COSI-230B: Natural Language Annotation for Machine Learning

Lecture 7: The Annotation Design Pipeline

Jin Zhao

Brandeis University

February 9, 2026

Logistics

1. **Office Hours**
   - Office hours are **by appointment only** this week
   - Email me to schedule: ✉ jinzhao@brandeis.edu

2. **Annotation RA Opportunity**
   - Kenneth Lai is hiring annotators for an **Action AMR Annotation Project**
   - Label actions in videos using Abstract Meaning Representation (AMR)
   - No prior AMR experience needed — they will train you
   - Up to 20 hours total, $20/hour, GUI-based annotation tool
   - Great chance to get hands-on experience with real annotation work!
   - Contact: klai12@brandeis.edu

Today's Plan

Three steps of annotation design:

1. **Task Formulation** — What are we predicting?
   Prediction targets, observability, reformulation, when not to annotate
2. **Schema Design** — What labels do we use?
   Inductive bias, structure, granularity, collapsibility, the "Other" trap
3. **Writing Guidelines** — How do annotators decide?
   Operationalization, decision procedures, examples, edge cases, common failures

+ Discussion + LLM diagnostics

Theme: Annotation quality is decided *before* anyone starts labeling

Part 1: Prediction Target vs. Label

Prediction target

The real-world thing you care about

- "Will this user churn?"
- "Is this news misleading?"
- "Is this patient depressed?"

Lives in the real world. Often not directly observable.

Annotation label

The proxy an annotator can actually assign

- "Did the user express dissatisfaction?"
- "Does the headline contradict the body?"
- "Does the text mention PHQ-9 symptoms?"

Lives in the annotation tool. Must be observable.

The Gap in Action

Example: Suppose your prediction target is "Identify content that causes harm."

Consider this text: "As a nurse, I think patients like you are exhausting."

Different operationalizations give different answers:

Label definition	**Agreement**	**Verdict**
Contains profanity or slurs	High	Not toxic
Insults a specific person	Moderate	Toxic
Would violate Reddit's content policy	Moderate	Depends on subreddit
Would make the target feel unwelcome	Low	Toxic
A reasonable person would find it offensive	Very low	Who knows

Same text, five operationalizations, five different answers.

"Detect toxicity" is not a task. It is a family of tasks.

Can You Actually See It?

Observability = Can the annotator find the answer in the data you show them?

Observable in text:

- Specific words used
- Who is addressed
- Explicit claims made
- Text structure

NOT observable:

- Speaker's true intent
- Reader's emotional reaction
- Whether a claim is actually true
- What happened after

Quick Test

Text: "I love how this project is going."

Sarcastic? **You can't tell.** Intent is not in the text.

Fix: Ask "Does this *read as* sarcastic?" (reader perception) instead of "Is this sarcastic?" (speaker intent).

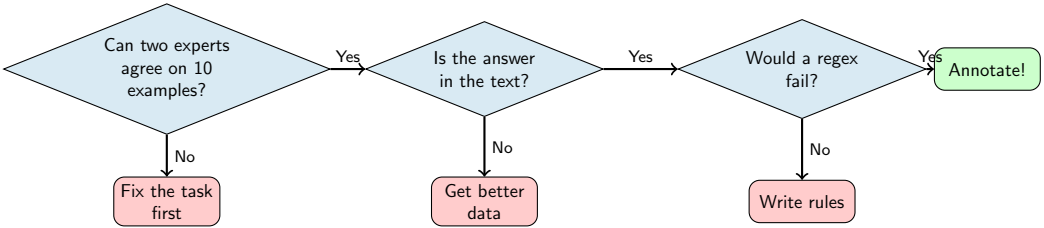
The Reformulation Move

When observability fails, reformulate the task:

Unobservable (bad)	**Observable (better)**
"Is this user depressed?"	"Does the text mention PHQ-9 symptoms?"
"Is this news article biased?"	"Does the headline use loaded language?"
"Will this comment start a fight?"	"Does this comment contain a personal attack?"
"Is this review fake?"	"Does the review describe specific product features?"

The left column asks annotators to be mind-readers.
The right column asks them to be text-readers.

When Annotation Is the Wrong Move



The first question is the most important.

If you can't get two experts to agree on a handful of examples, you have a *formulation* problem, not an annotation problem.

Annotation is expensive. Don't use it until you've earned the right to.

Part 2: Schemas = Inductive Bias

Schema: the set of labels annotators choose from

The schema decides:

- What categories exist
- What gets collapsed together
- What gets ignored entirely

Consequences:

- Model can only learn distinctions the schema encodes
- Merged categories can never be split later
- Missing categories are invisible

Key Claim

A schema looks like a simple list. It is not. It is a set of commitments about what matters.

Same Task, Three Schemas

Sentiment analysis — three schemas for the same data:

A: {Pos, Neg}

Theory: sentiment is binary. Pick a side.

B: {Pos, Neg, Neutral}

Theory: absence of sentiment is its own category.

C: {Pos, Neg, Neutral, Mixed}

Theory: a text can be positive *and* negative at once.

Test case: "Great food but terrible service."

- Schema A: Positive? Negative? (forced choice)
- Schema B: Neutral (sentiments cancel out?)
- Schema C: Mixed (both present simultaneously)

Same text, three schemas, three different training examples.

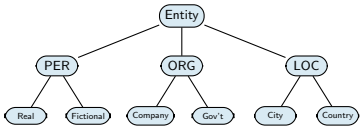
Flat vs. Hierarchical Labels

Flat: all labels are peers

PER ORG LOC MISC

Fast, high agreement, simple.
But: "Amazon" (company) and "WHO" (gov't org) get the same label.

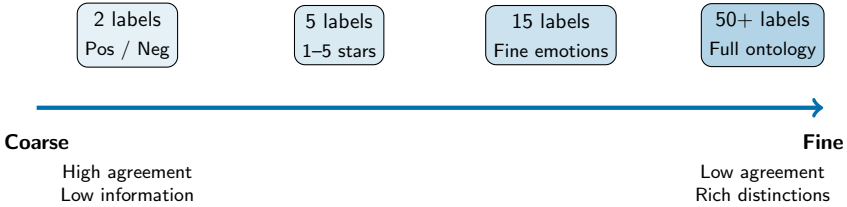
Hierarchical: labels in a tree



Richer distinctions, evaluate at multiple levels.
But: slower, needs more annotator training.

Rule of thumb: Start with the simplest structure that captures the distinctions your task actually needs.

The Granularity Trade-off



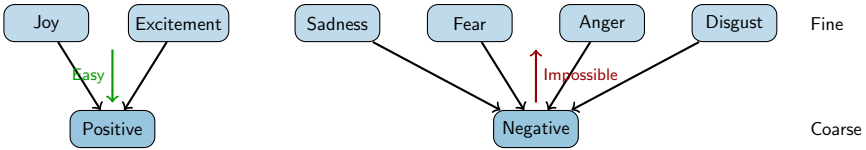
When to go coarse: spam filter (Spam / Not Spam), crowdworkers, limited data, need a reliable baseline

When to go fine: medical coding (ICD-10), expert annotators, research benchmarks, error analysis

No "correct" granularity. It depends on what you'll *do* with the labels.

The Collapsibility Principle

You can always merge fine labels into coarse ones. You can never split coarse labels into fine ones.



Practical advice: If in doubt, annotate one level finer than you think you need.

Once two things share a label, you have lost the distinction forever.

The "Other" Trap

Almost every schema has "Other." In poorly designed schemas, it grows to 20–40% of labels.

Why "Other" inflates:

1. Schema is missing categories
   (data has phenomena you didn't anticipate)
2. Boundaries are unclear
   (annotators bail out when unsure)
3. It's a shortcut
   (hard items → "Other" saves time)

Better alternatives:

- "Unclear / Cannot decide"
  (separates ambiguity from gaps)
- Free-text "Other" with required justification
  (forces explanation; feeds revision)
- No "Other" at all
  (force a choice; analyze disagreements)

Rule

A large "Other" means your schema needs work. In pilots, require a note for every "Other" — those notes are gold for revision.

Part 3: The Intuition Trap vs. Operationalization

Bad: relies on intuition

"Label as **toxic** if it is rude, disrespectful, or likely to make someone leave a conversation."

- "Rude" — by whose standards?
- "Likely to make someone leave" — mind-reading
- Three vague criteria joined by "or"

Better: operationalized

Label as **toxic** if *any* of:

1. Direct insult at a person ("you're an idiot")
2. Slur or dehumanizing term for a group
3. Explicit threat of harm

Not toxic: disagreement without insults, sarcasm without a target, profanity for emphasis.

What changed: vague adjectives → observable criteria, implicit logic → explicit, missing scope → explicit exclusions.

The Operationalization Test

A guideline is operationalized if:

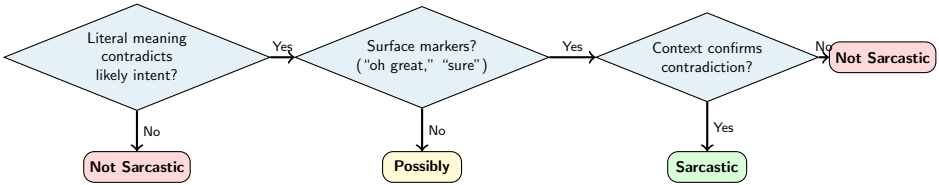
1. **Two strangers could follow it independently and mostly agree**
   Without training, calibration, or talking to each other
2. **The criteria are observable in the data**
   Not inferred, imagined, or dependent on external knowledge
3. **The logic is explicit**
   AND vs. OR, necessary vs. sufficient, scope of each rule
4. **Boundary cases are addressed, not deferred**

Rule

"Use your best judgment" is the white flag of guideline design. Every time you write it, you are transferring the design problem from you to the annotator.

Decision Procedures

Definitions tell annotators what. Procedures tell annotators how to decide.



Key features: each step = one yes/no question · every path → a label · ordered easiest to hardest

Procedures Localize Disagreement

Without a procedure: "Is 'You people are the worst' hate speech?"

- Annotator A: Yes (group targeting) Annotator B: No (no slur, no threat)
- Both using the definition correctly — just interpreting differently

With a procedure:

1. Contains a slur or dehumanizing term? → **Both say No**
2. Targets a protected group by name? → **Disagreement here** ("you people" = which people?)
3. Targeting explicit in context? → Check context

Now you know *exactly* where they disagree (step 2) and *exactly* what to clarify.

Procedures don't eliminate disagreement. They **localize** it.

Examples That Constrain, Not Just Illustrate

Three kinds of examples — you need all three:

1. **Prototypical** — clear, central cases
   - "I hate you and your stupid family" → **Toxic** (confirms basic understanding)
2. **Boundary** — near the edge, resolved with explanation
   - "You people never learn" → **Toxic** (targets group, even without slurs)
   - "This policy is idiotic" → **Not Toxic** (targets policy, not person)
3. **Contrastive pairs** — minimal pairs that flip the label
   - "You're an idiot" → **Toxic** vs. "That idea is idiotic" → **Not Toxic**
   - One feature changes (person → idea), label flips

Rule: At least half your examples should be boundary or contrastive. Prototypes alone only cover easy data.

Edge Cases First

Conventional approach:

1. Define labels
2. Write guidelines for typical cases
3. Pilot $\rightarrow$ discover edge cases
4. Patch guidelines
5. Repeat (slowly, painfully)

Edge-cases-first approach:

1. Collect 20–30 real examples
2. Identify the 10 hardest
3. Design labels & procedures for *those*
4. Easy cases take care of themselves

Principle

If your guidelines work on the hard cases, they will work on the easy ones. The reverse is not true.

Where to find edge cases: sample your data, adversarial brainstorming, prior work, ask an LLM, two-annotator pilot.

Common Guideline Failures

1. **Definition by synonym**
   "Toxic = harmful, offensive, or inappropriate." Synonyms are not definitions.
2. **Circular definition**
   "Hate speech is speech that expresses hate." Says nothing.
3. **Only positive examples**
   Annotators know what *is* toxic but not what *isn't*. Boundary undefined.
4. **Missing scope**
   Does "the text" mean the post, the thread, or the conversation?
5. **Implicit priorities**
   "Consider both tone and content." Which wins when they conflict?
6. **Guidelines too long to use**
   30-page doc no one reads. Keep reference to 3–5 pages; details in appendix.

Discussion: Three Schemas for Hate Speech

Same task — hate speech detection. Three competing schemas:

Binary

- Hate Speech
- Not Hate Speech

Typed

- Racist
- Sexist
- Homophobic
- Other Hate
- Not Hate Speech

Severity

- 0: No hate
- 1: Implicit bias
- 2: Derogatory
- 3: Dehumanizing
- 4: Call to violence

Questions to discuss:

1. A post is both racist *and* sexist. Which schema handles this? Which doesn't?
2. You're building a content moderation system. Which schema do you pick and why?
3. Where will annotators disagree the most? What does that tell you?

The Trade-offs

	Binary	**Typed**	**Severity**
Agreement	High	Moderate	Low
Expressiveness	Low	Moderate	High
Handles overlap?	No	Poorly	Implicitly
Actionable?	"Remove or not"	"What kind?"	"How bad?"
Collapsible?	Already flat	$\rightarrow$ Binary	$\rightarrow$ Binary

There is no universally "best" schema. The right choice depends on:

- What action you will take on the labels
- How much annotator training you can afford
- Whether you need to explain *why* something was flagged

Discussion: Diagnose These Guidelines

Three real-world guideline excerpts. What's broken?

#1 Sentiment

"Positive = the reviewer liked the product. Negative = did not like. Neutral = no strong opinion."

What happens with "Great battery but the screen cracked after a week"? What is "no strong opinion"?

#2 Named Entities

"Mark all entities. An entity is a real-world object denoted with a proper name. Use your best judgment for edge cases."

Is "Monday" an entity? "The White House" — building or administration?

#3 Helpfulness

"Rate 1–5 where 1 = not helpful, 5 = very helpful. Consider accuracy, completeness, and clarity."

What do 2, 3, 4 mean? Accurate but incomplete — what score? Clear but wrong?

LLM Demo: Same Prompt, Three Models

Try this at home (or right now on your laptop)

Prompt: "Is the following text biased? Answer yes or no."
Text: "Studies show women are underrepresented in STEM fields."

What you'll get:

- **Model A:** "No" — interprets "biased" as *factually wrong*
- **Model B:** "Yes" — interprets "biased" as *one-sided framing*
- **Model C:** "It depends" — hedges due to safety training

None are wrong. The prompt didn't define "biased."

LLMs make task ambiguity *visible* in seconds — something that takes weeks to discover with human annotators.

LLM Disagreement = Your Task Is Vague

Use this as a cheap diagnostic:

1. Write your annotation prompt
2. Send it to 2–3 LLMs with 10 tricky examples
3. Where models **agree**: your task is probably clear (or trivially easy)
4. Where models **disagree systematically**: you have an ambiguity to resolve
5. Where models **disagree randomly**: your task may not be coherent

Do Not

Treat one model's output as ground truth. The point is to diagnose your *task*, not to get free labels.

30 minutes of this replaces weeks of discovering the same problems through human pilot annotation.

LLMs and Schema Design

LLMs have different schema constraints than humans:

- **Label count:** 5–8 well-defined labels in a single prompt is the sweet spot
  20+ labels $\rightarrow$ instruction-following accuracy drops sharply
- **Label names:** "Personal Attack" $>$ "T2a" $>$ "Category 7"
  LLMs need descriptive names; they never had your training session
- **Overlapping labels:** {Hate, Offensive, Rude} $\rightarrow$ inconsistent output
  LLMs guess silently where humans would ask a question
- **Ordinal scales:** "Rate 1–5" without anchors $\rightarrow$ central tendency
  "3 = adequate but with notable flaws" is far better than just "3"
- **Few-shot cost:** $\geq$2 examples per label $\times$ 20 labels = 4,000+ prompt tokens
  Schema size is a prompt-engineering constraint

If you plan to use LLM annotation: design for LLM constraints from the start. Retrofitting rarely works.

LLM Stress-Testing: The Protocol

Before human pilots, test your guidelines with LLMs:

1. Paste your complete guidelines into a prompt
2. Add 10–15 tricky examples (boundary cases, not easy ones)
3. Prompt: "Label this and explain your reasoning step by step"
4. Send the *same* examples to 2–3 different LLMs
5. Compare the *reasoning*, not just the labels

Why this works

LLMs fill in gaps silently — just like annotators do. But LLMs generate a chain of reasoning you can inspect. Where humans struggle quietly, LLMs expose the problem explicitly.

What LLM Stress-Testing Reveals

Four diagnostic patterns:

1. **LLMs disagree with each other** $\rightarrow$ your guideline is *ambiguous* on that case
   Each model picked a different valid interpretation
2. **LLM invents criteria** not in your guidelines $\rightarrow$ your guidelines are *incomplete*
   The model had to fill a gap you didn't address
3. **LLM restates the definition** instead of applying it $\rightarrow$ *circular* definition
   "This is offensive because it could be offensive to readers" = no operational content
4. **LLM cites two of your rules** that give opposite answers $\rightarrow$ *contradiction*
   Your rules conflict and you didn't specify priority

Each pattern maps to a specific fix: clarify the ambiguity, add the missing rule, rewrite with observable criteria, or add priority ordering.

LLM Compliance $\neq$ Clarity

Dangerous trap: "The LLM followed my guidelines, so they must be clear."

Compliance

- LLM produces *an* answer
- Tests: are guidelines parseable?
- LLMs comply with almost anything
- Vague guidelines $\rightarrow$ silent interpretation

Clarity

- Multiple LLMs produce the *same* answer
- Tests: are guidelines unambiguous?
- Requires **disagreement analysis**
- Clear guidelines $\rightarrow$ convergent output

Rule

Use **disagreement between models** as your diagnostic, not compliance from one model.

Key Takeaways

1. **Prediction target $\neq$ label.** The label is always a proxy. Reformulate until observable.
2. **Don't annotate until you've earned the right to.** Two experts, ten examples, agreement.
3. **Schemas are inductive bias.** Every label set encodes assumptions about what matters.
4. **Design for collapsibility.** Fine $\rightarrow$ coarse is easy. Coarse $\rightarrow$ fine is impossible.
5. **"Other" is a diagnostic, not a solution.** A large "Other" means schema revision.
6. **Operationalize, don't intuit.** If two strangers can't follow your guidelines, rewrite.
7. **Decision procedures tell annotators how, not just what.**
8. **Design from the edge cases in.** Hard cases first; easy cases follow.
9. **LLMs are cheap diagnostic tools.** Use disagreement, not compliance.

Questions?

Office Hours: Wednesdays 1-3pm, Volen 109

jinzhao@brandeis.edu