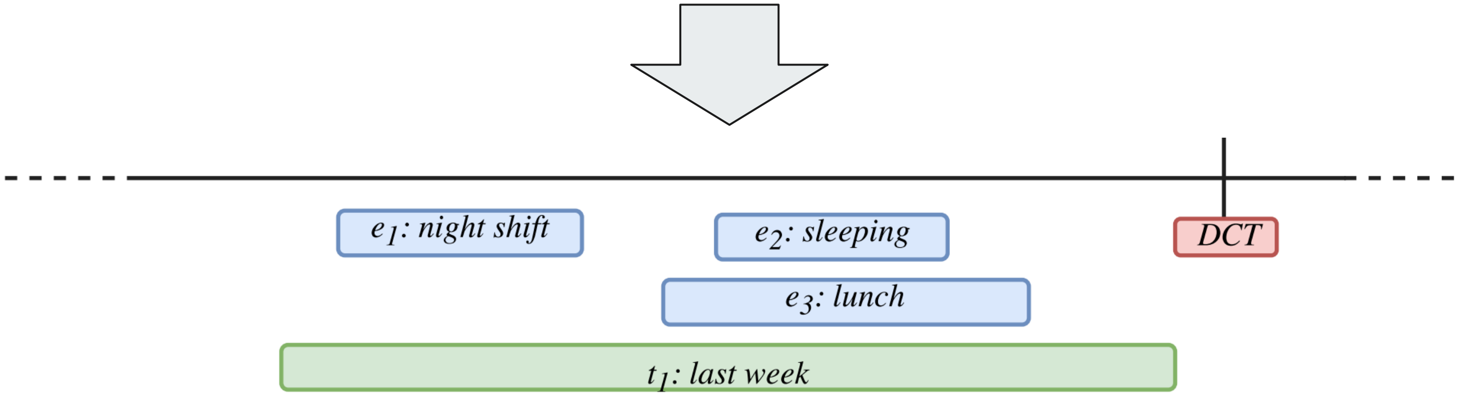
Temporal Information Extraction

Temporal Information Extraction

Last week, John had a night shift, so he slept during lunch.

Temporal Information Extraction

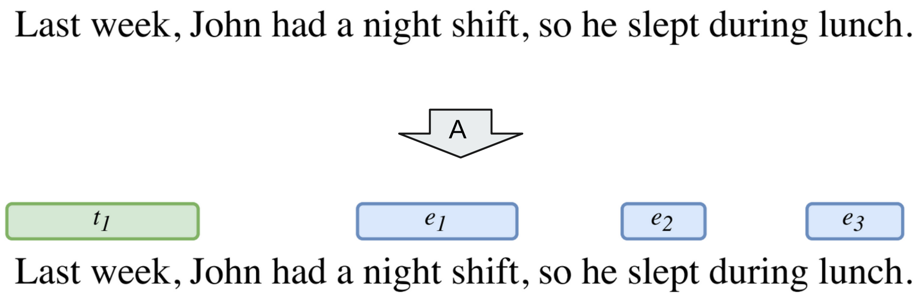
Last week, John had a night shift, so he slept during lunch.



Last week, John had a night shift, so he slept during lunch.

Three Steps

Three Steps

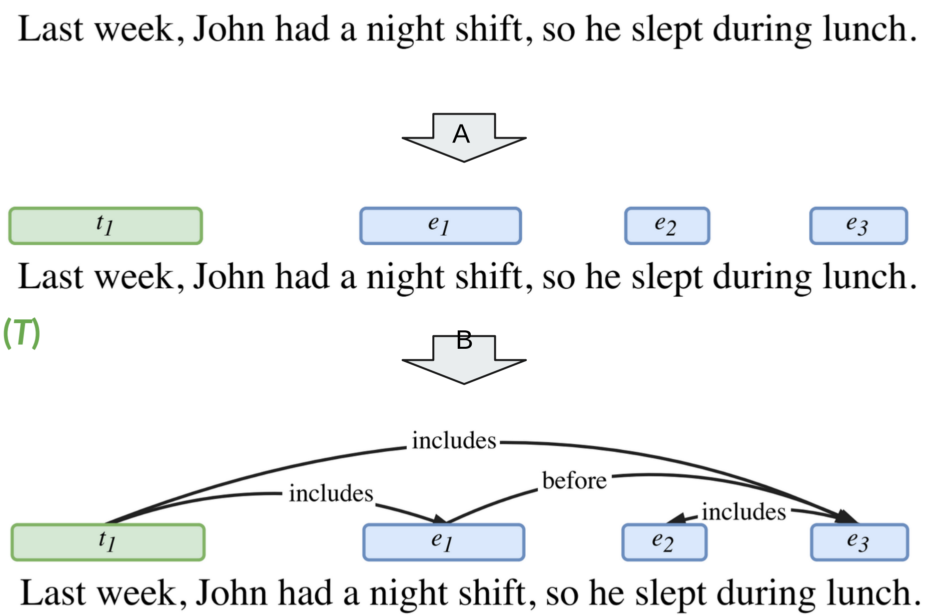


A. Recognize **events (*E*)** and **temporal expressions (*T*)**

Three Steps

A. Recognize **events (*E*)** and **temporal expressions (*T*)**

B. Recognize **temporal relations (*R*)**

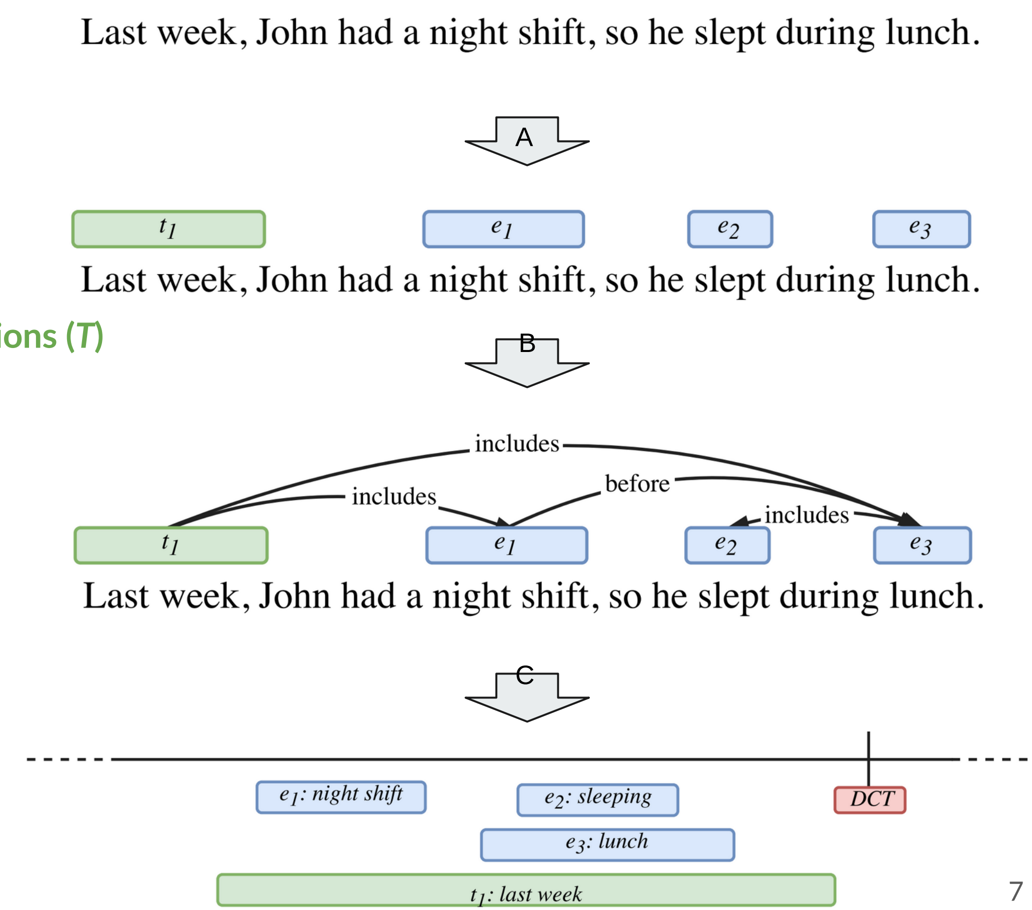


Three Steps

A. Recognize **events (*E*)** and **temporal expressions (*T*)**

B. Recognize **temporal relations (*R*)**

C. Construct a timeline

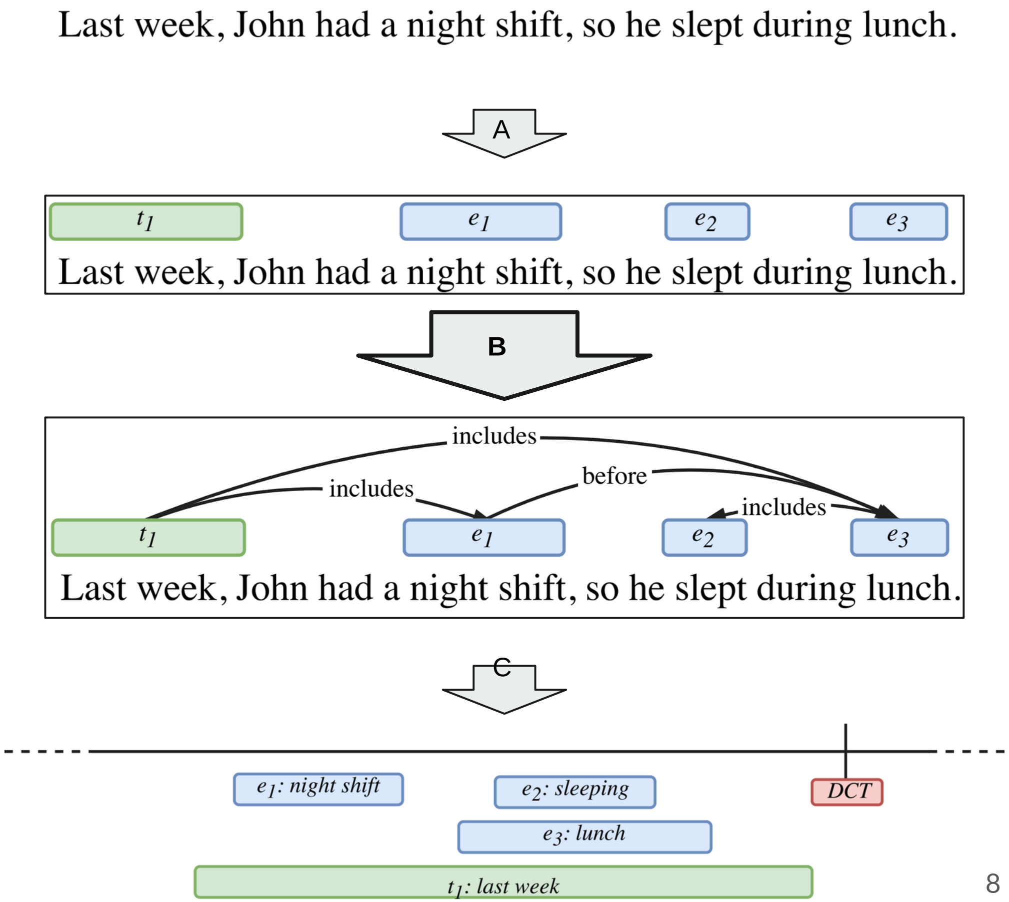


Three Steps

A. Recognize **events** and **temporal expressions**

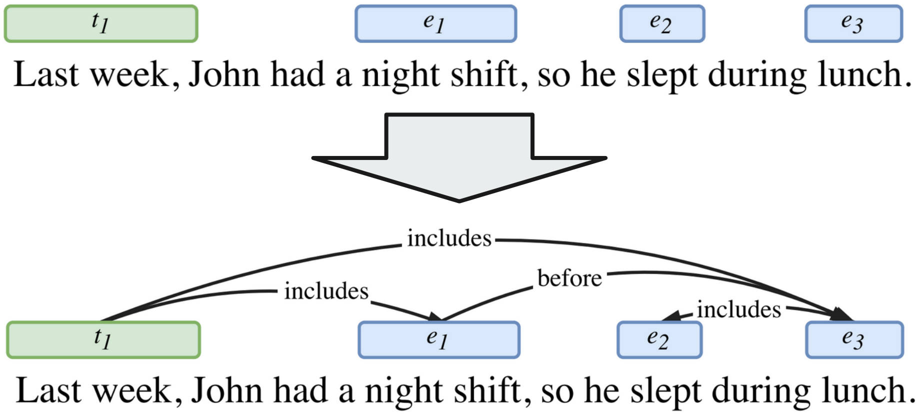
B. Recognize **temporal relations** (***R***)

C. Construct a timeline



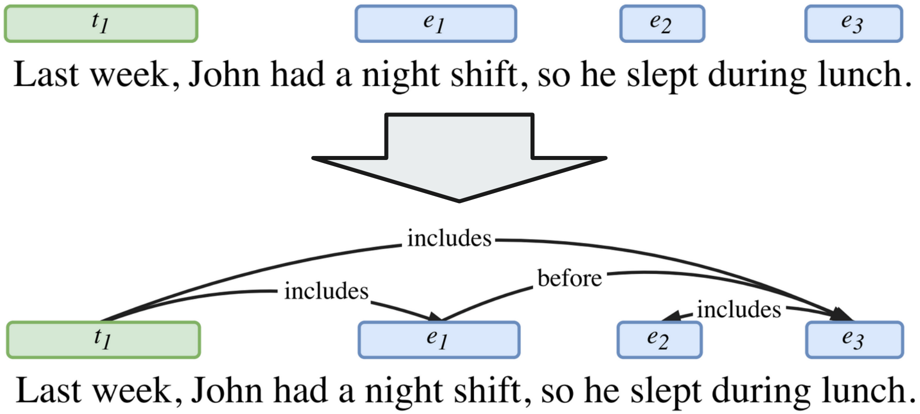
Temporal Relation Extraction

- Pairwise Classification
  - EUT x EUT



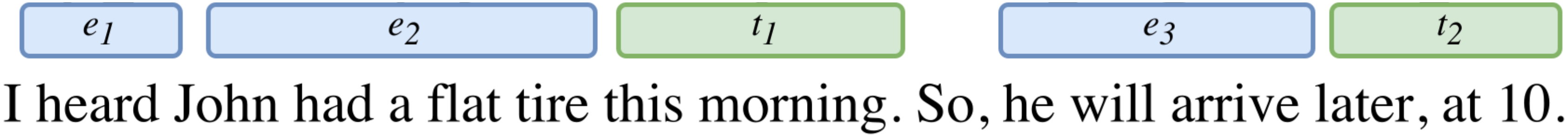
Temporal Relation Extraction

- Pairwise Classification
  - E∪T x E∪T
- Relation Types:
  - 13 Allen's Interval Relations
  - Today in the Exercises:
    - BEFORE
    - INCLUDES
    - SIMULTANEOUS
    - NONE (no arrow)



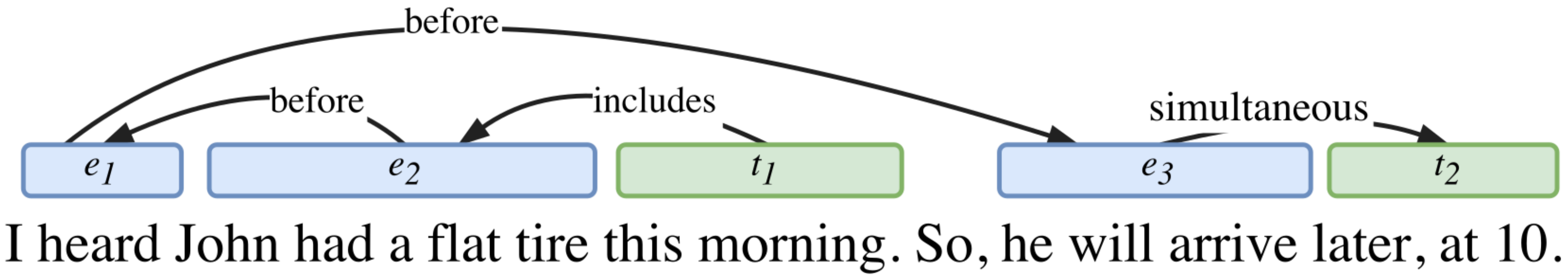
Temporal Relation Annotation (Ex 1.1)

- BEFORE
- INCLUDES
- SIMULTANEOUS
- NONE (no arrow)

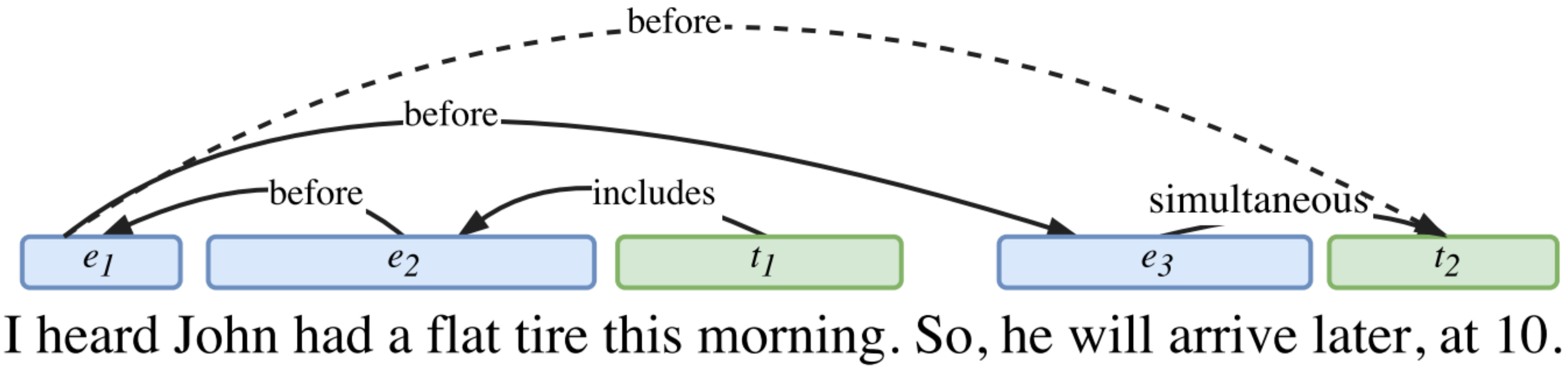


Temporal Relation Annotation (Ex 1.1*)

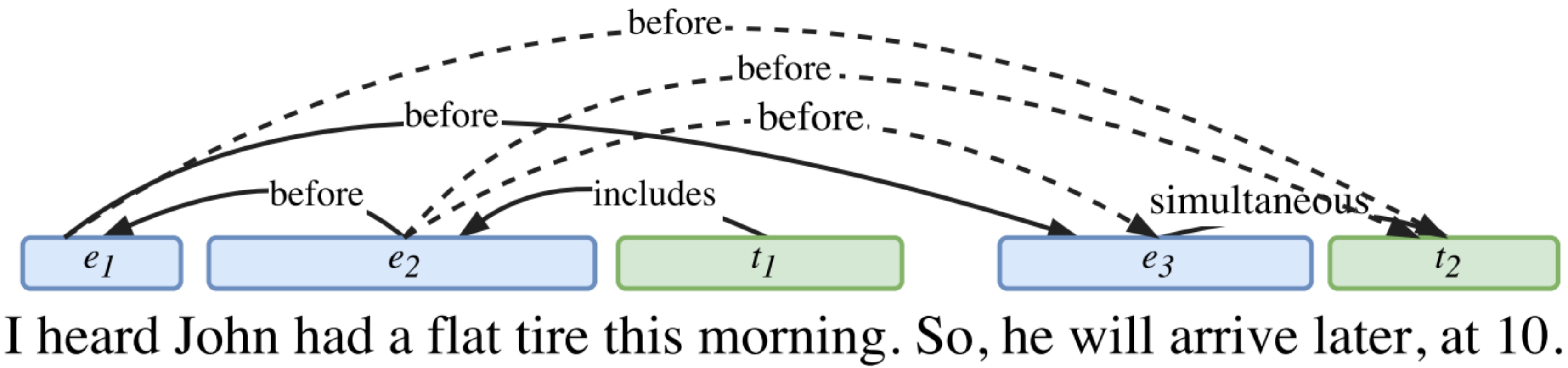
- BEFORE
- INCLUDES
- SIMULTANEOUS
- NONE (no arrow)



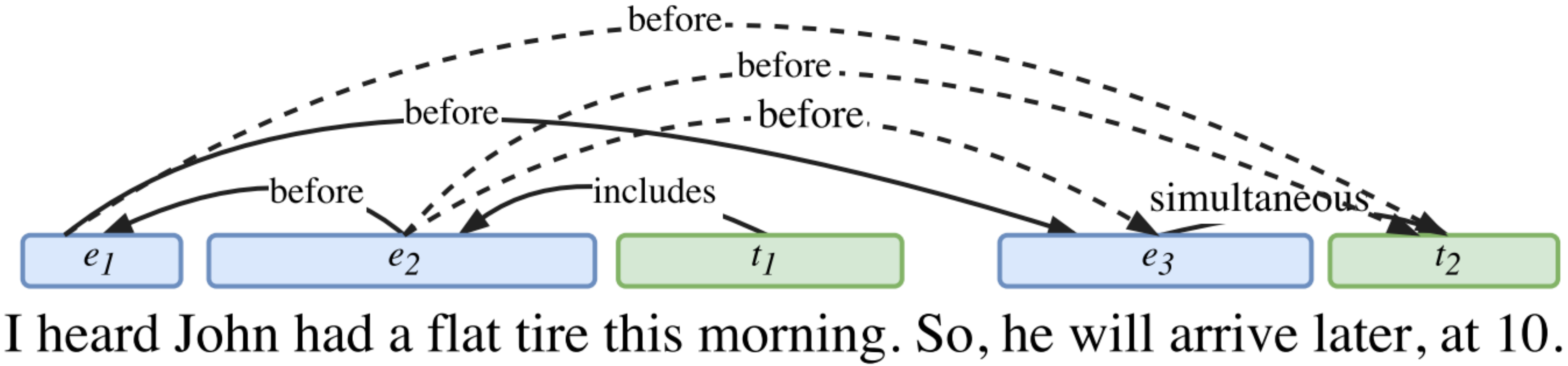
Temporal Relation Annotation



Temporal Relation Annotation

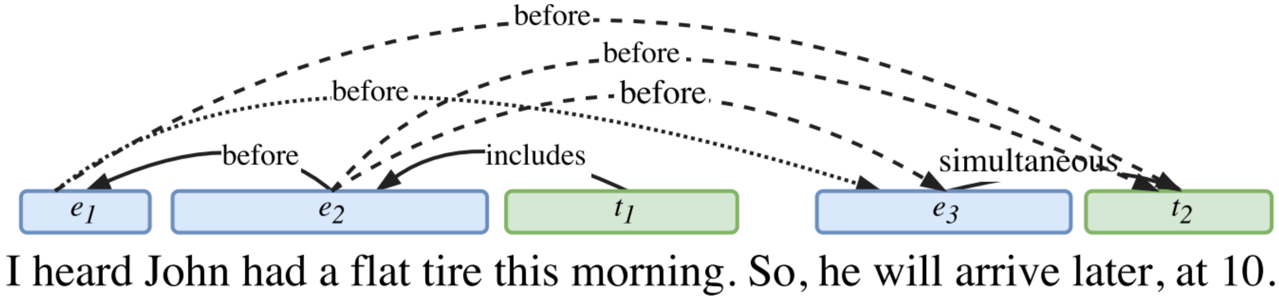


Temporal Relation Annotation



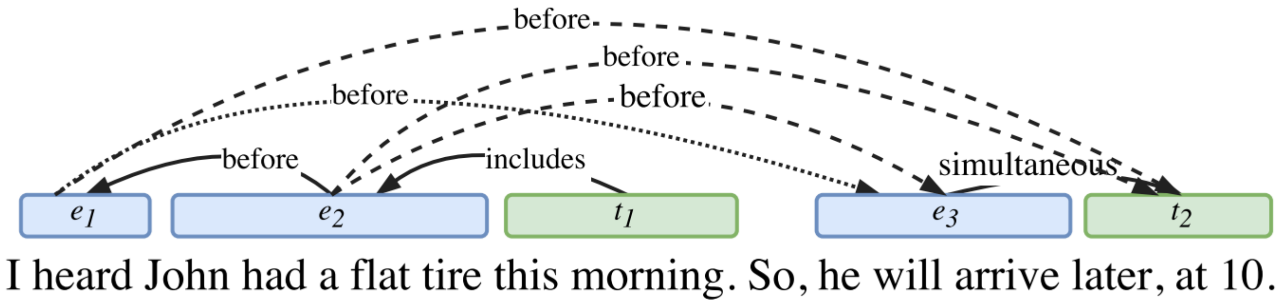
- Inter-Annotator Agreement

Temporal Relation Annotation (Ex 1.2)



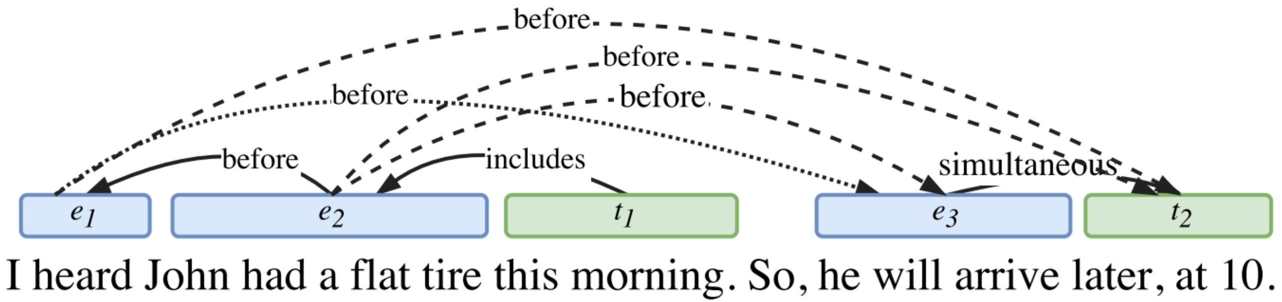
- Inter-Annotator Agreement as F_1-measure
  - A_1: full + dotted
  - A_2: full + dashed
- F_1 = (2 * Precision * Recall) / (Precision + Recall) = **?**
  - Precision = $|A_1 \cap A_2| / |A_1|$
  - Recall = $|A_1 \cap A_2| / |A_2|$

Temporal Relation Annotation (Ex 1.2*)



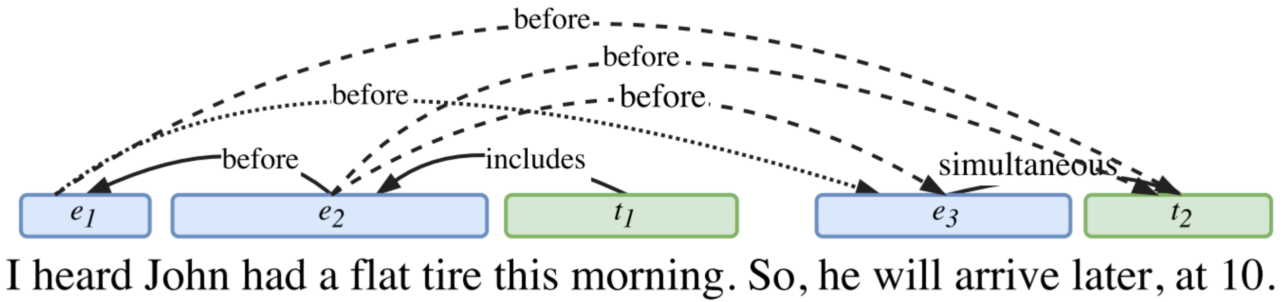
- Inter-Annotator Agreement as F_1-measure
  - A_1: full + dotted
  - A_2: full + dashed
- F_1 = (2 * Precision * Recall) / (Precision + Recall) = (2 * .75 * .5)/(.75 + .5) = .6
  - Precision = $|A_1 \cap A_2| / |A_1|$ = 3 / 4 =.75
  - Recall = $|A_1 \cap A_2| / |A_2|$ = 3 / 6 = .5

Temporal Reasoning (Ex. 1.3)



- What rules can you come up with that infer A2's (dashed arrows) relations from A1's relations (dotted + solid arrows)?

Temporal Reasoning (Ex. 1.3*)



- What rules can you come up with that infer A2's (dashed arrows) relations from A1's relations (dotted + solid arrows)?
  - before(i,j) and before(j,k) → before(i,k)
  - before(i,j) and simultaneous(j,k) → before(i,k)

Temporal Reasoning

- Training
- Inference
- Evaluation

Temporal Reasoning

- Training
- **Inference**
- Evaluation

Greedy Inference

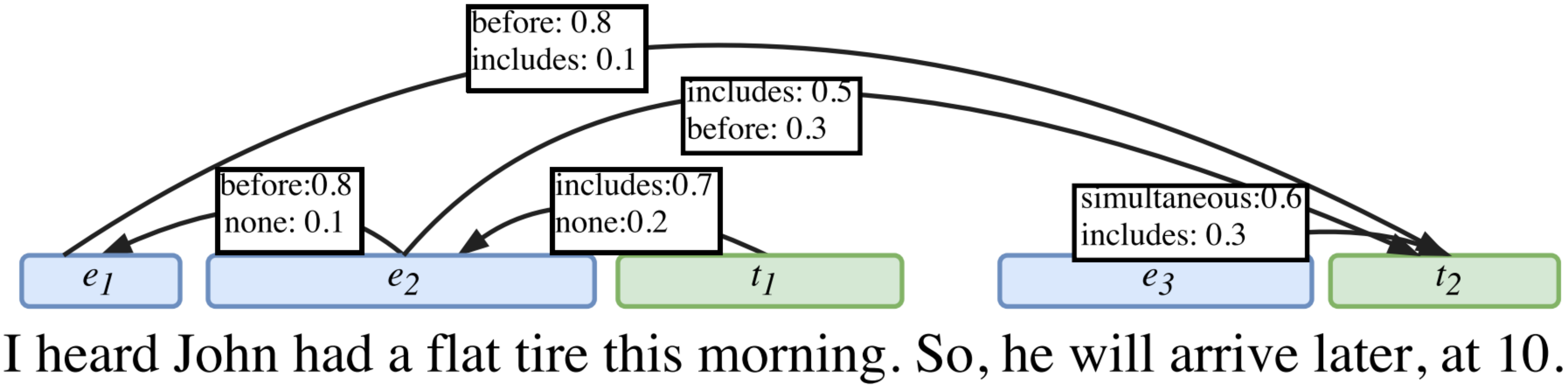
- Pairwise Classification
  - Candidates $c_{ij} \in E \cup T \times E \cup T$
  - Relation types $r \in$ {before, includes, simultaneous, none}
  - Assume we already trained a model $P(r \mid c_{ij}, s)$

Greedy Inference

- Pairwise Classification
  - Candidates $c_{ij} \in E \cup T \times E \cup T$
  - Relation types r $\in$ {before, includes, simultaneous, none}
  - Assume we already trained a model $P(r \mid c_{ij}, s)$

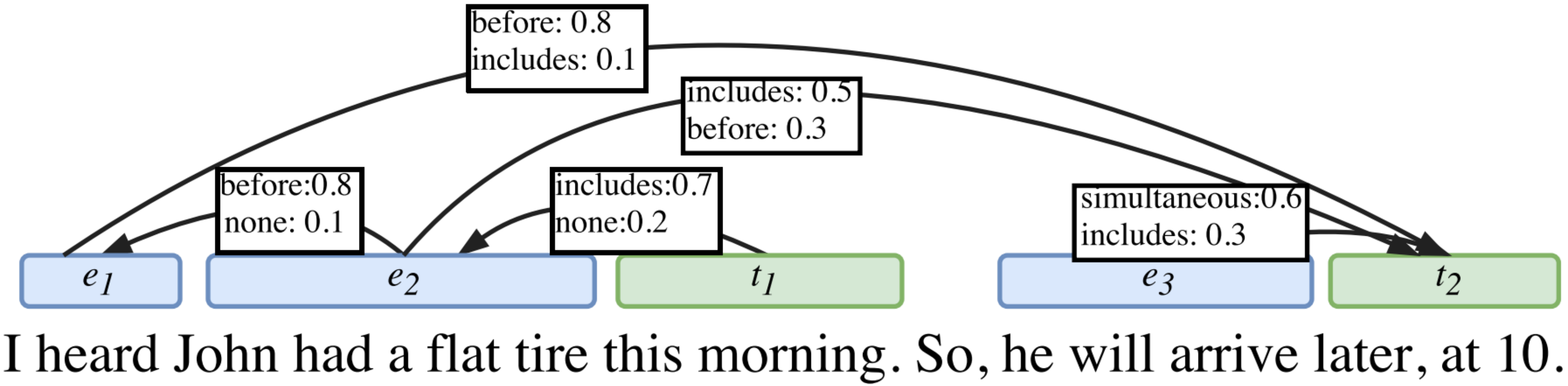
- For each sentence *s*:
  - For each candidate c_{ij}:
    - Predicted relation type = $\mathrm{argmax}\ P(r \mid c_{ij}, s)$

Greedy Inference: *argmax* $P(r \mid c_{ij}, s)$ (Ex.



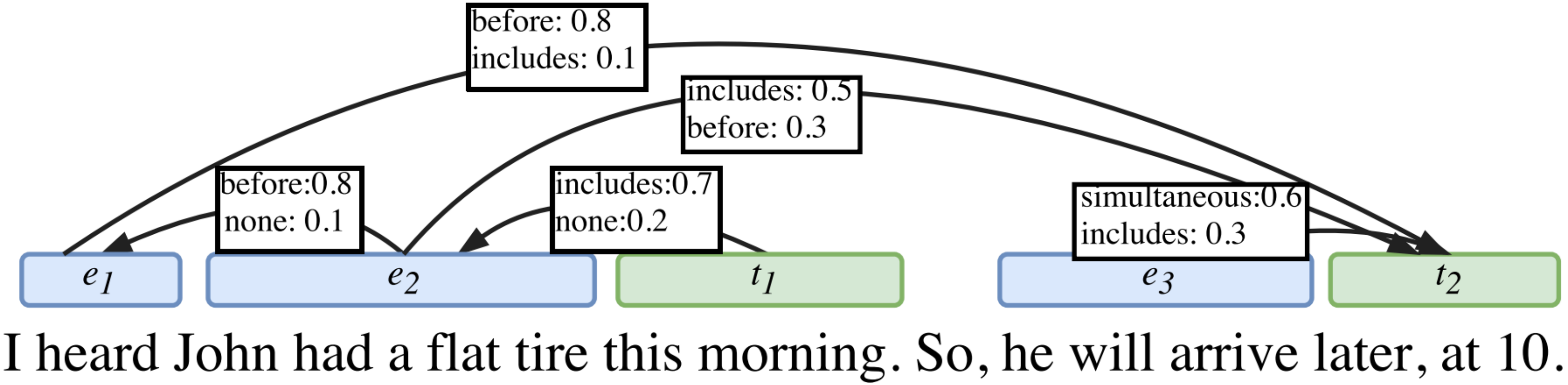
- What is the problem for this model output if we use greedy inference?

Greedy Inference: *argmax P(r | c_{ij}, s)* (Ex.



- What is the problem for this model output if we use greedy inference?
  - Not possible to create a timeline! (because $e_2 \rightarrow t_2$ is 'includes')

Greedy Inference: *argmax* $P(r \mid c_{ij}, s)$



- Possible Solution: Inform inference about temporal rules

Temporal Inference with ILP

- Integer Linear Programming (ILP)
  - Formulate inference as linear objective
    - Possibly integer or boolean variables
    - Constraints on variable values (>, <, =)

  - Solving by maximizing objective

Temporal Inference with ILP

- Greedy inference as ILP:
  - r_{ij} ~ binary decision variable saying that c_{ij} is of type r

$$Objective = \max \sum_{c_{ij}}^{C} \sum_{r}^{R} r_{ij} \cdot P(r|c_{ij}, s)$$

$$\forall c_{ij} : \sum_{r}^{R} r_{ij} = 1$$

Temporal Inference with ILP (Ex. 2.2)

- Formulate the rules from Ex. 1.3* as ILP constraints:
  - before(i,j) and before(j,k) → before(i,k)
  - before(i,j) and simultaneous(j,k) → before(i,k)

$$Objective = \max \sum_{c_{ij}}^{C} \sum_{r}^{R} r_{ij} \cdot P(r|c_{ij}, s)$$

$$\forall c_{ij} : \sum_{r}^{R} r_{ij} = 1$$

?

Logical Rule as ILP expression (Ex. 2.2)

A	**B**	**C**	**A and B → C**	**?**
0	0	0	True	
0	0	1	True	
0	1	0	True	
0	1	1	True	
1	0	0	True	
1	0	1	True	
1	1	0	False	
1	1	1	True	

Logical Rule as ILP expression (Ex. 2.2*)

A	B	C	A and B → C	C - A - B > -2
0	0	0	True	0 - 0 - 0 > -2
0	0	1	True	1 - 0 - 0 > -2
0	1	0	True	0 - 0 - 1 > -2
0	1	1	True	1 - 1 - 0 > -2
1	0	0	True	0 - 1 - 0 > -2
1	0	1	True	1 - 0 - 1 > -2
1	1	0	False	0 - 1 - 1 > -2
1	1	1	True	1 - 1 - 1 > -2

Logical Rule as ILP expression (Ex. 2.2*)

You might have come up with:

A * B <= C

However, this is not **linear** in the decision variables!!!

A	B	C	A and B → C	C - A - B > -2
0	0	0	True	0 - 0 - 0 > -2
0	0	1	True	1 - 0 - 0 > -2
0	1	0	True	0 - 0 - 1 > -2
0	1	1	True	1 - 1 - 0 > -2
1	0	0	True	0 - 1 - 0 > -2
1	0	1	True	1 - 0 - 1 > -2
1	1	0	False	0 - 1 - 1 > -2
1	1	1	True	1 - 1 - 1 > -2

Temporal Inference with ILP (Ex. 2.2*)

- Formulate additional ILP constraints that incorporate the rules from Ex. 1.3:
  - before(i,j) and before(j,k) → before(i,k)
  - before(i,j) and simultaneous(j,k) → before(i,k)

$$Objective = \max \sum_{c_{ij}}^{C} \sum_{r}^{R} r_{ij} \cdot P(r|c_{ij}, s)$$

$$\forall c_{ij} : \sum_{r}^{R} r_{ij} = 1$$

$$\forall c_{ij} \forall c_{jk} : before_{ik} - before_{ij} - before_{jk} > -2$$

$$\forall c_{ij} \forall c_{jk} : before_{ik} - simultaneous_{ij} - before_{jk} > -2$$

SpanBERT

Possible Spans

- Random sub-word tokens
- Random whole words
- Noun Phrases
- Verb Phrases
- Named Entities
- Geometric Spans (Best one from paper, some fancy)

Does it make sense?

- Arguments usually cover a span of words
- SpanBERT should perform strong with this type of predictions
- SpanBERT might be too complex for the SRL
- It would do the same thing as achieved by the signal from the labels itself

Write down objective function

- We combine the SBO loss and the Categorical prediction loss

- Given N sentences, where each sentence X^n is a sequence of words $x_1^n, \ldots, x_T^n$
- Each sentence has a set of S^n masked spans
    - each span s_i^n has a length of K, *so it* starts at $s_{i,1}^n$ and ends at $s_{i,K}^n$

So s_0^n is the word at the position just before the span, and s_{K+1}^n after the span

- For the span s_i^n we have position annotations $p_{i,1}^n$ through $p_{i,K}^n$
- y_k^n is the word for masked token $s_{i.k}^n$
- z_t^n is the SRL label for word x_t^n
- The objective models have the sets of parameters Θ_1 and Θ_2

$$\mathcal{L} = \sum_{n \in N} \mathcal{L}^n$$

$$\mathcal{L}^n = \sum_{s_i \in S^n} \sum_{k \in K} \mathcal{L}_{SBO}(y_{i,k}^n) + \sum_{t \in T} \mathcal{L}_{CAT}(z_t^n)$$

$$= \sum_{s_i \in S^n} \sum_{k \in K} \left[-\log P(y_{i,k}^n | s_{i,0}^n, s_{i,K+1}^n, p_{i,k}^n; \Theta_1) \right] + \sum_{t \in T} -\log P(z_t^n | X^n; \Theta_2)$$

Where have we seen it before?

- With the BIO labels for SRL

- For argument1 we have: B-ARG1, I-ARG1
  For argument2 we have: B-ARG2, I-ARG2
  …

- These arguments already indicate precisely where spans begin