Blog Post

Ofer Mendelevitch

Ofer Mendelevitch leads developer relations at Vectara. He has extensive hands-on experience in machine learning, data science and big data systems across multiple industries, and has focused on developing products using large language models since 2019. Prior to Vectara he built and led data science teams at Syntegra, Helix, Lendup, Hortonworks and Yahoo! Ofer holds a B.Sc. in computer science from Technion and M.Sc. in EE from Tel Aviv university, and is the author of "Practical data science with Hadoop" (Addison Wesley).

Retrieval and Search

Retrieval Augmented Generation (RAG) Done Right: Retrieval

Share

power your retrieval matters!...

October 10 , 2023 by Ofer Mendelevitch

In part 1 of "RAG Done Right," we covered the importance of text chunking and how this apparently simple operation during data ingestion might have significant implications on the performance of your RAG pipeline.

We learned that text chunks need to be small enough so as to have focused semantic meaning and not too much noise (often one or more complete sentences), and can be augmented with sentences before or after during LLM generation to accomplish an optimal setting.

In this, we assumed that retrieving the best-matching text chunk at query time is a black box that just works. As it turns out, the embedding model that is used in neural (vector) search during the retrieval step can have a significant impact on overall RAG performance, and not all embedding models are created equal.

In this blog post, we introduce Vectara's new embedding model Boomerang and demonstrate the differences when using other embedding models such as the ones provided by OpenAI or Cohere.

Let's dig in.

What is an Embedding model?

An embedding model is a type of model that converts words, phrases, or even entire sentences into fixed-size vectors of numbers (floats). These vectors capture the semantic meaning of the words or phrases, making it easier for machine learning algorithms to understand and process natural language.

augmented generation.

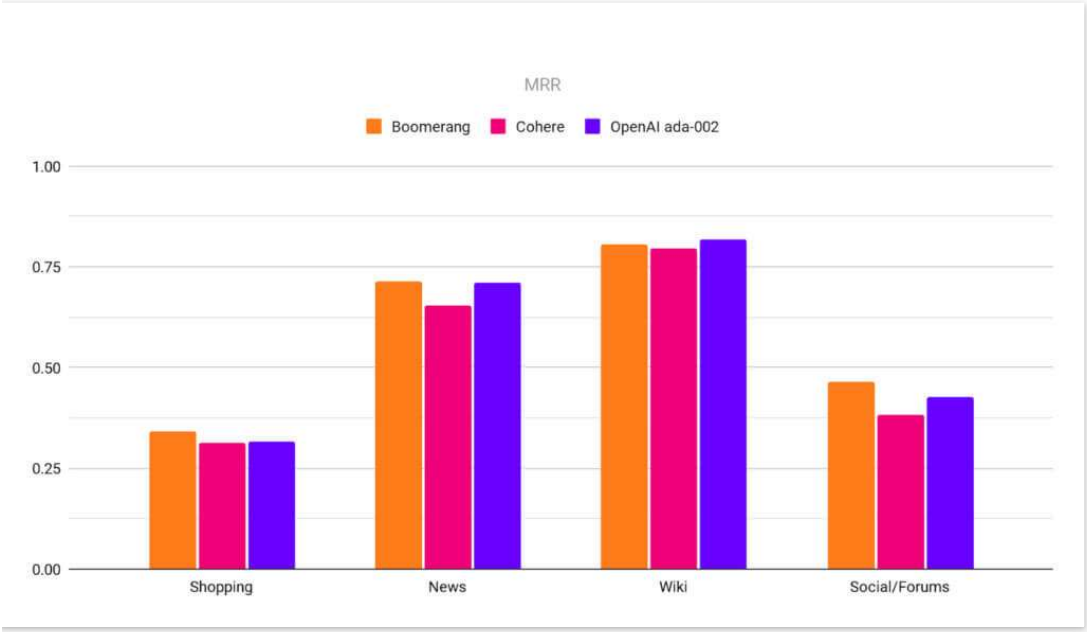
Specifically, embedding models are used as follows:

- During data ingestion, every chunk of text (see chunking) is converted into a vector and stored in a vector store alongside the text itself
- At query time, the user query (or prompt) is itself converted into a vector embedding and we use a form of similarity search to match similar vectors that have similar semantic meaning

There are two commercially available embedding models (OpenAI's Ada2 and Cohere's co:embed) and a few publicly available embedding models (SBERT, USE-QA, mContriever, etc).

At Vectara, we've been hard at work to improve the performance of our embedding model, and have recently announced our new embedding model: Boomerang.

Our retrieval benchmarks demonstrate significant gains in Boomerang over multiple benchmarks, including English and non-English datasets.



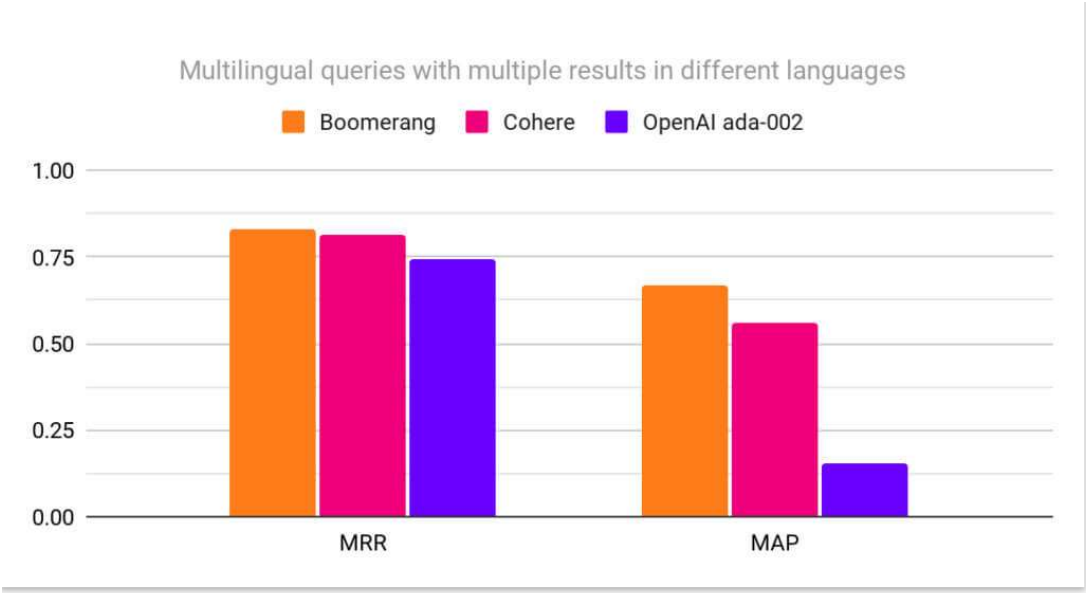


Figure 2: *Cross-lingual benchmark comparing Boomerang to OpenAI and Cohere.*

Retrieval benchmarks are important, but how does a good retrieval engine impact overall end-to-end results in RAG? Let's explore this with some example implementations.

Embedding model benchmarks

To compare some end-to-end RAG implementations, we'll construct a question-answering pipeline using the contents of the LLAMA2 paper, which has 77 pages.

We consider the following queries:

- What learning rate was used for pre-training?
- Was RLHF used?
- Which models are released for commercial use?
- Was red teaming used?

implementation for LlamaIndex users. For embedding we will use both the OpenAI and Cohere models.

First, we extract the text from the PDF file using the unstructured.io PDF extractor (during early experiments we noticed that PyPDF did rather poorly in extracting text from the PDF):

```
file_name = 'llama2.pdf'
elements = partition(fname)
parts = [str(t) for t in elements if
type(t)!=us.documents.elements.Title]
documents = [Document(text=' '.join(parts), metadata={})]
```

Then, we create a VectorStoreIndex using the SentenceWindowNodeParser node parser with a window size of 3 (in the example below using the OpenAIEmbeddings):

```
lm = ChatOpenAI(model_name='gpt-3.5-turbo-16k', temperature=0.0)
Embedding = OpenAIEmbeddings()
node_parser = SentenceWindowNodeParser.from_defaults(
        window_size=3,
        window_metadata_key="window",
        original_text_metadata_key="original_text",
)
nodes = node_parser.get_nodes_from_documents(documents)
ctx = ServiceContext.from_defaults(llm=llm, embed_model=embedding,
node_parser=node_parser)
openai_index = VectorStoreIndex(nodes, service_context=ctx)
```

Now that the index is constructed, we can create a query engine while using the MetadataReplacementPostProcessor post processor to ensure surrounding information is provided to the LLM for generation:

```
query_engine = openai_index.as_query_engine(similarity_top_k=5,
        node_postprocessors=[
```

```
    )
response = query_engine.query(query)
```

Next, we construct a similar RAG pipeline with Vectara. Here we simply generate a VectaraIndex and provide the same extracted document text as input. The text is indexed into Vectara and embedded using Boomerang.

```
vectara_index = VectaraIndex.from_documents(documents)
```

Then we can directly query the Vectara index:

```
response = str(vectara_index.as_query_engine(similarity_top_k=5,
                                             n_sentences_before=3,
n_sentences_after=3).query(query))
```

We run each of our queries in 3 languages: English, Hebrew, and Turkish, and the results are shown in tables 1, 2, and 3, respectively. See the full code in this notebook for additional languages.

Question	Cohere	OpenAI	Vectara
What learning rate was used for pre-training?	The learning rate used for pre-training is a constant learning rate of 10^-6.	The learning rate used for pre-training is not mentioned in the given context information.	The learning rate used for pre-training was 5 × 10−6 for the 70B parameter Llama 2-Chat and 1 × 10−5 for the rest.
Was RLHF used?	Yes, RLHF (Reinforcement Learning from Human	Yes, RLHF (Reinforcement Learning from Human	Yes, RLHF (Rejection Sampling fine-tuning and PPO

	context information.	context information.	
Which models are released for commercial use?	Llama 2 is the model that has been released for commercial use.	Llama 2 is the model that is released for commercial use.	Llama 2 is the model that is released for commercial use.
Was red teaming used?	Yes, red teaming was used in the study.	Yes, red teaming was used as a proactive risk identification method in the development of the LLMs	Yes, red teaming was used in the context described.

Table 1: *Results in English*

Question	Cohere	OpenAI	Vectara
What learning rate was used for pre-training? **באיזה קצב למידה נעשה שימוש בהכשרה מוקדמת?**	According to the information in the context, there is no specific information on the rate of use of early training.	The rate at which pretraining is used is not mentioned in the given information	The learning rate schedule used for pretraining is not mentioned in the given context information.

האם נעשה שימוש ב-RLHF?	mentioned in the context information	Sampling fine-tuning) is one of the algorithms explored in the context.	Learning from Human Feedback) is mentioned in the context information.
Which models are released for commercial use? **אילו דגמים משוחררים לשימוש מסחרי?**	The context information does not provide any specific information about released models for commercial use	Noam Shazeer's "Fast transformer decoding: One write-head is all you need" from 2019 and "Glu variants improve transformer" from 2020, Gabriel Synnaeve's "Growing up together: Structured exploration for large action spaces" from 2019, Yarden Tal, Inbal Magar, and Roy Schwartz's "Fewer errors, but more stereotypes? ... (shortened for display) are all released	Llama 2, Llama 2-Chat, and their variants with 7B, 13B, and 70B parameters are released for commercial use

Was red teaming used? **האם נעשה שימוש בצוות אדום?**	It is not possible to determine whether a red team was used based on the given information.	According to the information in the context, there is no specific information or answer to the question of whether a red team was used	Yes, red teaming was conducted as part of the safety investigations and efforts. Various groups of internal employees, contract workers, and external vendors, including domain experts in cybersecurity, election fraud, social media misinformation, legal, policy, civil rights, ethics, software engineering, machine learning, responsible AI, and creative writing, were involved in the red teaming exercises. These exercises aimed to identify proactive risks and patterns in order to improve model safety. The red team participants were given risk

			interactions with the language model, and they annotated various attributes, including risk areas and degree of risk.

Table 2: *Results in Hebrew*

Question	Cohere	OpenAI	Vectara
What learning rate was used for pre-training? **Ön eğitim için hangi öğrenme oranı kullanıldı?**	No information is given about the learning rate	The learning rate used for pretraining is not specified.	The learning rate used for pre-training is stated to be 5 × 10−6 for Llama 2-Chat with 70B parameters, where the maximum learning rate is 5 × 10−6, and 1 × 10−5 for the rest.
Was RLHF used? **RLHF kullanıldı mı?**	Yes, RLHF was used in the study mentioned in the context information	Yes, RLHF (Reinforcement Learning from Human Feedback) was used in the study mentioned in the	Yes, RLHF (Rejection Sampling fine-tuning) was used in the iterative fine-tuning process of the RLHF models.

Which models are released for commercial use? **Hangi modeller ticari kullanıma sunuluyor?**	MPT models, Falcon models and Vicuna models are available for commercial use.	Llama 2 model is being released to the general public for research and commercial use	Llama 2 and Llama 2-Chat models are available for commercial and research purposes.
Was red teaming used? **kırmızı takım kullanıldı mı?**	There is no information in the given context about whether the red team was used or not.	Yes, the red teaming was performed to further understand and improve model safety.	Yes, the red team was used.

Table 3: *results in Turkish*

As we can see – in English, even though some of the responses differ slightly between Cohere, OpenAI or Vectara, the responses to each of the 4 questions are reasonably good across the board with OpenAI, Cohere and Vectara. The only exception is the response to "What learning rate was used for pre-training?" with OpenAI which is not as good as with Cohere or Vectara. This reflects the fact that the embedding models work pretty well in English, and retrieve relevant information from the paper.

For Hebrew and Turkish – things are different.

Let's look first at the query "Which models are released for commercial use?". In Hebrew, for both the OpenAI and Cohere embedding model, the RAG pipeline is not able to answer this question, whereas Vectara's Boomerang model does pretty well, with an accurate response from the RAG pipeline "Llama 2, Llama 2-Chat, and their variants with 7B, 13B, and 70B parameters are released for commercial use".

Forth both OpenAI and Cohere the matching text chunks are:

"Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, and Omer Levy. "

"Kilem L. Gwet."

"Yarden Tal, Inbal Magar, and Roy Schwartz."

"Noam Shazeer. "

"Noam Shazeer."

Clearly, none of these chunks are relevant for the question posed. In contrast, for Vectara, the first most relevant chunk is:

"We are releasing the following models to the general public for research and commercial use."

...which is clearly relevant.

As another example, let's look at the question "Was red teaming used?" in Turkish.

Here the RAG setup with Cohere fails with

"There is no information in the given context about whether the red team was used or not."

In this case, both OpenAI and Vectara successfully pull the relevant information to then respond correctly to the question.

Conclusions

Creating a robust RAG pipeline that provides good responses, in multiple languages, is often more complicated than it initially appears,

putting it all together.

In this blog post we witnessed how Vectara's Boomerang model, integrated into Vectara's "RAG as a service" architecture, helps our users build effective GenAI applications. When compared to OpenAI and Cohere's embedding models Boomerang is on par with these other models in English but seems to outperform those models in some examples when using other languages like Hebrew or Turkish.

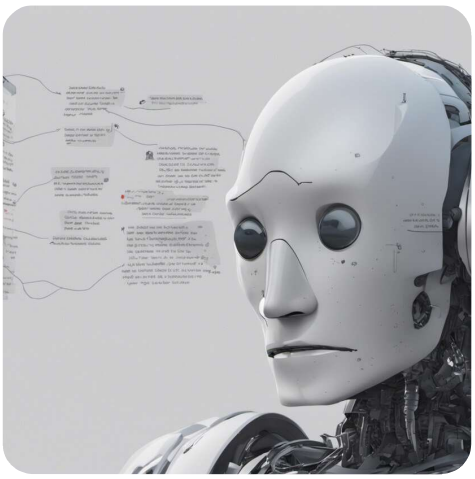
To try Boomerang with Vectara:

1. Sign up for a free account if you don't have one already
2. Follow the quickstart guide to create a corpus and API key. Boomerang is enabled by default for new corpora.
3. Ingest your data into the corpus using Vectara's Indexing API or use the open source vectara-ingest project.

If you need help, check out our forums and Discord server.

The full code for this blog is available here.

Recommended Content



September 13, 2023 by Ofer Mendelevitch & Logan Markewich | 6 min Read

Generation (RAG) Done Right: Chunking

September 20, 2023 by Ofer Mendelevitch | 7 min Read

– Vectara's New and Improved Retrieval Model

September 26, 2023 by Suleman Kazi & Vivek Sourabh | 9 min Read

CUSTOMER STORIES

SonoSim "Success Story"

SonoSim is dedicated to transforming medical care through ultrasound education & training. SonoSim needed to build an advanced search interface for question-answering to allow users to find content across multiple corpora collections. The team needed the ability to filter out low search relevant results and only deliver actionable intelligence to their end users. However, finding exactly what they need, on-demand, as an instructor or learner has become an increasing challenge as the SonoSim libraries of content and scanning cases continue to expand. With Vectara, SonoSim was able to demonstrate the value of AI for their central use case, increase efficiency and adoption of their training platform, and much more!

Get the Success Story →