MATTER Cycle Deep Dive

Task Formalization

Jin Zhao

Brandeis University

February 9, 2025

Today's Agenda

1. Review of sequence labeling
2. Deep dive into task formalization
3. Types of annotation tags
4. Document Type Definitions (DTDs)
5. JSON Schema for annotations
6. Prompts as task specifications
7. Configuring annotation tools

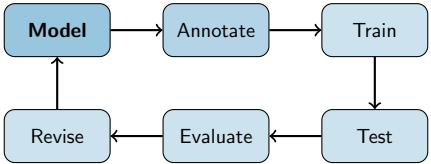
Quick Review

From last lecture on sequence labeling:

- NER identifies named entities (PER, ORG, LOC, etc.)
- BIO tagging scheme: Beginning, Inside, Outside
- Span boundaries are the main challenge
- Tokenization must be done before annotation

Today: How do we formally specify annotation tasks?

The MATTER Cycle Revisited



Focus today: The "Model" phase

- Defining what you want to annotate
- Formalizing your annotation schema
- Creating specifications that tools can use

Why Formalize?

Benefits of formal task specification:

1. **Validation:** Check that annotations are well-formed
2. **Consistency:** Ensure annotators follow the schema
3. **Tool configuration:** Set up annotation interfaces
4. **Documentation:** Clear record of what was annotated
5. **Reproducibility:** Others can replicate your work

Key insight: A formal specification is a contract between you and your annotators (human or LLM)

Four Types of Annotation Tags

1. **Non-consuming tags:** Document or sentence-level labels
   - Not tied to specific text spans
   - Example: Document sentiment, genre classification
2. **Span/Extent tags:** Labels on text spans
   - Anchored to character or token offsets
   - Example: Named entities, noun phrases
3. **Link/Relation tags:** Connections between spans
   - Can be directed or undirected
   - Example: Coreference, semantic relations
4. **Attribute tags:** Properties of other tags
   - Attached to existing annotations
   - Example: Entity type, relation polarity

Non-Consuming Tags

Definition: Tags not directly associated with a span

Common uses:

- Document-level classification (sentiment, topic)
- Sentence-level labels (grammaticality, quality)
- Metadata (author, date, source)

Example:

Movie Review Annotation

- Document: "Roger Dodger is one of the most compelling variations on this theme."
- Sentiment: Positive
- Genre: Drama

Format: Often stored as document metadata or in a separate column

Span/Extent Tags

Definition: Labels anchored to contiguous text

Specification requires:

- Tag name (e.g., PERSON, LOCATION)
- Start offset (character or token index)
- End offset
- Optional: attributes

Example (standoff format):

```
T1   PERSON 0 12   ‘‘Barack Obama’’
T2   LOCATION 26 36   ‘‘Washington’’
```

Key decisions:

- Character offsets vs. token indices?
- Inclusive vs. exclusive end offsets?
- How to handle whitespace?

Link/Relation Tags

Definition: Associations between spans

Types of relations:

- **Directed:** Source → Target (e.g., "works_for")
- **Undirected:** Symmetric (e.g., "related_to")
- **N-ary:** More than two arguments

Example:
Text: "Tim Cook is the CEO of Apple."

- T1: PERSON "Tim Cook"
- T2: ORG "Apple"
- R1: CEO_OF Arg1:T1 Arg2:T2

Challenge: Relations can cross sentence boundaries

Attribute Tags

Definition: Properties attached to existing annotations

Common attributes:

- Entity subtype (PER $\rightarrow$ politician, athlete, actor)
- Polarity (positive/negative)
- Confidence level
- Temporal information (past/present/future)

Example:

- T1: NAME "Julie Delpy"
- A1: Type T1 Actor

Benefit: Separate core annotation from secondary properties
Can annotate in multiple passes (first spans, then attributes)

Knowledge Check

What type of tag is each example?

1. Document-level movie genre classification
2. Named entity recognition (NER)
3. Semantic role labeling (SRL)
4. Syntactic dependencies
5. Part-of-speech tags and verb tense
6. Document sentiment score

Knowledge Check

What type of tag is each example?

1. Document-level movie genre classification
2. Named entity recognition (NER)
3. Semantic role labeling (SRL)
4. Syntactic dependencies
5. Part-of-speech tags and verb tense
6. Document sentiment score

Answers:

1. Non-consuming
2. Span
3. Span + Link (predicate + arguments)
4. Link
5. Span + Attribute
6. Non-consuming

Document Type Definitions (DTDs)

DTDs formally specify XML annotation schemas

Used by tools like MAE (Multi-document Annotation Environment)

Example DTD for movie review annotation:

```
<!ENTITY name "movieReviewTask">

<!-- Span tags -->
<!ELEMENT Name ( #PCDATA ) >
<!ELEMENT Movie ( #PCDATA ) >

<!-- Link tags -->
<!ELEMENT Acts_In EMPTY >
<!ELEMENT Directs EMPTY >

<!-- Attribute tags -->
<!ATTLIST Name Is_A (Actor|Director|Writer) #IMPLIED >
```

DTD Components

Key DTD elements:

- `<!ELEMENT>` – Define tag types
- `<!ATTLIST>` – Define attributes for tags
- `#PCDATA` – Parsed character data (text content)
- `EMPTY` – No text content (links)
- `#IMPLIED` – Attribute is optional
- `#REQUIRED` – Attribute is mandatory

Example attribute definition:

```
<!ATTLIST Sentiment
    Polarity (Positive|Negative|Neutral) #REQUIRED >
```

Benefit: Validation – reject invalid annotations automatically

JSON Schema for Annotations

Modern alternative to DTDs

Used by Label Studio, Hugging Face datasets, APIs

Example JSON annotation:

```
{
  "text": "Apple announced a new iPhone.",
  "entities": [
    {"start": 0, "end": 5, "label": "ORG"},
    {"start": 22, "end": 28, "label": "PRODUCT"}
  ],
  "sentiment": "neutral"
}
```

Advantages over DTD:

- Native to most programming languages
- Easier to read and write

JSON Schema Definition

Formally specify your JSON structure:

```
{
  "type": "object",
  "properties": {
    "text": {"type": "string"},
    "entities": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "start": {"type": "integer"},
          "end": {"type": "integer"},
          "label": {"enum": ["PER","ORG","LOC"]}
        },
        "required": ["start", "end", "label"]
```

ISO Linguistic Annotation Framework (LAF)

Standard model for linguistic annotation projects

Key principles:

1. **Standoff annotation:** Store annotations separately from text
2. **Character offsets:** Reference text by position
3. **Layered annotation:** Each level in separate document
4. **Standard vocabularies:** Use established labels when possible

Benefits:

- Original text remains unchanged
- Multiple annotation layers can coexist
- Easy to add/remove annotation types
- Supports overlapping annotations

Resource: ISOCat data categories – `https://datcatinfo.net`

Prompts as Task Specifications

New paradigm: LLM prompts as lightweight specifications

Traditional specification:

- Formal DTD/Schema
- Detailed guidelines document
- Annotator training
- Tool configuration

Prompt-based specification:

- Natural language instructions
- Few-shot examples
- Output format template
- Iterative refinement

Key insight: The prompt IS the task specification for LLM annotation
Clear prompt = clear task definition

From Guidelines to Prompts

Converting human guidelines to LLM prompts:

Human guideline:
"Annotate all person names in the text. Include full names, first names only, and nicknames. Do not include titles like 'Dr.' or 'President' unless they are part of the proper name."

LLM prompt:

```
Extract all person names from the text. Include:
- Full names (e.g., "John Smith")
- First names (e.g., "John")
- Nicknames (e.g., "Johnny")

Do NOT include titles (Dr., Mr., President) unless
part of proper name.

Return as JSON: {"entities": [{"text": "...",
"start": N, "end": N, "label": "PERSON"}]}
```

Configuring Annotation Tools

Your specification must be operationalized

Tool configuration requires:

1. Define label set (tags available to annotators)
2. Set up annotation types (spans, relations, attributes)
3. Configure validation rules
4. Design annotation interface
5. Set up export format

Example tools:

- **brat:** `annotation.conf` file
- **Label Studio:** XML labeling config
- **MAE:** DTD file
- **Prodigy:** Python recipe

Choosing an Annotation Tool

Considerations when selecting a tool:

1. **Task support:** Does it handle your annotation types?
2. **Platform:** Web-based vs. desktop? Team access?
3. **Export formats:** BIO, JSON, standoff?
4. **LLM integration:** Pre-annotation support?
5. **Cost:** Open source vs. commercial?

No perfect tool exists – choose based on your specific needs

Tip: Check GitHub for tools that match your task

Best Practices for Task Formalization

1. **Start simple:** Begin with minimal schema, add complexity as needed
2. **Use examples:** Real annotated examples clarify definitions
3. **Document everything:** Future you will thank present you
4. **Validate early:** Test your schema before full annotation
5. **Version control:** Track schema changes over time
6. **Consider the model:** What will your ML system actually use?

Golden rule: If annotators are confused, your specification is unclear

Next Class: Overview of Tasks III

Lecture 9 (Feb 11): Relation and Complex Annotation

Topics:

- Relation extraction
- Coreference resolution
- Semantic role labeling
- Event extraction
- Complex annotation structures

Reading: Pustejovsky & Stubbs, Chapter 6

Key Takeaways

1. **Four tag types:** Non-consuming, span, link, attribute
2. **DTDs** formally specify XML annotation schemas
3. **JSON Schema** is the modern alternative for most workflows
4. **ISO LAF** provides principles for annotation design
5. **Prompts** serve as task specifications for LLM annotation
6. **Tool configuration** operationalizes your specification

Questions?

Office Hours: Wednesdays 1-3pm, Volen 109

jinzhao@brandeis.edu