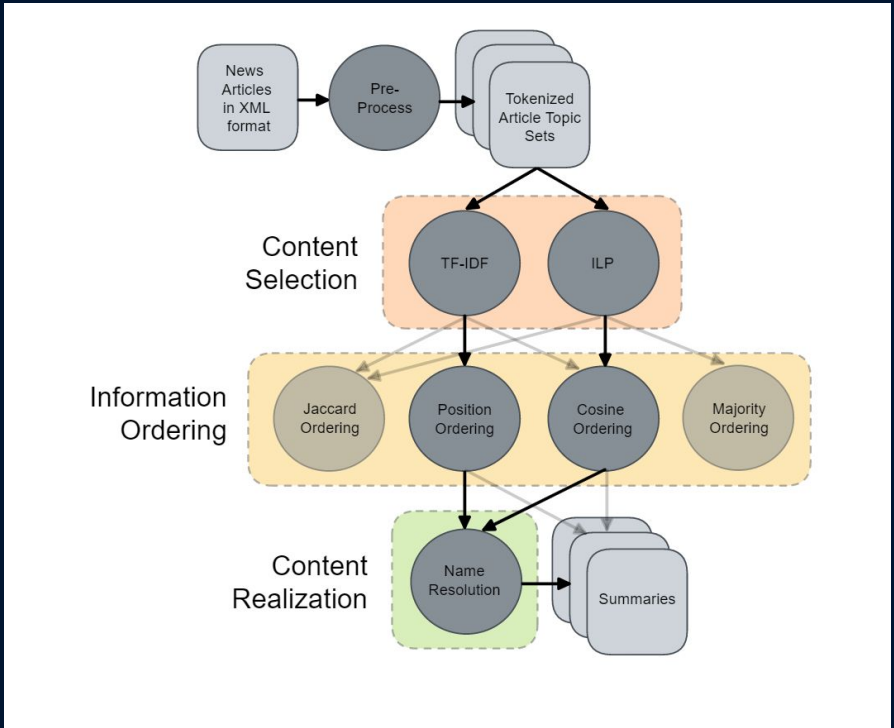
Systems on Evaltest

Ablation Study

WorkSplit

- Content Realization - Rachel & Chenxi
- Updated Pipeline - Yi-Chien
- Evaluation on Evaltest - Yi-Chien, Yian & Tashi
- Ablation Study - Yian
- Report & Slides - Everyone

System Architecture



Components and/or methods not utilized within the best systems are shown greyed-out.

Inputs and outputs are shown as rounded rectangles, while system components are shown as circles.

Content Realization

Part 1 - build Find_People module

- Retrieve people entities
- Retrieve people with modifiers

Part 2 - resolve names in the summary

- Extract names
- Find longest name
- Rules of replacement

Results

- ROUGE score
- Error analysis

Find_People Module

1. Retrieve People Entities

```
people = [entity.text for entity in doc.ents if entity.label_ == 'PERSON']
```

- Operates on a single sentence; uses spaCy NER pipeline component

e.g.

“Beagle dog *Snoopy* loves to dance.”

people = [“Snoopy”]

Find_People Module

2. Retrieve People with Modifiers

```
people_tokens = " ".join(people).split()  # all tokens for people

only_people_with_modifiers = []  # list for only people with modifiers

for noun_phrase in doc.noun_chunks:
    noun_head = noun_phrase.root

    # checks that the head of this noun phrase was found in people tokens
    if noun_head.text not in set(people_tokens):
        continue
    # only stores noun phrases that are people with modifiers
    if noun_phrase.text not in people:  # Since, people list does not include modifiers
        only_people_with_modifiers.append(noun_phrase.text)
```

- Uses spaCy dependency parser to get noun phrases
- Checks that noun phrases are headed by people nouns
- And that they are modified in some way

"Beagle dog *Snoopy* loves to dance."

only_people_with_modifiers = ["Beagle dog Snoopy"]

Find_People Module

2. Retrieve People with Modifiers

```
# selects all modified versions of that person
modified_person = [
    modified_person for modified_person in only_people_with_modifiers if person in modified_person
]

if modified_person:
    # appends first modified person
    people_with_modifiers.append(modified_person[0])
    # removes that modified person so that it is not appended again
    only_people_with_modifiers.remove(modified_person[0])
else:
    # if there was never a modifier for that person, just appends un-modified person
    people_with_modifiers.append(person)
```

- For each person, gets a modified_person list
- Stores first modified_person
- Removes that modified_person from only_people_with_modifiers

```
return {
    "people": people,
    "people_with_modifiers": people_with_modifiers
}
```

{ “people”: [“Snoopy”],
“people_with_modifiers”: [“Beagle dog Snoopy”] }

Content Realization

Part 1 - build find_people module

- Retrieve people entities
- Retrieve people with modifiers

Part 2 - resolve names in the summary

- Extract co-indexed name lists
- Find longest name
- Rules of replacement

Results

- ROUGE score
- Error analysis

Name Resolution - Step 1

- *Snoopy* is a cute dog.
- Beagle dog *Snoopy* loves to dance.
- Charlie Brown's best friend Snoopy, is always ready for an adventure.

List of Names		List of Modified Names
['Snoopy,'		['Snoopy',
'Snoopy',	co-indexed	'Beagle dog Snoopy',
'Snoopy']		'Charlie Brown's best friend Snoopy']

Name Resolution - Step 2 & 3

- 2. Find Snoopy with longest modifier
  - Eg. ‘Charlie Brown’s loyal companion Snoopy’
  - 3 at each position - unmodified/modified/longest

3 versions of Snoopy in sentence ‘Beagle dog *Snoopy* loves to dance.’

[*Snoopy*, Beagle dog *Snoopy*, Charlie Brown’s loyal companion Snoopy]

- 3. Rules of replacement
  - 1st time appearing - replaced with longest name
  - Later occurrences - replaced with unmodified name

Name Resolution - Example

- *Snoopy* is a cute dog.
- Beagle dog *Snoopy* loves to dance.
- Charlie Brown's best friend Snoopy, is always ready for an adventure.

- Charlie Brown's loyal companion Snoopy, is a cute dog.
- *Snoopy* loves to dance.
- *Snoopy* is always ready for an adventure.

Content Realization

Part 1 - build find_people module

- Retrieve people entities
- Retrieve people with modifiers

Part 2 - resolve names in the summary

- Extract co-indexed name lists
- Find longest name
- Rules of replacement

Results

- ROUGE score
- Error analysis

ROUGE Score Results

TF-IDF	Metric	CS + IO	CS + IO + CR
Rouge-1	Recall	**0.31028**	**0.31293**
	Precision	**0.37183**	**0.36833**
	F1-score	**0.33793**	**0.33796**
Rouge-2	Recall	**0.07246**	**0.07350**
	Precision	**0.08654**	**0.08591**
	F1-score	**0.0788**	**0.07913**

ILP	Metric	CS + IO	CS + IO + CR
Rouge-1	Recall	**0.34492**	**0.34691**
	Precision	**0.38148**	**0.37779**
	F1-score	**0.36202**	**0.36143**
Rouge-2	Recall	**0.09455**	**0.09514**
	Precision	**0.10488**	**0.10353**
	F1-score	**0.09936**	**0.09908**

Analysis

Some produce good re-written results

- Bush → President Bush
- Hunter → Boulder district attorney Alex Hunter

Undesirable results attributed to:

- Over/under-recognition
- Co-referencing issue
- Inaccurate extracting
- Unwanted clipping

https://docs.google.com/spreadsheets/d/1Q4KnKplC1O6hEpjjWYdcdC3bBl5ZU4SiTjN2myEbLq4/edit#gid=144644483

Best System on Devtest

TFIDF and ILP was tested on Devtest both with & without CR, baseline was tested on Devtest without CR

- ILP outperforms TF-IDF in terms of overall performance
- slight reduction in precision and F-1 scores after CR for ILP
- slight reduction in precision after CR for TFIDF

Best System on Devtest

N		CS + IO		CS + IO + CR		Baseline
		TF-IDF	ILP	TF-IDF	ILP	
1	Recall	0.31028	0.34492	0.31293	0.34691	0.23824
	Precision	0.37183	0.38148	0.36833	0.37779	0.33036
	F1-Score	0.33793	0.36202	0.33796	0.36143	0.27436
2	Recall	0.07246	0.09454	0.07350	0.09514	0.05136
	Precision	0.08654	0.10488	0.08591	0.10353	0.07092
	F1-Score	0.07880	0.09936	0.07913	0.09908	0.05906

System on Evaltest

TFIDF and ILP was tested on Evaltest with CR, baseline was tested on Evaltest without CR

- ILP method remained the optimal choice for our system
- baseline approach yielded an impressive rouge-score

System on Evaltest

N	Metric	CS + IO + CR		Baseline
		TF-IDF	ILP	
1	Recall	0.32825	**0.36769**	0.29203
	Precision	0.38140	0.39139	**0.39313**
	F1-Score	0.35214	**0.37873**	0.33294
2	Recall	0.08051	**0.11572**	0.07610
	Precision	0.09366	**0.12296**	0.10465
	F1-Score	0.08641	**0.11909**	0.08748

Ablation Study

The purpose is to understand the contribution of different components to the overall system by removing each component at a time.

N	Metric	No CS	No IO	No CR	Complete System
1	Recall	0.23906	**0.34732**	0.34492	0.34725
	Precision	0.32811	0.37744	**0.38148**	0.37796
	F1-score	0.27400	0.36139	**0.36202**	0.36164
2	Recall	0.05162	0.09520	0.09454	**0.09530**
	Precision	0.07043	0.10334	**0.10488**	0.10368
	F1-score	0.05904	0.09899	**0.09936**	0.09922