The Practitioner's Guide To E5

Arjun Patel

Jun 24, 2024

Deep Dives

A good embedding model makes a huge difference in the performance of a search application. When picking the first model to use with Pinecone Inference API, we were looking for a model that would be a good choice for a wide variety of applications. We picked E5 because it's small, open source, natively multilingual, and performs well on benchmarks across languages.

In this article, we'll give you a look under the hood of the E5 model: how it was trained, how to best use it, and a working example that shows off some of the handy properties of a multilingual model.

Why are embedding models important?

Most of the world's information is locked up in unstructured data like text, images, and audio.

These data can’t readily be interpreted by software applications unless they are converted to vectors (also known as embeddings), which are numerical representations of data.

Embedding models are the engines that do this conversion.

Multilingual models can do this within and across languages, acting as a sort of universal language semantic layer.

Embedding models turn input text into vectors, which are used for vector search in vector databases

Why host a model in the first place?

Building with vector databases involves a lot of moving parts, and one of the hardest ones is choosing a model to embed your data with.

Typically, this required you to:

1. Find a set of models, open source and third party APIs
2. Learn how to use the models, best practices and implementation
3. Compare their performance, and choose one
4. Deploy it for inference

5. Embed your data
6. Upsert into Pinecone
7. Build!

Now with the Pinecone Inference API, you can get started embedding and upserting your data within the Pinecone SDK. It’s as easy as one function call:

```
from pinecone import Pinecone

pc = Pinecone(API_KEY)

texts = [
  "This is the first sentence to be embedded",
  "This is the second sentence",
  ... # more sentences
  "This is the last sentence"
]

embeddings = pc.inference.embed(
  model="multilingual-e5-large",
  inputs=texts,
  parameters={"input_type": "passage"}
)
```

The Pinecone Inference API takes care of hosting best-in-class models and tokenization for you as well.

What is E5?

The E5 model series was developed by Microsoft researchers, and stands for ***E****mbe****E****ddings from bidir****E****ctional* ***E****ncoder r****E***presentations, and is meant to be a performant embedding model for semantic search, retrieval and classification tasks [1]. The specific instantiation of the E5 model hosted on Pinecone is called *multilingual-e5-large*.

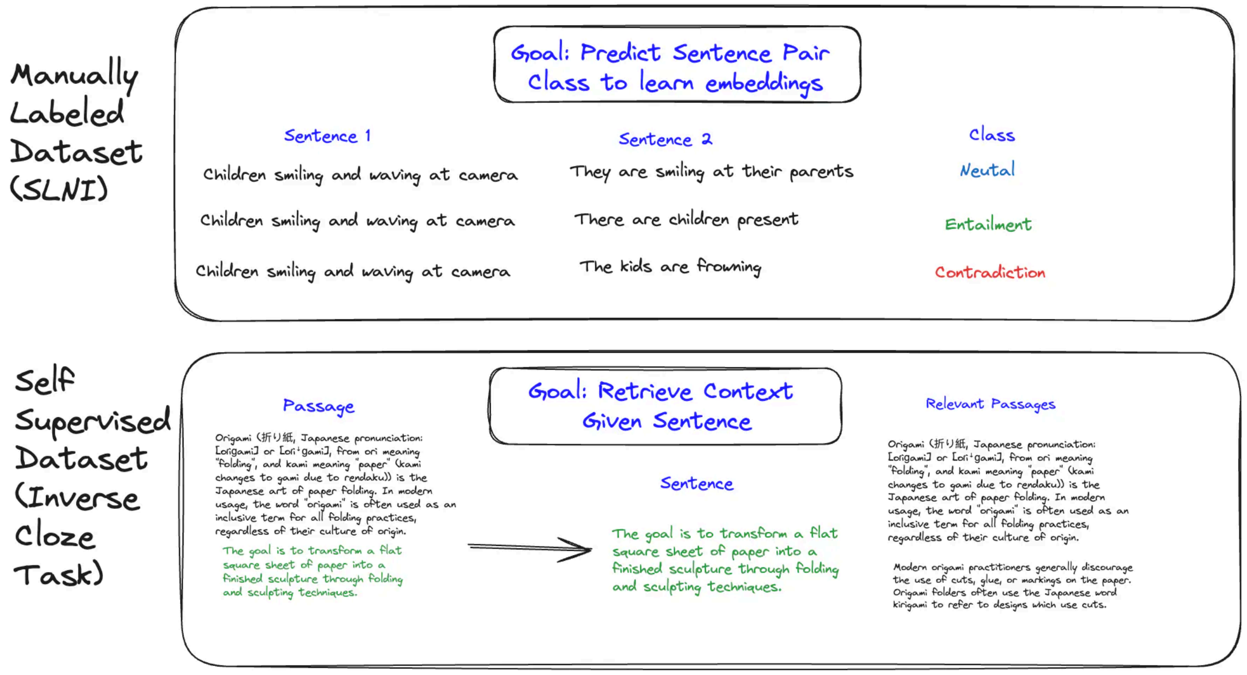
Although multilingual embedding models existed prior to E5, E5 itself stands out for its **unique training dataset, training processes, and performance.**

Dataset Creation and Training Process

The researchers behind E5 pointed out that prior models were developed using two kinds of datasets:

1. **Manually labeled datasets,** like MS-MARCO or SNLI, which are high quality but expensive to scale upward for better performance
2. **Synthetic dataset tasks** like the Inverse Cloze Task, which take advantage of text structure to generated labeled data, but are inherently lower quality [1, 2].

Additionally, most embedding models are not multilingual, and obtaining data that satisfies the above criteria demands a corresponding increase in dataset size.


Manually Labeled Dataset (SLNI)
Goal: Predict Sentence Pair Class to learn embeddings
Self Supervised Dataset (Inverse Cloze Task)
Goal: Retrieve Context Given Sentence
Passage
Sentence
Relevant Passages

Sentence 1	Sentence 2	Class
Children smiling and waving at camera	They are smiling at their parents	Neutal
Children smiling and waving at camera	There are children present	Entailment
Children smiling and waving at camera	The kids are frowning	Contradiction

Origami (折り紙, Japanese pronunciation: [origami] or [oɾiꜜgami], from ori meaning "folding", and kami meaning "paper" (kami changes to gami due to rendaku)) is the Japanese art of paper folding. In modern usage, the word "origami" is often used as an inclusive term for all folding practices, regardless of their culture of origin.
The goal is to transform a flat square sheet of paper into a finished sculpture through folding and sculpting techniques.

The goal is to transform a flat square sheet of paper into a finished sculpture through folding and sculpting techniques.

Origami (折り紙, Japanese pronunciation: [origami] or [oɾiꜜgami], from ori meaning "folding", and kami meaning "paper" (kami changes to gami due to rendaku)) is the Japanese art of paper folding. In modern usage, the word "origami" is often used as an inclusive term for all folding practices, regardless of their culture of origin.

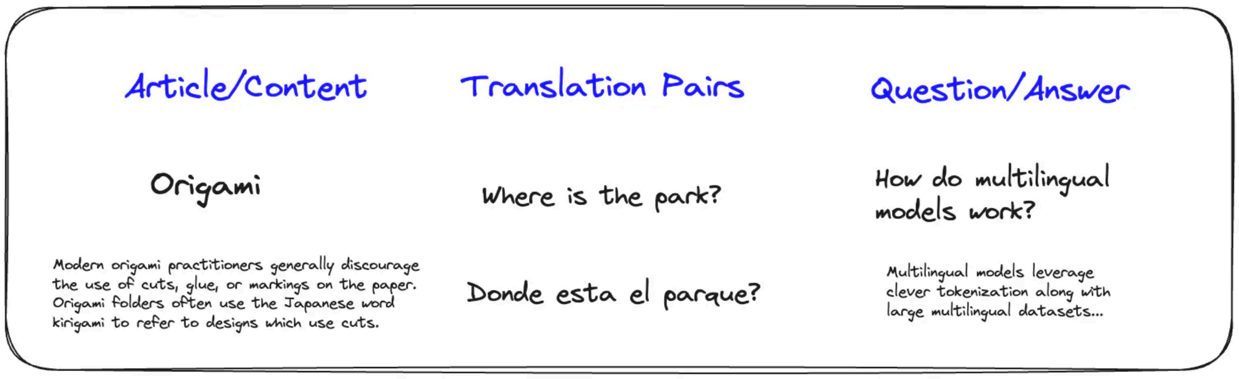
Modern origami practitioners generally discourage the use of cuts, glue, or markings on the paper. Origami folders often use the Japanese word kirigami to refer to designs which use cuts.

A comparison of manually labeled datasets and synthetic tasks possible for training models

So, they set out to curate a massive multilingual dataset which would allow for the best properties of both manually labeled datasets and synthetic ones [3].

This dataset was constructed by collecting naturally occurring text pairs across varying sources and languages. A text pair can be thought of as two texts that would normally exist together in the “wild”, such as titles of articles and corresponding sections, translation pairs, and questions/answer pairs.

Another important quality of these pairs is that they can be symmetric or asymmetric. You might have a pair of texts that are roughly the same length (such as translation pairs) or different lengths (such as document titles and paragraphs). The researchers decided to differentiate these as “query” texts and “passage” texts, so much so that they prefix the corresponding data when training the model, to better perform on certain retrieval tasks [3].


Article/Content
Translation Pairs
Question/Answer
Origami
Where is the park?
How do multilingual models work?
Modern origami practitioners generally discourage the use of cuts, glue, or markings on the paper. Origami folders often use the Japanese word kirigami to refer to designs which use cuts.
Donde esta el parque?
Multilingual models leverage clever tokenization along with large multilingual datasets...

Example dataset pairs that may be used in the training of E5

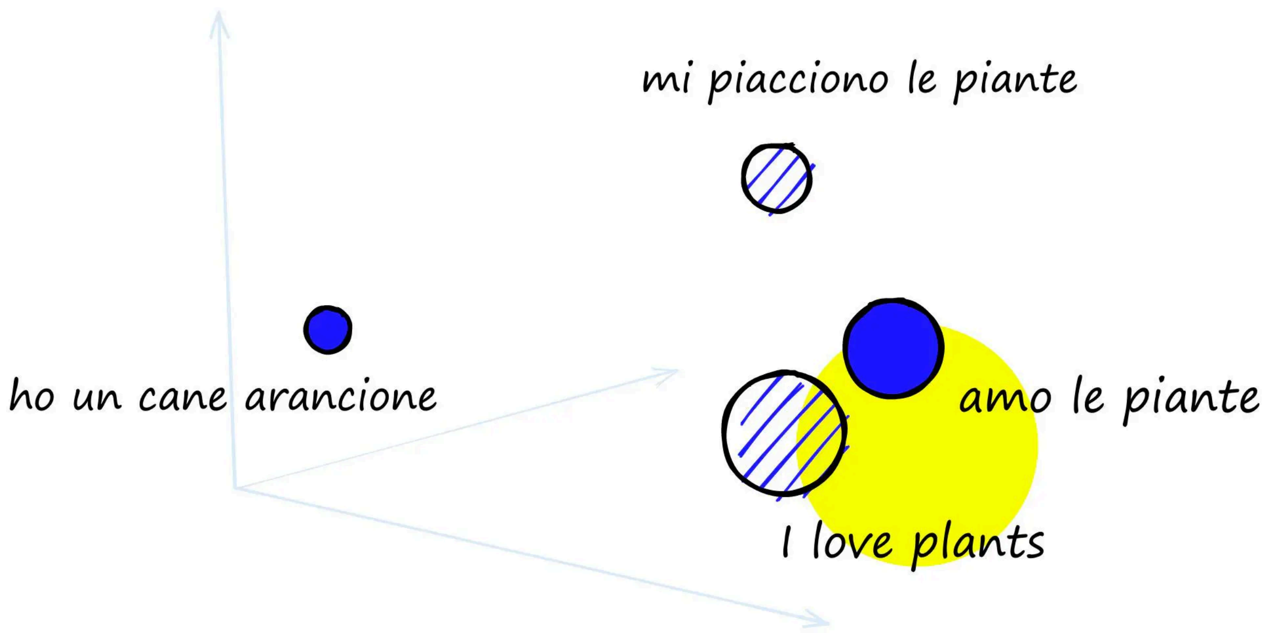
If we were to train embedding models for semantic search and retrieval, we’d want embeddings of these pairs to be close together in vector space. This would correspond to the closeness of their semantic meaning and relevance to one another, which are desirable properties to encode in vectors.

And since they occur across the internet in large numbers, it’s easy to grab and identify these pairs.

By using these pairs in the training data for embedding models, we can take advantage of the inherent relevance and semantic closeness of these objects, as a signal for embedding models.

And how was multilingualism achieved? Multilingual models like E5 leverage clever tokenization schemes that allow for all sorts of languages to be processed during training.

E5’s dataset and tokenization procedures ensure that any of around 100 languages can be embedded seamlessly with the model [4].



An example of sentences in different languages clustering in vector space

A great primer on multilingual training of embedding models is located here.

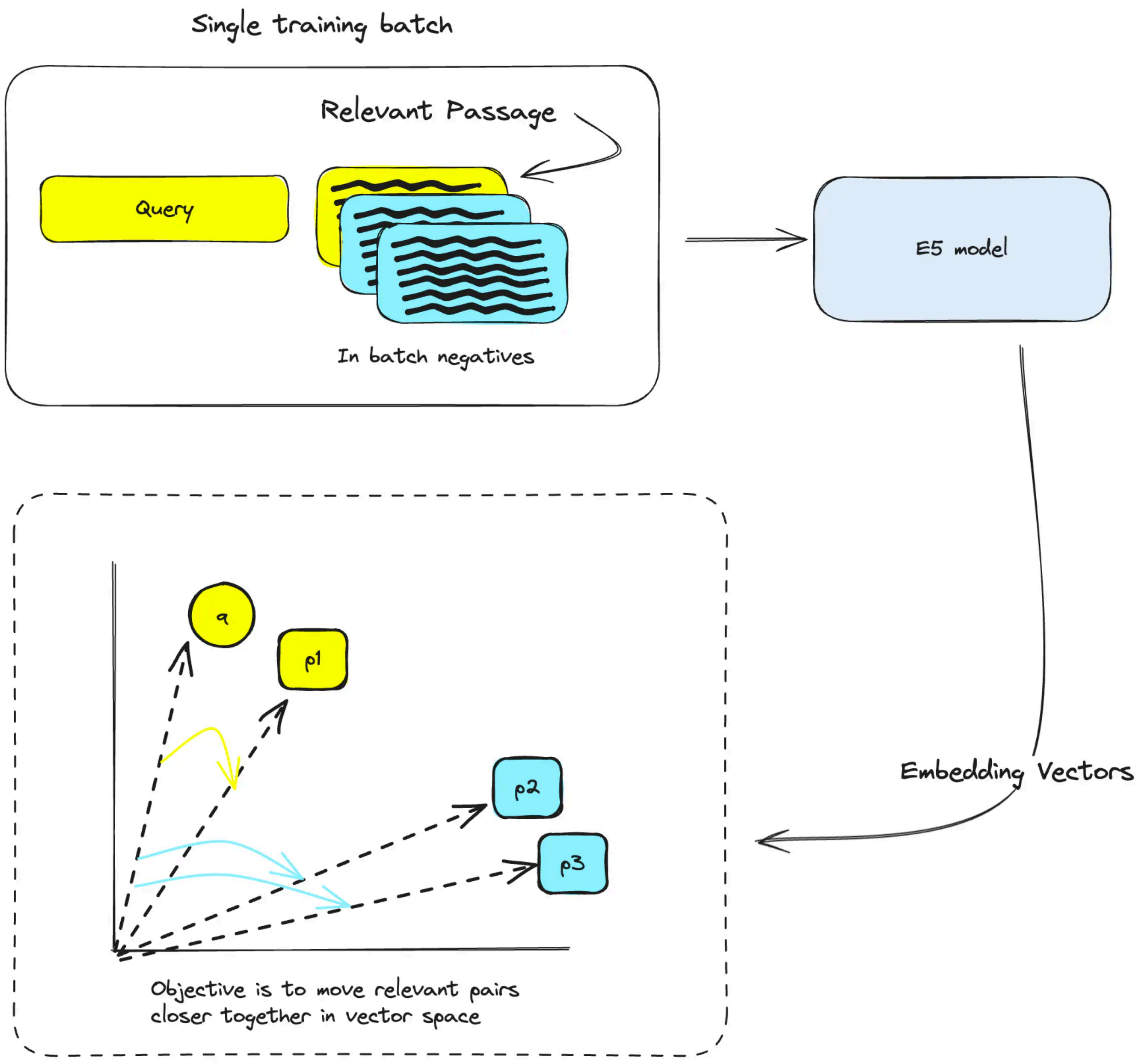
The Pinecone Inference API tokenizes input text for you, so there is no need to be concerned with implementation details for different languages; just pass the text to the endpoint.

After suitable datasets are gathered, the next step is model training.

Learning with Weakly Supervised Contrastive Pretraining

Remember those natural text pairs we mentioned? Finding labels in this fashion is called “weak supervision”, as they can still be noisy, as opposed to “supervised” datasets with clear labels.

The E5 training process begins by using a pretrained XLM-Roberta model, and trains additionally on a *billion* multilingual text pairs. Even though the XLM-Roberta model is already multilingual, this additional pretraining in a pairwise fashion is critical for downstream tasks [5].



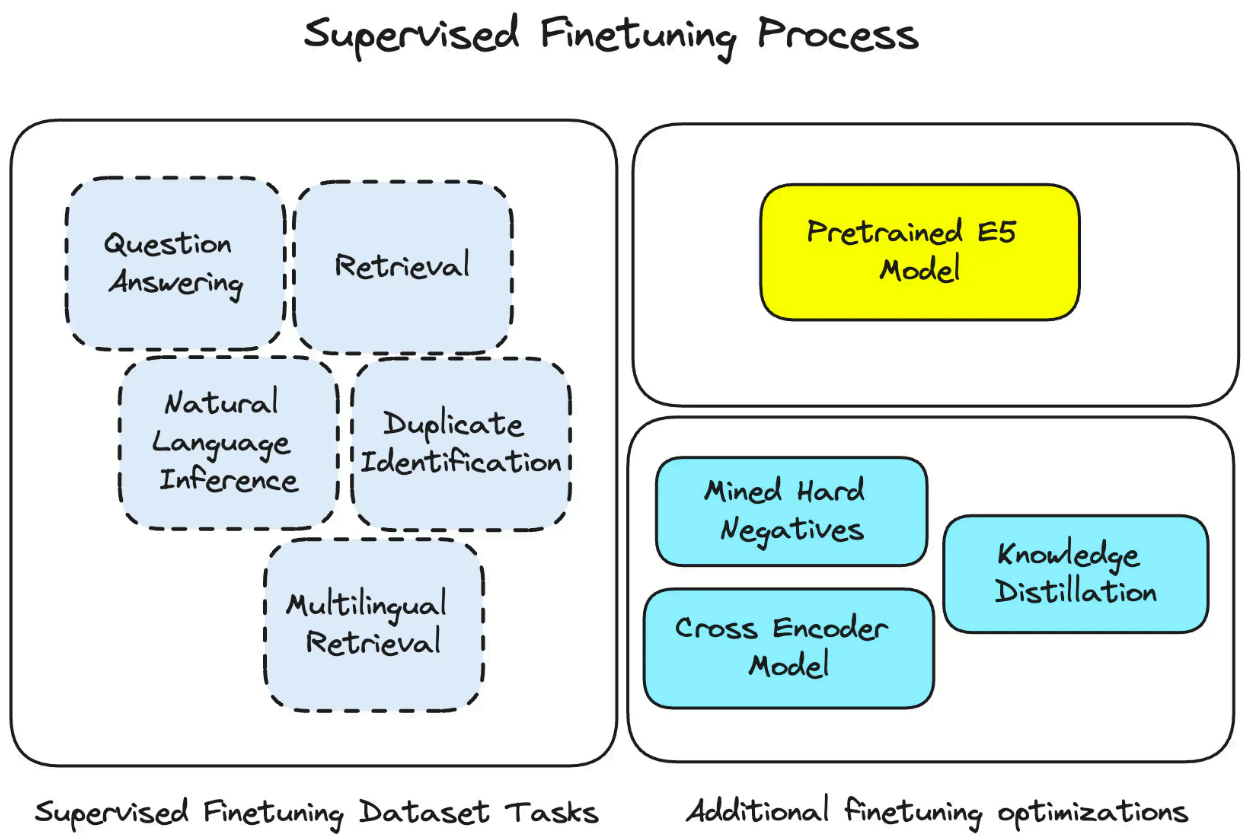
Visual of contrastive pretraining and the desired embedding outcome

“Contrastive pretraining” just means training so that the resulting embedding representations of two neighboring text pairs cluster together in vector space.

However, we also need to train with mismatched text pairs too, which we aim to push far away from one another in vector space. These are also known as in-batch negatives, as they are created with the data within a batch.

Together, this is “weakly supervised contrastive pretraining”, as the model learns on a huge amount of weakly labeled data in a contrastive fashion.

Specializing with Supervised Finetuning



The dataset tasks, modeling, and finetuning optimizations needed for multilingual E5 finetuning

After imbuing the model with the information gained from weakly supervised contrastive pretraining, the supervised finetuning step focuses the model on a curated multilingual labeled dataset spanning topics such as retrieval, semantic similarity, and question answering.

The dataset being used here is similar to what we'd expect for other manually labeled datasets, with the exception that it also spans several languages. Using human labeled data in this stage ensures the model is learning exactly what it needs to for these useful tasks. This dataset, alongside some clever techniques such as knowledge distillation and mined hard negatives really supercharges the multilingual capabilities of the model.

The net effect is that E5 provides great performance along with multilingual capability and relatively small size. Even at only 500 million parameters, E5 maintains competitive scores on the Massive Text Embedding Benchmark English leaderboard, and also on MIRACL, a benchmark dataset for multilingual retrieval [3]. This demonstrates that the model doesn't compromise its English performance due to multilingual training.

Tips for building with E5

Query and Passage Embeddings

Using the Pinecone Inference API with E5 requires specifying a required input type parameter. This input type parameter allows the API to serve different embeddings for queries and passages.

This is because during the training process, the researchers behind E5 prompted queries and passages differently to help differentiate the kind of embeddings learned used in different tasks.

You can enable this prompting by passing the corresponding *'query'* or *'passage'* to the input type parameter in Pinecone. **Generally, data upserted to Pinecone will need the "passage" input type, and queries will get the "query" input type.**

```python
from pinecone import Pinecone

pc = Pinecone(API_KEY)

# embed a single query this
query_embedding = pc.inference.embed(
  model="multilingual-e5-large",
  inputs=[query],
  parameters={"input_type": "query"}
)

# embed passages like this
passage_embeddings = pc.inference.embed(
  model="multilingual-e5-large",
  inputs=texts,
  parameters={"input_type": "passage"}
)
```

When using E5, it would be good to keep in mind the following rules of thumb:

- **Use Query and Passage for "asymmetric" tasks** that involve aligning a search or question with longer passages chunks, such as retrieval augmented generation, semantic search, or question answering
- **Use Query prefix exclusively for tasks that are "symmetric"**, such as bitext mining, paraphrase retrieval, similarity search, etc
- **Use Query prefix for using embeddings as features,** such as for classification or data labeling with Pinecone [4]
- **Skip tokenizing, but still chunk:** the Inference API handles tokenization for you.
- **Migrate your embeddings:** In general, you will need to use the same embedding model to embed your data and the vector you're using to query. If you have existing embeddings with a different model provider, you will have to re-embed those using E5
- **Know E5's model parameters**: Batch size is currently limited to 96, with an input size of 1024 tokens.
- **Stick with cosine similarity:** The E5 model was trained using cosine similarity based loss function, so use the "cosine" distance metric when creating your index [1].

A worked out example: Language Learning

With the launch of Pinecone Inference in public preview, we've provided an example notebook to help you get started using the Inference API and the multilingual E5 model. In the notebook, we add semantic search to a multilingual language learning dataset, to enable faster learning in English and Spanish sentences.

And there's more you can do with E5 and the Inference API:

- **Embed** your data for your current semantic search or RAG workflow with Pinecone
- **Enable** effortless multilingual search over your monolingual dataset
- **Expand** existing capabilities over your single language search application for global customers
- **Explore** the multilingual data you already have, without being limited by lack of translations
- **Enjoy** the ease of mind with a managed, multilingual API

We’re excited to put the power of a multilingual embedding model like E5 in your hands. It’s the first of several to come that will be made available through the Inference API. Stay tuned for more!

References

[1] L. Wang et al., “Text embeddings by weakly-supervised contrastive pre-training,” arXiv.org, https://arxiv.org/abs/2212.03533 (accessed Jun. 21, 2024).

[2] K. Lee, M.-W. Chang, and K. Toutanova, “Latent retrieval for weakly supervised open domain question answering,” arXiv.org, https://arxiv.org/abs/1906.00300 (accessed Jun. 21, 2024).

[3] L. Wang et al., “Multilingual E5 text embeddings: A technical report,” arXiv.org, https://arxiv.org/abs/2402.05672 (accessed Jun. 21, 2024).

[4] “Multilingual-E5-large Model Card ,” Hugging Face, https://huggingface.co/intfloat/multilingual-e5-large (accessed Jun. 21, 2024).

[5] A. Conneau et al., “Unsupervised cross-lingual representation learning at scale,” arXiv.org, https://arxiv.org/abs/1911.02116 (accessed Jun. 21, 2024).

Share:

Further Reading

LangGraph and Research Agents

Build Privacy-aware AI software using Pinecone

Interactive Introduction to Tokenization

Pinecone

Product

Overview

Documentation

Integrations

Trust and Security

Solutions

Customers

RAG

Semantic Search

Multi-Modal Search

Candidate Generation

Classification

Resources

Learning Center

Community

Pinecone Blog

Support Center

System Status

What is a Vector Database?

What is Retrieval Augmented Generation (RAG)?

Company

About

Partners

Careers

Newsroom

Contact

Legal

Customer Terms

Website Terms

Privacy

Cookies

Cookie Preferences