Episode Book

EPISODE #6712

Between BM25 and Vector Search

PUBLISHED	20 November 2026 at 14:35 UTC
DURATION	29:58
LISTEN	myweirdprompts.com/episode/learned-sparse-retrieval-embeddings-58372/
HOSTS	Herman, Corn, Larry, Hilbert
SCRIPT MODEL	moonshotai/kimi-k3.1-pro
VOICES	omnivoice
PIPELINE	V5.36
BOOK GENERATED	20 November 2026 at 21:20 UTC

SYNOPSIS

Information retrieval is not just a binary choice between BM25 keyword matching and dense vector embeddings. Across the middle rungs of the retrieval ladder, learned sparse methods like SPLADE expand vocabularies onto inverted indexes while late-interaction systems like ColBERT preserve token-level nuance. A look into how LSA's matrix factorization contrasts with modern neural embedders, and why sparse term weights often assign search relevance to semantically unrelated tokens.

TRANSCRIPT



Herman

Seven percent. That is the entire share of queries, across seven standard test collections, where keyword search found something useful that the modern neural embedder missed. The other direction runs as high as twenty-three percent.



Corn

Which sounds like the end of the argument until you find out the seven percent is the part with the part numbers in it. Daniel's prompt today picks up where we left the transcript archive: indexed, ranked with BM25, and still occasionally defeated by someone remembering the idea of an episode but none of its words. He wants to know what actually lives between that and semantic search, whether it is a ladder or a leap. And embeddings most of all, because everyone says they use them every day in Qdrant, but LSA and word2vec were embeddings too, and they are clearly not the same animal. Also, can you just use a modern retrieval embedder as your basic search box and be done with it?



Herman

That last one is where I think people get the wrong answer fastest.



Corn

Then let's give him the slow answer.



Herman

The popular picture is a binary. BM25 over here, counting terms with some very clever statistics about how rare they are. Vector search over there, doing meaning. Empty space in between. The literature has a whole populated neighborhood in that empty space, and the name to know is learned sparse retrieval.



Corn

A neighborhood most people drive past on the way to the vector database.



Herman

There is a ladder, roughly. BM25 at the bottom. Learned sparse models like SPLADE above it. Then single-vector dense retrieval, the thing most people mean when they say embeddings. Then multi-vector late interaction, ColBERT-style, one vector per token. And at the top, cross-encoder reranking, where a model actually reads the query and the candidate document together. Each rung trades exactness for semantics, and index size for compute.



Corn

And each rung costs you the ability to explain to anyone why document twelve came back.



Herman

Sometimes literally, and we will get to that, because the middle rung has a strange secret.



Corn

Before the secret, the bottom of the ladder has rungs of its own. Last time BM25 matched words exactly as written, and half of the vocabulary mismatch is just the same word in a different coat.



Herman

That is the cheap end, and it is old. Stemming chops words back to a crude root, so renting, rented and rents all land on rent. Martin Porter's stemmer, from nineteen eighty, is still the default in a lot of search engines. Lemmatisation is the politer cousin: it uses a dictionary to map a word to its real base form, so mice goes to mouse, which no suffix-chopper will ever manage.



Corn

And stemming overreaches. A stemmer that folds university and universe into the same root has made a decision on your behalf.



Herman

Then character n-grams. Index overlapping slices of three or four characters instead of whole words, and a misspelled surname still shares most of its slices with the right one. That is what rescues typos and transliterations, and it is a cousin of the fuzzy matching we did a whole episode on. And then synonym and query expansion: a hand-built list that says leasing also means renting, or pseudo-relevance feedback, where you take the top few results, pull their common terms back into the query and search again.



Corn

All of which is still counting words. Just more of them, or shorter ones.



Herman

Every one of those still sits on the inverted index, and every one needs a person or a rule to say which words belong together. The next rung is where a model learns that instead.



Corn

Learned sparse. Sell it to me.



Herman

SPLADE is the canonical version, from twenty twenty-one. Sparse lexical and expansion model. Here is what it does. Instead of producing one dense vector of, say, a thousand mysterious dimensions, it produces a vector over the vocabulary. Actual tokens. And it is sparse, meaning almost every entry is zero, because the training applies explicit sparsity regularization. There is a log-saturation effect on the term weights so repeated terms do not run away with the score.



Corn

So far that is a very fancy bag of words.



Herman

Except it learns which terms to expand to. Your query says feline, the document says cat, and BM25 dies right there, the vocabulary mismatch we spent last time on. SPLADE has learned, from training, to put weight on cat when it sees feline. And crucially it still runs on an inverted index with exact term matching underneath. That is the middle, and it is not a metaphor. It is a different data structure with a learned layer on top.



Corn

An inverted index that has been to night school.



Herman

And the payoff is you keep the operational shape of keyword search. Posting lists, exact matching, the whole Lucene world. You are not rebuilding your infrastructure around nearest-neighbor search.



Corn

Now the strange secret.



Herman

You cannot read the vector. That is the secret. The intuition everyone has is sparse means interpretable, because the dimensions are words, so you should be able to look at the weights and see what the model thought mattered. A paper at SIGIR this year called Wacky Weights dissected exactly that and found SPLADE assigning weight to expansion terms that appear semantically unrelated to the input. Bigger vocabularies produce more of these wacky tokens. Stricter sparsity regularization produces fewer. And they seem to be used mostly for in-domain effectiveness rather than generalizing out of domain.



Corn

So the readable bag of words is unreadable.



Herman

It goes back further than that. Work in twenty twenty-three constrained SPLADE's vocabulary to stopwords, and even to random words, terms with no traditional ranking value at all, and it still encoded useful ranking signals in the documents and still ranked well. Which tells you the vocabulary dimensions are being used as a convenient high-dimensional addressing scheme, not as a statement about language. The model is not saying this document is about the word the. It is putting weight where the training found weight useful.



Corn

The interpretability was a property of the labels, not the model.



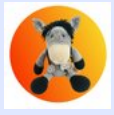
Herman

Say more about that, because I think that is right and I had not phrased it that way to myself.



Corn

If the dimensions were arbitrary numbers nobody would call it interpretable. Somebody printed words along the axis and everyone relaxed. But the model never agreed to mean those words. It agreed to minimize a loss. So the comfort came from the packaging.



Herman

Hm. Yes. And there is a practical consequence, which is that debugging a weird SPLADE ranking by eyeballing the top weighted terms will send you chasing ghosts. You will find a semantically unrelated token carrying real weight and conclude the model is broken when it is doing exactly what made it effective.



Corn

I have debugged enough of your installations to know that the ghost usually wins.



Herman

One time. And it was a stale index, which I found.



Corn

You found it after rebuilding the index, which is how you find everything.



Herman

Moving up a rung before I get defensive. Late interaction. ColBERT and its descendants. Instead of crushing a whole passage into one vector, you keep one vector per token, for the query and for the document, and at query time you score with MaxSim. For each query token, find its best-matching document token, take that maximum similarity, sum across the query tokens.



Corn

So nothing gets averaged away before the comparison.



Herman

That is exactly the motivation. The PyLate paper puts the complaint about single-vector search plainly: it suffers from compressing all the information into a single vector, and that compression causes notable degradation on out-of-domain, long-context and reasoning-intensive retrieval. A long transcript about six topics has to fit all six into one point. Late interaction lets the relevant tokens speak for themselves.



Corn

The whole archive episode is six topics, three of them Hilbert.



Herman

And there is a lovely hybrid of the hybrid here, called SLIM, which maps the contextualized token vectors into a sparse high-dimensional lexical space before doing late interaction, so the whole multi-vector thing can run on Lucene inverted indexes. The middle is not one rung. People keep bolting the rungs together.



Corn

And the moment you bolt two things together, someone has to choose the mixture, which is an argument we are saving for the last part of this series. The middle rung is learned sparse retrieval, and that is where I'd like to pause for a word from someone.

L

Larry

Friends, is your retrieval feeling sparse? Do your vectors refuse to bridge the gap between keyword and meaning? Ask your doctor about Splaindex, the learned sparse remedy that expands your terms, saturates your logs, and keeps an inverted index humming in the background. Splaindex is not for everyone. Side effects may include expansion terms semantically unrelated to your query, stopwords quietly carrying your entire ranking signal, random words gaining wacky weights, a sudden urge to refit your corpus nightly, transductive nostalgia for the year nineteen eighty eight, cosine similarity in inappropriate settings, and a single dense vector that compresses everything you love into one small, lonely point. Tell your producer if your index grows larger than expected or your late interaction lasts more than four hours.



Herman

The refitting point is actually backwards, because the whole advantage of the modern...



Corn

Stopwords carrying the ranking signal is documented, and you are fact-checking a medicine.



Herman

Fine. But the refitting claim is the interesting error, because that is the old world, and it is exactly where Daniel's next question lives.



Corn

Then go back to the bottom of the ladder. LSA. What is it actually doing, mechanically, not poetically.



Herman

Latent semantic analysis, or latent semantic indexing in retrieval contexts. The information-retrieval technique was patented in nineteen eighty-eight by Deerwester, Dumais, Furnas, Harshman, Landauer, Lochbaum and Streeter. You build a matrix of word counts per document. Terms down one side, documents along the other. Then you apply singular value decomposition and reduce the rows while preserving the similarity structure among the columns. Documents become points in a much smaller space, and you compare them by cosine similarity.



Corn

So it is a factorization of a table you already counted. Nothing is predicting anything. There is no training loop in the modern sense.



Herman

Linear algebra over observed co-occurrence. And the deep property, the one that matters for Daniel, is that it is transductive. The decomposition is of this matrix, this corpus. The latent space you get is fitted to these documents and these terms. Add a pile of new documents with new vocabulary and the honest move is to refit.



Corn

Which is why the medicine bottle says refit your corpus nightly. Fine, he got one right.



Herman

Its probabilistic successor, PLSA, has the same corpus-specificity. And then word2vec, Mikolov and colleagues at Google, presented at ICLR in twenty thirteen. That is a genuine change in kind, not just a better decomposition. A learned neural mapping. You train on surrounding context and out come reusable per-word vectors. And the efficiency was the breakthrough, it made learning good word vectors from enormous corpora practical.



Corn

Reusable is doing the work there. I can take the vectors somewhere else, to a corpus they have never seen.



Herman

Right, and fastText extended the idea using character sequences within words, so morphology and rare words got a foothold. But notice what it still is. Per-word. If you want a vector for a passage, you average the word vectors or do something similar, and averaging is exactly the compression late interaction complains about.



Corn

So walk me to the modern embedder. What is the object in Qdrant, precisely.



Herman

A transformer encoder, fine-tuned with contrastive learning objectives on query-passage pairs, producing one vector per passage. NV-Retriever describes these models exactly that way and then says the hard part of training them is selecting high-quality hard negatives. Not the architecture. The negatives. You need passages that look relevant and are not, so the model learns a boundary instead of a blur.



Corn

That is a different job from LSA in a way that has nothing to do with transformers.



Herman

Yes, and I want to be careful here because the fashionable account is old versus new, small versus big. The structural break is a fixed function versus a decomposition. LSA gives you a space derived from your matrix. A modern embedder gives you a function you can point at a brand new document, tonight, with vocabulary it never saw during your indexing, and get a vector that lives in the same space as everything you indexed last month. No refit. That is what makes it usable for a growing transcript archive at all.



Corn

And the training is asymmetric. Query on one side, passage on the other. Short question, long document. Which is a search-shaped problem, not a language-shaped one.



Herman

Which is also why the instruction prefixes exist, and we will get there. But here is the thing that makes the lineage real rather than a story. LSA and the modern embedder share an assumption. The distributional hypothesis. Words close in meaning occur in similar pieces of text. That is the philosophical foundation under both of them. Word2vec is distributional semantics made efficient, the modern embedder is the same bet with a retrieval objective. They share a premise, not a mechanism.



Corn

Same religion, different plumbing.



Herman

I am keeping that.



Corn

You may not. Now, the part I actually want to argue about. Contextual versus static. Because I think that is the division people feel without being able to name. Word2vec gives cat one vector, always. Bank gets one vector whether I mean a river or a place that rejects my mortgage. The transformer reads the sentence.



Herman

That is right, and it is a separate axis from transductive versus fixed-function, which is why people talking past each other on this is so common. There are at least three divisions bundled into the phrase modern embeddings. Static word vectors versus contextual token representations. Corpus-fitted decomposition versus a reusable learned function. And generic representation learning versus contrastive training for asymmetric retrieval. Daniel's Qdrant embedder is on the far side of all three, and LSA is on the near side of all three, and word2vec is in the middle of the first and the far side of the second.



Corn

So when someone says embeddings are an old idea, they are right about the premise and wrong about the object.



Herman

Well put... no, I mean, that is the correct content. MTEB is worth invoking here because its whole existence proves embedding model is not one thing. Eight task types, fifty-eight datasets, a hundred and twelve languages in the original benchmark. And the finding is that no particular text embedding method dominates across all tasks. The model you want for clustering is not automatically the model you want for query-to-document retrieval.



Corn

A leaderboard with eight events and no decathlon champion. Which brings me to the search box question, and I have used the archive search more than you have. I think a modern embedder can be the search box, and I think it will feel magical for a week. Because the failures at the start are all the failures BM25 had. Somebody remembers an episode as the one about renting when we said leasing, and the embedder sails through it and everyone declares victory. Then someone searches a model number, or an unusual surname, or the exact odd phrase they half remember, and it returns something warmly adjacent and wrong. And the wrongness is worse because it is plausible.



Herman

Plausible wrongness is the right thing to be worried about. BM25 failing is legible. No results, or results sharing a stray token. You can see what happened. Dense failure gives you three documents that are about the neighborhood of your query and not the thing.



Corn

So the answer is partly, and it depends on the query. Which is a very you answer.



Herman

It is the only answer the evidence supports. The split we opened with says dense covers nearly everything BM25 covers. The keyword-only slice never goes above seven percent. So yes, you can use a retrieval embedder for basic search and most queries will be fine. But BM25 covers a slice dense misses, and that slice is concentrated in exactly the paraphrase-light, identifier-heavy lookups that basic search boxes exist for. And here is the honest limit: nobody has published the direct experiment. Nobody has taken a modern embedder and benchmarked it as a plain keyword engine on exact-phrase, identifier and product-code lookup against BM25. The evidence is all indirect.



Corn

That is a strange hole. It is the first thing any of us would try.



Herman

It is. I suspect the interesting version is too per-corpus to publish. But I want to be clear that I am inferring the identifier behavior from that asymmetry and the known paraphrase strength, not quoting a lookup benchmark.



Corn

Good. Now the operational reasons drop-in replacement is false even where the quality is fine. Because I have watched you install these things.



Herman

Instruction sensitivity. On Quora, running BGE-base and BGE-large without the query instruction costs zero point zero six nDCG, and putting the instruction back restores it. These models expect to be told what kind of text they are embedding. Embed a query bare because you think of it as a bag of keywords and you can be measurably off the model's trained distribution.



Corn

So plain search with a modern embedder includes a small liturgy you must recite at the query.



Herman

And the liturgy differs by model. Which is not a property of BM25.



Corn

Domain next, because this is the one that bit me. FinMTEB is the financial benchmark, sixty-four datasets, and a plain bag-of-words approach beats the sophisticated dense embeddings on financial semantic textual similarity. Also general-benchmark performance has limited correlation with financial-domain performance. So the



Corn (continued)

leaderboard placement of your embedder is close to a rumor about your corpus.



Herman

CliniQ is stranger, and I say this as someone who spent years reading medical notes. A thousand discharge summaries, over twelve hundred queries, more than seventy-seven thousand relevance judgments. BM25 sets a strong baseline and stays competitive with the dense retrievers. And the general-domain dense retrievers outperform the ones built for the medical domain.



Corn

The specialist model loses to the generalist on its own ward.



Herman

Which should make anyone humble about buying a domain-adapted embedder on the strength of the label. Domain match in the name is not domain match in the weights.



Corn

And meanwhile BM25 is not even a fixed baseline. BEIR called it robust, with reranking and late interaction best on zero-shot average at high computational cost. BEIR-PL found it significantly weaker for Polish than English, because of the inflection and morphological complexity. Keyword search itself is corpus- and language-dependent. The floor moves. So when someone says just use BM25 for English and just use embeddings for everything else, both halves are folklore.



Herman

Polish is the example I would hand them. A language where the same concept arrives wearing twelve suffixes is hard on exact term matching and has nothing to do with neural versus not neural.



Corn

One more deflation before I let you be enthusiastic again. The SPLADE-v3 query expansion study. Generated query expansion on top of a strong learned sparse retriever improved things across the board. Best relative gains around five to nine percent depending on the collection. Lovely. Then the controls.



Herman

Ah, the controls. Shuffled text and a non-contextual lexical bag, both of which destroy any claim that a model understood the query, also stayed above baseline in all twenty-four aggregate comparisons. The added vocabulary carries most of the benefit. More terms, more chances for the inverted index to catch. You do not need a story about meaning to explain it.



Corn

Does that damage your middle rung? Because the middle rung's sales pitch is learned semantic expansion, and now the shuffled bag is nipping at it.



Herman

It damages the interpretation more than the rung. SPLADE still gives you a sparse neural retriever on an inverted index with strong results. The engineering is real. What the controls tell you is to stop narrating it as the model understanding your query.



Corn

A machine that works for reasons you are not allowed to enjoy.



Herman

You enjoy that kind of machine more than anyone I live with.



Corn

I enjoy a machine that works while nobody watches it. Different pleasure. What I keep bumping into is that the archive is small and weird, and every number we have leaned on is from big, comparatively well-behaved collections. BEIR, financial filings, medical notes. We have thousands of transcripts where the same nonsense word can be the single best key to an episode.



Herman

Which cuts both ways, and this is worth being precise about. Rare vocabulary is where lexical matching shines, so it argues for keeping BM25 central. But it is also where a general embedder has the least reliable semantics, because the term was rare in its training data too. So the embedder's advantage over keyword search, which is real and large on paraphrase-heavy corpora, could be smaller for us.



Corn

And the seven percent keyword-only slice might be bigger for us. Nobody knows, because it is a property of the query distribution, and our queries are Daniel trying to find something he half remembers while doing something else.



Herman

The part I am least sure about is whether a query can even be paraphrase-light in a transcript archive. People remember episodes topically.



Corn

They remember episodes topically until they remember one phrase. Then it is the only thing they have. You do this. You searched for a paper last week by an exact clause from its abstract, and you were furious when the paraphrase came back.



Herman

I was not furious.



Corn

You re-indexed.



Herman

I re-indexed because the index was stale.



Corn

Mm.



Herman

There is a real point under your enjoyment, though, which is that query type is not stable within a single user, let alone a single corpus. The same person issues a topical query and an exact-string query an hour apart.



Corn

So the honest engineering position is that keyword search never gets removed, and how much the vectors get trusted is something you measure on your own collection, not something you import. A long way from the binary Daniel started with. One thing I want to nail down before we drift. Is a retrieval embedder a different tool from one you would use for plain search, or the same tool used plainly. Because we have been saying partly for ten minutes and I want the sentence.



Herman

It is a different tool in training and in expectations, and a usable one for most plain queries. The contrastive asymmetric training makes it a query-to-passage matcher, not a term matcher. The instruction prefixes mean it has a protocol. The domain results mean its competence is not portable on the strength of a leaderboard. But nothing about it forbids exact-ish lookups, and the split we opened with says it usually covers what BM25 covers. So, different object, overlapping job.



Corn

I can work with that.



Herman

And the LSA comparison is the cleanest way to see why the object is different, because...



Hilbert

Because the matrix only knows what you already counted, Herman. That is the part people skip when they say semantic. I kept a concert index for years. Ticket stubs in a shoebox, setlists on napkins, and a spiral notebook where every band got a number and I drew lines between the ones that turned up together. Same bill, same support slot, same bass player, same line. The Fall was number one. Everything since is derivative, so the numbering is no use to anybody else.



Herman

The lines are co-occurrence. You built the count matrix by hand and left out the decomposition.



Hilbert

I left out a lot of things. It was a notebook. What I wanted to ask, and you two have just about answered it, is whether the lines were doing the work or whether the notebook understood anything. Because everyone says semantic and it never sat right with me.



Herman

The lines. Nothing in that notebook represented meaning except that things kept turning up near each other, and that is the distributional bet in its purest form. LSA is the same bet with linear algebra. The modern embedder adds a retrieval objective, but it does not add understanding as a separate ingredient you could point to.



Hilbert

That settles it, then. It was the cross-references.



Corn

You made the notebook unusable to anyone else by assigning The Fall the number one, and unusable is a form of accuracy.



Hilbert

It is on the shelf by the desk, plugged into nothing, and it still gets a line when a support act turns up twice. Nobody else reads it. They would read it wrong.



Herman

Read it wrong how, though?



Hilbert

They would start at number two. A man called Derek mended my amplifier in seventy-four and charged me three pounds ten, and he is the only person who ever looked at it and went straight to the right page.



Herman

Derek got the right page because he was not looking for meaning, he was looking for an entry.



Corn

Which is the seven percent in a notebook.



Herman

It is, a little. The lookup that has no paraphrase in it. And notice the thing Hilbert's index and our archive share: the cross-references are the asset, and they were expensive to collect, and no model gets to invent them afterward. His lines were observations about what actually appeared together. Our inverted index is the same kind of observation. The learned part sits on top of it or it does not, but the observation is not replaceable by better weights.



Corn

You have found a way to be sentimental about a posting list.



Herman

I have found a way to say the corpus is in the system whether you call it semantic or not. LSA is corpus-fitted outright. A modern embedder hides the corpus in its training data instead of in your matrix. That is a better deal operationally and a worse one for knowing what the thing knows.



Corn

That is a good reason to distrust the word semantic without giving up the machinery. And the experiment I would actually like answered is still the missing one: a modern embedder against BM25, on exact phrases and odd surnames, on a weird little archive like ours.



Herman

And there is no consensus to borrow either. MTEB says no embedding method dominates across tasks, and every comparison we have comes from somebody else's collection. Next time we build the embedding layer over the archive by hand, chunk by chunk and vector by vector, and start finding out how it behaves on ours.



Corn

Three pounds ten, and Derek went straight to the right page. Some lookups do not want meaning.



Herman

Hilbert Flumingtop produces this show, and keeps the desk running while we argue about data structures.



Corn

This has been My Weird Prompts, the human-AI collaboration podcast.



Herman

We will be back soon.



Corn

See you tomorrow.