CS – 6320: Natural Language Processing

Report : Information Extraction System on Academic Research Papers

Team Name : The seekers

- ***Nidhi Vaishnav (ntv170030)***
- ***Maitri Shah (mxs172030)***

1. Problem:

Expectation from computer to process human language (natural language) and process tasks like Information retrieval, information extraction, question answering, summarization etc. is quite old. To perform these tasks computer must have the knowledge of the domain as well as ability to comprehend the human language. To perform this task, we are creating an Information Extraction application to perform template extraction of natural language, which can be useful as the base data to perform the above natural language tasks.

As we know in the field of research that a researcher must have to go through multiple research papers to keep up with current field, to review them for conferences or class, or for the literature survey of the new field which takes hundreds of hours. We are proposing a system which can summarize the details of a research paper in terms of template.

2. Corpus:

To perform information extraction, huge corpus is required. To collect the corpus, we have focused on two major domains.

1. Computer science research paper
2. Medical Research paper

We have collected corpus out of 10 computer science research papers and 5 medical research papers (abstracts). Our corpus is of the size 50k words. To collect the corpus we have used IEEE, ACM, Cornell university Library for computer science papers and National Center for Biotechnology Information (part of United States National Library of Medicine) for medical papers.

3. Information Extraction:

We have created 10 templates which covers basics of all the research papers. Templates with their respective examples are as per below.

Information Extraction template Table

No.	Template	Attributes
1	Aim (Problem, domain, hypothesis)	3
2	Contribution (work, researchers, research time, research institute, relevance)	5
3	Compare (subject1, subject2, about, domain, outcome)	4
4	Observation (object, action, observation, hypothesis)	4
5	Terminology (Key, definition, reference (citation/general field term etc.))	3
6	Process (method name, purpose, process, assumption (if any), tools)	5
7	Selection (sample, stage, selection criteria, justification)	4
8	Results (Domain, task, results)	3
9	Conclusion (task, results, achievements, limitations, future work)	5
10	Acknowledgement (Authors, Paper, publication, publish year, reference page)	5
		41

1. **Aim (problem, domain, hypothesis)**
   - Given template provides Aim of the research paper.
   - We have 3 information properties for the given template
     - Problem: what is the problem that is intended to be solved
     - Domain: domain of the problem
     - Hypothesis: an explanation based on limited evidence as a starting point for the research

   Example:
   "In this paper, we aim to resolve these difficulties by proposing AVOD, an Aggregate View Object Detection architecture for autonomous driving."
   Template:
   Aim (to resolve these difficulties**,** this paper**,** by proposing AVOD, an Aggregate View Object Detection architecture for autonomous driving.**)**

   Code Output:
   ```
   {'problem': 'resolve these difficulties', 'domain': In this paper, 'hypothesis': 'by proposing AVOD, an Aggregate View Object Detection architecture for autonomous driving'}
   ```

   Accuracy: 75%

2. **Contribution (work, researchers, research time, research institute, relevance)**
   - Given template provides contribution of researchers
     - Work: paper/topic name
     - Researcher: name of the researchers
     - Research time: required time for research (null is valid)
     - institute: institute/publication of research or presentation of research (null is valid)
     - relevance: How it is related to given paper

   Example:

   This work was begun and brought to fruition when we were at Cambridge University; we dedicate this paper to the memory of Professor David G. Crighton, who first interested one of us (L.M.) in this problem nearly a decade ago.

   Template:
   contribution (this work, David G. Crighton, a decade ago, Cambridge University, begun and brought to fruition)

3. **Comparison (object1, object2, domain, outcome)**
   - Given template provides comparison between 2 objectives
     - Object 1: the object which is to be compared

- Object 2: the object which is to be compared with
- Domain: domain of comparison
- Outcome: comparison result

Example:

On the validation set (Table I), our architecture is shown to outperform MV3D by 2.09% AP on the moderate setting and 4.09% on the hard setting.

Template:

Comparison (Our architecture**,** MV3D**,** validation set**,** 2.09% AP on the moderate setting and 4.09% on the hard setting.**)**

4. **Observation (domain, problem, observation)**
   - Given template provides observation details
     - Object: what we are observing
     - Action: what are we doing
     - Observation: what we observed
     - Hypothesis: observation time details

Example:

"Computer vision is an attractive solution for automated assessment of PD, by making possible recent advances in computational power and deep learning algorithms."

Template:

Observation (Computer vision**,** is, an attractive solution for automated assessment of PD, by making possible recent advances in computational power and deep learning algorithms**)**

Code Output:

```
{'object': 'Computer vision', 'action': is, 'observation': an attractive solution for automated assessment of PD, 'hypothesis': by making possible recent advances in computational power and deep learning algorithms}
```

Accuracy: 92%

5. **Terminology (Key, definition, reference)**
   - Given template provides details of abbreviations with their reference
     - Key: abbreviation
     - Definition: expansion of the term
     - Reference: how we are using it or where it is referred

Example:

“We present AVOD, an Aggregate View Object Detection network for autonomous driving scenarios."

Template:

Terminology (AVOD, an Aggregate View Object Detection, network for autonomous driving)

Code Output:

```
{'key': 'AVOD', 'definition': 'an Aggregate View Object Detection network', 'reference': ' for autonomous driving scenarios'}
```

Accuracy: 81.5%

6. **Process (method name, purpose, process, assumption (if any), tools)**
    - Given template provides details of a process
        - Method name: it provides name of the method used
        - Purpose: it provides the purpose of use
        - Process: what process/task is happening
        - Assumption: any special conditions
        - Tools: used tools

Example:

Movement trajectories of individual joints were extracted from videos of PD assessment using Convolutional Pose Machines, a pose estimation algorithm built with deep learning.

Template:

Process (Convolutional Pose Machines**,** a pose estimation algorithm, extraction from videos of PD assessment**,** Movement trajectories of individual joints**,** deep learning.**)**

7. **Selection (sample domain, sample size, selection criteria, justification)**
    - Given template provides selection criteria of the sample
        - Sample: Object of the Sample
        - Stage: sample selection place in process
        - Selection criteria: selection criteria – how did we select
        - Justification: selection process and measure

Example:

"Forty-five features of the three categories (i.e., foods and drinks, food adjectives, and ambience adjectives) were selected at this step using xyz method."

Template:

Selection (Forty-five features**,** at this step, the three categories (i.e., foods and drinks, food adjectives, and ambience adjectives)**,** using xyz method**)**

Code Output:

```
{'sample': 'Forty-five features', 'stage': 'at this step', 'sample criteria': 'of the three categories (i.e., foods and drinks, food adjectives, and ambience adjectives)', 'justification': 'using xyz method'}
```

Accuracy: 89%

8. **Results (Domain, task, result)**
    - Given problem provides results of a task
        - Domain: Domain of task
        - Task: The process/task
        - result: result of the process

Example:

"For LID, the communication task yielded the best results (detection: AUC = 0.930, severity estimation: r = 0.661)."

Template:

Results(For LID, the communication task, the best results (detection: AUC = 0.930, severity estimation: r = 0.661))

9. **Conclusion (problem, solution, achievements, limitations, future work)**
    - Given template provides the conclusion.
    - problem: problem that we handled for given work
    - solution: provided solution
    - achievement: significant achievement
    - limitations: limitation of problem
    - future work: suggested future work that can be done on the current problem

Example:

Conclusion (proposed architecture**,** state of the art results on the KITTI 3D object detection benchmark**,** real time with a low memory footprint**, -,** deployment on autonomous vehicles**)**

Template:

Conclusion (proposed architecture**,** state of the art results on the KITTI 3D object detection benchmark**,** real time with a low memory footprint**, -,** deployment on autonomous vehicles**)**

10. **Acknowledgement (Authors, Paper, publication, publish year, reference page)**
    - Given templates provides acknowledgement list from the acknowledgement
    - Authors: Authors of the paper
    - Paper: Paper name
    - Publication : publication organization or conference name

- Publish year: year of publication
- Reference page: referred pages

Example:

"A. Geiger, P. Lenz, and R. Urtasun ; “Are we ready for autonomous driving? the kitti vision benchmark suite,”; Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference; 2012; pp. 3354–3361"

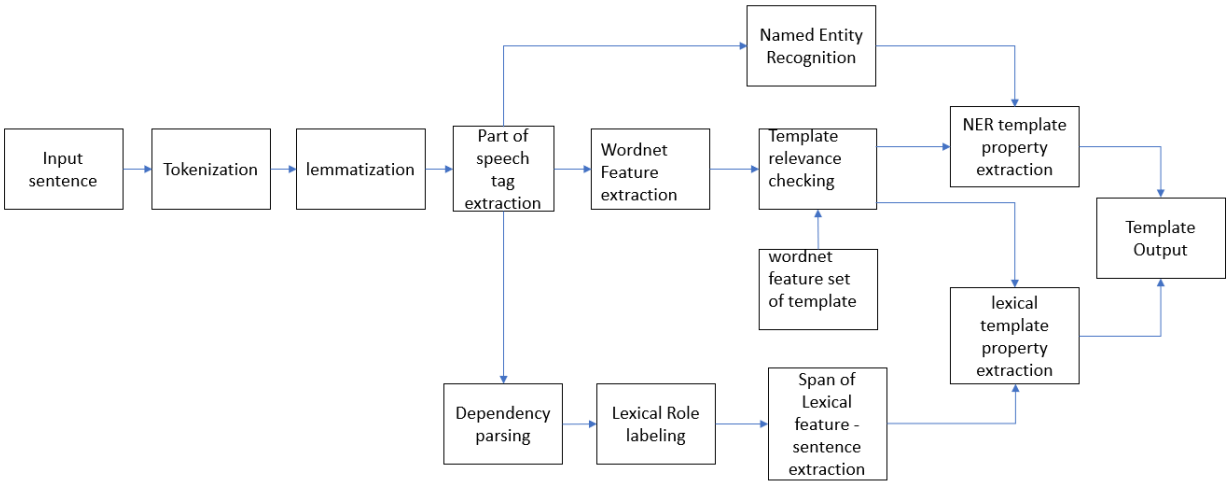
Template:

Acknowledgement (A. Geiger, P. Lenz, and R. Urtasun **;** “Are we ready for autonomous driving? the kitti vision benchmark suite,”**;** Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference; 2012**;** pp. 3354–3361**)**

4. Architecture:

To implement the given system, we are using Python language with libraries like nltk and spaCy. In the image bellow the architecture of the template detection is provided.
Here we have divided template properties in 2 parts:

1. NER template properties – these properties can be extracted by applying NER
2. Semantic template properties – these properties can be extracted using dependency parse tree



Information Extraction application for Academic Research papers – pipeline diagram

- **Note:** for above diagram, consider data from the previous state is also available.
- Ex. Transition line is displayed from POS to NER, but along with POS, tokens and lemma are also available for NER if required.

➢ **Input sentence:**

- Here sentence is provided as input

➢ **Tokenization**

- Sentence is tokenized into tokens using spaCy library
- Tokenization is useful as we can use the tokens to get the higher-level features
- Here stop words are not removed as they have propositions and pronouns which are useful for the current application to extract the dependency-based templates
- Input: “I am vising Dallas.”
- Output: [‘I, ‘am’, ‘visiting’, ‘Dallas’, ‘.’]

➢ **Lemmatization**

- Tokens are Lemmatized into lemma form using spaCy library
- It provides the basic dictionary form of the word
- Template wordnet features and sentence wordnet features are compared in their lemmatized form to check for the relevance

- Input: Tokens: [‘I, ‘am’, ‘visiting’, ‘Dallas’, ‘.’]
- Output: lemma: [‘I’, ‘be’, ‘visit’, ’Dallas’]

➢ **Part of speech tag extraction**

- Part of Speech tags are extracted from using spaCy library
- Part of speech tags are very important features to understand the tokens and they are used in parallel with semantic role labeling to get the templates
- Input: Tokens: [‘I, ‘am’, ‘visiting’, ‘Dallas’, ‘.’]
- Output: lemma: [‘PRON’, ‘VERB, ‘VERB’, ’PROPN’, ‘PUNCT’]

➢ **Wordnet feature extraction**

- Here we are extracting wordnet features like Hypernyms, hyponyms, meronyms, holonyms, entail and synonyms from the sentence.
- We are using this list for the template relevance checking.
- **Input:** token list : [‘outcome’]
- **Output:** wordnetFeatureDict:

```
['outcome']

current Synset: Synset('result.n.03')
hypernyms: [Synset('ending.n.04')]
hyponyms:       [Synset('consequence.n.02'),       Synset('deal.n.07'),
Synset('decision.n.03'),                        Synset('decision.n.04'),
Synset('denouement.n.01'),                Synset('poetic_justice.n.01'),
Synset('separation.n.07'), Synset('sequel.n.01'), Synset('worst.n.01')]
holonyms: []
meronyms: []
entail = []

current Synset: Synset('consequence.n.01')
hypernyms: [Synset('phenomenon.n.01')]
hyponyms:     [Synset('aftereffect.n.01'),     Synset('aftermath.n.01'),
Synset('bandwagon_effect.n.01'),                Synset('brisance.n.01'),
Synset('butterfly_effect.n.01'),              Synset('by-product.n.01'),
Synset('change.n.04'),                  Synset('coattails_effect.n.01'),
Synset('coriolis_effect.n.01'),                     Synset('dent.n.01'),
Synset('domino_effect.n.01'),                    Synset('harvest.n.02'),
Synset('impact.n.02'),     Synset('influence.n.04'),     Synset('knock-
on_effect.n.01'),  Synset('offspring.n.02'),  Synset('outgrowth.n.01'),
Synset('placebo_effect.n.01'),           Synset('position_effect.n.01'),
Synset('product.n.05'),                     Synset('repercussion.n.01'),
Synset('response.n.01'),                     Synset('side_effect.n.02'),
Synset('spillover.n.01')]
holonyms: []
meronyms: []
entail = []

Synonym:    ['result',    'resultant',    'final_result',    'outcome',
'termination', 'consequence', 'effect', 'outcome', 'result', 'event',
'issue', 'upshot']
```

- **Wordnet feature set of templates**

- Here we are extracting wordnet features like Hypernyms, hyponyms, meronyms, holonyms, entail and synonyms for selected words from the template relevant to the template.
- We are using this list for the template relevance checking.
- Input: tokenList of pre-selected word: For result template: “result”
- Output: wordnet features

```
current Synset: Synset('consequence.n.01')
hypernyms: [Synset('phenomenon.n.01')]
hyponyms:     [Synset('aftereffect.n.01'),     Synset('aftermath.n.01'),
Synset('bandwagon_effect.n.01'),                Synset('brisance.n.01'),
Synset('butterfly_effect.n.01'),              Synset('by-product.n.01'),
Synset('change.n.04'),                  Synset('coattails_effect.n.01'),
Synset('coriolis_effect.n.01'),                     Synset('dent.n.01'),
Synset('domino_effect.n.01'),                    Synset('harvest.n.02'),
Synset('impact.n.02'),     Synset('influence.n.04'),     Synset('knock-
on_effect.n.01'),  Synset('offspring.n.02'),  Synset('outgrowth.n.01'),
Synset('placebo_effect.n.01'),           Synset('position_effect.n.01'),
Synset('product.n.05'),                     Synset('repercussion.n.01'),
Synset('response.n.01'),                     Synset('side_effect.n.02'),
Synset('spillover.n.01')]
holonyms: []
meronyms: []
entail = []

current Synset: Synset('solution.n.02')
hypernyms: [Synset('statement.n.01')]
hyponyms: [Synset('denouement.n.02')]
holonyms: []
meronyms: []
entail = []

current Synset: Synset('result.n.03')
hypernyms: [Synset('ending.n.04')]
hyponyms:       [Synset('consequence.n.02'),       Synset('deal.n.07'),
Synset('decision.n.03'),                       Synset('decision.n.04'),
Synset('denouement.n.01'),                Synset('poetic_justice.n.01'),
Synset('separation.n.07'), Synset('sequel.n.01'), Synset('worst.n.01')]
holonyms: []
meronyms: []
entail = []

current Synset: Synset('resultant_role.n.01')
hypernyms: [Synset('semantic_role.n.01')]
hyponyms: []
holonyms: []
meronyms: []
entail = []

current Synset: Synset('result.v.01')
hypernyms: [Synset('prove.v.01')]
hyponyms:         [Synset('be_due.v.01'),         Synset('come.v.13'),
Synset('follow.v.03'), Synset('follow.v.06')]
```

```
holonyms: []
meronyms: []
entail = []

current Synset: Synset('leave.v.07')
hypernyms: [Synset('produce.v.03')]
hyponyms: [Synset('lead.v.03')]
holonyms: []
meronyms: []
entail = []

current Synset: Synset('result.v.03')
hypernyms: [Synset('happen.v.01')]
hyponyms: []
holonyms: []
meronyms: []
entail = []

Synonym: ['consequence', 'effect', 'outcome', 'result', 'event',
'issue', 'upshot', 'solution', 'answer', 'result', 'resolution',
'solvent', 'result', 'resultant', 'final_result', 'outcome',
'termination', 'resultant_role', 'result', 'result', 'ensue', 'leave',
'result', 'lead', 'result']
```

- **Template relevance checking**

- Based on template wordnet features are compared to make sure that the sentence is relevant to the template
- Features like synonyms, hypernyms, hyponyms, holonyms, meronyms etc. are compared based on the requirement
- Ex. For “outcome”, we have “results” as synonyms, which is in the synonym set of the template feature word “result”.

- **Name Entity recognition**

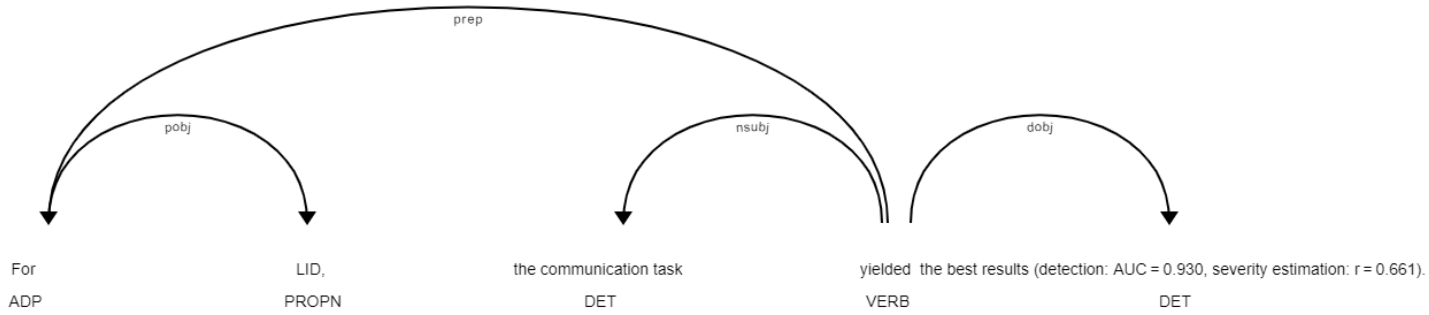
- Name Entity recognition feature is used to identify features such as author name, location, year, cardinal values etc.
- Ex. Input: 2012
- Output: DATE

- **NER template property extraction**

- Based on the relevance of the template and NER, data is added in the property of the template using spaCy library.
- Ex. IEEE conference, New York, so now we have location, we have IEEE word in our template relevance list (conference list) so we can combine this data to add it into publisher detail.

➢ **Dependency parsing**

- Here we are creating dependency parse tree to understand the structure of a sentence using spaCy.
- Input: token list
- Output:



➢ **Lexical role labeling**

- Using the parse tree, we are getting the **Lexical** roles like nsubj, dobj, prep, pobj etc.
- Using these roles, we are identifying the template properties.
- Ex. For the task property in the result statement for above sentence shown in tree, based on nsubjpass or nsubj for the child of root word (yield) “task”, we are identifying that task is part of the given template.

➢ **Span of lexical feature – sentence extraction**

- Now from the “task” we need to get the entire word, that is “the communication task” as the template.
- For that we are finding the phrase span of the word “task” which provides its sentence spanning across the children nodes in the dependency parse tree.

➢ **lexical template property extraction**

- Based on the **Lexical** role labeling we might get multiple objects, Ex there can be multiple objects, so to make sure that we detect the relevant words only, based on some specific indicator words for the template we are selecting the template property.

➢ **Template output**

- By combining all the above tasks, templates are extracted and provided as output as dictionary.
- Input: "For LID, the communication task yielded the best results (detection: AUC = 0.930, severity estimation: r = 0.661)."
- Output: {'domain': ' For LID', 'task': 'the communication task', 'result': ' the best results (detection: AUC = 0.930, severity estimation: r = 0.661)'}

5. Challenges:

Corpus & template creation:

- One of the challenging tasks was to collect the corpus of 50k words and create template out of those.
- Initially we thought of keeping domain of papers only for computer science, but after guidance of the faculty we moved in the medical domain (NCBI) as they were providing PubMed format.
- But creating templates we need to read the papers, which was a tedious task due to lack of background in the medical field, and which slowed us down.
- But after some analysis, we decided to work with the medical domain as we already made some progress and computer science domain, which was our dream work.

Implementation difficulties:

- Here in our system, we had very limited fields which could use the NER features which are easy to implement.
- In most of our sentence templates, dependency tree has been used to parse the sentence.
- But traversing the dependency tree and to identify the appropriate semantic role was a difficult task for us as we received many options for a single template property to parse based on just semantic roles.
- We overcome this challenge by providing relevant dataset, which helped to filter some of the objects.
- Moreover, we have tried to implement the template detections such that we can detect active and passive voices. Which was interesting task if not challenging.

6. Future work:

- We have implemented our templates just based on a single statement.
- So, if we encounter “For given research”, which is parsed as the topic, we are not finding the discourse. (Which research from the previous sentences)
- This can be implemented using discourse structure algorithms.