Episode Book

EPISODE #6539

When Real Engineering Teams Still Choose Encoders

PUBLISHED	12 November 2026 at 13:36 UTC
DURATION	28:26
LISTEN	myweirdprompts.com/episode/encoder-decoder-production-models-53507/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	z-ai/glm-5.3
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.26
SOURCES	25 cited by the research (listed at the end)
BOOK GENERATED	13 November 2026 at 09:46 UTC

SYNOPSIS

While decoder models dominate headlines, production engineering teams frequently rely on bidirectional encoders for classification, tagging, and retrieval. A close look at real deployments explains the economics: Roblox reached 25,000 moderation checks per second on CPUs, Itaú built BERTaú from scratch to gain 22 percent in retrieval, and PayPal weighed custom pretraining against standard fine-tuning. The operational constraints of latency, tokenization costs, and unlabeled text volume dictate when encoders remain the strictly superior architecture.

TRANSCRIPT



Corn

Three thousand inferences per second on a thirty-six-core Xeon. That's what Roblox was getting out of a fine-tuned BERT running on CPUs, back when everyone else was reaching for GPUs. The guy with the cost-equivalent Tesla card got five hundred.



Herman

And that gap is not a story about hardware. It's a story about architecture. Daniel sent us a prompt this week that I have been looking forward to, because it's the kind of question where the answer is surprising if you've only followed the headlines. He wants to know who actually uses encoder-only versus decoder-only models in the real world — and he's very specific: concrete examples, not theory. Teams that trained their own custom encoder for a real job, teams that took an off-the-shelf model and were right to, and — this is the part I like — when picking the other architecture would have been the mistake.



Corn

He wants the counterfactual. Not just "Roblox used BERT," but "Roblox used BERT, and here's what would have broken if they hadn't."



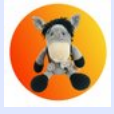
Herman

And the counterfactual cuts both ways, which is what makes it fun.



Corn

Before we get into the cases, the mechanical difference — because it explains almost everything that follows.



Herman

The mechanical difference is attention direction. An encoder-only model, BERT and its descendants, is bidirectional — every token gets to look at every other token, left and right, simultaneously, and what comes out the far end is one representation per token. That's why encoders are natural for classification, tagging, and embeddings — the whole input informs one fixed output shape. A decoder-only model, GPT and its descendants, generates causally — token by token, each one conditioned only on what came before. Which is why it's natural for anything that continues text.



Corn

There's a quote from a benchmark paper this year that compresses the whole economics in one sentence: "Unlike discriminative encoders, LLMs must generate output token by token — even when producing a single class label — introducing inherent latency and usage-based costs."



Herman

That single sentence is the reason Roblox picked an encoder and Copilot picked a decoder. We should mention, for completeness, that there's a third category — encoder-decoder, like Whisper for speech recognition. But the real-world fork, the one Daniel's asking about, lives between the two poles.



Corn

Let's start with the one case where the physics made the answer obvious. Roblox, twenty nineteen, content moderation.



Herman

They were building text classifiers and named entity recognition for a platform whose user base is mostly kids, so the moderation has to actually work. They fine-tuned off-the-shelf BERT for it — not trained from scratch, just fine-tuned — and the F1 scores improved by ten percentage points or more over their previous models. That part is the standard BERT story. The interesting part is what came after.



Corn

The production requirement.



Herman

Over twenty-five thousand inferences per second. Over a billion a day. Under twenty milliseconds of latency. On that scale, the accuracy was the easy half — they chose accuracy first and speed second, and then spent what they described as tense months making BERT fast enough. They got a thirty-times-plus inference speedup through the optimization work, and along the way they discovered DistilBERT, which improved their speed dramatically.



Corn

I was going to ask about the GPU fleet, but the punchline is that they skipped it.



Herman

They skipped it on purpose, and the reasoning is worth quoting because it's a real production insight. Real-time request-response traffic doesn't batch well — their words were that batching these requests "would only create overhead and complexity." So they went CPU. And after the optimization, throughput per dollar was six times better on CPU than GPU — over three thousand inferences per second on a thirty-six-core Xeon versus five hundred per second on a cost-equivalent Tesla V100.



Corn

A six-to-one beat from the cheaper hardware.



Herman

Six to one. And that's the counterfactual, Corn — a decoder generating a label token by token cannot hit twenty-five thousand per second at under twenty milliseconds on commodity CPUs. It's not a preference. It's arithmetic.



Corn

Twenty-five thousand classifications a second, and the entire output vocabulary is maybe a dozen labels. Generating "spam" one character at a time would be a formality for a machine that can't do formality.



Herman

That's the whole thing. You're running a label stamp at industrial speed, and the encoder is the shape of a label stamp.



Corn

Which sets up PayPal, because PayPal had almost the same problem and made a different choice about training. Customer service — dispute filing, transaction status, refund status.



Herman

Right. Their chatbot resolves forty-four percent of customer contacts automatically, and the NLP underneath that is classification over customer utterances with a fixed label set. They'd been running LSTMs, and they moved to BERT-Base. But instead of just fine-tuning, they did what they called a gap analysis between BERT's general training and their chatlog context, then customized the BERT-Base weights on PayPal chatlog data — self-supervised learning on their own text, then a fine-tune on top.



Corn

And the payoff for all that?



Herman

Roughly three percent absolute micro F1 over stock BERT-Base.



Corn

Three percent.



Herman

And I want to sit with that, because it's a real number from a real deployment, and it's modest. That's the calibration point for anyone asking "should we domain-adapt our own encoder?" — the honest answer from this case is: expect real but not dramatic returns if what you have is a modest amount of in-domain text.



Corn

Which takes us to the bank that had all the text. Itaú, in Brazil.



Herman

BERTaú. Itaú trained a BERT from scratch on Portuguese — specifically, on roughly eighteen months of their own chatbot data. Fourteen and a half gigabytes, twenty-two and a half million words, a million training steps. And the gains were not three percent. Twenty-two percent improvement in FAQ retrieval MRR. Two-point-one on sentiment F1, four-point-four on NER F1.



Corn

The twenty-two is the one that jumps out.



Herman

It should, because retrieval is exactly where domain vocabulary mismatch hurts most. Portuguese financial customer-service language is far enough from the English web text that general models were trained on that the vocabulary fragments — the model just doesn't have good representations for the words your customers actually use. Train from scratch on the right text and the retrieval gain is enormous.



Corn

So PayPal and Itaú had the same broad problem — customer service classification — and made opposite choices. PayPal adapted BERT-Base for three percent. Itaú built fresh for twenty-two.



Herman

That's the decision rule. It's not ambition, it's not prestige, it's a single variable: how much unlabeled in-domain text do you have? PayPal had limited chatlog text and adapted. Itaú had twenty-two million words and built fresh. The gap between three and twenty-two is the evidence.



Corn

We should mention we did an episode a while back on fine-tuning ASR and what happens after — the deployment side. Roblox's tense months of optimization is exactly that part nobody budgets for.



Herman

Nobody budgets for it! You fine-tune the model, you celebrate the F1, and then someone says "great, now serve it a billion times a day under twenty milliseconds." Now, the strongest "train from scratch" argument in the whole literature is Microsoft Research's PubMedBERT — and I want to walk through it, because the paper actually picks a fight with the prevailing assumption.



Corn

Please. You've been vibrating since I mentioned it.



Herman

Because it's a rare case of a paper saying "everyone is doing this wrong." The received wisdom was that you always start from a general model and continue pretraining on domain text — because general models have seen so much. PubMedBERT challenged that directly: for domains with abundant unlabeled text, like biomedicine, pretraining from scratch beats starting from a general model. Because general vocabularies fragment biomedical terms — the tokenizer splits domain terms into pieces that never add up to the concept.



Corn

And they had the text to back it.



Herman

Fourteen million PubMed abstracts. Three-point-two billion words, twenty-one gigabytes. Trained with an in-domain vocabulary on sixteen V100s on a DGX-2, about five days. Result: BLURB score of eighty-one point one six versus seventy-six point four six for RoBERTa, eighty point three four for BioBERT, seventy-six point one one for plain BERT.



Corn

Five points over the field. In benchmark terms that's a landslide.



Herman

It is, and it's driven entirely by the vocabulary and the from-scratch pretraining. Now, the same pattern shows up in regulated industries — and I'll keep this brisk, because it's a parade, not a story. SEC-BERT: pretrained on two hundred sixty thousand, seven hundred seventy-three 10-K filings from 1993 through 2019. And the variant I love is SEC-BERT-NUM, which replaces every number token with a single NUM pseudo-token.



Corn

Because filings are drowning in numbers, and every distinct number fragments into rare tokens.



Herman

A model trained on filings spends its vocabulary on numeric expressions it will never see again. Replace every number with one placeholder and the vocabulary goes to the words that matter. It's a tokenizer-level fix that only makes sense if you're building a domain encoder for that corpus. And LEGAL-BERT, pretrained on twelve gigabytes of English legal text, where the authors found that the standard guidelines for pretraining and fine-tuning — their words, "often blindly followed" — don't generalize well in the legal domain.



Corn

Now, before we get into Roblox and PayPal — no, wait, we've done those. Before we get into the off-the-shelf side, let's hear from a listener. Franklin, on the forty-seven bus. Franklin: Hi Corn, hi Herman, this is Franklin, I'm on the forty seven bus into town, sitting upstairs near the window, and I wanted to say I loved the episode about the BERT style models versus the ones that generate word by word, so thank you for that. My question is, and I know it's not really the point, but when you have a model that only outputs a label, does it still count as understanding the sentence the way we do, or is the label just a shortcut, because my nephew keeps telling me these things don't understand anything at all and I'd like to have an answer for him at Sunday dinner.



Herman

I love that Franklin rides the forty-seven bus thinking about this. My answer for Sunday dinner: the label isn't a shortcut, it's the job. Nobody asks the thermometer whether it understands fever.



Corn

And the honest half for the nephew: it doesn't understand the way he does. Nobody's claiming it does. What it does is use the whole sentence to place it among labels, better than anything else at that speed. If he wants understanding, he can have it, at two thousand dollars a million requests — we'll get to that.



Herman

Okay — flip it. Where the off-the-shelf call was right, and where a decoder was the only architecture that could do the job.



Corn

BloombergGPT is where I'd start, because it's the counter-example that proves the rule. Same domain as SEC-BERT, opposite architecture.



Herman

Opposite architecture, and for a reason. BloombergGPT is a fifty-billion-parameter decoder-only model, trained on three hundred sixty-three billion financial tokens plus three hundred forty-five billion general tokens. It averaged sixty-two point five one across five financial tasks versus fifty-one point nine zero for GPT-NeoX of similar size, winning four of five.



Corn

And the reason it's a decoder is in the task list.



Herman

Sentiment, NER — the stuff SEC-BERT does — and question answering. Generation. A bidirectional encoder has no output mechanism for generating an answer. The moment your task set includes generation, you've left encoder country. That's the cleanest illustration of the fork in the whole literature: same domain, opposite architecture, because the tasks diverged.



Corn

One caveat I'd flag, because nobody else does: BloombergGPT is documented as a research model and a press release. Whether it's scoring anything in production, the public record doesn't say. The paper is real; the deployment is folklore.



Herman

Fair. The deployment we know about in detail is GitHub Copilot — and it's the mirror image of Roblox. Peak load of about eight thousand requests per second, hundreds of millions of requests a day, average response time under two hundred milliseconds. Same latency discipline as Roblox's twenty milliseconds. Opposite architecture, because the output is code, not a label. You cannot produce a continuation with an encoder-only model — there's no mechanism. The task is "write the next forty characters of this function," and that is generation by definition.



Corn

There's a detail from their engineering talks that I find quietly damning. The original model size was chosen — their words — "largely based on intuition a few years ago."



Herman

I find that reassuring, honestly. It means the constraint that actually governed was the two-hundred-millisecond budget, and the model was sized to fit it — with bigger models reserved for test generation and documentation, because those aren't as time-sensitive. That's architecture following output shape and latency, not the other way around. And when they shipped their own custom completions model in October twenty twenty-five: three times the tokens-per-second, thirty-five percent lower latency, twenty percent more accepted and retained characters, twelve percent higher acceptance.



Corn

They rebuilt the model for the same reason they picked a decoder in the first place. The job hadn't changed, the margins had.



Herman

Now — off-the-shelf encoders, and the biggest deployment class of all: retrieval.



Corn

The bi-encoder pattern.



Herman

You encode queries and documents independently, so you can precompute all the document embeddings ahead of time, store them, and do fast approximate nearest-neighbor search at query time. NVIDIA's Retrieval QA Embedding Model is a twenty-four-layer encoder based on E5-Large-Unsupervised — thousand-and-24-dimensional embeddings, five-hundred-twelve-token max input. IBM's Granite embedding models are bi-encoders under Apache 2.0 — English, then multilingual. And the production pattern is two stages: embeddings shortlist about a hundred chunks, then a cross-encoder reranker rescores them.



Corn

Say the part that makes it an encoder.



Herman

You cannot precompute a decoder's answer for every document in advance. That's the entire business. The bi-encoder's independence — query and document encoded separately — is what makes indexing possible at all. Retrieval is off-the-shelf encoder territory because the architecture is the data structure.



Corn

My own gripe with that pipeline — I've said this before — is that near-duplicate documents get near-identical bi-encoder scores and the ordering is basically a coin flip. Which is the reranker's real job: stability, not just precision.



Herman

And here's the underreported story, the one I'd put at the center if I had to pick. The two camps are converging. NVIDIA's NeMo AutoModel documentation states plainly that it includes a training recipe for converting Llama decoder-only models into encoder architectures with bidirectional attention. The industry is taking decoders and surgically converting them into encoders when it wants embeddings — because bidirectional attention is the right shape for that job, and the starting weights happen to be in a decoder.



Corn

That's the tell, isn't it. Nobody writes a conversion recipe out of loyalty.



Herman

Nobody writes one at all unless they need it. The reverse pressure exists too: an August twenty twenty-five paper argues decoder-only models like Qwen2.5-Coder-3B "suffer from high latency, particularly due to expensive prefill computations when leveraging long, repository-scale contexts" — and bolts a lightweight encoder onto the front to fix it. Both camps admit the other's shape is right for part of the job.



Corn

And that gets us to the economics, and I want to slow down here, because this is where the argument actually gets settled. The cost benchmark from this year — encoders against GPT-4o and Claude Sonnet 4.5 on classification.



Herman

This is the paper that quoted about generating a label token by token. They compared fine-tuned BERT, RoBERTa, and DistilBERT against both LLMs, zero-shot and few-shot, across four standard tasks, with encoders deployed as stateless services and pricing snapshotted in January. AG News is the clearest case: the encoders cluster at ninety-four to ninety-five macro-F1 — best RoBERTa at ninety-four point six three — while the best LLM configuration reaches ninety-one point three five. Every encoder beat every LLM, including few-shot.



Corn

And the cost side?



Herman

This is the number. Five dollars seventy-three to ten dollars forty-four per million requests for the encoders. Two hundred seventy-six dollars to over twelve hundred seventy dollars per million for LLM prompting. Thirty to two hundred times the cost, for worse accuracy.



Corn

Two hundred times the price to lose by three points. That's not a tradeoff, that's a tax.



Herman

On DBPedia it's near-saturation: BERT and DistilBERT both hit ninety-nine point four macro-F1 versus ninety-eight point eight three for the best LLM — costs of seven to eleven dollars versus four hundred sixty to two thousand seven hundred dollars per million. On SST-2, BERT at ninety-four point four two essentially matches Claude few-shot at ninety-four point four one — a dead heat — with median latencies of roughly a hundred to a hundred fifty milliseconds versus three hundred twenty-six to nearly fourteen hundred.



Corn

The one place the LLM wins outright is IMDB, I assume.



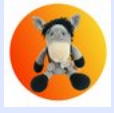
Herman

IMDB, yes. Claude few-shot gets ninety-six point four eight against RoBERTa's ninety-four point eight four. Under two F1 points better. It costs two thousand one hundred twenty dollars per million versus thirty-three dollars. The paper's conclusion is blunt: indiscriminate use of large language models for standard text classification workloads leads to suboptimal system-level outcomes — and they call prompting, in this setting, a dominated strategy under realistic throughput.



Corn

A dominated strategy. There's a term doing referee work.



Herman

And there's one more result I want on the record, because it kills a tempting middle path. The lazy hybrid is "encoder first, LLM safety net for the uncertain cases." An EMNLP paper this year tested exactly that for emotion classification — encoder predicts, uncertain examples escalate to a compact LLM verifier — and they ran a Pareto search over thresholds and seven different verifier setups. Zero-shot, LoRA-tuned, voting, self-consistency. The title is the finding: "Why Compact LLM Verifiers for Emotion Classification Fail." No compact-LLM verifier improved the encoder's cost-accuracy frontier.



Corn

The safety net costs more than the accidents.



Herman

That's the plain reading. And it connects back to your instinct about what a model is for. The Smol Training Playbook lists exactly three reasons a company can't use an off-the-shelf model — domain specificity, deployment constraints, and governance — and domain specificity is the one that maps onto everything we've covered today. It's not a prestige question. It's a data question. Do you have the text?



Corn

PayPal didn't, and got three percent. Microsoft had three billion words, and got a landslide.



Hilbert

It's not a trick, the NUM token. Corn called it a tokenizer trick earlier. It's a filing-room habit. You blank the numbers because the numbers are not the sentence. I ran a document job once, late nineties, mortgage paperwork, thousands of pages a week, and the search would not work until every figure was replaced with a placeholder. Before that, the same amount with different trailing zeros looked like different facts. You'd pull three documents and get three answers. My colleague Petro kept the vocabulary printout on the wall and counted what was wasted on numbers by hand. Forty-one percent. He wrote it in the margin.



Herman

Forty-one percent of the vocabulary.



Hilbert

Herman, you said from-scratch beats adapting when you have the text. I kept thinking — it's not really the volume, it's whether the thing looks like a form or looks like a conversation. The mortgage job was all forms. We tried a prose approach on a form job once, some consultant's idea, and it produced beautiful sentences that answered nothing. A form wants a label. I'd take the placeholder argument further than the paper does, but Petro was not a man to trust on methodology.



Corn

Was Petro the one who trusted the beautiful sentences?



Hilbert

No, that was the consultant. Petro never trusted anything after 1994. Hold on — Herman, you're clipping on the last word again, I'll ride the level. Anyway. Corn, the way you described the bi-encoder, precompute the documents and forget them — that's the lot, I'd never thought of it that way. I've been blanking numbers for twenty years without knowing why it generalizes.



Herman

The form-versus-conversation framing is not as far from the literature as you think. The AI Engineering Handbook makes almost the same point — moderation, intent, ticket routing don't need the model to write long answers, and forcing a generative model into the job buys you overhead. And your mortgage story is the NUM insight from the other end. SEC-BERT arrived at blank-the-numbers by measuring fragmentation; Petro arrived at it by pulling three documents and getting three answers.



Corn

Petro ran the ablation study. He just didn't call it that.



Hilbert

He wouldn't have called it anything. He wrote it in the margin.



Corn

On the label question Franklin raised, there's a detail from the benchmark I'd want at his Sunday dinner. On SST-2, BERT and Claude few-shot landed within one-hundredth of an F1 point of each other. One is a hundred-millisecond CPU job, the other is a frontier model writing its opinion out longhand. If that's a shortcut, it's the shortcut that finishes first.



Herman

The cutting-room floor item I'd keep: Copilot's first version did retrieval with just open files plus string similarity — no neural encoder at all. The largest code completion service on earth, and version one's context system was basically find-and-grep. They bolted the embeddings on later, when the accuracy demanded it. Even the generative maximalists start with string matching.



Corn

Here's the question I'd leave open. The published evidence runs one direction — encoders holding their ground, decoders getting converted into encoders, nobody we could find documenting a major product that replaced a working encoder classifier with a decoder-only LLM and reported a win. Is that because it hasn't happened, or because nobody writes up the migration that failed? The discourse is thinner than the practice, and I don't know which way that gap points.



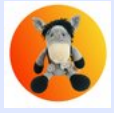
Herman

And the convergence suggests the fork itself may be dissolving — not two model families at war, but two attention patterns that production systems now mix and match. A decoder converted to an encoder for the index, an encoder bolted onto a decoder for the prefill. The question stops being "which camp are you in" and becomes "which attention shape does this stage of the pipeline need."



Corn

Which is, quietly, the most engineering answer available.



Herman

It is! Thanks as always to our producer, Hilbert Flumingtop, for keeping us on the air and the levels honest. If this was your kind of episode, go back for episode nine, Benchmarking Custom ASR Tools - Beyond The WER; episode four, If Your Voice Ages, Does Your Fine-Tune Become Useless; and episode twenty, Architectural AI. This has been My Weird Prompts — no question too obscure, no rabbit hole too deep. If you enjoyed this one, the review button takes four seconds and it helps. We'll be back soon.



Corn

See you tomorrow.

SOURCES

What the research for this episode read before the script was written. Primary sources first.

1. **arXiv 2602.06370, Cost-Aware Model Selection for Text Classification: Multi-Objective Trade-offs Between Fine-Tuned Encoders and LLM Prompting in Production (Valdes) [primary]**

   retrieved 2026-11-12; pricing snapshot 2026-01-22.
   https://arxiv.org/html/2602.06370

2. **Roblox Newsroom, How We Scaled Bert To Serve 1+ Billion Daily Requests on CPUs, 2020-05 [primary]**

   retrieved 2026-11-12.
   https://about.roblox.com/newsroom/2020/05/scaled-bert-serve-1-billion-daily-requests-cpus

3. **PayPal Technology Blog (Moein Saleh), Building PayPal-BERT using Transfer Learning and Self-Supervised Learning Techniques, 2020-12-15 [primary]**

   retrieved 2026-11-12.
   https://medium.com/paypal-tech/building-paypal-bert-using-transfer-learning-and-self-supervised-learning-techniques-42b8fa55a45d

4. **Itaú, BERTau: Itau BERT for digital customer service, LXAI at NeurIPS, published 2021-01-28 [primary]**

   retrieved 2026-11-12.
   https://doi.org/10.52591/lxai202312107

5. **Microsoft Research Blog, Domain-specific language model pretraining for biomedical natural language processing [primary]**

   retrieved 2026-11-12.
   https://www.microsoft.com/en-us/research/blog/domain-specific-language-model-pretraining-for-biomedical-natural-language-processing/

6. **Bloomberg, Introducing BloombergGPT, Bloomberg's 50-billion parameter large language model, purpose-built from scratch for finance [primary]**

   retrieved 2026-11-12.
   https://www.bloomberg.com/company/press/bloomberggpt-50-billion-parameter-llm-tuned-finance/

7. **Hugging Face, nlpaueb/sec-bert-shape model card [primary]**

   retrieved 2026-11-12.
   https://huggingface.co/nlpaueb/sec-bert-shape

8. **GitHub Blog, The road to better completions: Building a faster, smarter GitHub Copilot with a new custom model, 2025-10-23 [primary]**

   retrieved 2026-11-12.
   https://github.blog/ai-and-ml/github-copilot/the-road-to-better-completions-building-a-faster-smarter-github-copilot-with-a-new-custom-model/

9. **NVIDIA NeMo AutoModel docs, Embedding Models [primary]**
retrieved 2026-11-12.
https://docs.nvidia.com/nemo/automodel/latest/model-coverage/embedding-models/overview

10. **NVIDIA Technical Blog, Build Enterprise Retrieval-Augmented Generation Apps with NVIDIA Retrieval QA Embedding Model [primary]**
retrieved 2026-11-12.
https://developer.nvidia.com/blog/build-enterprise-retrieval-augmented-generation-apps-with-nvidia-retrieval-qa-embedding-model/

11. **NVIDIA Technical Blog, Develop Production-Grade Text Retrieval Pipelines for RAG with NVIDIA NeMo Retriever [primary]**
retrieved 2026-11-12.
https://developer.nvidia.com/blog/develop-production-grade-text-retrieval-pipelines-for-rag-with-nvidia-nemo-retriever

12. **IBM Granite Embedding models, GitHub [primary]**
retrieved 2026-11-12.
https://github.com/ibm-granite/granite-embedding-models/

13. **Hugging Face, The Smol Training Playbook [primary]**
retrieved 2026-11-12.
https://huggingfacetb-smol-training-playbook.hf.space/

14. **Fish Audio, Launching Fish Audio S1, 2025-11-20 [primary]**
retrieved 2026-11-12.
https://fish.audio/blog/introducing-s1/

15. **ACM DL, Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing, 10.1145/3458754**
retrieved 2026-11-12.
https://dl.acm.org/doi/10.1145/3458754

16. **NVIDIA Technical Blog, Microsoft Research Unveils New BERT-Based Biomedical NLP AI Model**
retrieved 2026-11-12.
https://developer.nvidia.com/blog/microsoft-unveils-new-bert-based-biomedical-nlp-ai-model/

17. **arXiv 2303.17564v3, BloombergGPT: A Large Language Model for Finance**
retrieved 2026-11-12.
https://arxiv.org/html/2303.17564v3

18. **arXiv 2010.02559, LEGAL-BERT: The Muppets straight out of Law School**
retrieved 2026-11-12.
https://arxiv.org/abs/2010.02559

19. **ACL Anthology, Why Compact LLM Verifiers for Emotion Classification Fail..., EMNLP 2026**
retrieved 2026-11-12.
https://aclanthology.org/2026.emnlp-main.1049.pdf

20. **InfoQ, How GitHub Copilot Serves 400 Million Completion Requests a Day (Cheney presentation)**
retrieved 2026-11-12.
https://www.infoq.com/presentations/github-copilot/

21. **neurosymbolic.org reading-group notes, Productivity Assessment of Neural Code Completion**
retrieved 2026-11-12.
https://www.neurosymbolic.org/reading-group-notes/Productivity%20Assessment%20of%20Neural%20Code%20Completion%20%201316e8b77df7492eb000c6146ceee1f8.html

22. **AANN 2025, RepoFusion-in-Decoder: Efficient Cross-File Code Completion via Lightweight Encoder Fusion, 2025-08-15**
retrieved 2026-11-12.
https://doi.org/10.1109/aann66429.2025.11257694

23. **Unstructured, Best Embedding Model for RAG: What You Need to Know**
retrieved 2026-11-12.
https://unstructured.io/blog/understanding-embedding-models-make-an-informed-choice-for-your-rag

24. **AI Engineering Handbook, Reading the Landscape of Production Models**
retrieved 2026-11-12.
https://aiengineeringhandbook.com/llm/5.common_models.html

25. **fastino.ai, A Guide to Small Language Models**
retrieved 2026-11-12.
https://fastino.ai/blog/a-guide-to-small-language-models