

# **NLP for Annotation, Extraction and Classification of Pulmonary Embolism MIMIC III clinical notes**

Soyoung An

# Goal

- **Using Regular Expression (d-dimer) and PyContextNLP (PE) to:**
  - **Extract and annotate mentions and relations of d-dimer and pulmonary embolism.**
  - **Classify the clinical documents into: Affirmed and Negated.**
  - **Evaluate the system classified documents with manually classified ones.**

# Document Selection

- MIMIC-III documents
- Using MIMIC-III MySQL server
  - Selected from NOTEVENTS containing **d-dimer**, **CT**, and **pulmonary emboli (PE)**
    - 335 notes were selected and saved into Brat directories.
    - 30 notes for training
    - 20 notes for testing

# The annotation scheme

## the reference standard annotation

**New Annotation**

**Text**

**Number**

**Search**

Google, Wikipedia

**Entity type**

- ☒ Low\_Ddimer
- ☐ High\_Ddimer
- ☐ Ddimer
- ☐ Value
- ☐ Unit
- ☐ PE
- ☐ positive\_DOC
- ☐ negative\_DOC

**Entity attributes**

Level: ? Negation: ? Experencer: ? Temporality: ?

**Notes**

Negation: ?

Negation: Negated

Negation: Affirmed

OK Cancel

	negative_DOC
1	[**2771-8-30**] 1:08 AM
2	CTA CHEST W&W/O C & RECONS; CT 100CC NON IONIC CONTRAST
3	Reason: PLEURITIC CP. +DDIMER.?PE
4	Field of view: 36 Contrast: OPTIRAY Amt: 100
5	
6	UNDERLYING MEDICAL CONDITION:
7	42 year old man with pleuritic CP.
8	+d-dimer
9	REASON FOR THIS EXAMINATION:
10	eval for PE
11	No contraindications for IV contrast
12	
13	WET READ: FKh MON [**2771-8-30**] 1:49 AM
14	No PE.

## Design (d-dimer pipeline)

### D-dimer: Regular Expression

D-dimer 910  
d-dimer (530)  
D-DIMER-3827  
D-Dimer-6229\*\*  
d-dimer of/to/was 3013  
D-Dimer-417  
D-Dimer-[\*Numeric Identifier 5683\*\*]\*  
D-dimer:[\*Numeric Identifier 5409\*\*] ng/mL

High d-dimer  
pos ddimer  
d dimer  
elevated ddimer  
+ D-dimer  
+DDimer  
increased d-dimer  
d-dimer returned positive

- 3 rules for avoiding cross/repeat extract d-dimer/ddimer.
- Some notes have more than 20 other values (Fibrino 449, Glucose 194, Lipase 160, ...), to separately match value will be hard.

```
rule=r'(?P<name>(d-dimer | ddimer))  
      (?P<n1>.{1,25}?)  
      (?P<value>[0-9]{1,4}(\.[0-9]{0,3})?\s*)  
      (?P<n2>\W*)  
      (?P<unit>(ug\| | ng\|ml | mg\| | nmol\|)?)'
```

```
rule1=r'(elevated | pos | positive | increased | high | \+)(.{1,20})?(d-dimer | d\s?dimer)'
```

```
rule2=r'(d-dimer | d\s?dimer)([^\0-9-:]{1,15})?(positive | pos)'
```

# D-dimer pipeline

- Instantiation pipeUtils Annotation and Document
  - Read and pre-process the clinical notes
  - Apply one d-dimer rule
  - Check the unit
  - Normalize the numeric value
  - Compare with the threshold (500 ng/mL)
  - Annotate d-dimer mention
  - Sequentially apply next rules, annotate mentions
  - Append/write annotations to each annotation files (same as ann file with different ext)
  - Classify the document
- 
- Mention level evaluation
    - Read annotation file and convert to data frame
    - Convert data frame to annotation object
    - Mention level evaluation



## Design (PE pipeline)

- Target:

PE

Pulmonary embolism/li/lus

septic embolism/li/lus

embolism/li/lus

pulmonary artery embolism/li/lus

- Modifier: tried many different modifiers
- According to the modifier to set the type of target PE
- According to the types of targets to classify the document

PyContextNLP

Developed

- PE target rules
- PE Modifiers
- PE Feature inferences
- PE Document inferences
- Revised DocumentClassifier module
- Built a process module
- Write annotation to file for later validation
- Built functions to read annotation file and convert to dataframe and annotation object for validation.



# PE pipeline

- Read rules
- Instantiation MyPipe (wrapper of PyContextNLP using PyRush as sentence segmenter)
- Save annotations dataframe to corresponding files
- Document level evaluation
  - By compare with manually annotated BRAT files, which has document level marks inside
- Mention level evaluation (for fp, fn)
  - Read saved annotation file and standard ann file to compare. (training process)





# Results

- Trained, revised by 30 training set notes
- Tested by 20 test set notes.
- D-dimer pipeline can correctly extract, annotate and classify all test notes.
- The result of PE pipeline:
  - Training set:
    - Precision: 1.0
    - Recall: 1.0
    - F1: 1.0
  - **Testing set: (2 fn out of 20 notes)**
    - **Precision: 1.0**
    - **Recall: 0.8**
    - **F1: 0.89**

# Discussion and challenges

## For d-dimer pipeline

- Most importance is not overlap between rules
- Changed from 2 rules to 3 regular expression rules
- Example: `rule1 = (d-dimer)(Value)(Unit)`  
`rule2_old = (pre_modifier)?(d-dimer)(post_modifier)?`
- Try to use less greedy rules (match more than wanted)
- The value should normalize according to the units


# Discussion and challengers (Cont.)

## For PE pipeline

- Use restricted target rules and modifier rules.
  - `\bno\s{1,10}(?=pe)`
  - `\bno\s{1,10}(?=pulmonary embolism)` instead of `\bno\b`
- Challenge
  - Right/left no .....left/right yes
- Example: **(one fn)**
  - No central PE on L, but .... Not excluded...
  - No central pulmonary embolus is identified on the **left**, .... Subsegmental pe involving the **right** low lobe ...
- Possible solution: May use IMPRESSION part, Conclusion part for classification.
- Sentence segment/context window is very important. **(one fn)**
- Possible solution: Define context window in rule definition.
- Modifier: tiny



# Future work

- Modify rules
  - Test more notes
  - Clean the code
- 



**THANK YOU**