

Where We Are, and Where We Are Going

A Positioning Note

James Pustejovsky

Brandeis University
Department of Computer Science

CS 135: Computational Semantics

Fall 2026

A Fair Question

We have spent six sessions on lambda calculus, types, models, quantifiers, monotonicity, and possible worlds.

Meanwhile, a large language model answers all of these questions.

So: why are we doing this?

Today's answer, in one line:

Because the structure we have been building is what those systems are spending enormous resources to approximate — and it is not clear they have it.

What We Have Built

Session	Machinery	Guarantees
1–2	typed lambda calculus	composition is exact and total
3	models, satisfaction	truth is defined, and computable
4	the fragment	the whole thing runs
5	generalized quantifiers	scope, and what determiners can mean
5A	monotonicity	inference falls out of the denotations
6	worlds, context change	belief, modality, discourse

The word to notice is guarantees. These are not tendencies or high-probability behaviors. Given the lexicon, the entailments follow — always, for every sentence, including ones nobody has written down.

That is an unusual property for a system to have, and worth knowing the price of.

Monotonicity, Concretely

Today's material is the sharpest example.

No student who owns a car commutes by bus.

From *no* being $\downarrow\text{MON}\downarrow$, it follows — with no training, no data, no examples — that this entails:

- *No first-year student who owns a car commutes by bus*
- *No student who owns a sports car commutes by school bus*
- and every other narrowing of either argument, without bound

One lexical fact licenses an infinite family of inferences. The generalization is not learned from examples; it is a consequence of what *no* means.

A system that learns these one at a time is doing something fundamentally different — and paying for it.

How a Transformer Does This Instead

Attention computes, for each pair of positions, a learned compatibility:

$$\text{Attn}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right) V$$

What it is	a learned, soft, weighted mixture over positions
-------------------	--

What it is not	a typed operation with a determinate result
-----------------------	---

Relations are encoded in the learned matrices W_Q, W_K, W_V — one set per head, per layer. Whatever structure the model has is distributed across them.

This is genuinely powerful, and it is how every deployed system works. The question is not whether it works. The question is what it costs, and what it guarantees.

Three Things Attention Does Not Give You

- ❶ **Typing.** Nothing prevents combining two representations that should not compose. Every pair of positions gets a score; there is no undefined.
- ❷ **An order on meanings.** Monotonicity needs $\subseteq$. Dense vectors have no partial order — so nothing in the representation makes *jog* $\subseteq$ *move*.
- ❸ **Exactness.** Softmax returns a distribution. There is no step at which a derivation is *complete* and a result *determined*.

These are not bugs. They are the price of a representation that is continuous, learnable, and differentiable — which is exactly what buys similarity, robustness, and coverage. But they are the three properties Sessions 1–6 spent their time establishing.

Can a Distributional Model Get Monotonicity?

People have tried. It helps to see that **two different things** are being asked for.

	What is needed	Example
Lexical	an order on concepts	jog $\sqsubseteq$ move
Functional	how an operator <i>transforms</i> that order	<i>no</i> reverses it

These are separable problems, and the literature has made much more progress on the first.

But today's session was entirely about the second. Knowing that jogging is a way of moving tells you nothing about whether *No student jogs* follows from *No student moves*.

Approach 1: Polarity as an Input Feature

The most direct route: **mark each token with its polarity** and hand that to the model.

A determiner contributes a **two-bit signature** — one bit per argument:

Determiner	Restrictor	Scope
<i>every</i>	—	+
<i>some</i>	+	+
<i>no</i>	—	—
<i>not every</i>	+	—

Propagate those bits over the spans and you get a token-level annotation:

No student[—] who owns[—] a car[—] commutes[—] by bus[—]
Every student[—] who owns[—] a car[—] commutes⁺ by bus⁺

Then add a learned polarity embedding: $\mathbf{x}_i = \mathbf{e}_i + \mathbf{p}_{\pm}$. The model now has the information.

Does the Markup Work?

Yes — and that is exactly the problem.

Systems given polarity features improve substantially on downward-entailing inference. (Hu, Chen & Moss and related work on monotonicity marking from CCG or UD parses.)

But ask where the bits came from. **They were computed by a symbolic procedure** — the polarity calculus from Session 5A, run over a parse.

So the pipeline is: **parse** → **determiner lookup** → **polarity composition** → **neural model**.
The theory does the monotonicity; the network does everything else.

Three costs follow.

- You need a **parser**, and errors propagate.
- You need a **lexicon** of determiner signatures — written by hand.
- The model has not learned monotonicity. It has been told.

Approach 2: Put the Order Into the Geometry

Rather than annotating, change the representation so that inclusion *means* something.

Scheme	Entailment is	Idea
Order embeddings	$\mathbf{u} \leq \mathbf{v}$ componentwise	train in the positive orthant
Box embeddings	box containment	concepts as axis-aligned regions
Hyperbolic (Poincaré)	position in the hierarchy	trees embed with low distortion
Density embeddings	KL divergence	concepts as Gaussians

These genuinely solve the lexical problem. They give a partial order on concepts, learned from data, in a continuous space — which is precisely what plain dense embeddings lack.

Note what they have in common: each *chooses a geometry* in which containment is expressible. None of them gets an order for free from the vectors.

The Gap That Remains

Order embeddings tell you $\text{jog} \sqsubseteq \text{move}$.

They do not tell you what *no* does to $\sqsubseteq$.

Problem	Status	Solved by
Lexical order	largely solved	order / box / hyperbolic embeddings
Functional monotonicity	open	no standard account

What would be needed: a determiner represented as an operator that is *provably* order-preserving or order-reversing in each argument — a **monotone map between ordered spaces**, not merely a learned matrix.

Monotone networks exist: constrain weights to be non-negative and the function is increasing. But *decreasing* needs a sign flip, and choosing when to flip is a discrete decision the continuous machinery does not naturally make.

Why This Is a Good Example for This Course

- 1 **The property is precisely stated.** Not “does it understand negation” but: *is this operator order-reversing in its second argument?* That is checkable.
- 2 **The symbolic account is complete.** Session 5A derived every case from the denotations, in a few lines each.
- 3 **The distributional account is partial, and we can say exactly which part.** Lexical order, yes. Functional behaviour, not yet.
- 4 **And the repair attempts all do the same thing:** choose a richer geometry so that the structure has somewhere to live.

That last point is the whole argument for Parts III and IV. Order embeddings choose a cone; box embeddings choose regions; geometric algebra chooses grades and an invertible product. Each is a bet that the right geometry makes the structure expressible.

The Cost, in Resources

What does it take to approximate structure statistically?

Resource	Symbolic	Learned
Lexicon	written once	inferred from billions of tokens
A new entailment	derived	must appear, or be generalized to
Inference at test time	one substitution	a full forward pass
Chained reasoning	free, by transitivity	often needs explicit steps in context
Guarantee	by construction	empirical, per-input

Note the fourth row. A monotonicity chain of ten steps is ten substitutions and costs nothing extra. In a generative system it becomes ten reasoning steps, emitted as tokens, each an opportunity for error.

Chain-of-thought is, among other things, a way of buying back compositionality at inference time — by the token.

The Empirical Question

Do learned models actually acquire the structure?

Benchmark	Tests	Design
SCAN	compositional generalization	train on parts, test on combinations
COGS	sentence $\rightarrow$ logical form	novel structures at test time
CFQ	question $\rightarrow$ query	maximum compound divergence
Monotonicity NLI	today's material	DE/UE contexts, systematically varied

The construction is always the same. Train on *jump, walk twice, run left*; test on *jump twice*. A compositional system generalizes for free.

Results have improved a great deal and remain uneven — particularly on downward-entailing contexts, where the correct inference *reverses*.

That is a natural place for a statistical learner to fail: DE contexts are rarer in text, and the surface cue is a single word.

What These Benchmarks Presuppose

To ask whether a model composes, you need a theory of what composing *is*.

Every benchmark on the previous slide is built by someone who:

- wrote a **grammar** to generate the items;
- used a **compositional semantics** to compute the gold answers;
- and chose the train/test split by a **structural** criterion.

Sessions 1–6 are the machinery those people used. You are not learning an alternative to modern NLP; you are learning the standard against which it is measured.

The same holds for evaluation more broadly: an entailment dataset is only as good as the semantics used to label it.

The Honest Position

Two overclaims to avoid, in both directions.

Wrong: “Neural models don’t really understand; symbols are needed.”

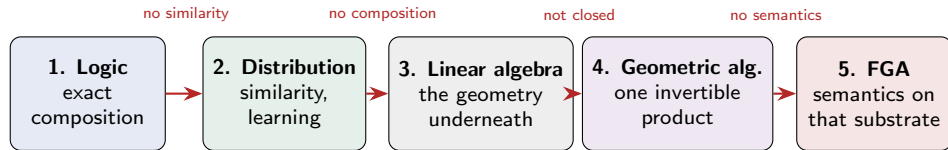
They work, at a scale and robustness no symbolic system approaches.

Also wrong: “Scale solved it; this material is history.”

The guarantees are not there, and the benchmarks that test for them say so.

The defensible position: each tradition has exactly what the other lacks. Logic has structure without similarity; distribution has similarity without structure. Whether one substrate can carry both is an open research question — and it is the question this course is organized around.

The Arc, Restated



We are finishing Theme 1. Everything from Session 1 to today has been building the left-hand box — and establishing, precisely, what is missing from it.

Next week Theme 2 begins, and the red label above the first arrow becomes the subject.

What Happens to Today's Material

Today	In Part II	In Parts III–IV
types	lost	candidate: grade
composition	lossy	candidate: contraction
monotonicity	no order, so gone	open
binding	approximate	candidate: wedge product
worlds, discourse	no counterpart	open

The middle column is why Part II is not the end of the course. Everything we built gets lost on the way into vector space.

The right column is the bet. Geometric algebra has operations with the right formal shape — graded, invertible, order-sensitive. Whether they can carry a semantics is what Sessions 17–24 examine, and what Problem Set 3 asks you to test.

Why You Are Learning Both

If you build systems	you will write evaluation sets, and the gold labels come from a compositional semantics.
If you do research	the open questions in this area are stated in the vocabulary of Sessions 1–6.
If you do neither	you will still need to judge claims about what models understand — and that requires a standard.

The skill this course is actually teaching is **knowing what a representation guarantees**.

That is a question you can ask of a lambda term, a knowledge graph, an embedding, an attention head, or a multivector. **The answer differs, and the difference is what determines what you can safely build on it.**

Two Questions to Carry Forward

① What does this representation guarantee?

Not: what does it usually do. What follows, by construction, for inputs nobody has seen?

② What does it cost to get that guarantee some other way?

In parameters, in tokens, in training data, in inference-time compute — and in confidence that it will hold next time.

Ask both of every framework in the rest of this course — including the one I am proposing in Part IV. Problem Set 3 explicitly asks you what the geometric account *cannot* do, and a well-argued negative result earns full marks.

Back to Work

Today: monotonicity in depth, then modality, intensionality, and dynamics.

Then: λ -DCS — the symbolic tradition at Freebase scale — and semantic types as vectors, which is the bridge.

Next week: Part II. Meaning from data.

Keep the monotonicity material in view when we get to Session 10.

When we show that vector composition loses structure, you will already know one precise thing it loses — and you will be able to say exactly which inferences go with it.

Questions?

Now: monotonicity