(/)Jay Alammar (/)
Visualizing machine learning one concept at a time.
@JayAlammar (https://twitter.com/JayAlammar) on Twitter. YouTube Channel (https://www.youtube.com/channel/UCmOwsoHty5PrmE-3QhUBfPQ)

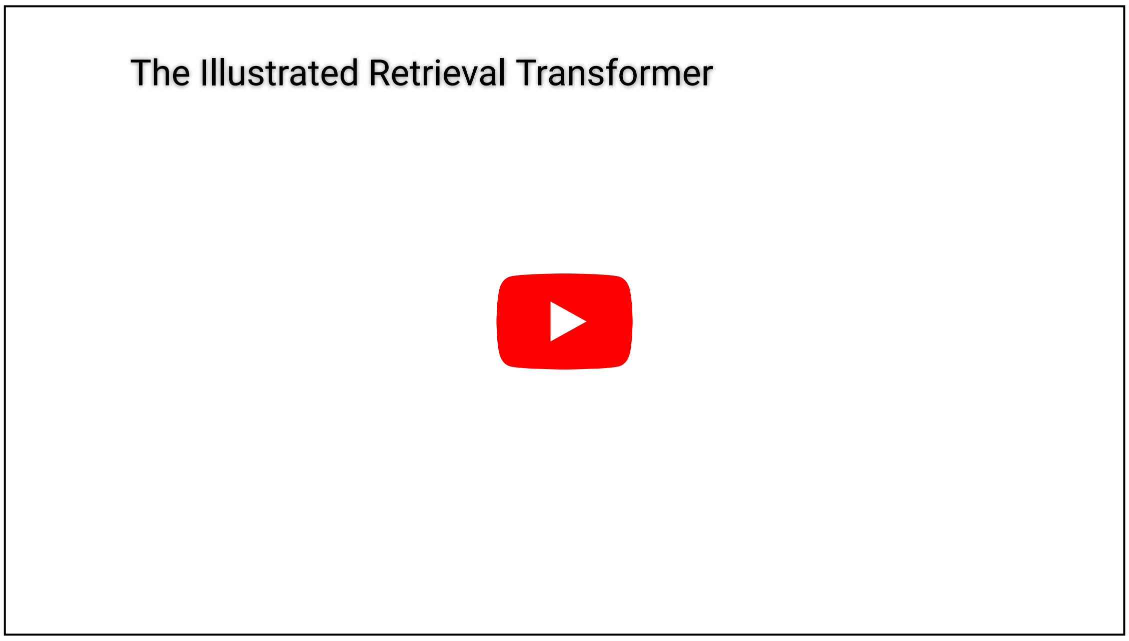
Blog (/) About (/about)

The Illustrated Retrieval Transformer

Discussion: Discussion Thread (https://github.com/jalammar/jalammar.github.io/discussions/21) for comments, corrections, or any feedback.
Translations: Korean (https://chloamme.github.io/2022/01/08/illustrated-retrieval-transformer-korean.html), Russian (https://habr.com/ru/post/648705/)

Summary: The latest batch of language models can be much smaller yet achieve GPT-3 like performance by being able to query a database or search the web for information. A key indication is that building larger and larger models is not the only way to improve performance.

Video

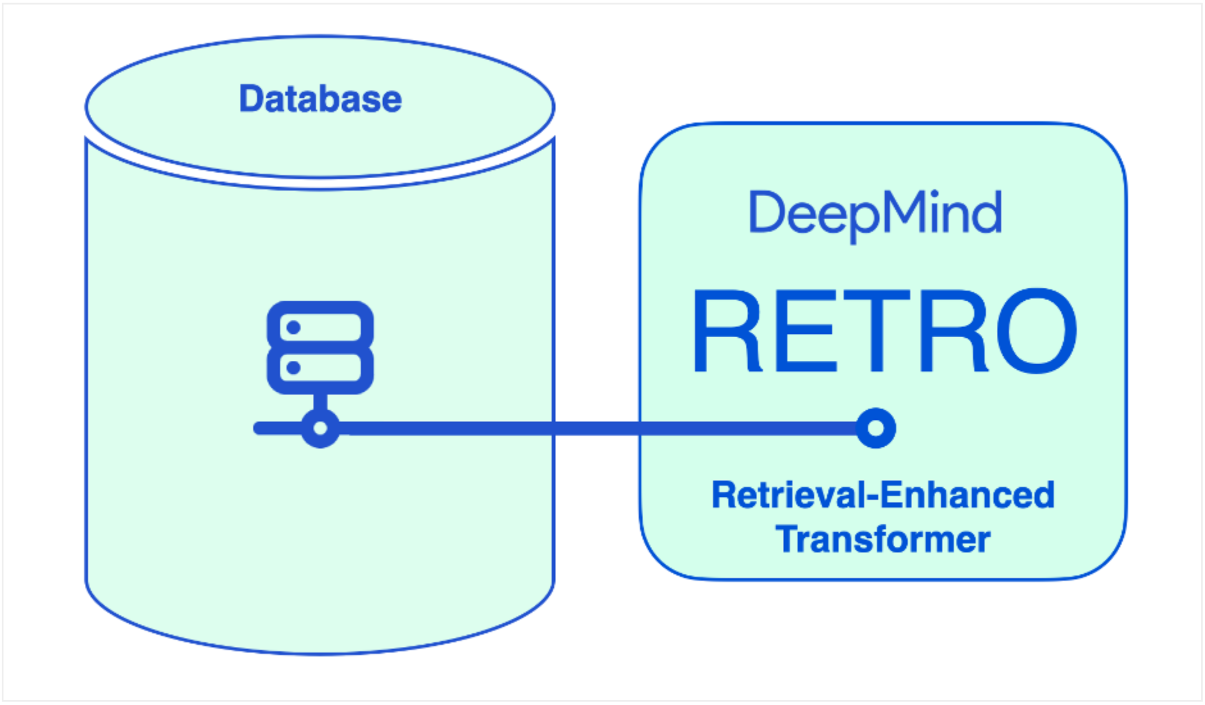


The last few years saw the rise of Large Language Models (LLMs) – machine learning models that rapidly improve how machines process and generate language. Some of the highlights since 2017 include:

- The original Transformer (/illustrated-transformer/) breaks previous performance records for machine translation.
- BERT (/illustrated-bert/) popularizes the pre-training then finetuning process, as well as Transformer-based contextualized word embeddings. It then rapidly starts to power Google Search (https://blog.google/products/search/search-language-understanding-bert/) and Bing Search (https://azure.microsoft.com/en-us/blog/bing-delivers-its-largest-improvement-in-search-experience-using-azure-gpus/).
- GPT-2 (/illustrated-gpt2/) demonstrates the machine's ability to write as well as humans do.
- First T5 (https://arxiv.org/abs/1910.10683), then T0 (https://huggingface.co/bigscience/T0pp) push the boundaries of transfer learning (training a model on one task, and then having it do well on other adjacent tasks) and posing a lot of different tasks as text-to-text tasks.
- GPT-3 (/how-gpt3-works-visualizations-animations/) showed that massive scaling of generative models can lead to shocking emergent applications (the industry continues to train larger models like Gopher (https://deepmind.com/research/publications/2021/scaling-language-models-methods-analysis-insights-from-training-gopher), MT-NLG (https://www.microsoft.com/en-us/research/blog/using-deepspeed-and-megatron-to-train-megatron-turing-nlg-530b-the-worlds-largest-and-most-powerful-generative-language-model/)…etc).

For a while, it seemed like scaling larger and larger models is the main way to improve performance. Recent developments in the field, like DeepMind's RETRO Transformer (https://deepmind.com/research/publications/2021/improving-language-models-by-retrieving-from-trillions-of-tokens) and OpenAI's WebGPT (https://openai.com/blog/improving-factual-accuracy/), reverse this trend by showing that smaller generative language models can perform on par with massive models if we augment them with a way to search/query for information.

This article breaks down DeepMind's RETRO (**R**etrieval-**E**nhanced **TR**ansf**O**rmer) and how it works. The model performs on par with GPT-3 despite being 4% its size (7.5 billion parameters vs. 185 billion for GPT-3 Da Vinci).



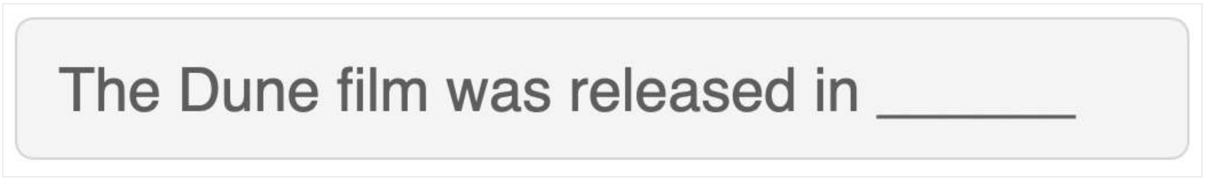
RETRO incorporates information retrieved from a database to free its parameters from being an expensive store of facts and world knowledge.

RETRO was presented in the paper Improving Language Models by Retrieving from Trillions of Tokens (https://arxiv.org/abs/2112.04426). It continues and builds on a wide variety of retrieval work (http://www.crm.umontreal.ca/2018/Langue18/pdf/Cheung.pdf) in (https://ai.facebook.com/blog/retrieval-augmented-generation-streamlining-the-creation-of-intelligent-natural-language-processing-models/) the (https://openreview.net/forum?id=HklBjCEKvH) research (https://arxiv.org/abs/2102.02557) community (https://openreview.net/forum?id=B184E5qee). This article explains the model and not what is especially novel about it.

Why This is Important: Separating Language Information from World Knowledge Information

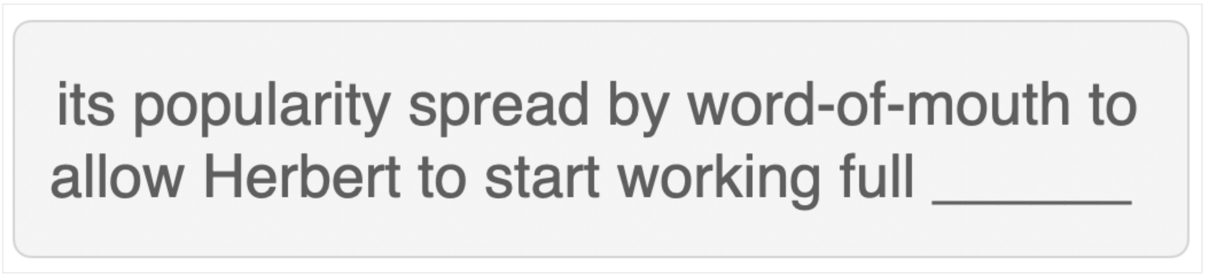
Language modeling trains models to predict the next word–to fill-in-the-blank at the end of the sentence, essentially.

Filling the blank sometimes requires knowledge of factual information (e.g. names or dates). For example:


The Dune film was released in ________

Input prompt: The Dune film was released in ....

Other times, familiarity with the language is enough to guess what goes in the blank. For example:

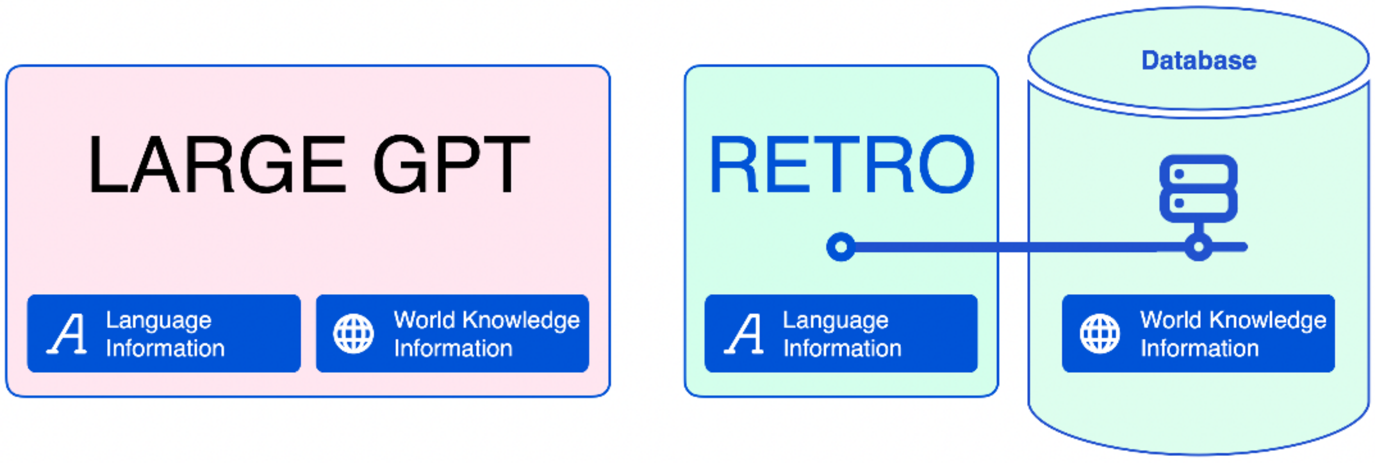


its popularity spread by word-of-mouth to allow Herbert to start working full _______

Input prompt: its popularity spread by word-of-mouth to allow Herbert to start working full ....

This distinction is important because LLMs encoded everything they know in their model parameters. While this makes sense for language information, it is inefficient for factual and world-knowledge information.

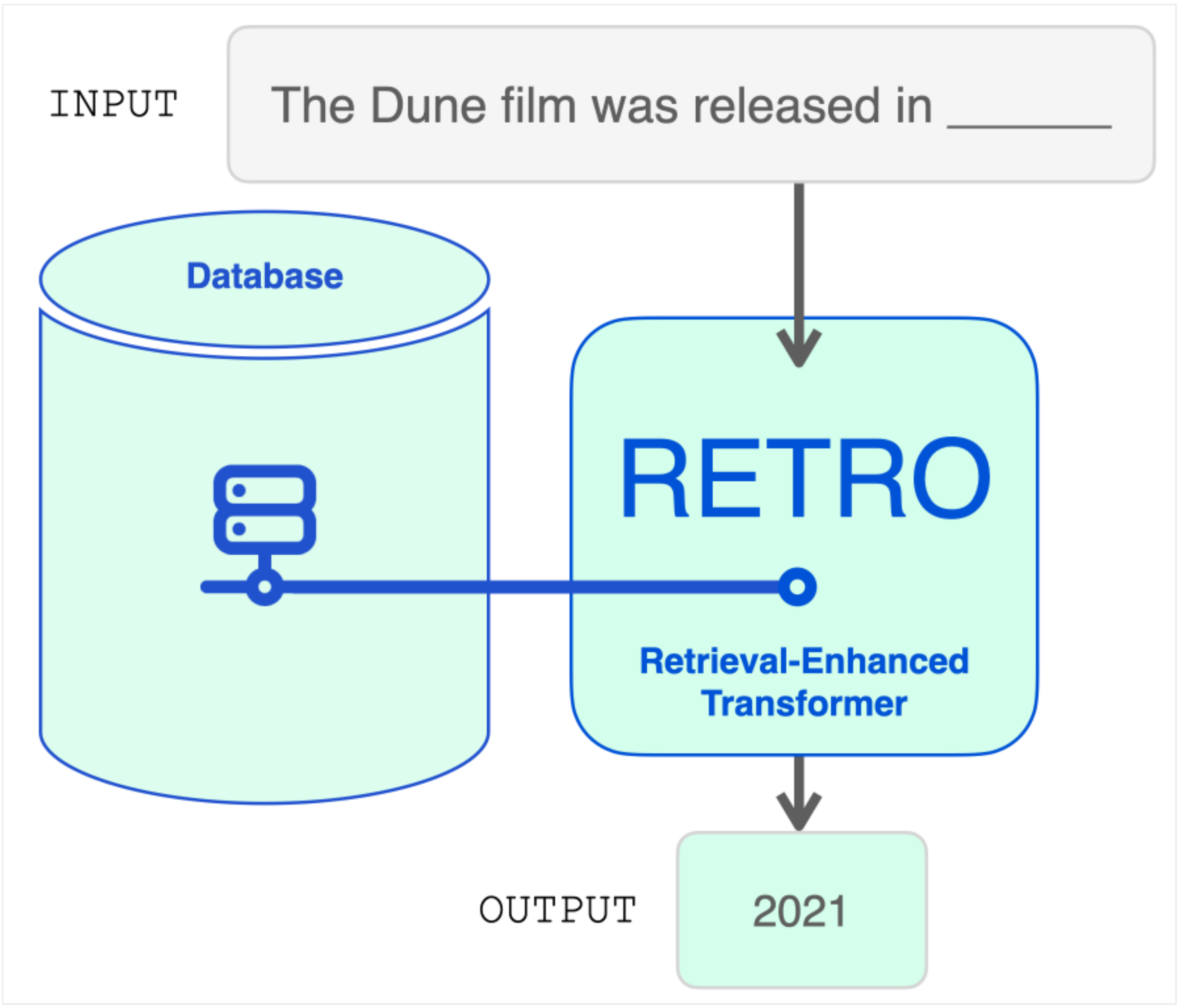
By including a retrieval method in the language model, the model can be much smaller. A neural database aids it with retrieving factual information it needs during text generation.



Aiding language models with retrieval methods allows us to reduce the amount of information a language model needs to encode in its parameters to perform well at text generation.

Training becomes fast with small language models, as training data memorization is reduced. Anyone can deploy these models on smaller and more affordable GPUs and tweak them as per need.

Mechanically, RETRO is an encoder-decoder model just like the original transformer. However, it augments the input sequence with the help of a retrieval database. The model finds the most probable sequences in the database and adds them to the input. RETRO works its magic to generate the output prediction.


INPUT
The Dune film was released in ______
Database
RETRO
Retrieval-Enhanced Transformer
OUTPUT
2021

RETRO utilizes a database to augment its input prompt. The prompt is used to retrieve relevant information from the database.

Before we explore the model architecture, let's dig deeper into the retrieval database.

Inspecting RETRO's Retrieval Database

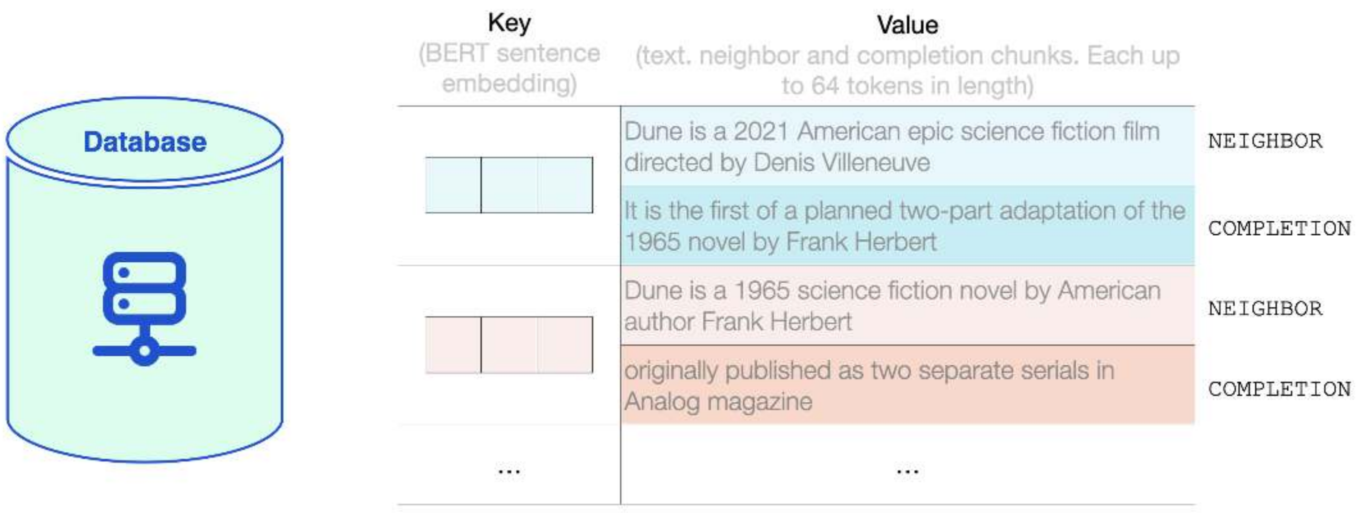
The database is a key-value store.

The key is a standard BERT sentence embedding.

The value is text in two parts:

1. Neighbor, which is used to compute the key
2. Completion, the continuation of the text in the original document.

RETRO's database contains 2 trillion multi-lingual tokens based on the *MassiveText* dataset. Both the neighbor and completion chunks are at most 64 tokens long.

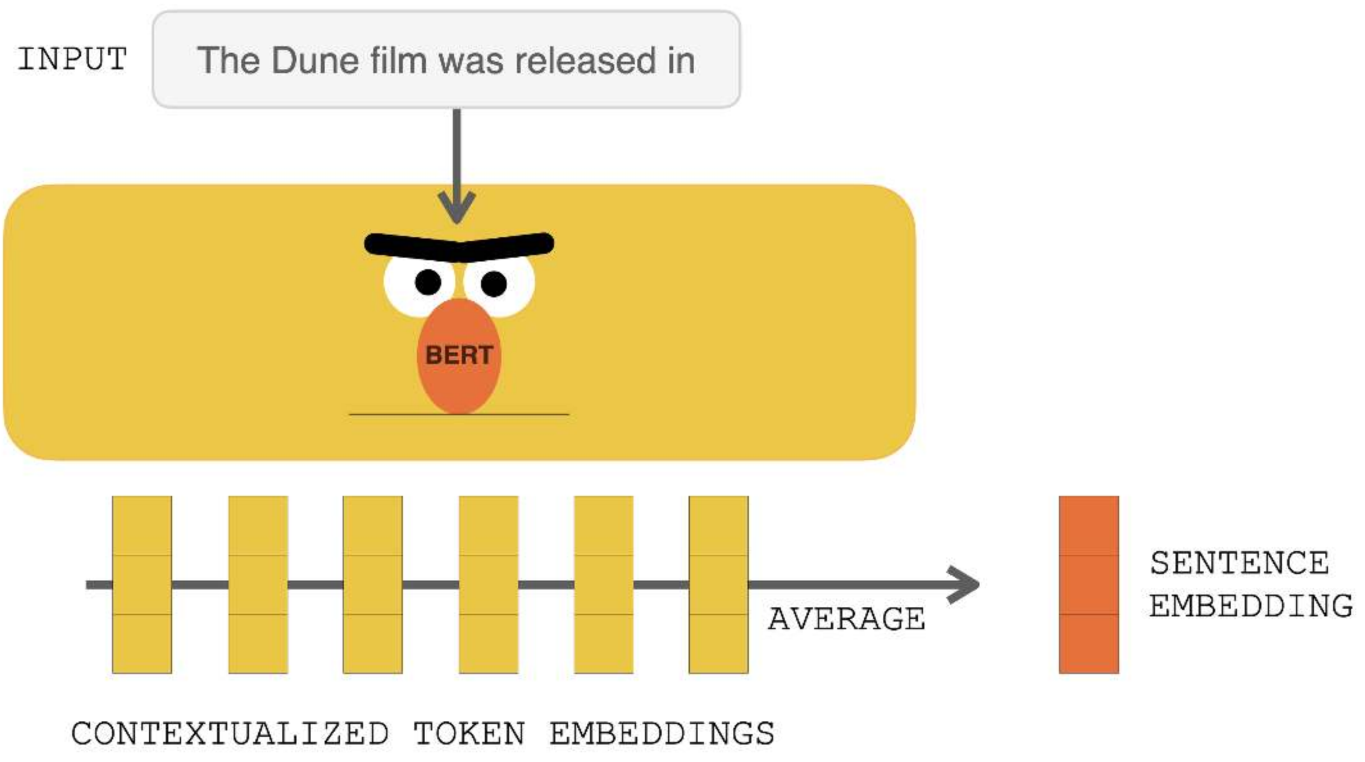

Database
Key
(BERT sentence embedding)
Value
(text. neighbor and completion chunks. Each up to 64 tokens in length)
Dune is a 2021 American epic science fiction film directed by Denis Villeneuve
NEIGHBOR
It is the first of a planned two-part adaptation of the 1965 novel by Frank Herbert
COMPLETION
Dune is a 1965 science fiction novel by American author Frank Herbert
NEIGHBOR
originally published as two separate serials in Analog magazine
COMPLETION
...
...

A look inside RETRO's database shows examples of key-value pairs in the RETRO database. The value contains a neighbor chunk and a completion chunk.

RETRO breaks the input prompt into multiple chunks. For simplicity, we'll focus on how one chunk is augmented with retrieved text. The model, however, does this process for each chunk (except the first) in the input prompt.

The Database Lookup

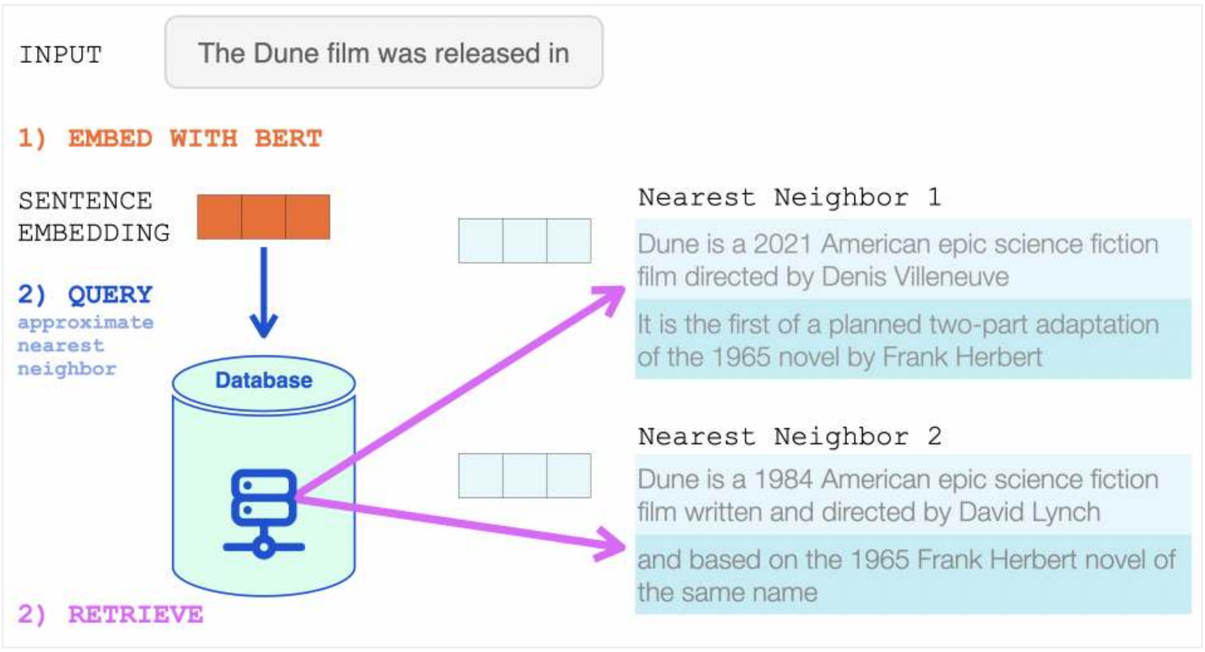
Before hitting RETRO, the input prompt goes into BERT. The output contextualized vectors are then averaged to construct a sentence embedding vector. That vector is then used to query the database.


INPUT
The Dune film was released in
BERT
AVERAGE
SENTENCE EMBEDDING
CONTEXTUALIZED TOKEN EMBEDDINGS

Processing the input prompt with BERT produces contextualized token embeddings. Averaging them produces a sentence embedding.

That sentence embedding is then used in an approximate nearest neighbor search (https://github.com/google-research/google-research/tree/master/scann (https://github.com/google-research/google-research/tree/master/scann)).

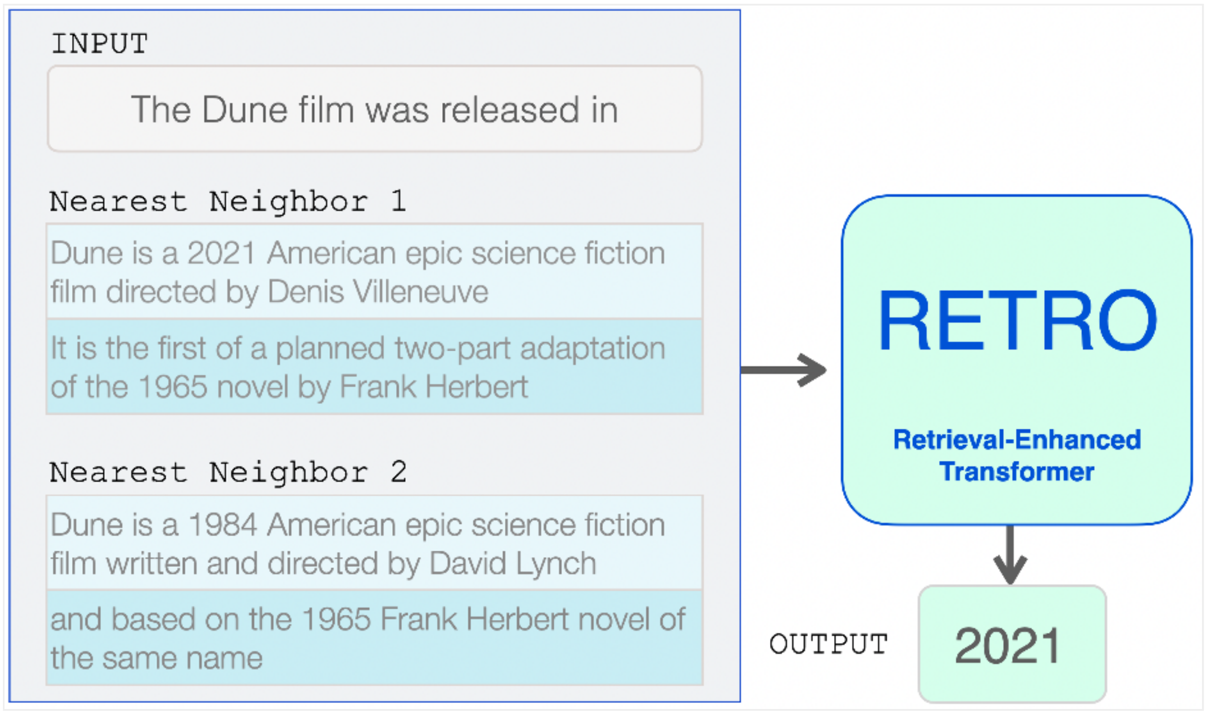
The two nearest neighbors are retrieved, and their text becomes a part of the input into RETRO.


INPUT
The Dune film was released in
1) EMBED WITH BERT
SENTENCE EMBEDDING
2) QUERY
approximate nearest neighbor
Database
2) RETRIEVE
Nearest Neighbor 1
Dune is a 2021 American epic science fiction film directed by Denis Villeneuve
It is the first of a planned two-part adaptation of the 1965 novel by Frank Herbert
Nearest Neighbor 2
Dune is a 1984 American epic science fiction film written and directed by David Lynch
and based on the 1965 Frank Herbert novel of the same name

The BERT sentence embedding is used to retrieve the nearest neighbors from RETRO's neural database. These are then added to the input of the language model.

This is now the input to RETRO. The input prompt and its two nearest neighbors from the database (and their continuations).

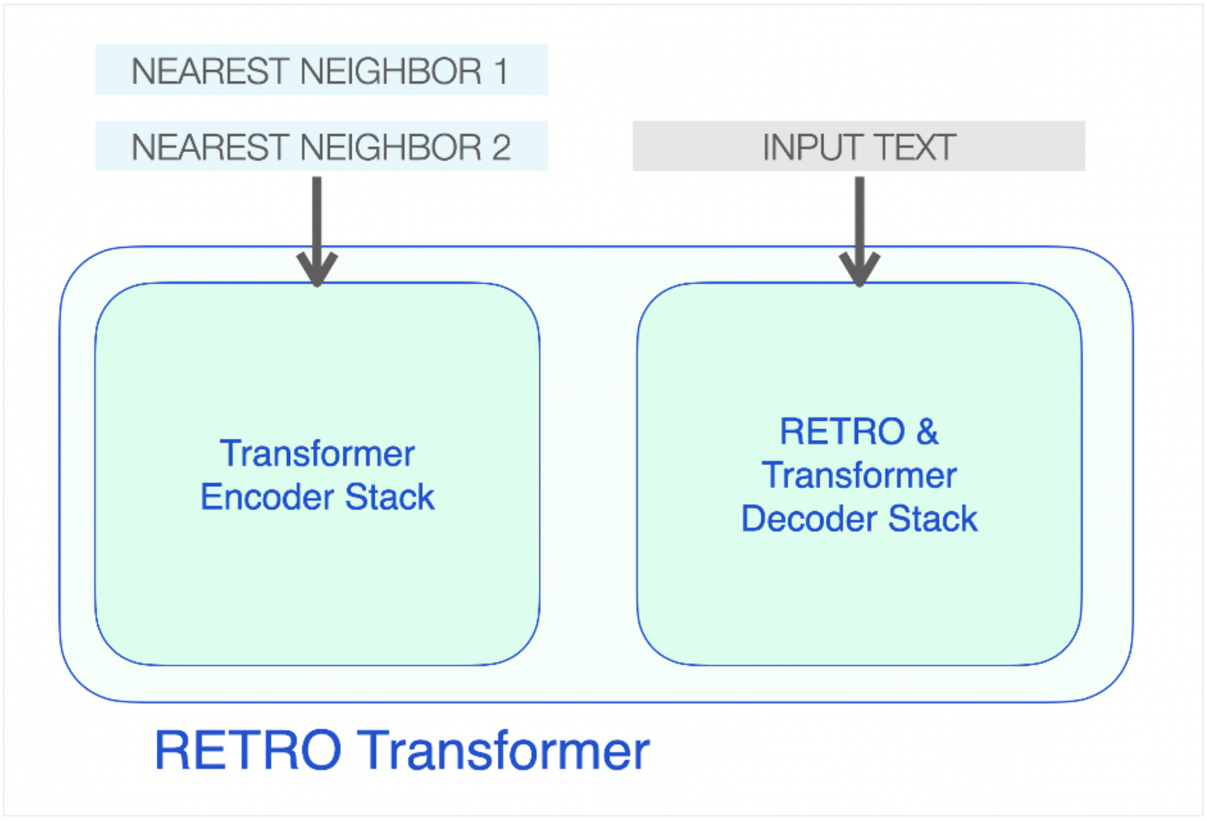
From here, the Transformer and RETRO Blocks incorporate the information into their processing.


INPUT
The Dune film was released in
Nearest Neighbor 1
Dune is a 2021 American epic science fiction film directed by Denis Villeneuve
It is the first of a planned two-part adaptation of the 1965 novel by Frank Herbert
Nearest Neighbor 2
Dune is a 1984 American epic science fiction film written and directed by David Lynch
and based on the 1965 Frank Herbert novel of the same name
RETRO
Retrieval-Enhanced Transformer
OUTPUT
2021

The retrieved neighbors are added to the input of the language model. They're treated a little differently inside the model, however.

RETRO Architecture at a High Level

RETRO's architecture is an encoder stack and a decoder stack.

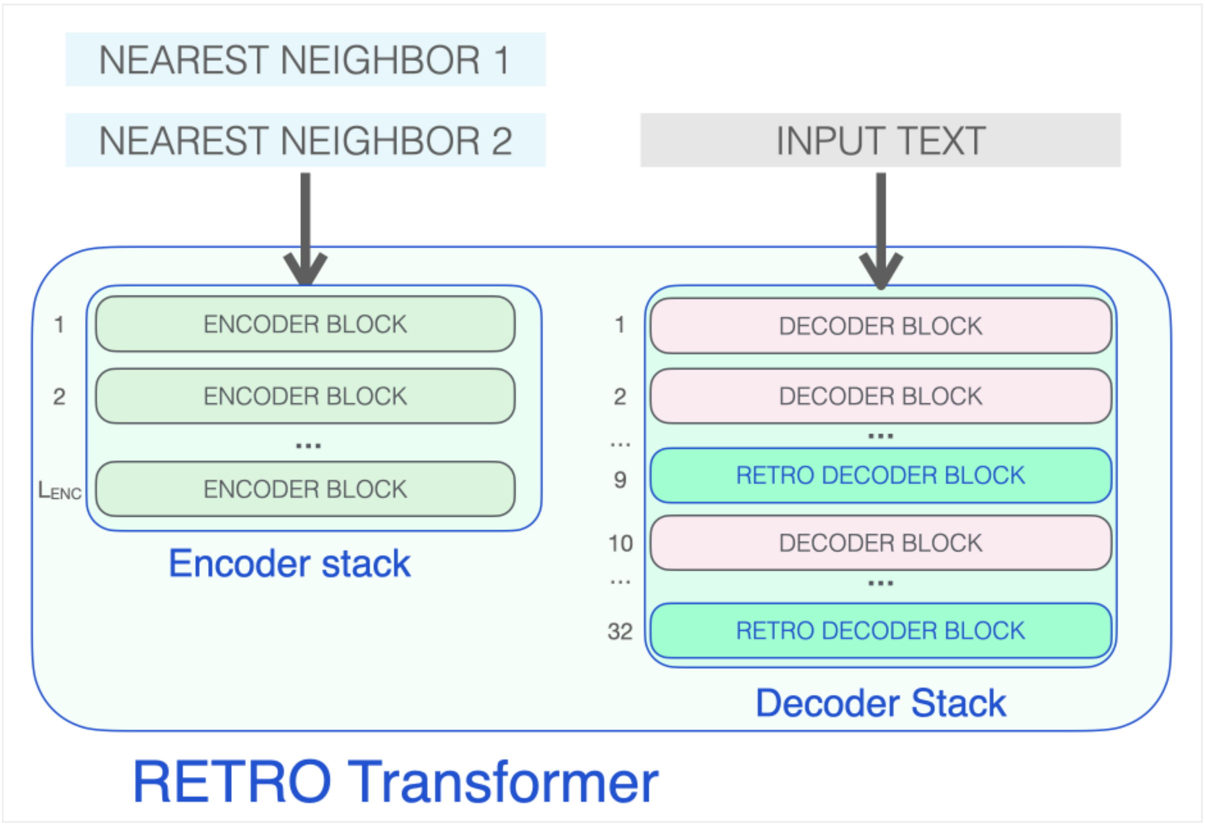


A RETRO transformer consists of an encoder stack (to process the neighbors) and a decoder stack (to process the input)

The encoder is made up of standard Transformer encoder blocks (self-attention + FFNN). To my best understanding, Retro uses an encoder made up of two Transformer Encoder Blocks.

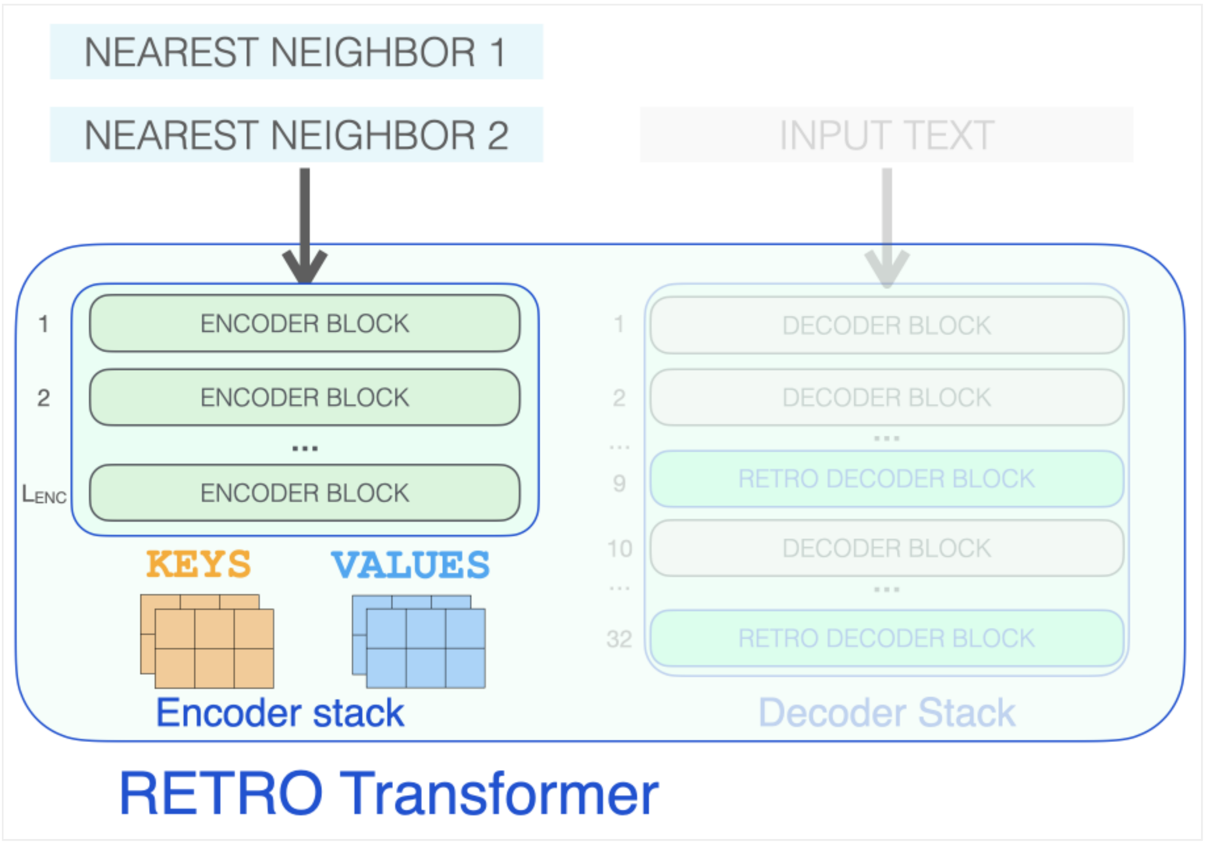
The decoder stack interleaves two kinds of decoder blocks:

- Standard transformer decoder block (ATTN + FFNN)
- RETRO decoder block (ATTN + Chunked cross attention (CCA) + FFNN)



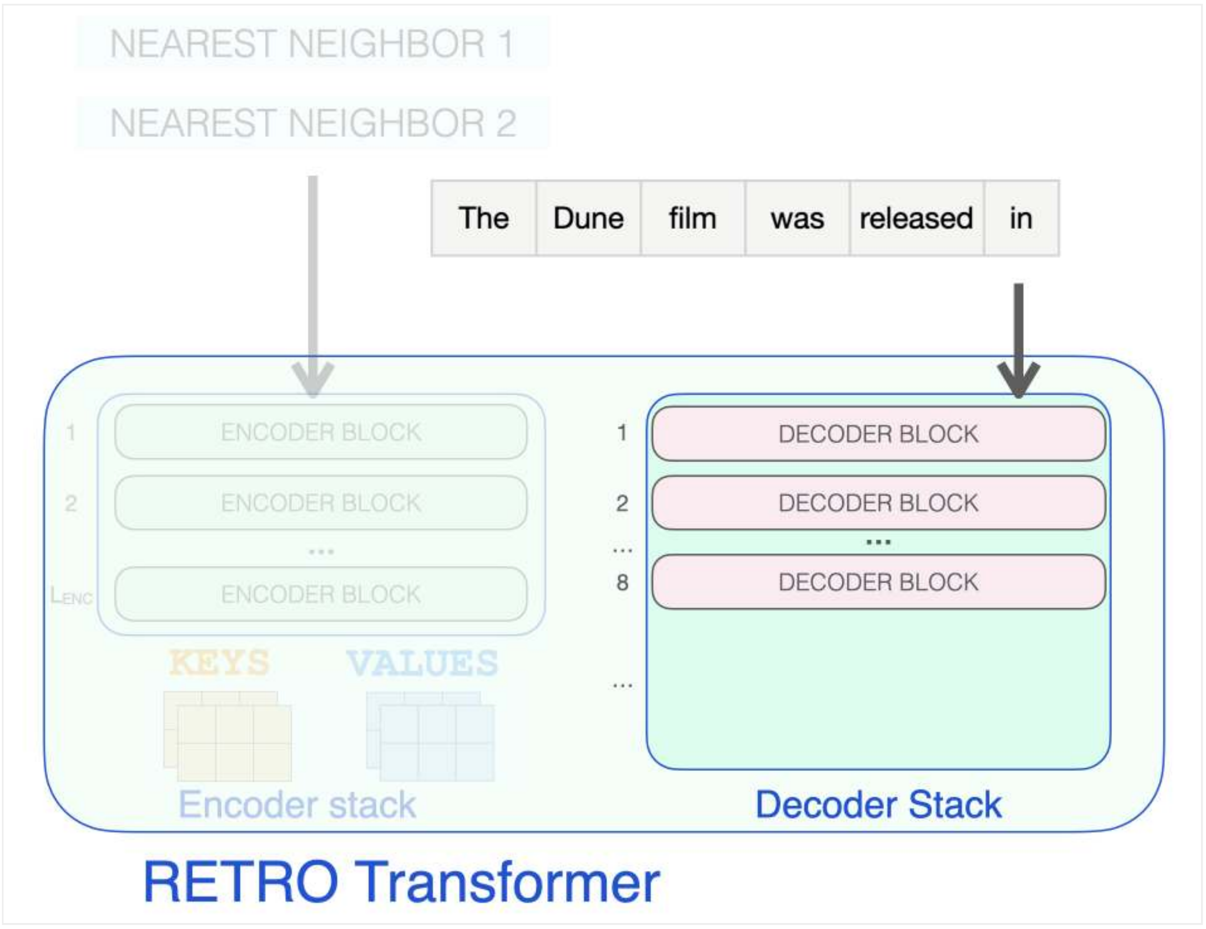
The three types of Transformer blocks that make up RETRO

Let's start by looking at the encoder stack, which processes the retrieved neighbors, resulting in KEYS and VALUES matrices that will later be used for attention (see The Illustrated Transformer (https://jalammar.github.io/illustrated-transformer/) for a refresher).



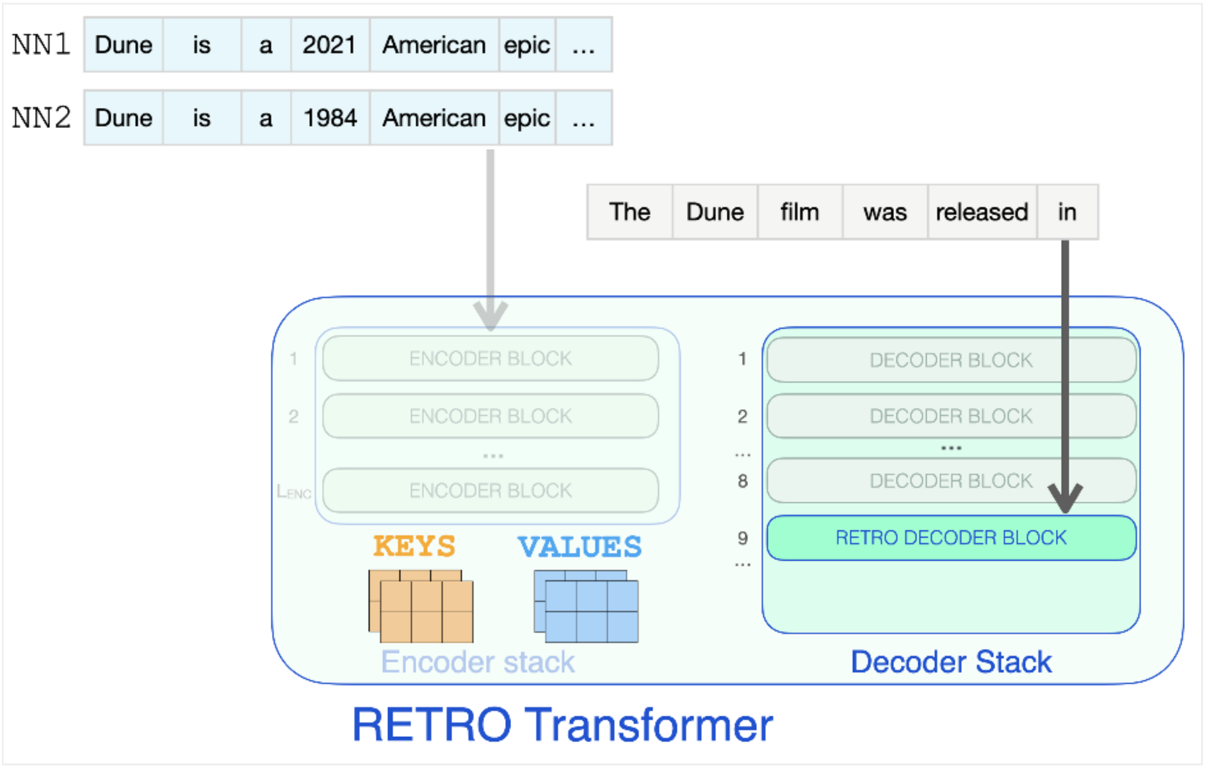
The encoder stack processes the retrieved neighbors resulting in KEYS and VALUE matrices

Decoder blocks process the input text just like a GPT would. It applies self-attention on the prompt token (causally, so only attending to previous tokens), then passes through a FFNN layer.



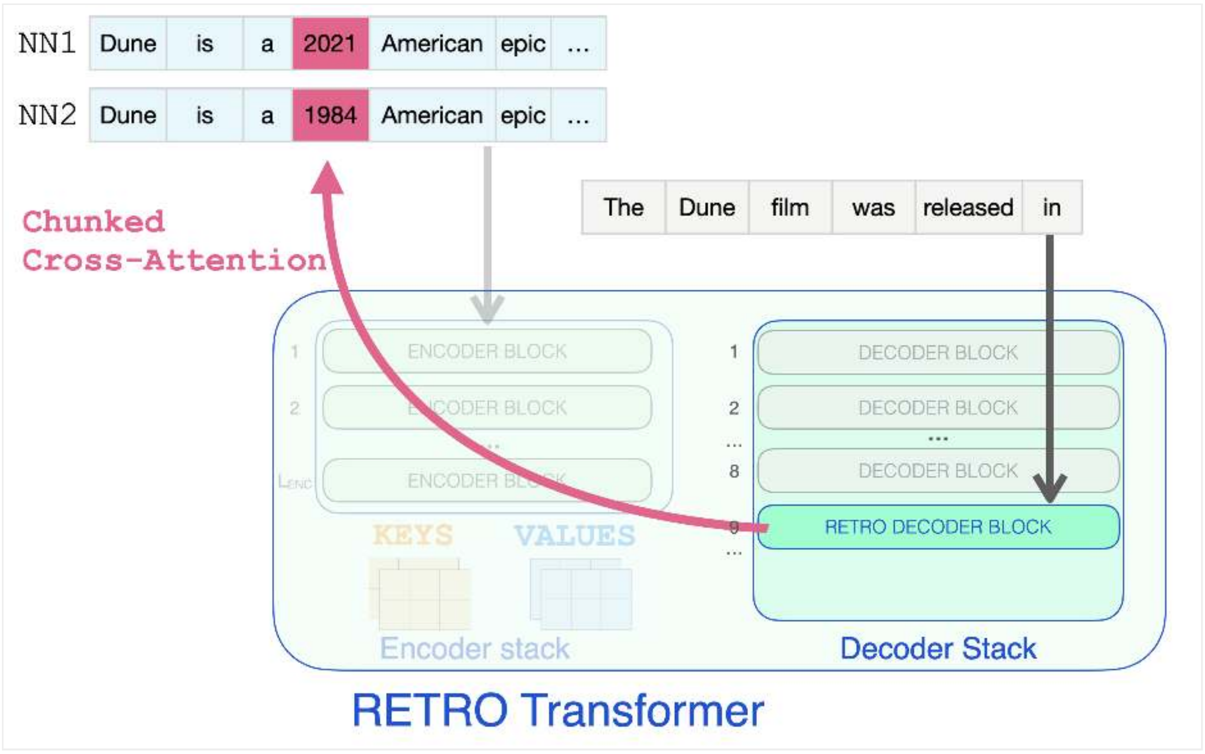
Input prompt passes through standard decoder block containing self-attention and FFNN layers

It's only when a RETRO decoder is reached do we start to incorporate the retrieved information. Every third block starting from 9 is a RETRO block (that allows its input to attend to the neighbors). So layers 9, 12, 15…32 are RETRO blocks. (The two smaller Retro models, and the Retrofit models have these layers starting from the 6th instead of the 9th layer).


NN1 Dune is a 2021 American epic ...
NN2 Dune is a 1984 American epic ...
The Dune film was released in
ENCODER BLOCK
DECODER BLOCK
RETRO DECODER BLOCK
KEYS
VALUES
Encoder stack
Decoder Stack
RETRO Transformer

Input prompt reaches RETRO Decoder block to start information retrieval

So effectively, this is the step where the retrieved information can glance at the dates it needs to complete the prompt.



RETRO Decoder block retrieving information from nearest neighbour chunks using Chunked Cross-Attention

Previous Work

Aiding language models with retrieval techniques has been an active area of research. Some of the previous work in the space includes:

- Improving Neural Language Models with a Continuous Cache (https://openreview.net/forum?id=B184E5qee)
- Generalization through Memorization: Nearest Neighbor Language Models (https://openreview.net/forum?id=HklBjCEKvH)
- Read the Retrieval Augmented Generation (https://ai.facebook.com/blog/retrieval-augmented-generation-streamlining-the-creation-of-intelligent-natural-language-processing-models/) blog from Meta AI and go through Jackie Chi Kit Cheung's lecture on Leveraging External Knowledge in Natural Language Understanding Systems (http://www.crm.umontreal.ca/2018/Langue18/pdf/Cheung.pdf)
- SPALM: Adaptive Semiparametric Language Models (https://arxiv.org/abs/2102.02557)
- DPR: Dense Passage Retrieval for Open-Domain Question Answering (https://aclanthology.org/2020.emnlp-main.550/)
- REALM: Retrieval-Augmented Language Model Pre-Training (https://arxiv.org/abs/2002.08909)
- FiD: Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering (https://aclanthology.org/2021.eacl-main.74/)
- EMDR: End-to-End Training of Multi-Document Reader and Retriever for Open-Domain Question Answering (https://arxiv.org/abs/2106.05346)
- BlenderBot 2.0: Internet-Augmented Dialogue Generation (https://arxiv.org/abs/2107.07566)

Please post in this thread (https://github.com/jalammar/jalammar.github.io/discussions/21) or reach out to me on Twitter (https://twitter.com/JayAlammar) for any corrections or feedback.

Written on January 3, 2022

Subscribe to get notified about upcoming posts by email

Email Address

Thank you for subscribing!

Subscribe

Alammar, J (2018). The Illustrated Transformer [Blog post]. Retrieved from https://jalammar.github.io/illustrated-transformer/ (https://jalammar.github.io/illustrated-transformer/)

Note: If you translate any of the posts, let me know so I can link your translation to the original post. My email is in the about page (/about).

(https://github.com/jalammar) (https://www.linkedin.com/in/jalammar) (https://www.twitter.com/jayalammar)