EEP 596: LLMs: From Transformers to GPT || Lecture 10

Dr. Karthik Mohan

Univ. of Washington, Seattle

February 4, 2026

Logistics

- MP2 to be released this Friday and due 2 weeks later
- MP2 - Work in teams of 2
- Anything else?

Upcoming Lecture Topics

- Chatbots and their Design - Industry use-cases
- Prompt Engineering Techniques
- Pre-Training, SFT, PEFT, LoRA, DPO (new this year)
- Multi-Modal AI Models
- Agents, Agentic AI (New this year with a new MP3)
- Other topics (time-permitting): Prompt Injection Attacks, LLM Guardrailing

Last Lecture

- Training objectives for BERT
- Moltbook
- AI Safety

Today's Lecture

- Instacart Search Relevance use-case
- Prompt Engineering Best Practices
- Paper Presentations

Deep Learning References

Deep Learning

Great reference for the theory and fundamentals of deep learning: Book by Goodfellow and Bengio et al Bengio et al

Deep Learning History

Embeddings

SBERT and its usefulness SBert Details Instacart Search Relevance

Instacart Auto-Complete

Attention

Illustration of attention mechanism

The LLM Agents Stack

Increasing layers of abstraction: Python → Agent systems

Layer #	Abstraction Layer	What it abstracts away	What you *explicitly* control	Typical usage at that layer	Example libraries /
0	**Python / Math**	Nothing	Everything (loops, math, data structures)	Implement algorithms from scratch	NumPy, SciPy
1	**ML Algorithms**	Optimization logic	Features, loss, training loop	Classical ML models	scikit-learn
2	**Deep Learning Frameworks**	Backprop, GPU ops	Model architecture, training	Neural networks	PyTorch, TensorFlo
3	**Model Architectures**	Layer wiring	Hyperparameters, data	Reusable NN blueprints	CNNs, RNNs
4	**Transformers**	Sequence modeling mechanics	Tokenization, layers, attention	NLP & vision foundation	Transformer encoder/decoder
5	**Foundation Models**	Pretraining at scale	Prompting, fine-tuning	General-purpose intelligence	BERT, GPT-style m
6	**LLMs (APIs / checkpoints)**	Training + inference infra	Prompts, parameters	Text generation & reasoning	GPT-4, LLaMA
7	**Prompting & Templates**	Raw prompt crafting	Prompt structure	Reliable LLM usage	Prompt templates

The LLM Agents Stack

8	**Tool Calling / Functions**	API invocation logic	Tool definitions	LLM ↔ real world	Function calling
9	**RAG (Retrieval)**	Knowledge grounding	Indexing strategy	QA over private data	Vector DBs
10	**Chains**	Call sequencing	Workflow order	Multi-step LLM apps	LangChain
11	**Graphs (State Machines)**	Control flow & retries	State schema	Deterministic agents	LangGraph
12	**Tool Protocols**	Tool interfaces	Security + permissions	Model-agnostic tools	MCP
13	**Single Agents**	Reason–act loop	Goals & tools	Autonomous tasks	ReAct agents
14	**Multi-Agent Systems**	Coordination logic	Roles & policies	Planning & debate	Planner–Executor-
15	**Agentic Platforms**	Orchestration & memory	High-level objectives	Production AI systems	Agent frameworks

Application of Sentence Transformer Embeddings to Instacart Recommendations

Instacart Recommendations (ITEMS)

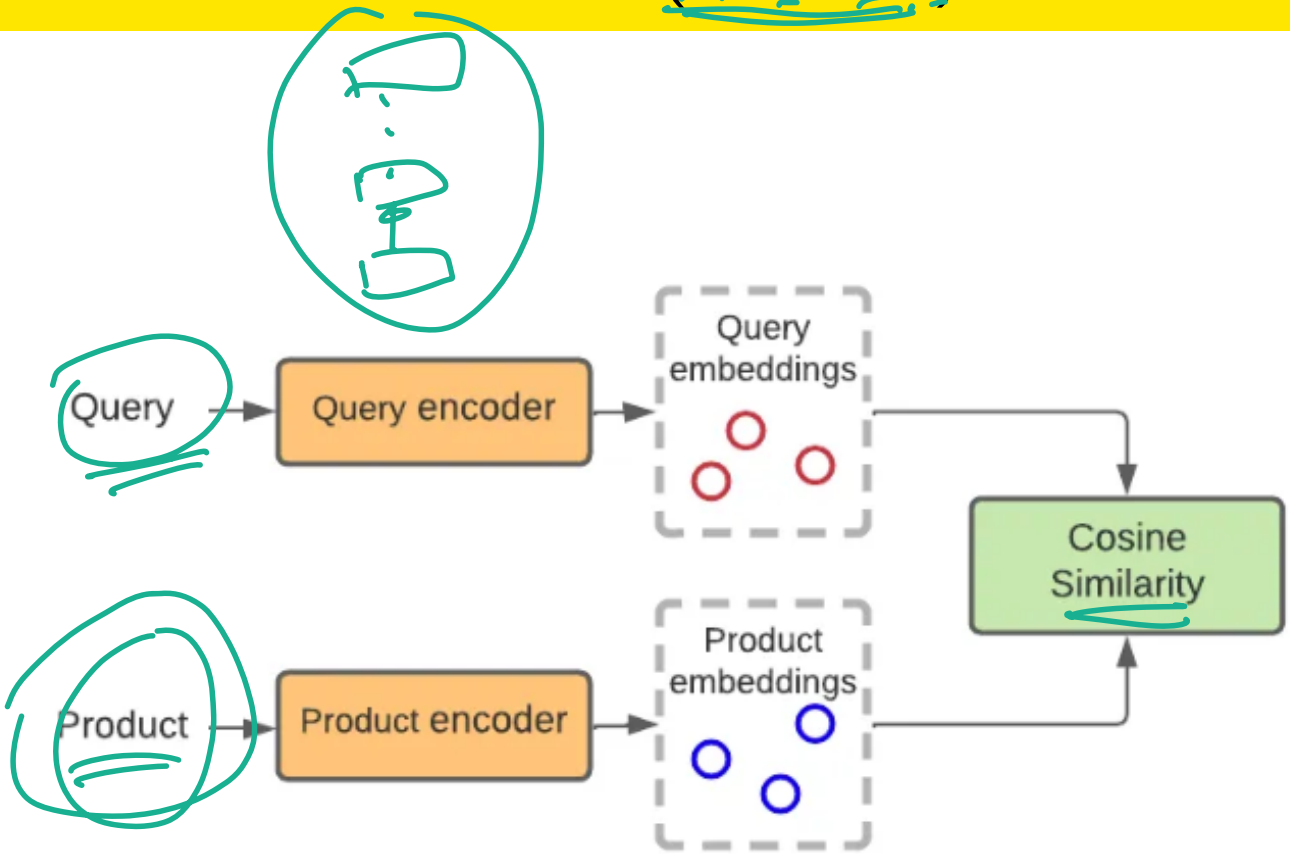


Figure 1. Conceptual diagram of a two-tower model

Two Tower Architecture

Two Towers

Self-explanatory, but there are two towers that represent two distinct objects (e.g. sentence A and sentence B or query and product or customer and product, etc).

Two Tower Architecture

Two Towers

Self-explanatory, but there are two towers that represent two distinct objects (e.g. sentence A and sentence B or query and product or customer and product, etc).

SBERT Two Tower

Is a **Siamese Two Tower**, where the weights and layers of the two towers are *identical*. In the training of a Siamese two-tower, the weights are said to be tied together between the two towers and gradients are computed keeping the tying in place.

Two Tower Architecture

Two Towers

Self-explanatory, but there are two towers that represent two distinct objects (e.g. sentence A and sentence B or query and product or customer and product, etc).

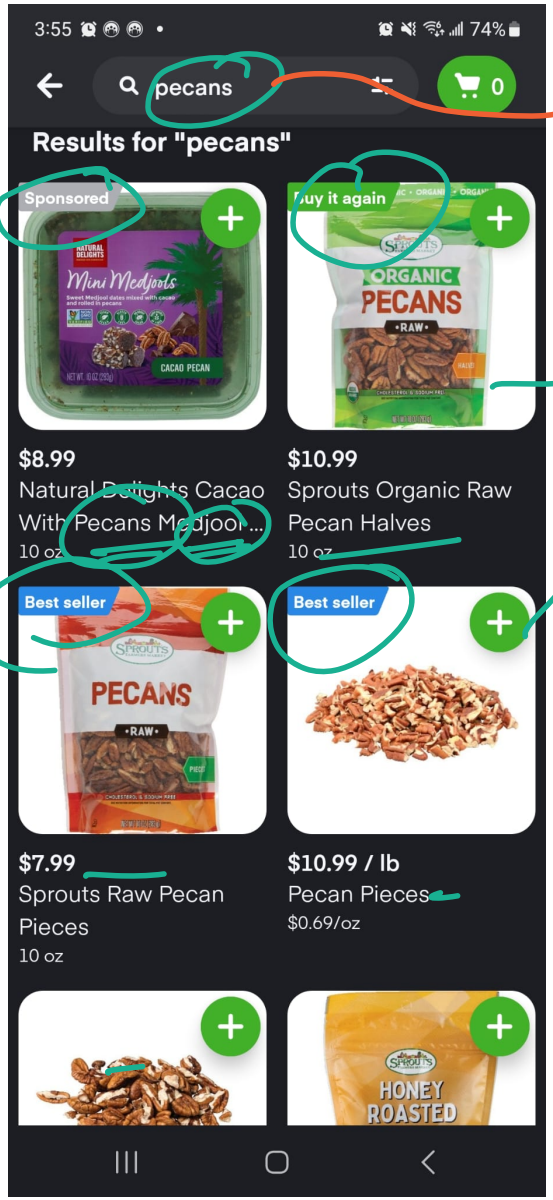
SBERT Two Tower

Is a **Siamese Two Tower**, where the weights and layers of the two towers are *identical*. In the training of a Siamese two-tower, the weights are said to be tied together between the two towers and gradients are computed keeping the tying in place.

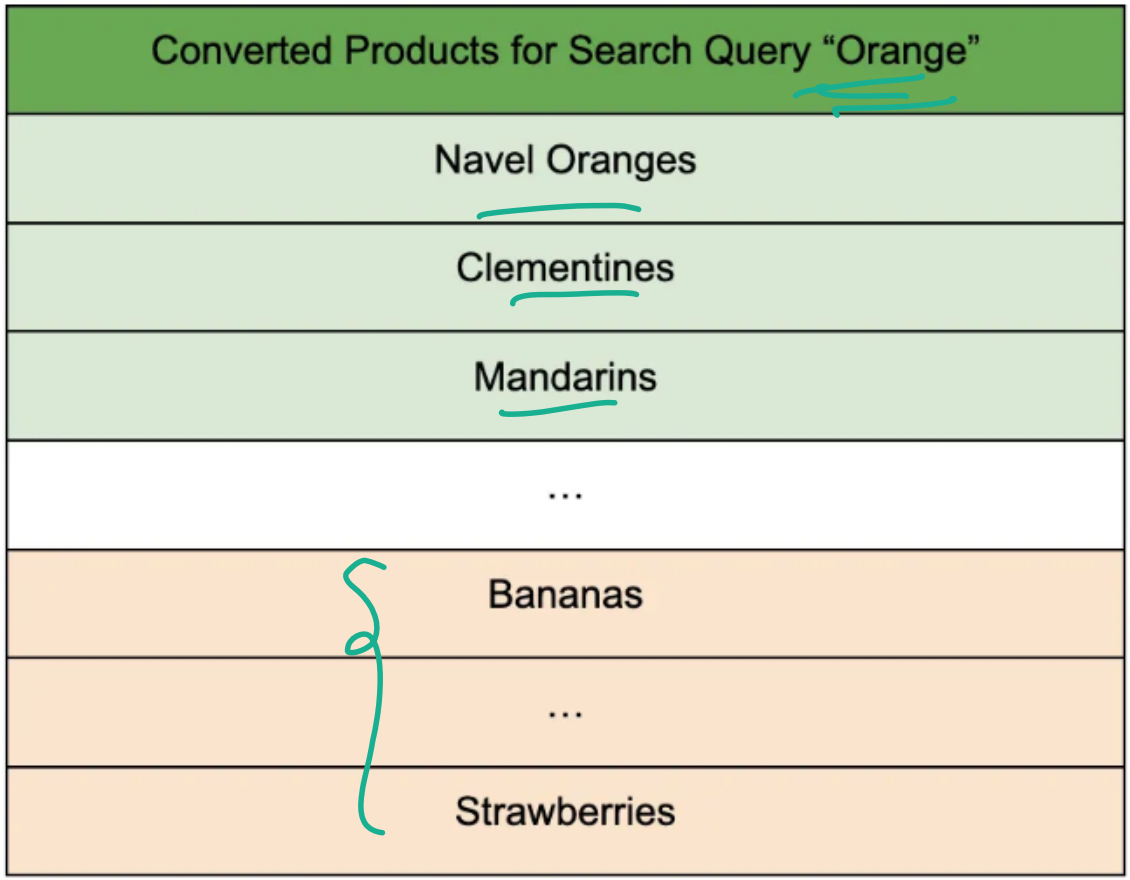
Instacart/Recommendations Two Tower

In this example, the two towers don't refer to the same kind of object (e.g. sentence) but refer to a product and query. Hence the two towers have distinct weights learned from the data.

Positive Examples



High-quality Positive Examples



Negative Examples

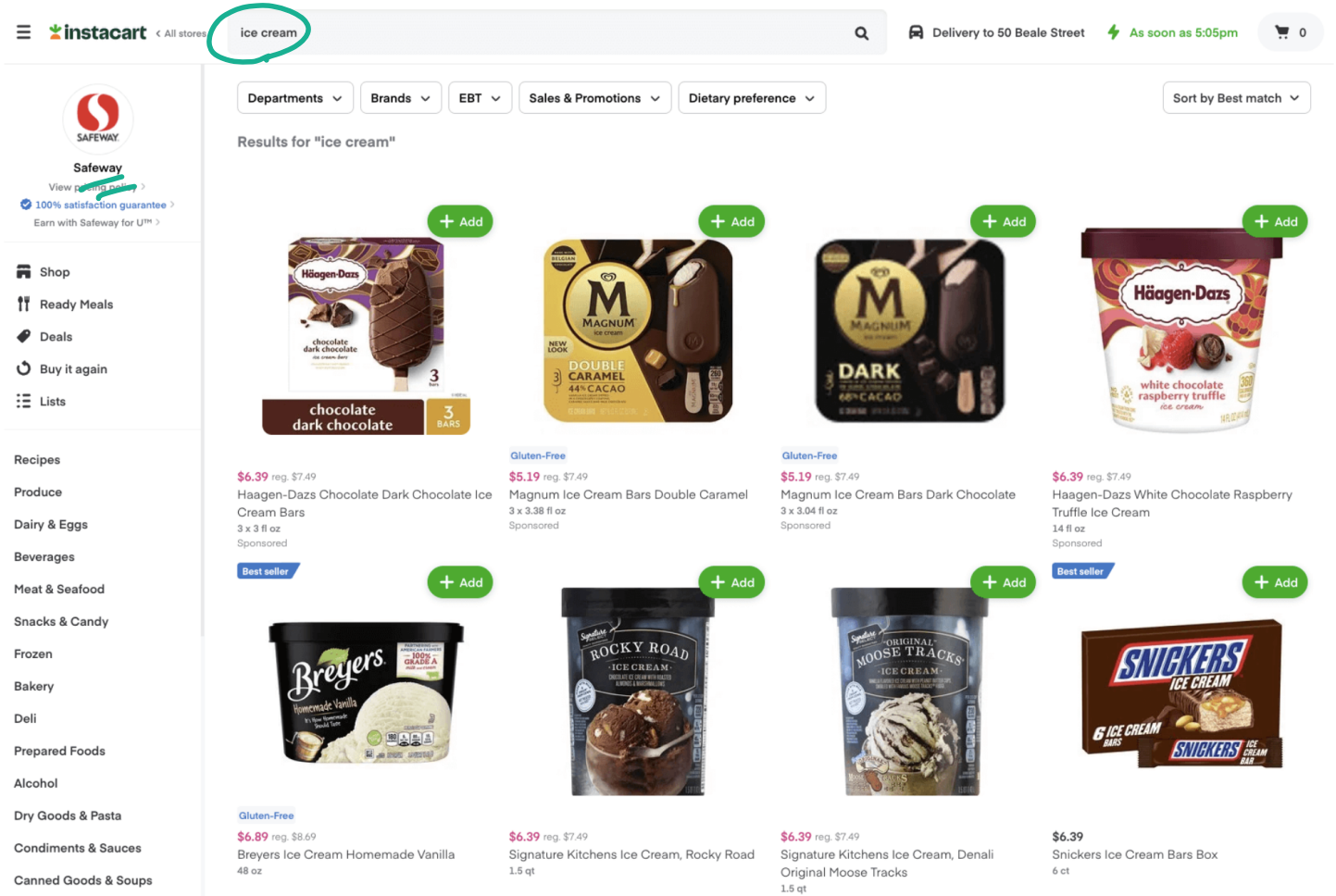


Figure 2. Top products for the query "ice cream" at Safeway. While a customer may have a preference for one of these over the others, all products are highly relevant to the query.

Negative Examples

Figure 3. (Left) In the vanilla implementation of in-batch negative, all off-diagonal negative samples are given the same weight. (Right) In our implementation with self-adversarial re-weighting, harder examples are given more weight (darker color), making the task more challenging for the model.

Negative Examples

Figure 3. (Left) In the vanilla implementation of in-batch negative, all off-diagonal negative samples are given the same weight. (Right) In our implementation with self-adversarial re-weighting, harder examples are given more weight (darker color), making the task more challenging for the model.

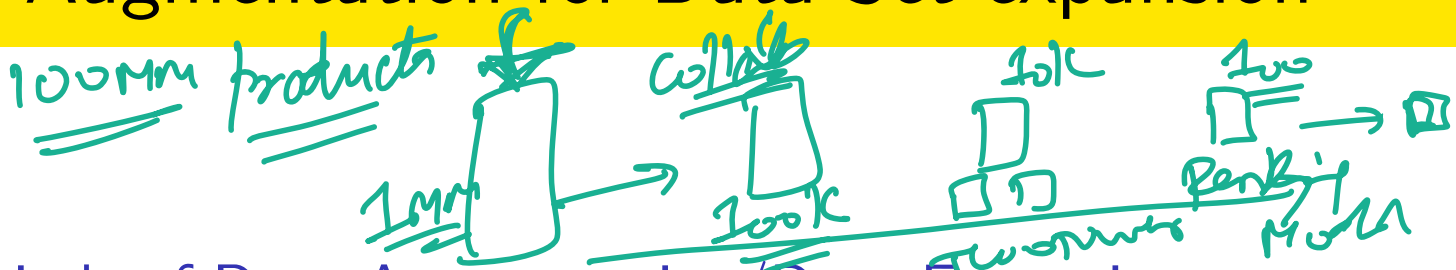
Self-adverserial data annotation

Query Product

Easy Negative examples: Tortilla $\rightarrow$ Coffee mug

Hard Negative examples: Tortilla $\rightarrow$ Tostitos Tortilla Chips

Data Augmentation for Data Set expansion



Two kinds of Data Augmentation/Data Expansion

1. **Expanding Product Signals:** This refers to not just using product titles but also product description or even images (multi-modal signals) for bettery *Product Embedding*.
2. E.g. 'Organic Milk' $\rightarrow$ 'Organic 2% Reduced Fat Milk'
3. **Expanding Cold Start Data:** Products that just got launched or are new to the Instacart ecosystem get surfaced through data augmentation. Here - (Query, Product) examples are **synthetically** created as training data for the model so it can learn to recognize and recommend new products.

Data Augmentation for Data Set exapansion

Data Augmentation in LLM context

This is a fairly common strategy that gets used in NLP tasks and in the use of LLMs. For instance - Microsoft's **Phi** model, which is a **Small Language Model**(SLM) was trained in part with high-quality *textbook data*, where the textbooks themselves got generated using a more powerful GPT model!

Data Augmentation for Data Set exapansion

Data Augmentation in LLM context

This is a fairly common strategy that gets used in NLP tasks and in the use of LLMs. For instance - Microsoft's **Phi** model, which is a **Small Language Model**(SLM) was trained in part with high-quality *textbook data*, where the textbooks themselves got generated using a more powerful GPT model!

LLMs as annotators and paraphrasers

Teacher → Student
(Large LM) (Small LM)

Also used often, analogous to the previous Phi model example is annotating inputs with targets using an accurate GPT model or generating more training data through paraphrase of the input.

Breakouts Time #1: Product Review Classification (12 mins)

Classifying product reviews

Let's say that you are a data scientist at Sambazon! Sambazon is an online retailer selling millions of products under tens of thousands of product categories. You work in the **Review Moderation and Insights** team that is responsible for deriving actionable insights from customer reviews data. Your team's charter includes understanding the **intent** of the reviews - esp. if its useful or obnoxious. Your team's product manager (PM) suggests that as part of this years roadmap, the product team would like to understand reviews from the lens of the following categories: highly useful, highly passionate, obnoxious and balanced. How would you as a scientist a) approach this problem b) What would be your sources of data? c) What would be the ML approach you would use? d) How would you train the model? e) What if you didn't have labels in the data as your PM suggested? f) What if you had labels for training but not enough data?

Model Training Architecture

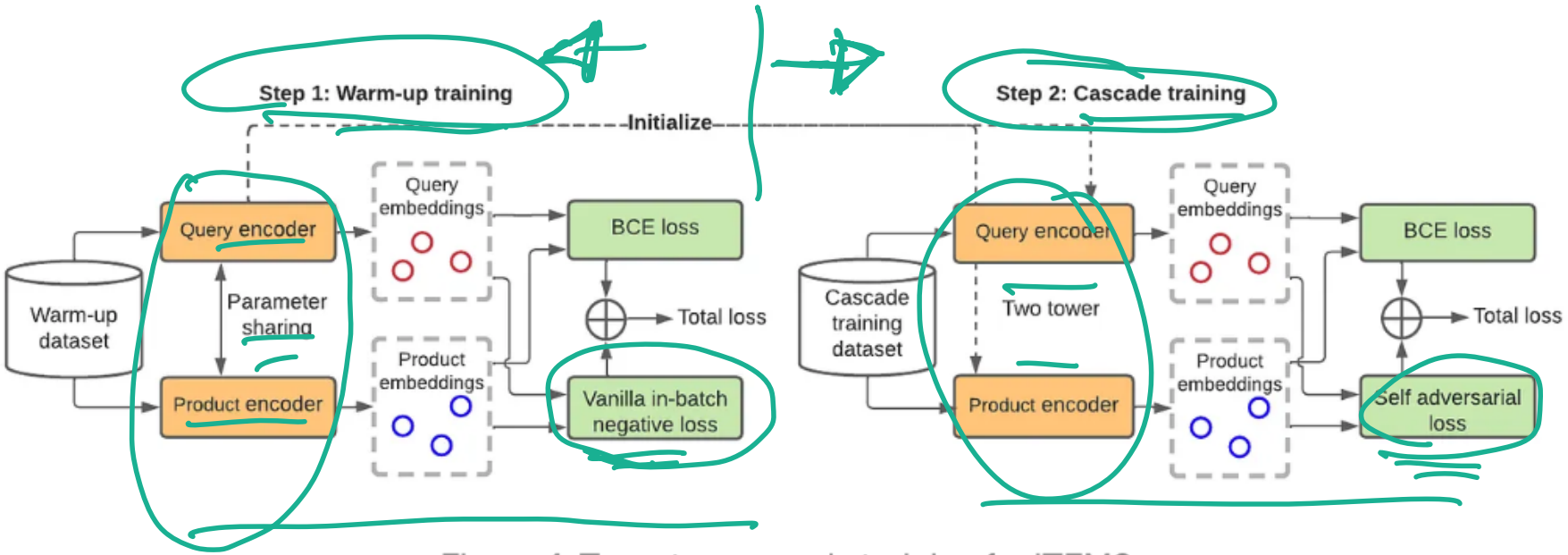


Figure 4. Two-step cascade training for ITEMS.

System Design

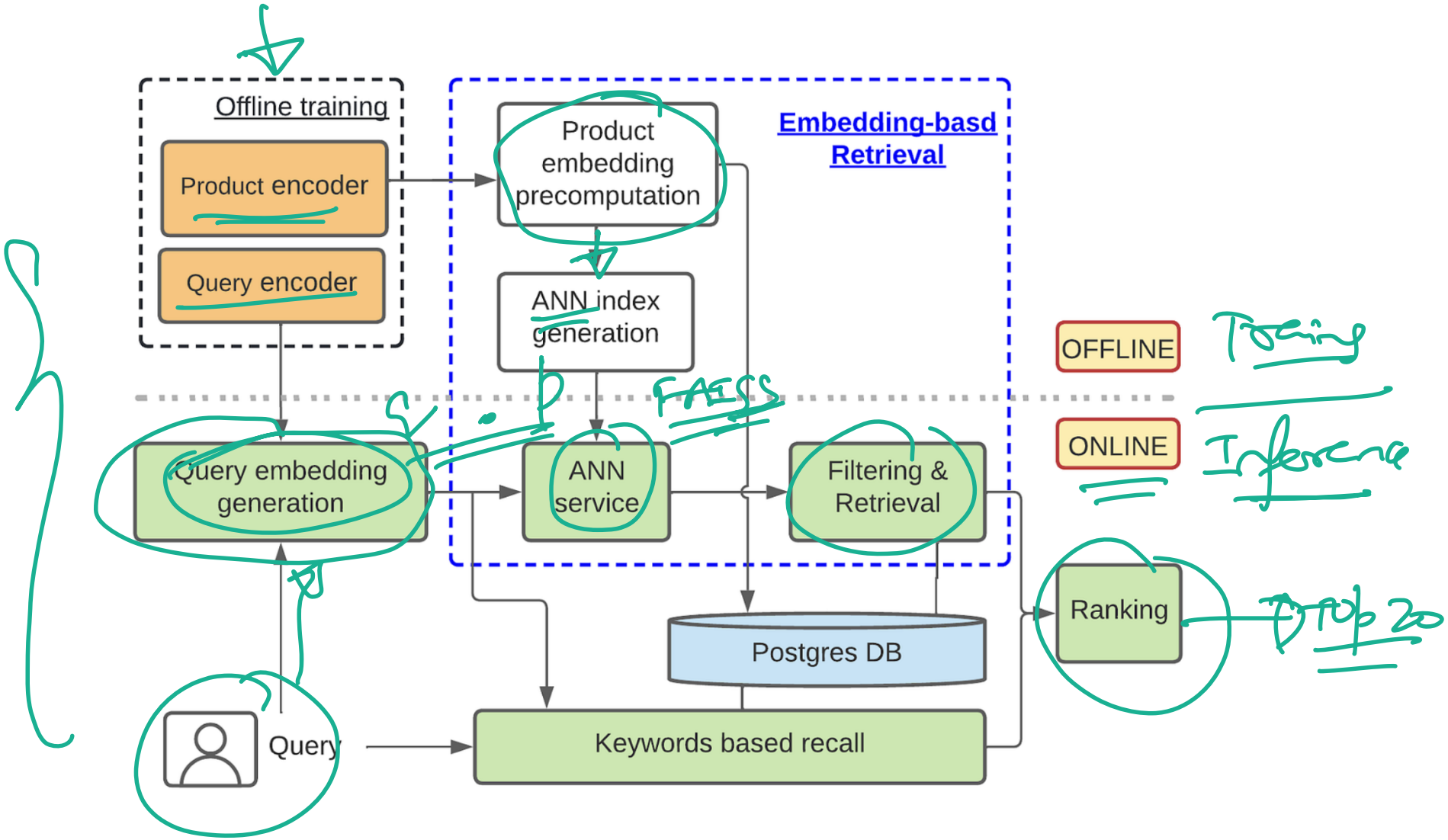
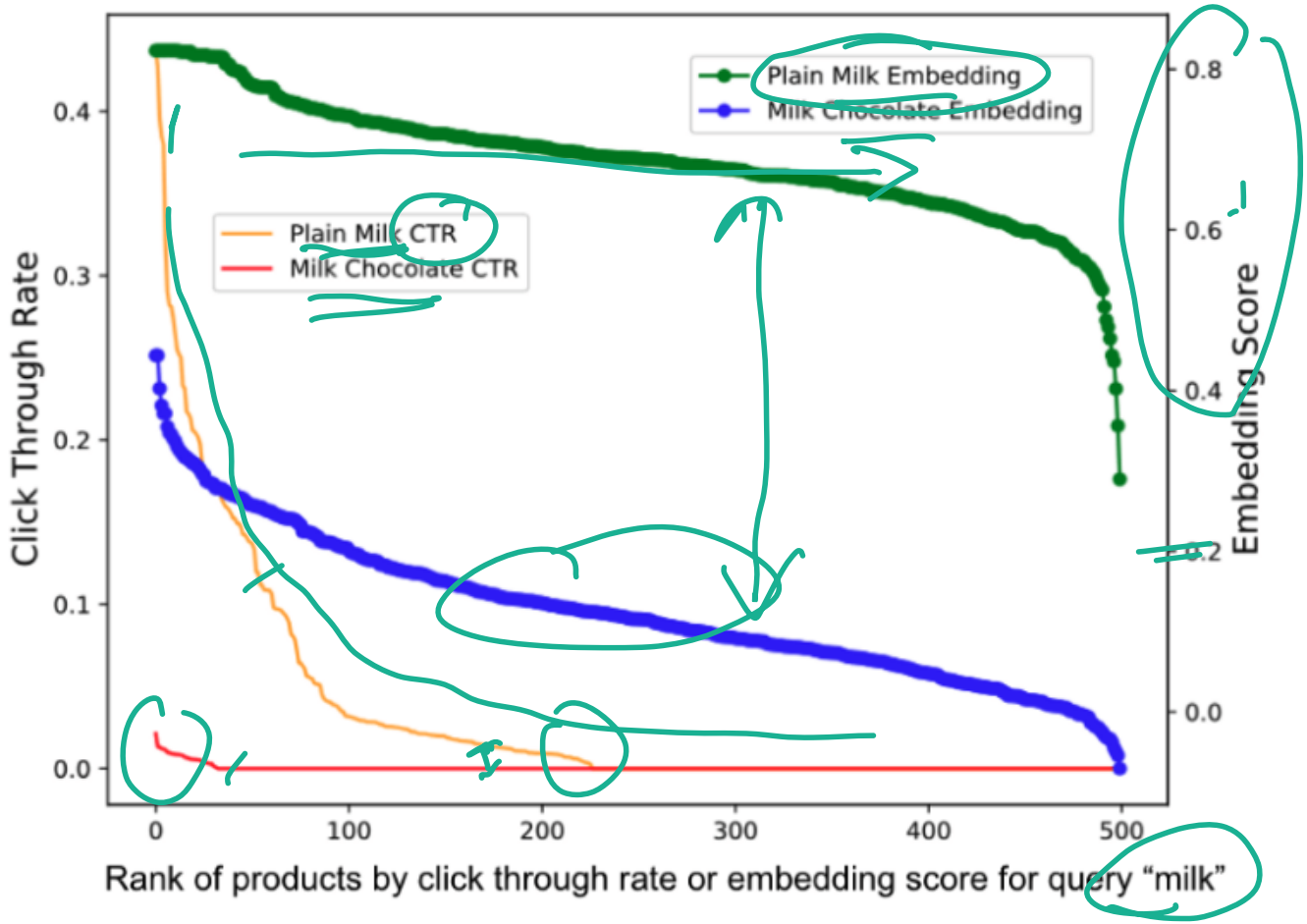


Figure 7. ITEMS system architecture.

CTR vs Embedding Score

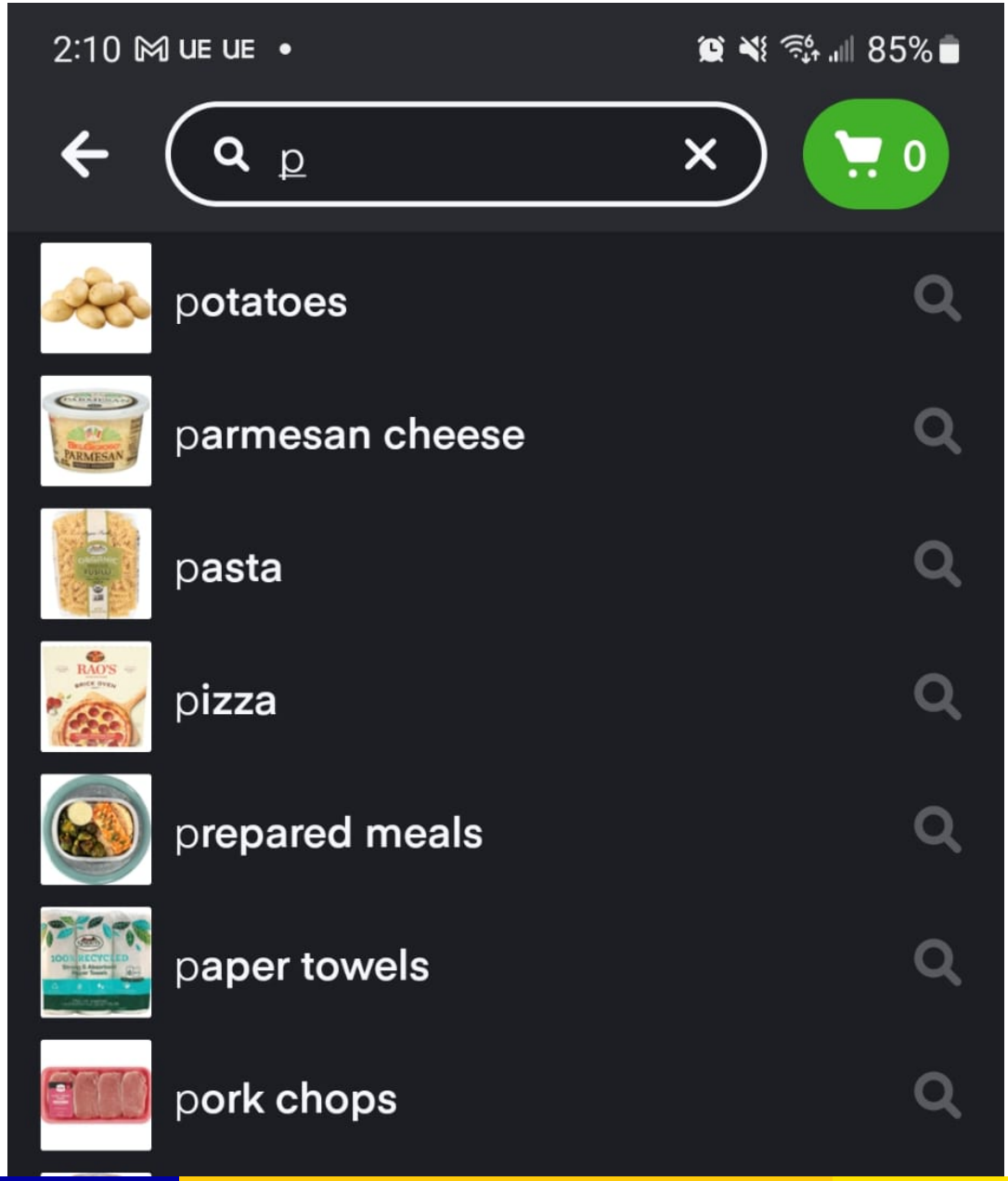


Breakouts Time #2

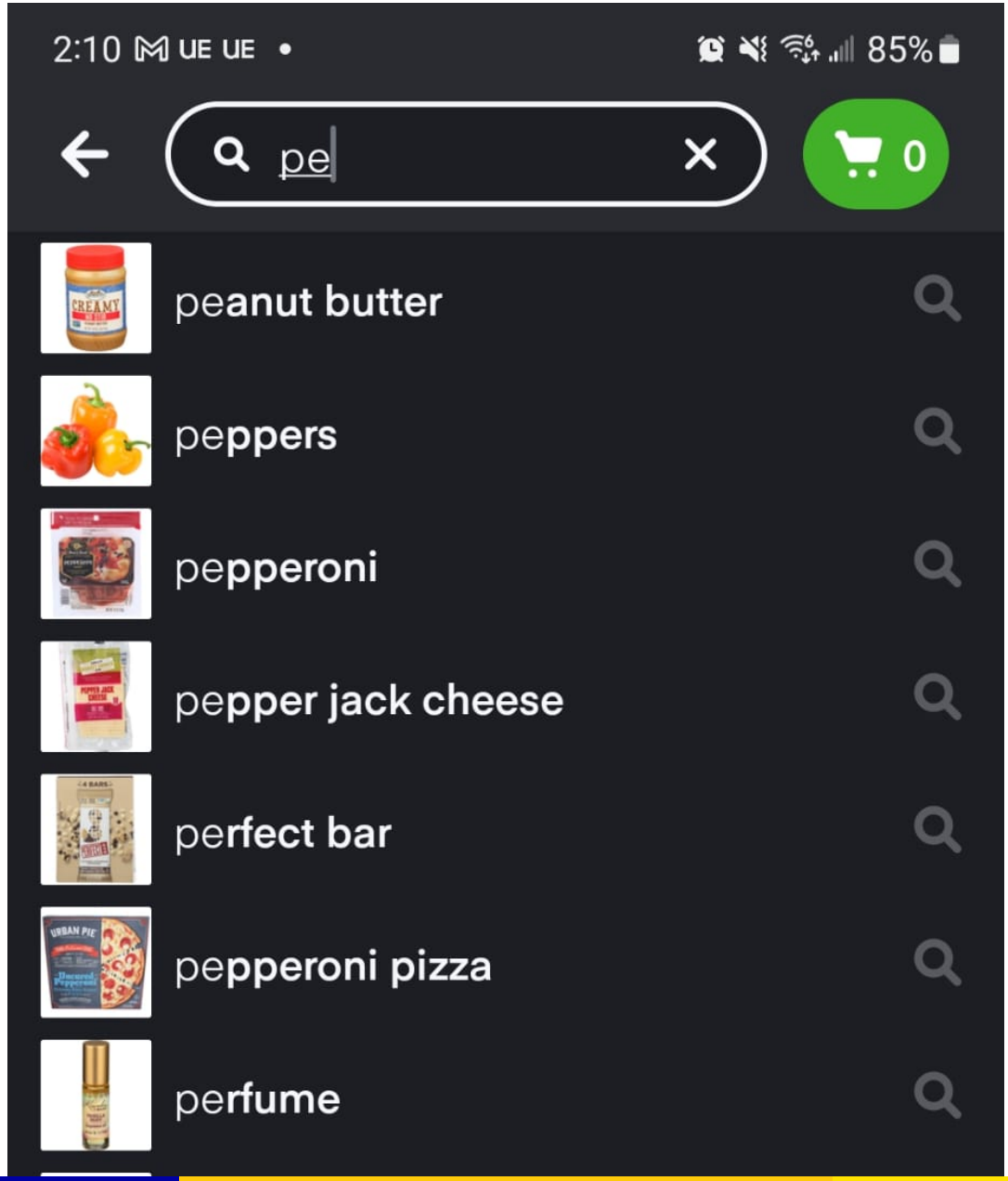
Instacart Auto-complete — 5 mins

Let's say you are tasked with building an auto-complete search for instacart, which can help complete partial words into full words/sentences through suggestions (auto-complete). How would you use what we have learned so far to model this? What architecture would you use? What would be your data? And what are some pitfalls or painpoints your model should address?

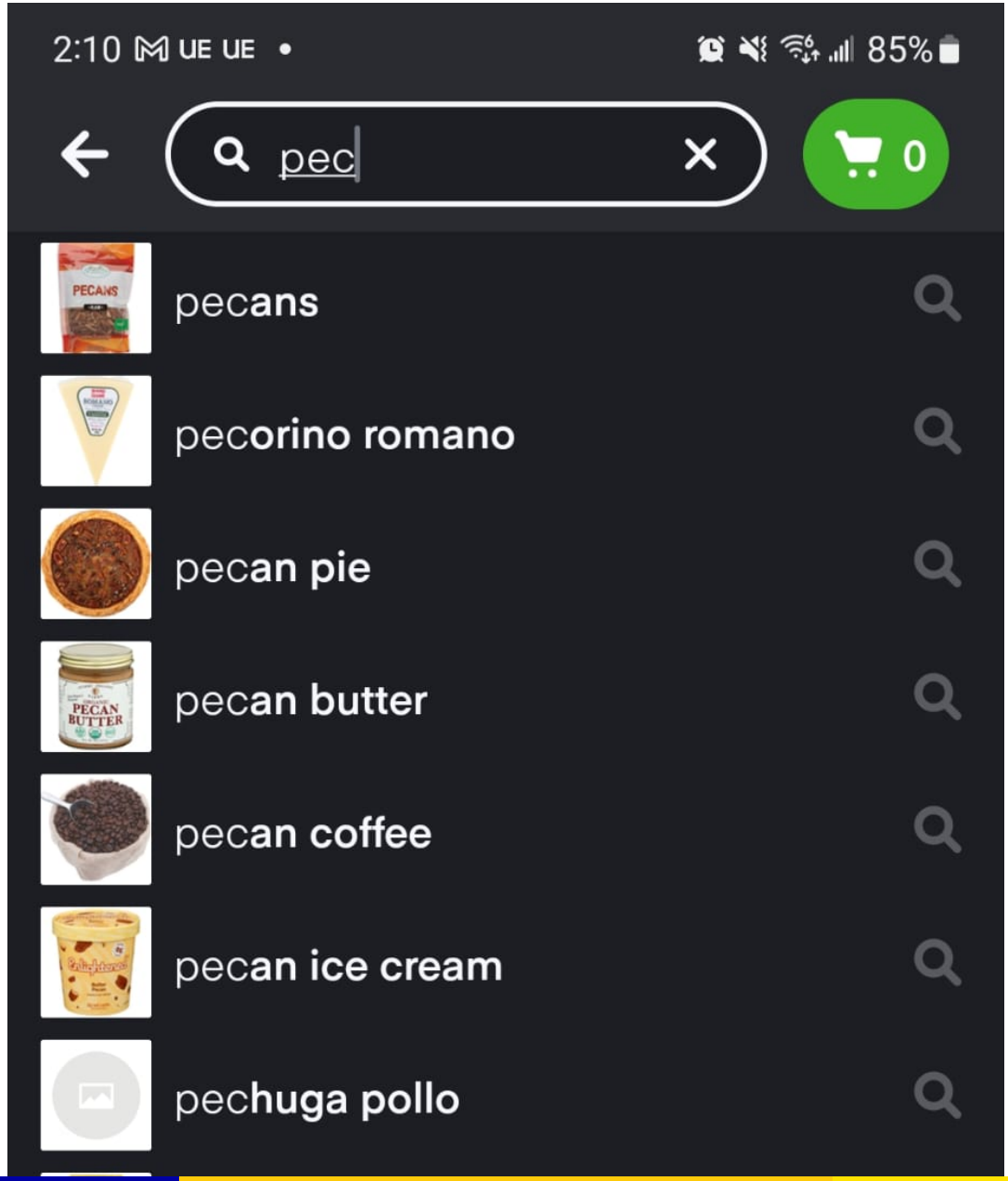
Instacart Auto-Complete and Search Relevance



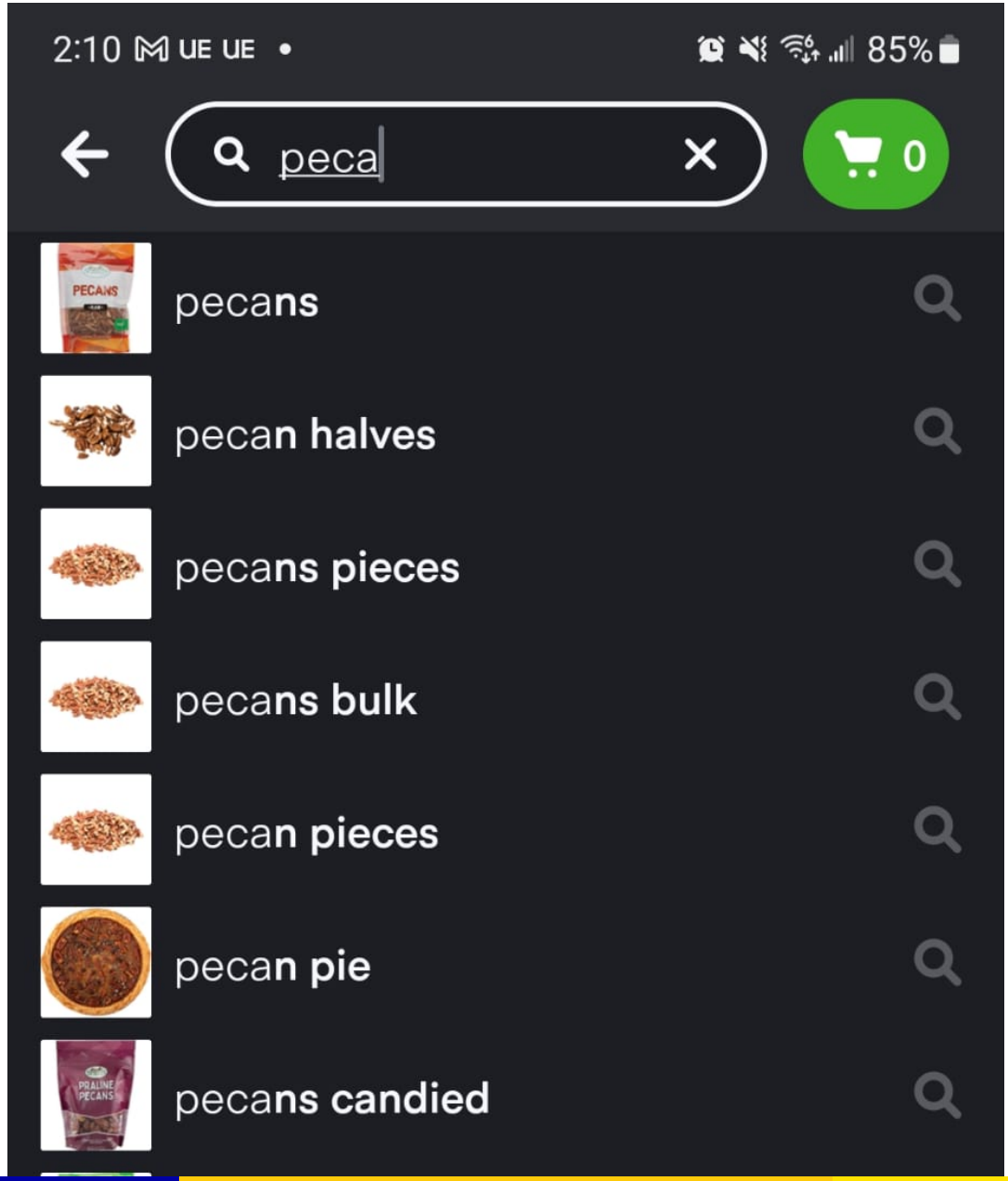
Instacart Auto-Complete



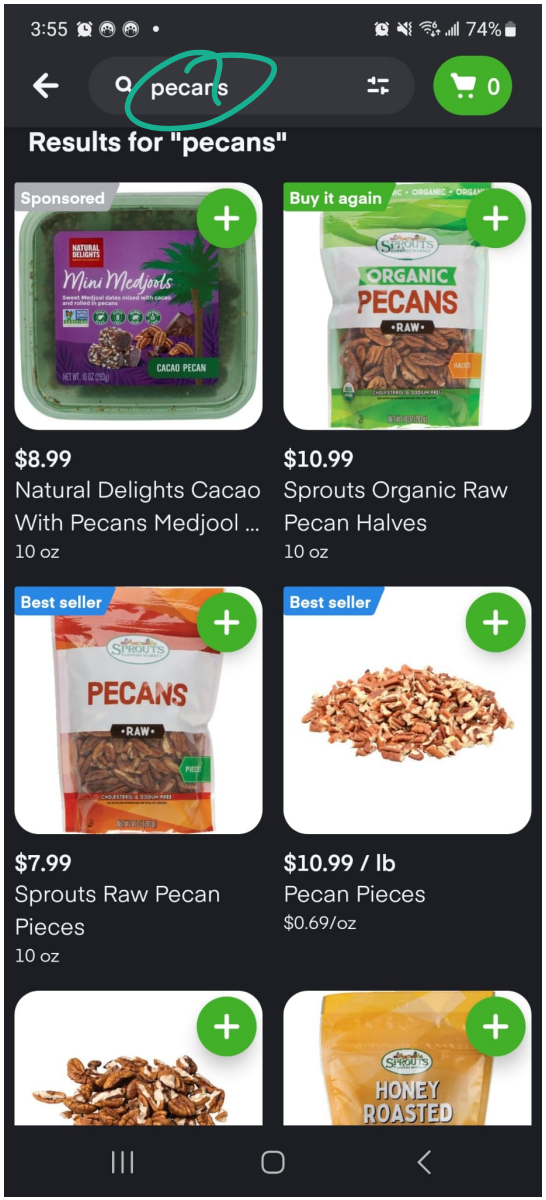
Instacart Auto-Complete



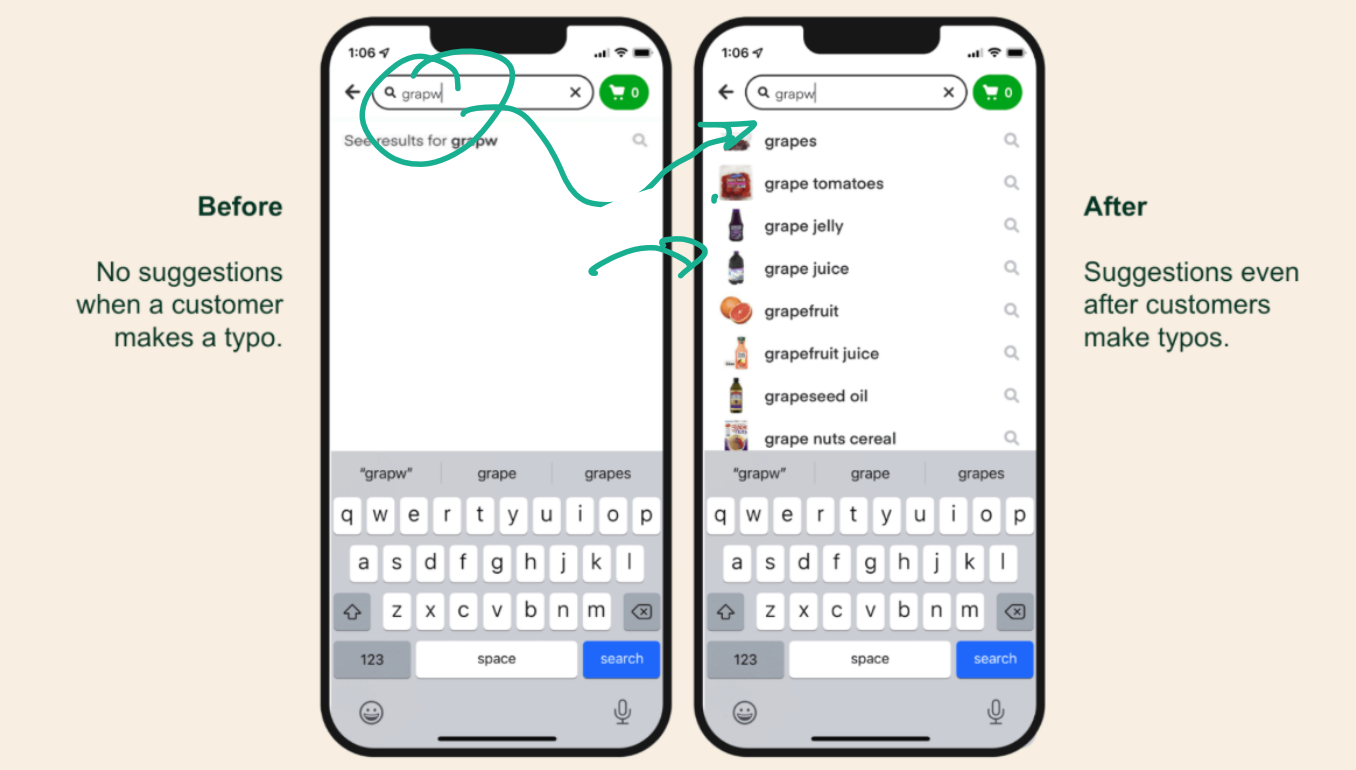
Instacart Auto-Complete



Instacart Auto-Complete and Search Results



HdanLing MSiPsellings



Instacart Diversifying Auto-Complete

Figure 9. Autocomplete when a customer searches for “mac”, before (left) and after (right) semantic deduplication.

Prompt Engineering Best Practices