ML Applications

Open Source

Retrieval Augmented Generation: Streamlining the creation of intelligent natural language processing models

September 28, 2020

Teaching computers to understand how humans write and speak, known as natural language processing (NLP), is one of the oldest challenges in AI research. There has been a marked change in approach over the past two years, however. Where research once focused on developing specific frameworks for specific tasks, today powerful general-purpose language models can be fine-tuned for a wide variety of different tasks. While promising, efforts to this point have applied these general-purpose models to tasks (such as sentiment analysis) for which a human could produce the solution without additional background knowledge.

Building a model that researches and contextualizes is more challenging, but it's essential for future advancements. We recently made substantial progress in this realm with our Retrieval Augmented Generation (RAG) architecture, an end-to-end differentiable model that combines an information retrieval component (Facebook AI’s dense-passage retrieval system) with a seq2seq generator (our Bidirectional and Auto-Regressive Transformers [BART] model). RAG can be fine-tuned on knowledge-intensive downstream tasks to achieve state-of-the-art results compared with even the largest pretrained seq2seq language models. And unlike these pretrained models, RAG’s internal knowledge can be easily altered or even supplemented on the fly, enabling researchers and engineers to control what RAG knows and doesn’t know without wasting time or compute power retraining the entire model.

Combining the strengths of open-book and closed-book

RAG looks and acts like a standard seq2seq model, meaning it takes in one sequence and outputs a corresponding sequence. There is an intermediary step though, which differentiates and elevates RAG above the usual seq2seq methods. Rather than passing the input directly to the generator, RAG instead uses the input to retrieve a set of relevant documents, in our case from Wikipedia.

Given the prompt “When did the first mammal appear on Earth?” for instance, RAG might surface documents for “Mammal,” “History of Earth,” and “Evolution of Mammals.” These supporting documents are then concatenated as context with the original input and fed to the seq2seq model that produces the actual output. RAG thus has two sources of knowledge: the knowledge that seq2seq models store in their parameters (parametric memory) and the knowledge stored in the corpus from which RAG retrieves passages (nonparametric memory).

These two sources complement each other. We found that RAG uses its nonparametric memory to “cue” the seq2seq model into generating correct responses, essentially combining the flexibility of the “closed-book” or parametric-only approach with the performance of “open-book” or retrieval-based methods. RAG employs a form of late fusion to integrate knowledge from all retrieved documents, meaning it makes individual answer predictions for document-question pairs and then aggregates the final prediction scores. Critically, using late fusion allows us to back-propagate error signals in the output to the retrieval mechanism, which can substantially improve the performance of the end-to-end system.

Combining a retrieval-based component with a generative component has advantages even in purely extractive tasks, such as the open-domain NaturalQuestions task. Performance improves when RAG has access to documents that contain clues to the correct answer but where the answer is never stated verbatim, and RAG even generates correct answers in certain situations where the correct answer is nowhere to be found in any of the retrieved documents. We obtained very strong results on NaturalQuestions, CuratedTrec, and WebQuestions with RAG, demonstrating that state-of-the-art machine reading performance can be achieved with a generative, rather than extractive, reader..

RAG truly excels at knowledge-intensive Natural Language Generation though, which we explored by generating "Jeopardy!" questions. The "Jeopardy!" questions that RAG generates are more specific, diverse, and factual than those of comparable state-of-the-art seq2seq models. We theorize this is because of RAG’s ability to synthesize a response using disparate pieces of information drawn from a number of sources.

RAG’s true strength lies in its flexibility. Changing what a pretrained language model knows entails retraining the entire model with new documents. With RAG, we control what it knows simply by swapping out the documents it uses for knowledge retrieval. We tested this behavior by replacing our original Wikipedia dataset with an older one and then asking questions like “Who is the prime minister of Iceland?” Results showed RAG leveraged the knowledge in the swapped-in corpus to adjust its answers, even though the parametric knowledge remained static. This adaptive approach is invaluable in situations where facts (or our understanding of the facts) evolve over time.

Removing training overhead from research

If AI assistants are to play a more useful role in everyday life, they need to be able not just to access vast quantities of information but, more important, to access the correct information. Given the speed at which the world moves, that proves challenging for pretrained models that require constant compute-intensive retraining for even small changes. RAG allows NLP models to bypass the retraining step, accessing and drawing from up-to-date information and then using a state-of-the-art seq2seq generator to output the results. This confluence should make future NLP models more adaptive, and indeed we’ve already seen strong results from a related Facebook AI research project, Fusion-in-Decoder.

We see broad potential for RAG, which is why we released it today as a component of the Hugging Face transformer library. Hugging Face's Transformers has become a de facto standard in open source NLP, thanks to its low barrier to entry and coverage of state-of-the-art models, and it integrates with the new Datasets library to provide the indexed knowledge source that RAG relies on. Now, with RAG's inclusion, we believe the community will be able to apply retrieval-based generation to both the knowledge-intensive tasks we already explored and some we haven't even imagined yet.

RAG frees researchers and engineers to quickly develop and deploy solutions to their own knowledge-intensive tasks with just five lines of code. We foresee the potential for future research into knowledge-intensive tasks that are just as easy and accessible as light-knowledge tasks like sentiment analysis today.

We will be presenting RAG at NeurIPS 2020 in December, and look forward in the meantime to seeing how the community responds to it. We also plan to continue pursuing these ideas by refining RAG's retrieval component, scaling the retrieval corpus and model, and improving RAG's overall performance on knowledge benchmarks, like the KILT benchmark we recently released.

Rag is created by Team in

RAG created by Sebastian Riedel(Research Manager)

Douwe Kiela(Research Scientist)

Patrick Lewis (FAIR PhD Student)

Aleksandra Piktus (Software Engineer)