COSI-230B: Natural Language Annotation for Machine Learning

Lecture 20: Synthetic Annotation and Self-Generated Supervision

Jin Zhao

Brandeis University

April 20, 2026

Today's Agenda

Part I: Foundations & Motivation

- Why synthetic annotation is tempting
- Self-Instruct pipeline deep dive
- Alpaca: cost, quality, and lessons

Part II: Methods & Variants

- WizardLM evolution strategies
- Toolformer: self-supervised tool-use
- UltraChat: scale vs. quality
- LIMA: the counter-argument

Part III: Risks & Theory

- Core risks taxonomy
- Model collapse: theory & evidence
- Provenance tracking in practice

Part IV: Practice & Discussion

- Live demos (noise propagation + diversity collapse)
- Mitigations
- In-class design activity (15 min)
- Wrap-up and next class

Goal: Understand when "the model becomes the annotator"—and what can go wrong.

Why Synthetic Annotation Is Tempting

Advantages:

- **Scale:** Generate millions of examples
- **Cost:** Orders of magnitude cheaper than human annotation
- **Speed:** Hours instead of months
- **Consistency:** No rater fatigue or drift
- **Coverage:** Generate for rare tasks/languages

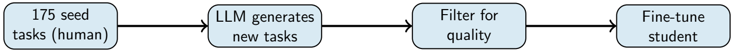
But:

- Scales *mistakes* as well as supply
- Introduces systematic biases from teacher model
- Can cause distribution drift and degeneration
- Provenance tracking becomes critical
- Quality ceiling bounded by teacher capability

"Synthetic annotation scales supply, but it also scales your mistakes. If you don't validate and mix carefully, you train the model on its own blind spots."

Self-Instruct: Generate → Filter → Fine-Tune

Wang et al. (2023): Bootstrap instruction data from a model itself.



Where artifacts enter:

- **Seed bias:** 175 seeds shape the entire distribution
- **Self-confirmation:** Model generates data it already "knows"
- **Filtering trade-off:** Strict filtering → less diversity; loose filtering → more errors
- **Style transfer:** Student inherits teacher's writing style, errors, and blind spots

Wang et al. (2023). Self-Instruct: Aligning Language Models with Self-Generated Instructions. ACL.
`https://aclanthology.org/2023.acl-long.754/`

Self-Instruct Deep Dive: Prompt Template & Filtering

Generation prompt template (simplified):

```
Come up with a series of tasks:

Task 1: {seed_task_1}
Task 2: {seed_task_2}
...
Task 8: {seed_task_8}

Task 9:
```

Filtering criteria applied:

1. ROUGE-L overlap < 0.7 with existing tasks (novelty)
2. Exclude tasks starting with "sorry," "image," "graph" (infeasible)
3. Instruction length: 3–150 words
4. Discard exact & near-duplicate instructions
5. Keyword blacklist for unsafe content

Self-Instruct Deep Dive: Prompt Template & Filtering (cont.)

Dataset statistics:

- 175 human-written seed tasks
- 82K generated instructions (pre-filter)
- 52K instructions retained (post-filter)
- $\sim$37% rejection rate
- Classification vs. generation split: $\sim$26% / 74%
- Average instruction length: 15.9 tokens

Key observation: The 8-shot in-context examples are randomly sampled from the growing pool, creating a snowball effect where early outputs shape later generations.

Alpaca: Semi-Synthetic Instruction Tuning

Taori et al. (2023): 52K instruction pairs from GPT-3.5 $\rightarrow$ fine-tune LLaMA-7B.

- Cost: $<$\$500 for the entire dataset
- Result: Surprisingly competitive with much larger models
- **But:** No human verification of generated outputs

Annotation artifact analysis:

- Outputs reflect GPT-3.5's style (verbose, hedging, US-centric)
- Factual errors in generated data propagate to student
- Template artifacts from the generation prompt persist
- Success masks quality concerns—cheap $\neq$ artifact-free

Taori et al. (2023). Alpaca: A Strong, Replicable Instruction-Following Model. Stanford CRFM.
`https://crfm.stanford.edu/2023/03/13/alpaca.html`

Alpaca: Cost Analysis & Quality Assessment

Cost breakdown ($500 total):

- 52K examples × ∼250 tokens/example
- GPT-3.5-turbo API cost: ∼$0.002 per 1K tokens
- Total API cost: ∼$100–$500
- Compute for LLaMA-7B fine-tuning: ∼$100 (cloud GPU)

Compare to human annotation:

- 52K human-written instructions: ∼$150K–$500K
- **100–1000× cheaper via synthetic**
- Time: days vs. months

Quality assessment:

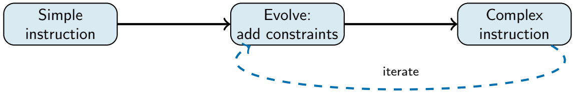
- Human eval: Alpaca-7B ≈ text-davinci-003 on simple tasks
- **Failure modes:**
  - Factual errors: ∼15–20% of outputs contain hallucinations
  - Reasoning: degrades sharply on multi-step problems
  - Safety: inherits GPT-3.5 refusal patterns verbatim

Vs. human-annotated alternatives:

- Dolly (Databricks): 15K human-written, competitive quality
- OpenAssistant: 161K human turns, higher diversity

WizardLM / Evol-Instruct: Complexity Evolution

Xu et al. (2024): Iteratively evolve instructions to increase complexity.



Evolution strategies: Add constraints, deepen reasoning, broaden scope, concretize, increase steps.

Risk: Evolution can create instructions that are complex but *unnatural*—artificial difficulty may not transfer to real-world tasks.

Xu et al. (2024). WizardLM: Empowering Large Pre-Trained Language Models. arXiv:2304.12244.

`https://arxiv.org/abs/2304.12244`

WizardLM Evolution Strategies: Examples

Five evolution operators with concrete before/after:

Strategy	**Simple (before)**	**Evolved (after)**
Add constraints	"Write a poem about spring."	"Write a poem about spring using only words with $\leq$ 5 letters, in haiku form."
Deepen	"Explain photosynthesis."	"Explain photosynthesis at the molecular level, including the Calvin cycle and electron transport chain."
Concretize	"Write a sorting algorithm."	"Write a quicksort in Python that handles duplicate keys and runs in-place with $O(\log n)$ stack space."

WizardLM Evolution Strategies: Examples (cont.)

Strategy	**Simple (before)**	**Evolved (after)**
Broaden	"Translate this sentence to French."	"Translate this sentence to French, Spanish, and Mandarin, and note key grammatical differences."
Increase steps	"Summarize this article."	"Summarize this article, then critique the author's argument, propose a counter-argument, and suggest follow-up research."

Caveat: After 3+ rounds of evolution, $\sim$20% of instructions become internally contradictory or infeasible.

Toolformer: Self-Supervised Tool-Use Annotations

Schick et al. (2023): Model annotates *when to call APIs* (calculator, search, etc.).

Key Insight

"Synthetic annotation" beyond labels: Toolformer creates annotations about **tool invocation** (when/how), shifting the unit of annotation to a *policy decision*.

Pipeline:

1. Sample API call positions in text
2. Execute API calls and check if they improve perplexity
3. Keep only "useful" tool annotations
4. Fine-tune model on annotated data

Annotation quality: Measured by downstream utility, not human agreement—a fundamentally different validation paradigm.

Schick et al. (2023). Toolformer: Language Models Can Teach Themselves to Use Tools. arXiv:2302.04761.

`https://arxiv.org/abs/2302.04761`

UltraChat: Large Synthetic Dialogue Generation

Ding et al. (2023): 1.5M multi-turn dialogues generated by two ChatGPT instances.

- **Scale:** Massive synthetic dialogue corpus
- **Curation:** UltraChat 200k filtered variant demonstrates importance of quality filtering
- **Design:** Systematic topic coverage across categories

Annotation concerns:

- Two instances of the same model → homogeneous dialogue patterns
- Conversational artifacts: unnaturally smooth turn-taking
- No human validation of factual claims in dialogues
- Filtered version (200k) substantially outperforms full set—**more is not always better**

Ding et al. (2023). Enhancing Chat Language Models by Scaling High-quality Instructional Conversations. EMNLP.
`https://aclanthology.org/2023.emnlp-main.183/`

UltraChat: Why Filtering Mattered

UltraChat 1.5M (unfiltered):

- 1.47M multi-turn conversations
- 30 topics across 3 sectors (world knowledge, creative writing, skill-based)
- Average turns per conversation: 4.6
- Contains repetitive, low-quality, and contradictory dialogues

Filtering pipeline (for 200K):

1. Remove conversations < 3 turns
2. Filter by GPT-4 quality score ($\geq 4/5$)
3. Deduplicate by embedding similarity (> 0.95)
4. Balance topic distribution

Performance comparison (MT-Bench):

Training Data	Score
UltraChat 1.5M (full)	5.48
UltraChat 200K (filtered)	**6.22**
Improvement	*+13.5%*

Key insight: 87% of the data was removed, yet the model *improved*. The removed data actively *hurt* performance through:

- Contradictory supervision signals
- Style homogenization
- Factual noise drowning out signal

LIMA: Less Is More for Alignment (The Counter-Argument)

Zhou et al. (2023): 1,000 carefully curated examples can beat 50K+ synthetic.

Setup:

- 1,000 training examples: 750 from forums (Stack Exchange, wikiHow, Reddit), 250 hand-written by the authors
- No RLHF, no PPO, no reward model
- Standard supervised fine-tuning of LLaMA-65B

Results (human eval):

- LIMA vs. Alpaca (52K synthetic): LIMA preferred 54% of the time
- LIMA vs. Davinci-003: LIMA preferred 43%
- Competitive with GPT-4 on 19% of prompts

Zhou et al. (2023). LIMA: Less Is More for Alignment. NeurIPS. `https://arxiv.org/abs/2305.11206`

LIMA: Less Is More for Alignment (cont.)

The "Superficial Alignment Hypothesis":

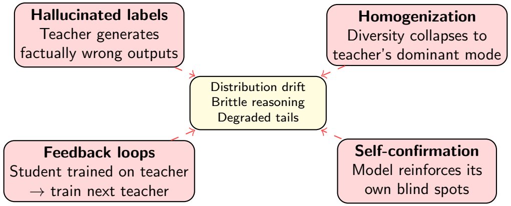
Core Claim

A model's knowledge and capabilities are learned almost entirely during pretraining. Alignment tuning teaches *style and format*, not knowledge.

Implication for synthetic data:

- If alignment is "superficial," then curating 1K high-quality examples may beat generating 50K mediocre ones
- Quality $\gg$ quantity for instruction tuning

Core Risks of Synthetic Annotation


Hallucinated labels
Teacher generates factually wrong outputs
Homogenization
Diversity collapses to teacher's dominant mode
Distribution drift
Brittle reasoning
Degraded tails
Feedback loops
Student trained on teacher → train next teacher
Self-confirmation
Model reinforces its own blind spots

All four risks share a root cause: The teacher model's errors and biases are treated as ground truth, then amplified through training.

Model Collapse: Why Recursive Training Is Dangerous

Shumailov et al. (2023, 2024): Training on recursively generated data causes degenerative loss of distributional tails.

What happens:

1. Train Model A on human data
2. Generate synthetic data with Model A
3. Train Model B on synthetic data
4. Repeat: each generation loses tail phenomena
5. Eventually: mode collapse, repetitive outputs

Why it matters:

- As web fills with LLM text, training data becomes recursive
- Rare phenomena erode first
- Diversity loss is *irreversible* without human data injection
- Published in *Nature* (2024)

Implication

Synthetic data requires intentional mixing with human-origin data and continuous audits.

Shumailov et al. (2023). arXiv:2305.17493; Shumailov et al. (2024). Nature. `https://arxiv.org/abs/2305.17493`

Model Collapse: Theoretical Framework

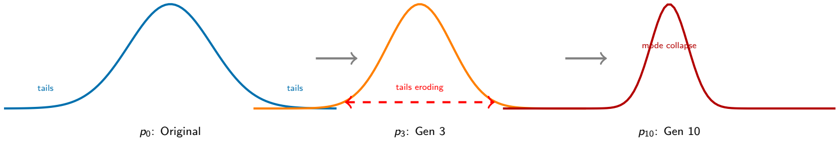
Shumailov et al.'s formal analysis:

Let p_0 be the true data distribution and p_n the distribution learned at generation n:

$$p_n = \mathcal{T}(\hat{p}_{n-1}) \quad \text{where } \hat{p}_{n-1} \text{ is estimated from finite samples of } p_{n-1}$$

Two sources of error compound at each generation:

- **Statistical error:** Finite sampling causes underrepresentation of low-density regions
- **Functional approximation error:** Model class cannot perfectly represent p_{n-1}



Result: Distributional tails vanish first. After n generations, rare events (minority dialects, unusual phrasings, edge-case reasoning) are systematically eliminated.

Key theorem: Even with infinite model capacity, *finite sampling* alone guarantees tail loss $\rightarrow$

Provenance Tracking in Practice

Why: If you can't distinguish synthetic from human data, you can't audit, filter, or mix properly.

Concrete example: Data card for a synthetic instruction dataset

Field	**Example Value**
Dataset name	`synth-instruct-medical-v2`
Generation date	2026-03-01
Teacher model	GPT-4-turbo (version 2025-12-01)
Generation prompt ID	`prompt-med-qa-v2.3` (linked to prompt registry)
Temperature / top-p	0.7 / 0.95
Num. raw generated	45,000
Num. post-filter	28,300 (63% retention)
Filtering criteria	Length 20–500 tokens; toxicity < 0.1; dedup > 0.85 cosine

Provenance Tracking in Practice (cont.)

Field	Example Value
Human-origin mixing	5,000 human examples (15% of final set)
Known limitations	Underrepresents pediatric cases; US-centric drug names
License / use restrictions	Internal research only; not for clinical deployment

Best practice: Every synthetic example should carry a metadata tag: {`‘‘origin’’: ‘‘synthetic’’, ‘‘teacher’’: ‘‘gpt-4-turbo’’, ‘‘prompt_id’’: ‘‘...’’, ‘‘filter_passed’’: [...]`}

Live Demo 1: Teacher Noise → Student Degradation

```python
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

rng = np.random.default_rng(0)
n = 2000
X = rng.normal(size=(n, 6))
w = rng.normal(size=6)
y_true = (X @ w > 0).astype(int)

def noisy_teacher(y, p):  # teacher makes errors
    return y ^ (rng.random(len(y)) < p)

y_teacher = noisy_teacher(y_true, 0.25)  # synthetic labels

# train student on synthetic labels, evaluate on true labels
clf = LogisticRegression(max_iter=2000).fit(X[:1500], y_teacher[:1500])
print("Student␣accuracy␣vs␣true:",
      accuracy_score(y_true[1500:], clf.predict(X[1500:])))
```

Takeaway: 25% teacher error rate propagates to student—the student can never exceed the

Live Demo 2: Measuring Diversity Collapse

```python
import collections, math

def type_token_ratio(texts):
    """Measure␣vocabulary␣diversity␣across␣a␣corpus."""
    all_tokens = "␣".join(texts).lower().split()
    return len(set(all_tokens)) / len(all_tokens)

def entropy(texts):
    """Shannon␣entropy␣of␣unigram␣distribution."""
    all_tokens = "␣".join(texts).lower().split()
    freq = collections.Counter(all_tokens)
    total = sum(freq.values())
    return -sum((c/total)*math.log2(c/total) for c in freq.values())

# Simulate: human vs synthetic corpora
human_texts = [...]       # diverse human-written responses
gen0_texts  = [...]       # first-generation synthetic
gen3_texts  = [...]       # third-generation (recursive)

for name, corpus in [("Human", human_texts),
                      ("Gen-0", gen0_texts),
                      ("Gen-3", gen3_texts)]:
    print(f"{name}:␣TTR={type_token_ratio(corpus):.3f},␣"
          f"H={entropy(corpus):.2f}␣bits")
```

Expected: TTR and entropy *decrease* with each synthetic generation, confirming vocabulary and style homogenization.

Mitigations for Synthetic Annotation Risks

1. **Filtering pipelines:** Remove invalid, duplicated, or low-quality synthetic items
   - Check for diversity, factuality, safety
   - Use multiple filtering criteria (not just perplexity)
2. **Human mixing:** Intentionally blend human-origin data with synthetic data
3. **Provenance tracking:** Record what portion is synthetic, from which generator, under which prompts and filters
4. **Evaluation guards:** Continuous evaluation on held-out human-labeled data; monitor for distribution drift
5. **Diversity audits:** Check that synthetic data covers the same distributional tails as human data

Key Definitions

Synthetic supervision: Labels/outputs generated by models. | **Filtering:** Quality gates for synthetic items.
Model collapse: Distributional degeneration from recursive training. | **Provenance:** Tracking

Discussion Questions

Discussion:

1. What must be true about your filtering pipeline to trust synthetic data (diversity, factuality, safety)?
2. Is model collapse inevitable if the web becomes saturated with generated text? What data should we prioritize collecting?
3. When does synthetic data improve generalization vs. just teach a model to imitate its teacher?
4. LIMA showed 1K curated examples can rival 52K synthetic. Does this mean synthetic data is unnecessary, or that it serves a different purpose?
5. How should provenance metadata change the way we build training pipelines in industry?

In-Class Activity: Design a Synthetic Data Pipeline (15 min)

Instructions: In groups of 3–4, design a synthetic data pipeline for **one** of the following tasks:

Option A:
Medical QA
(patient questions → doctor-quality answers)

Option B:
Code Review
(PR diffs → constructive feedback)

Option C:
Multilingual Sentiment Analysis
(5 languages)

Your plan must specify:

1. **Teacher model:** Which model? Why?
2. **Generation prompt:** Write the actual prompt template (2–3 sentences)
3. **Filtering criteria:** At least 3 concrete filters with thresholds
4. **Human mixing ratio:** What % human data? How sourced?
5. **Provenance documentation:** What metadata fields would you record?
6. **Top 3 risks:** Specific to your chosen task, with mitigations

Deliverable: One-page plan per group. Be ready to present your top risk to the class.

Activity Debrief: Common Patterns

As groups present, let's track common patterns:

Teacher model choices:

- Why did most groups pick the largest available model?
- Cost vs. quality trade-off
- Domain-specific fine-tuned vs. general-purpose

Filtering criteria:

- What filters were universal across groups?
- What was task-specific?
- Did anyone propose human-in-the-loop filtering?

Human mixing ratio:

- Range across groups? (Typical: 10–30%)
- LIMA suggests even 5% high-quality human data helps
- Where does human data come from for your task?

Risks identified:

- Medical: hallucinated drug interactions
- Code: plausible but buggy suggestions
- Multilingual: English-centric transfer artifacts

Key Takeaways

1. Synthetic annotation is powerful but requires rigorous quality controls
2. Model collapse is a real, documented phenomenon—mix with human data
3. Filtering is critical: UltraChat 200k $>$ UltraChat 1.5M
4. LIMA shows quality can trump quantity—1K curated $\geq$ 52K synthetic
5. Provenance tracking is essential—treat synthetic data as annotation metadata
6. The teacher's ceiling is the student's ceiling (without additional human signal)
7. Diversity collapses recursively: audit TTR, entropy, and tail coverage

Readings

Required readings for next class:

- Wang et al. (2023). *Self-Instruct: Aligning Language Models with Self-Generated Instructions.* ACL 2023.
- Shumailov et al. (2024). *AI Models Collapse When Trained on Recursively Generated Data.* Nature.
- Zhou et al. (2023). *LIMA: Less Is More for Alignment.* NeurIPS 2023.

Optional (recommended):

- Xu et al. (2024). *WizardLM: Empowering Large Pre-Trained Language Models.* arXiv:2304.12244.
- Ding et al. (2023). *Enhancing Chat Language Models by Scaling High-quality Instructional Conversations.* EMNLP.
- Taori et al. (2023). *Alpaca: A Strong, Replicable Instruction-Following Model.* Stanford CRFM.