

jxnl.co

@jxnlco

Systematically Improving RAG Applications

Session 2

If you're not fine-tuning, you're Blockbuster, not Netflix

Jason Liu

Overview of RAG Playbook

Sessions 1–3: Part 1

Building synthetic data, fine-tuning, and collecting user data and product

- ☐ Generate synthetic data
- ☐ Generate fast evaluations
- ☐ Use evals dataset to start finetuning
- ☐ Build a UX that feels fast and collects data for..
More evals!

Session 4–6: Part 2

Split, Map, Apply: Build search indices and query routing systems separately

- ☐ Segment Input queries to identity new topics and capabilities to invest in
- ☐ Refine and improve query routing system
- ☐ Explore new search indices or new metadata to better answer questions for a specific query segments (i.e., by generating synthetic questions)

Agenda

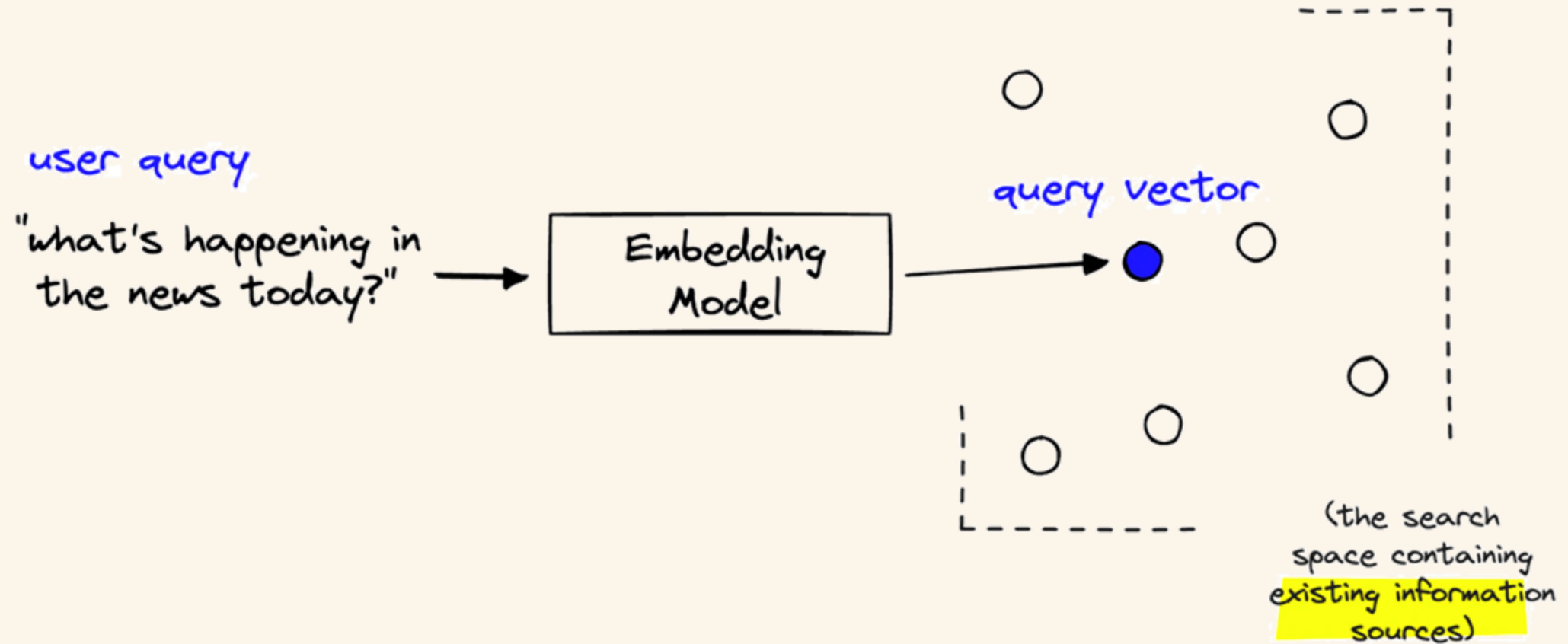
The challenge with using providers' existing embeddings

How to improve representations

Homework for this session









Next session

Source: <https://www.pinecone.io/learn/series/rag/embedding-models-rundown/>

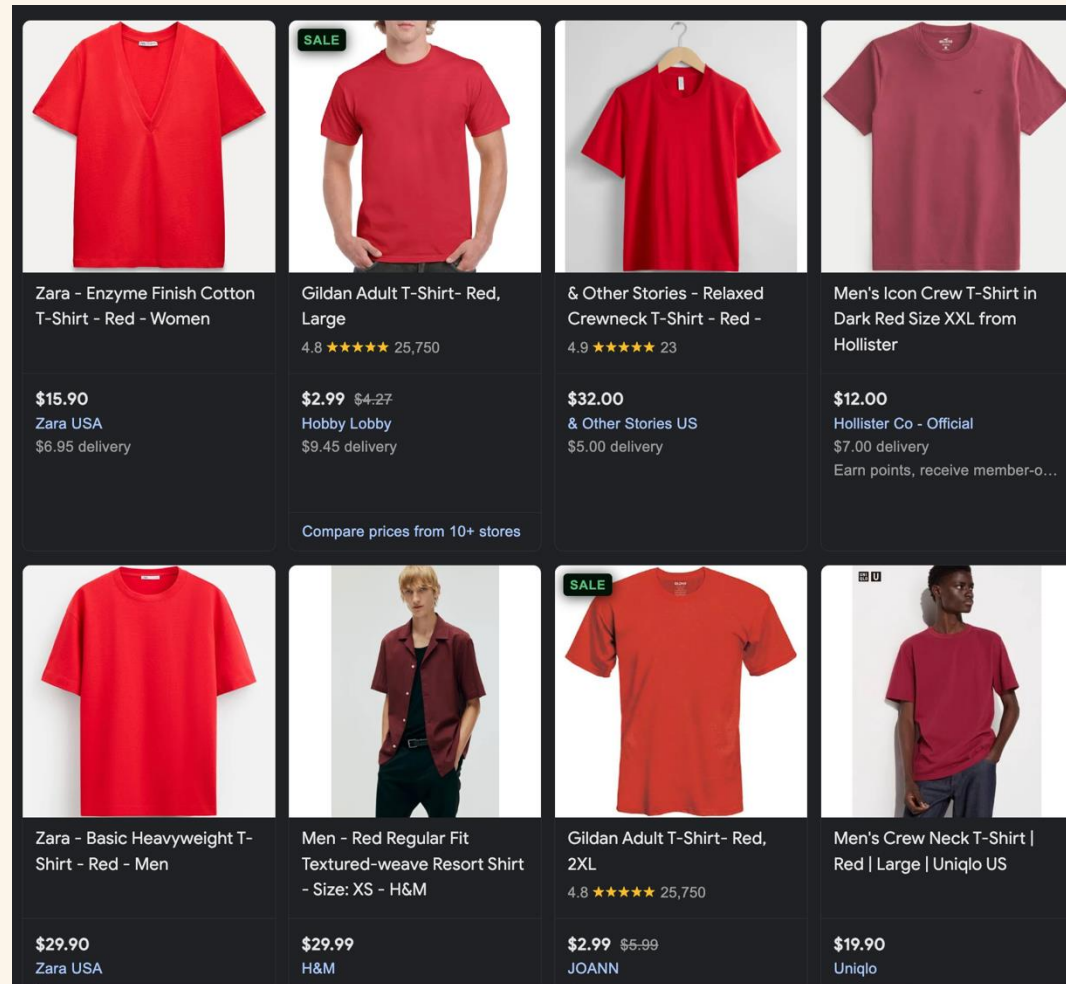


Embedding models translate human-readable text into machine-readable and searchable vectors.

Why does implementing a provider's existing embedding model fail?

			
Zara - Enzyme Finish Cotton T-Shirt - Red - Women	Gildan Adult T-Shirt- Red, Large	& Other Stories - Relaxed Crewneck T-Shirt - Red -	Men's Icon Crew T-Shirt in Dark Red Size XXL from Hollister
4.8 ★★★★★ 25,750	4.9 ★★★★★ 23		
\$15.90 Zara USA \$6.95 delivery	\$2.99 \$4.27 Hobby Lobby \$9.45 delivery	\$32.00 & Other Stories US \$5.00 delivery	\$12.00 Hollister Co - Official \$7.00 delivery Earn points, receive member-o...
	Compare prices from 10+ stores		
			
Zara - Basic Heavyweight T-Shirt - Red - Men	Men - Red Regular Fit Textured-weave Resort Shirt - Size: XS - H&M	Gildan Adult T-Shirt- Red, 2XL	Men's Crew Neck T-Shirt Red Large Uniqlo US
4.8 ★★★★★ 25,750	4.8 ★★★★★ 25,750		
\$29.90 Zara USA	\$29.99 H&M	\$2.99 \$5.99 JOANN	\$19.90 Uniqlo

Why does implementing a provider's existing embedding model fail?



We assume that the embedding model understands the user's intent and context

For example, after a user adds a red shirt to cart, we don't know if the user wants

- More similar red shirts
- More shirts of different colors in a similar fashion (e.g., V-neck)
- Pants, shoes, or a bag that may complement the red shirt as a part of an outfit

The model will only return high cosine similarity shirts

maven.com/applied-llms/rag-playbook

Why does implementing a provider's existing embedding model fail?

The Distance Assumption

- `distance(embed(query), embed(chunk))`
 - $\sim P(\text{relevant})$
- `distance(embed(user), embed(product))`
 - $\sim P(\text{purchase})$
- `distance(embed(product), embed(product))`
 - $\sim P(\text{substitutable})$
 - $\sim P(\text{complementary})$
 - $\sim P(\text{same cart}) \sim ???$
- `distance(embed(song), embed(user))`
 - $\sim P(\text{listen})$
 - $\sim P(\text{add to playlist})$
 - $\sim ???$
- `distance(embed(song), embed(song))`
 - $\sim P(\text{same playlist})$
 - $\sim P(\text{liked by same people})$
 - $\sim P(\text{same style})$
 - $\sim ???$



Using a providers embedding model bakes many assumptions on what similarity means

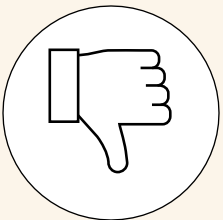
All that exists is the dataset we trained on

Dating app example

Are these similar or different?



“I love coffee”



“I hate coffee”

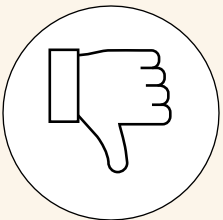


Dating app example

Are these different or similar?



“I love coffee”



“I hate coffee”

Answer: It depends

Different

- In the context of a dating profile , users who love coffee likely don't want to date users who hate coffee

Similar

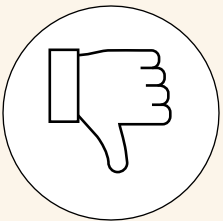
- In the context of a dating profile, both users may be foodies. This might demonstrate strong preferences for food and drink (e.g., one loves tea, one hates coffee)

Dating app example

Are these different or similar?



“I love coffee”



“I hate coffee”

Answer: It depends

Key takeaway:

- Ultimately, these questions don't matter in isolation. Even if OpenAI's embedding model has data on comparing texts, *the objective function is very different based on your product*
- For dating apps, we want to build a model that takes two profiles, compares them, predicts whether they will like each other (swipe left, send a super like), go on a date



What does this mean for you?

There is likely a significant gap between the text embeddings and the rankers you currently use, versus what would constitute a successful match.



What does this mean for you?

It's important to set up your logging now to collect the data needed to train and fine-tune an embedding model or reranker model.

Agenda

The challenge with using providers' existing embeddings

How to improve representations

Include more data or generate synthetic data

Finetuning and re-rankers

Homework for this session

Next session

Better Representations = More Data

Key takeaway:

- Don't assume the question embedding and the object embedding (e.g., the answer text embedding) will be similar for some definition
- The objective of similarity is weak and there's likely going to be something better

Example

- **Dating apps:** For two bios, do they like each other?
- **Music recommendation:** Given two songs, what's the likelihood they're on the same playlist?



What if I have no data?

... Synthetic data again

Leveraging Synthetic data

All the experience of generating synthetic data to test our precision and recall can now be reapplied to train a model that optimizes recall!

- For all the tasks where we evaluated the success of the search system using recall and embeddings we can now consider building finetuning data that is trained to improve recall
- Examples of datasets now look like:
 - Queries → Image summaries
 - Queries → Code Snippet
 - Queries → Table Summaries
 - Queries → Chunks
 - Queries → Table Chunks
 - Queries → Tool Descriptions

We can:

- We can also use this dataset to finetune
- We can also use LLMs to add better relevancy
- Allow us to improve everything we do over these 6 weeks

Leveraging Synthetic data

All the experience of generating synthetic data to test our precision and recall can now be reapplied to train a model that optimizes recall!

- For all the tasks where we evaluated the success of the search system using recall and embeddings we can now consider building finetuning data that is trained to improve recall
- Examples of datasets now look like:
 - Queries → Image summaries
 - Queries → Code Snippet
 - Queries → Table Summaries
 - Queries → Chunks
 - Queries → Table Chunks
 - Queries → Tool Descriptions

We can:

- We can also use this dataset to finetune
- We can also use LLMs to add better relevancy
- Allow us to improve everything we do over these 6 weeks

How this helps monitoring:

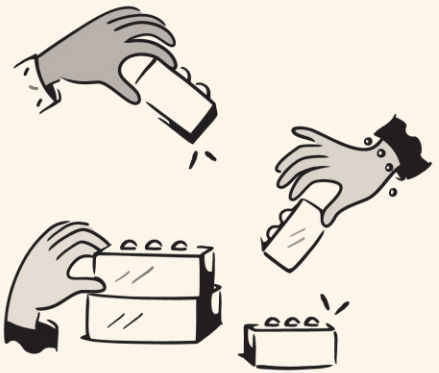
This embedding model, now equipped with the knowledge of your data, will likely perform better for:

- Embedding-based topic modeling
- Embedding-based classification



Everything we're using large language models (LLMs) for is what I had to pay data labeling teams hundreds of thousands of dollars for every year.

This is the ML playbook, but previously only possible at large companies



Historically, we build a product to collect data
Then use it to train a model
Then release a new product



LLMs just allow us to do this backwards

This is our wax on, wax off, Mr. Miyagi moment.



When someone says "This is our Wax on Wax off Mr Miagi moment," they typically mean:

1. They're in a situation where they're learning or practicing something through repetition.
2. The task at hand might seem mundane or unrelated to the ultimate goal.
3. There's an expectation that this repetitive practice will lead to mastery or understanding later on.
4. It's often used in contexts of mentorship or training.

This phrase has become a cultural shorthand for the idea that seemingly unrelated or repetitive tasks can have hidden benefits or lead to unexpected skill development.

 Copy  Retry  



Claude can make mistakes. Please double-check responses.

Agenda

The challenge with using providers' existing embeddings

How to improve representations

Include more data or generate synthetic data

Finetuning and re-rankers

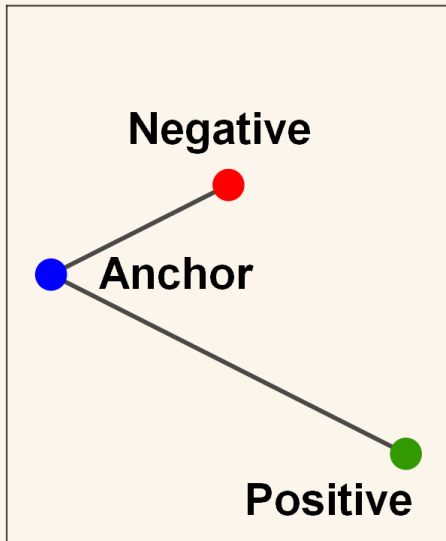
Homework for this session

Next session

So what happens when we finetune?

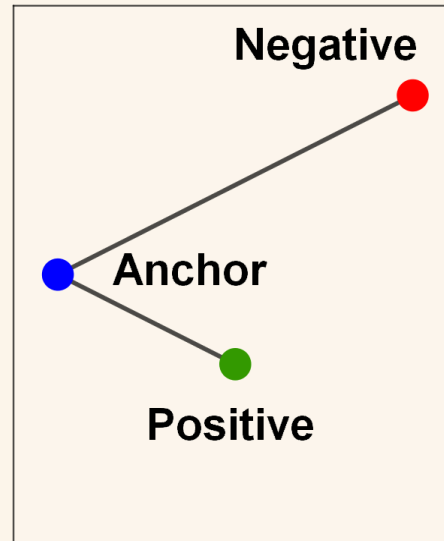
Create triplet examples (anchor and positive have same label, negative has different label)

Before fine-tuning



Learning
→

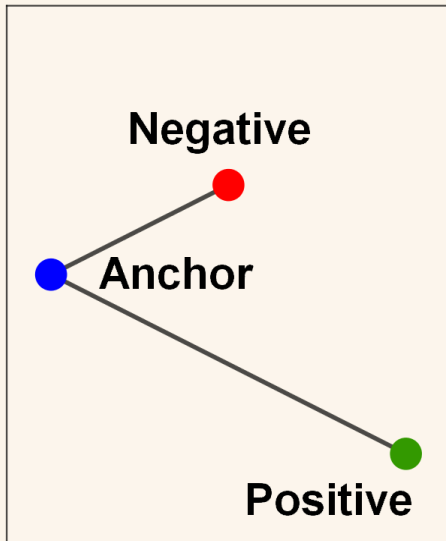
After fine-tuning



So what happens when we finetune?

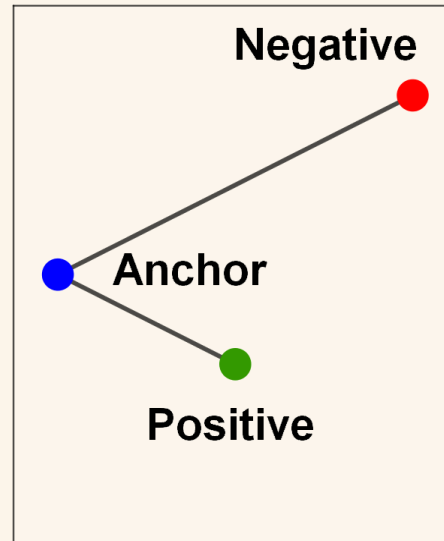
Create triplet examples (anchor and positive have same label, negative has different label)

Before fine-tuning



Learning
→

After fine-tuning

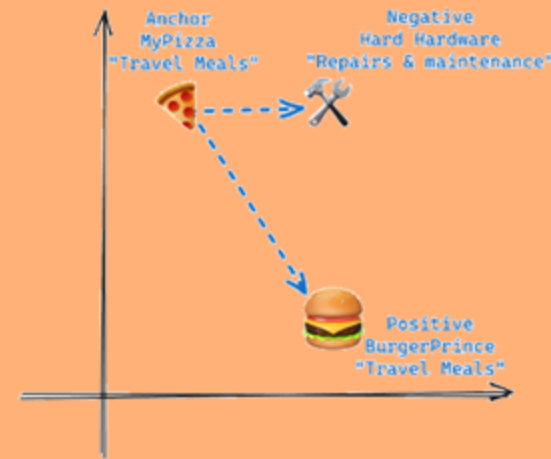


Case study: Improving Retrieval on Ramp with Transaction Embeddings

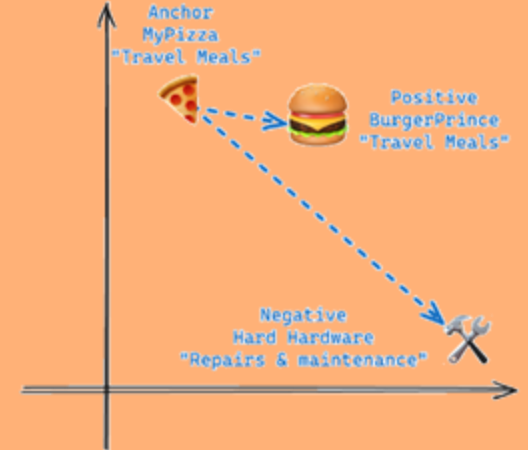
Additional reading:

<https://engineering.ramp.com/transaction-embeddings>

Before fine-tune



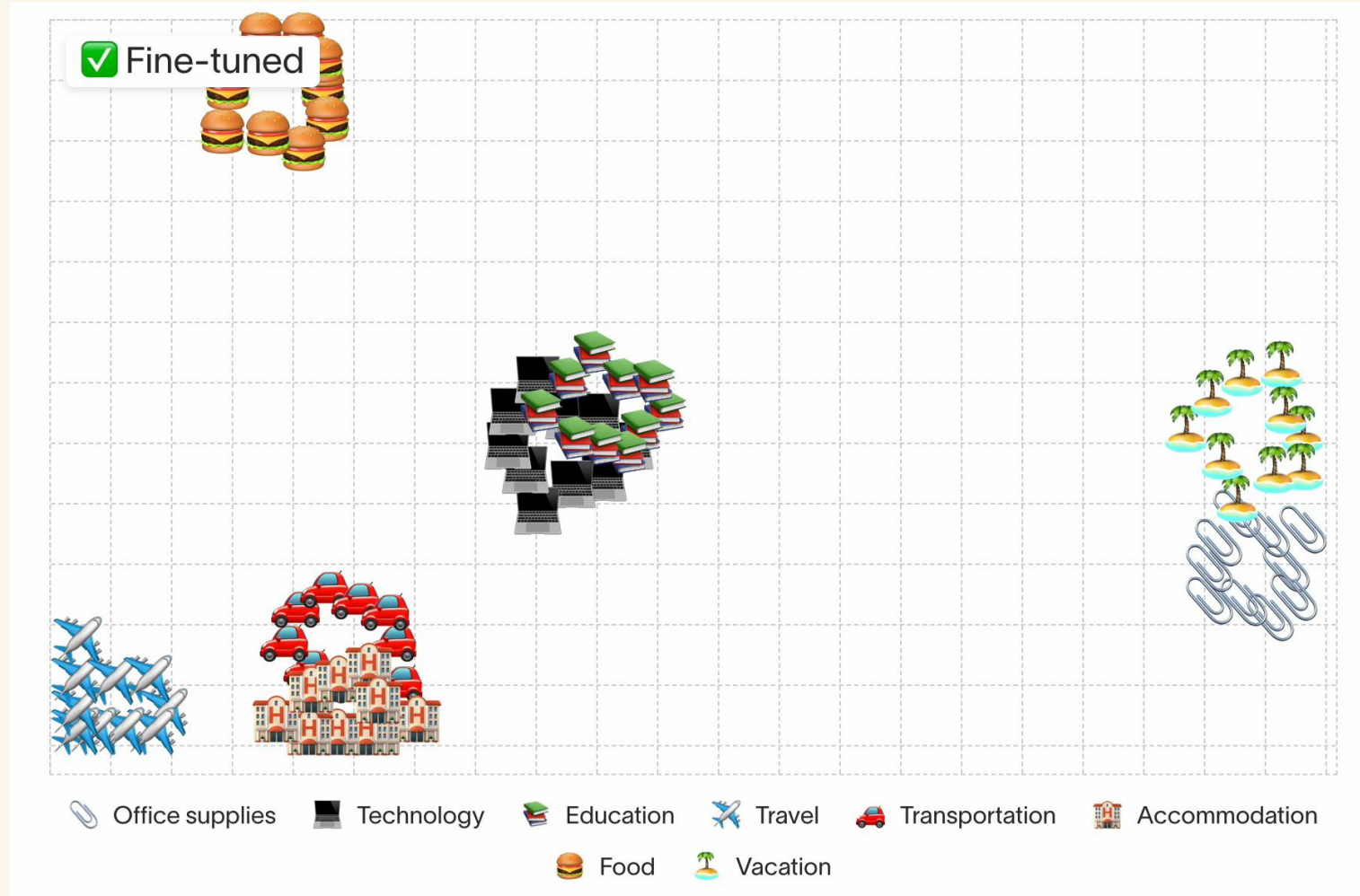
After fine-tune



Visualization of anchor, positive, and negative samples in triplet loss.

Case study: Improving Retrieval on Ramp with Transaction Embeddings

Additional reading: <https://engineering.ramp.com/transaction-embeddings>



What does the data look like?

1. With enough examples, your model will ultimately learn your objective
2. Examples:
 - query, passage, is_relevant (could be bootstrapped by LLM)
 - query, passage, is_cited (to provide even higher signal)
3. Consider both pair and triplet examples

```
question:str  
answer: str
```

```
question: str  
positive_examples: List[str]  
negative_examples: List[str]
```

4. If you have high quality data, you can still use the dataset to train both bi-encoders and cross-encoders, and even ColBERT models
 - You'll still need both!

What does the data look like?

1. With enough examples, your model will ultimately learn your objective
2. Examples:
 - query, passage, is_relevant (could be bootstrapped by LLM)
 - query, passage, is_cited (to provide even higher signal)
3. Consider both pair and triplet examples

```
question: str  
answer: str
```

```
question: str  
positive_examples: List[str]  
negative_examples: List[str]
```

4. If you have high quality data, you can still use the dataset to train both bi-encoders and cross-encoders, and even ColBERT models
 - You'll still need both!

@jxnico

Remember: your data is your moat, not the model!

1. Think about your models not as models but as data and unique application objectives
2. Especially with Cohere's re-ranking API, you can fine-tune and serve embedding models at scale optimized for your use case
3. You may also wonder how much data you need to fine-tune embeddings
 - [My experiment](#): with 1000 examples, you can start to outperform OpenAI (with the correct feedback)!

maven.com/applied-llms/rag-playbook

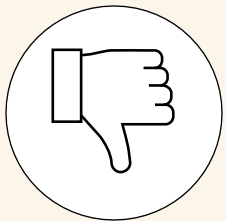
```
{
  "anchor": "What are the main causes of climate change?",
  "pos": [
    "Climate change is primarily caused by human activities that release greenhouse gases into the atmosphere. The burning of fossil fuels for energy, deforestation, and industrial processes are major contributors.",
    "The main drivers of climate change include the emission of greenhouse gases from burning fossil fuels, changes in land use such as deforestation, and industrial processes. These human activities increase the concentration of heat-trapping gases in the Earth's atmosphere."
  ],
  "neg": [
    "The Earth's climate has changed throughout history. Just in the last 650,000 years there have been seven cycles of glacial advance and retreat, with the abrupt end of the last ice age about 11,700 years ago marking the beginning of the modern climate era – and of human civilization.",
    "Climate change mitigation involves reducing greenhouse gas emissions and enhancing sinks that absorb greenhouse gases from the atmosphere."
  ]
}
```

Coffee example

Are these similar or different?



“I love coffee”



“I hate coffee”

Process


Push things that go together closer and push things that don't go together further apart

1. Initialize the 3 example vectors (anchor, positive, negative) with random numbers
2. The ML system will bring complementary things closer and distant things further
3. Depending on how we curate this data, we can start enforcing opinions of our model
 - Option 1: **“I love coffee”** & **“I love tea”** are similar to each other and dissimilar to **“I hate tea”**
 - Option 2: **“I love tea”** & **“I hate tea”** are the same because of positive vs. negative preferences, while **“I like coffee”** is different

Additional datasets:

https://sbert.net/docs/sentence_transformer/dataset_overview.html

maven.com/applied-llms/rag-playbook



SBERT.net

🏠 Sentence Transformers

👤 Star 14,712

Search docs

GETTING STARTED

Installation

Quickstart

SENTENCE TRANSFORMER

- Usage
- Pretrained Models
- Training Overview
- Dataset Overview
- Loss Overview
- Training Examples

CROSS ENCODER

- Usage
- Pretrained Models
- Training Overview
- Training Examples

PACKAGE REFERENCE

- Sentence Transformer
- Cross Encoder
- util

Quickstart

Sentence Transformer

Characteristics of Sentence Transformer (a.k.a bi-encoder) models:

1. Calculates a **fixed-size vector representation (embedding)** given **texts or images**.
2. Embedding calculation is often **efficient**, embedding similarity calculation is **very fast**.
3. Applicable for a **wide range of tasks**, such as semantic textual similarity, semantic search, clustering, classification, paraphrase mining, and more.
4. Often used as a **first step in a two-step retrieval process**, where a Cross-Encoder (a.k.a. reranker) model is used to re-rank the top-k results from the bi-encoder.

Once you have [installed](#) Sentence Transformers, you can easily use Sentence Transformer models:

```
from sentence_transformers import SentenceTransformer

# 1. Load a pretrained Sentence Transformer model
model = SentenceTransformer("all-MiniLM-L6-v2")

# The sentences to encode
sentences = [
    "The weather is lovely today.",
    "It's so sunny outside!",
    "He drove to the stadium.",
]

# 2. Calculate embeddings by calling model.encode()
embeddings = model.encode(sentences)
print(embeddings.shape)
# [3, 384]

# 3. Calculate the embedding similarities
similarities = model.similarity(embeddings, embeddings)
print(similarities)
# tensor([[1.0000, 0.6660, 0.1046],
#         [0.6660, 1.0000, 0.1411],
#         [0.1046, 0.1411, 1.0000]])
```

Documentation

1. [SentenceTransformer](#)
2. [SentenceTransformer.encode](#)
3. [SentenceTransformer.similarity](#)


Other useful methods and links:

- [SentenceTransformer.similarity_pairwise](#)
- [SentenceTransformer > Usage](#)
- [SentenceTransformer > Pretrained Models](#)
- [SentenceTransformer > Training Overview](#)
- [SentenceTransformer > Dataset Overview](#)
- [SentenceTransformer > Loss Overview](#)
- [SentenceTransformer > Training Examples](#)

[← Back to Articles](#)



Finally, a Replacement for BERT



▲ Upvote 525



 +513



Published December 19, 2024



[Update on GitHub](#)



[bwarner](#)
Benjamin Warner
 [answerdotai](#)



[NohTow](#)
Antoine Chaffin
 [lightonai](#)



[bclavie](#)
Benjamin Clavié
 [answerdotai](#)



[orionweller](#)
Orion Weller
 [guest](#)


[ohallstrom](#)
Oskar Hallström
 [lightonai](#)



[staghado](#)
Said Taghadouini
 [lightonai](#)



[alexisgallagher](#)
Alexis Gallagher
 [answerdotai](#)



[rbiswasfc](#)
Raja Biswas
 [guest](#)



[fladhak](#)
Faisal Ladhak
 [guest](#)



[tomaarsen](#)
Tom Aarsen

[ncoop57](#)
Nathan Cooper
 [answerdotai](#)

[griffin](#)
Griffin Adams
 [guest](#)

[jph00](#)
Jeremy Howard
 [answerdotai](#)

[johnowhitaker](#)
Jonathan Whitaker
 [answerdotai](#)

[iacolippo](#)
Iacopo Poli
 [lightonai](#)

TL;DR

This blog post introduces [ModernBERT](#), a family of state-of-the-art encoder-only models representing improvements over older generation encoders across the board, with a **8192** sequence length, better downstream performance and much faster processing.

ModernBERT is available as a *slot-in* replacement for any BERT-like models, with both a **base** (149M params) and **large** (395M params) model size.

► Click to see how to use these models with [transformers](#)

CHAT DASHBOARD PLAYGROUND DOCS COMMUNITY

LOG IN →

Guides and concepts

API Reference

Release Notes

LLMU

Cookbooks

Search...

/

Introduction

Fine-tuning with Web-UI

Programmatic Fine-tuning

Fine-tuning for Chat

Fine-tuning for Classify

Fine-tuning for Rerank

- Preparing the Rerank Fine-tuning Data
- Starting the Rerank Fine-Tuning
- Understanding the Rerank Fine-tuning Results
- Improving the Rerank Fine-tuning Results

FAQs / Troubleshooting

Going to Production

API Keys and Rate Limits

Going Live

How Does Cohere Pricing Work?

Fine-Tuning > Fine-tuning for Rerank

Understanding the Rerank Fine-tuning Results

In this section, we will explain the metrics for a fine-tuned model for Rerank.

Fine-tuned models for Rerank are trained using data consisting of queries mapping to relevant passages and documents and, for that reason, are evaluated using the same methods and performance metrics. You can also provide a test set of data that we will use to calculate performance metrics. If a test set is not provided, we will split your training data randomly to calculate performance metrics.

When you create a fine-tuned model for Rerank, you will see metrics that look like this:

81.25%

↑ 34%

Accuracy@1

Proportion of model predictions that contain a relevant passage as the first result

96.88%

↑ 34%

Accuracy@3

Proportion of model predictions that contain a relevant passage in the top three results

89.58%

↑ 33%

MRR@10

Mean Reciprocal Rank is a rank-aware relevance scoring in the first 10 results of a prediction

91.27%

↑ 32%

NDCG@10

Normalized Discounted Cumulative Gain is a rank-aware relevance scoring in the first 10 results of a prediction

Accuracy@1 and Accuracy@3

On this page

Accuracy@1 and Accuracy@3

MRR@10

Example of how MRR@k is calculated (when k=2)

nDCG@10

Example of how nDCG@k is Calculated

Example of how Cumulative Gain is Calculated

Example of how Discounted Cumulative Gain is Calculated

Example of how Ideal Discounted Cumulative Gain is Calculated

Example of How Normalized Discounted Cumulative Gain is Calculated

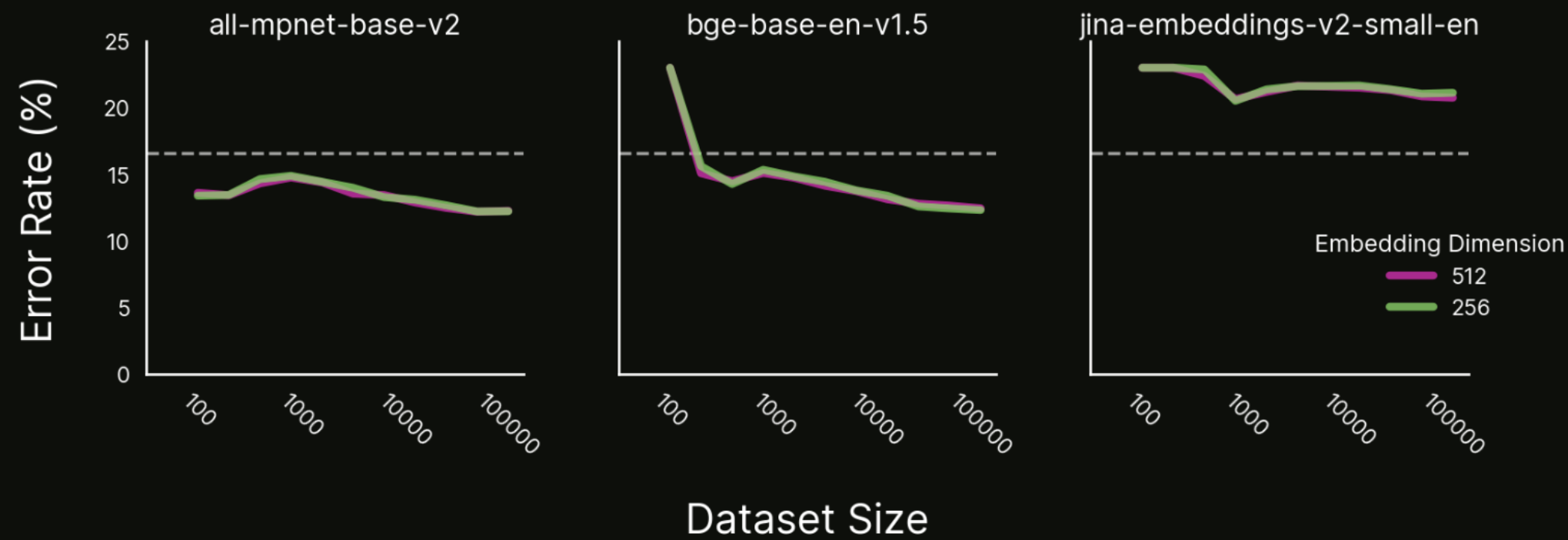
Putting it all Together

@jxnico

maven.com/applied-llms/rag-playbook

36

How much data do we need?



Error rate as a function of dataset size for three models fine-tuned on the Quora dataset.

Do we need to train task-specific models?

1. If you have multiple representations:
 - Questions and summaries
 - Questions and chunks
 - Questions and table/image summaries
2. These can all be mixed into a blend of training data to train a single model (eventually this can be used to replace OpenAI embeddings and the re-rankers)
3. In many cases multiple 'tasks' trained jointly can yield better results
4. It's unlikely that training the cosine distance or improving the cosine model will eliminate the need for the re-ranker
 - This can improve latency and precision recall trade-offs by having a two-tiered approach
5. Start collecting this data now because you'll need it some point in the future (as you expand your teams, you want to have the data ready for them to work with)

Fine-Tuning Encoders & Cross-Encoders in RAG: Case Studies and Implementations

Case Study / Implementation	Description	Performance Gains	Source
Enhancing Q&A Text Retrieval with Ranking Models: Benchmarking, fine-tuning and deploying Rerankers for RAG	Fine-tuned cross-encoder applied for reranking in a QA pipeline	14% accuracy boost over baseline retrieval	Link
Improving the Domain Adaptation of (RAG) Models for Open Domain Question Answering	Fine-tuning both question and passage encoders in a RAG system for domain adaptation.	~12% increase in Exact Match accuracy	Link
Re-ranking in Retrieval Augmented Generation: How to Use Re-rankers in RAG	Adding a fine-tuned cross-encoder on top of a dense retriever for customer support systems	20% boost in response accuracy; 30% reduction in irrelevant documents	Link

If you want to host your embedding models, a great place to check out is Modal Labs.

Check out my post on embedding all of Wikipedia in 15 minutes.

<https://modal.com/blog/embedding-wikipedia>

If you want to fine the best finetuned model
Check out my post on grid searching models using
50 GPUS

<https://modal.com/blog/fine-tuning-embeddings>

Agenda

The challenge with using providers' existing embeddings

How to improve representations

Homework for this session

Next session

Homework



Finish homework from last session

- Generate synthetic data to test your system (evaluation data set)
- **Establish a baseline to run experiments on**
 - Check BM25, Embeddings, Chunkers, and Rerankers (LanceDB)
 - With baselines, we can “do nothing” if experiments don’t pan out
- Review user queries: Sample some % of user traffic and run the LLM Ranker over retrieved text chunks and monitor questions that have low precision
- Start to build process and flywheel: Establish regular cadence to review questions or documents that get poor recall or even just low cosine or re-ranker scores



Homework for this session: Focus on improving representations

- For each set of subtasks we will have baseline recall metrics
 - Query → Chunk
 - Query → Snippets
 - Query → X
- Which ones need improvement the most?
- Can we prepare a triplets dataset?
 - Does Cohere Rerankers Finetuning improve recall for my task?
 - Do I have enough data (500–1000) to finetune and embedding model?
- Does it make sense to self host an embedding model?

Agenda

The challenge with using providers' existing embeddings

How to improve representations

Homework for this session

Next session

Overview of next week

- **Focus for last sessions:**
 - Understand the RAG playbook and how to implement it
 - Review how to use the RAG playbook steps (e.g., generate synthetic data, focus on recall-precision metrics) to tackle the query routing sub-problem while improving RAG applications
- **Focus of this sessions:**
 - Understand the major challenges with representations
 - Implement solutions to improve representations
- **Focus for next sessions:**
 - **Session 3:** The Art of RAG UX – Turning Design into Data