Episode Book

EPISODE #6014

Why Every AI Model Is Secretly a Decoder

PUBLISHED	15 October 2026 at 20:23 UTC
DURATION	21:53
LISTEN	myweirdprompts.com/episode/decoder-only-transformer-architecture-43038/
HOSTS	Corn, Herman, Larry, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.5
SOURCES	10 cited by the research (listed at the end)
BOOK GENERATED	19 October 2026 at 17:42 UTC

SYNOPSIS

Encoder-only, encoder-decoder, decoder-only — three shapes, one skeleton. This episode pulls apart the architectural fork at the center of modern language models: how attention masking and training objectives separate BERT from GPT, why the decoder-only design swallowed the field, and why encoder-only models still quietly run classification and retrieval at a fraction of the cost. Along the way: the four unrelated things this field calls "encoding," the parts bin that top labs actually assemble models from, and where the real architectural innovation is happening now.

TRANSCRIPT



Corn

Daniel has spent the last few weeks pulling this show apart layer by layer. Inference. Forward passes. Autoregressive decoding. Today he wants the skeleton underneath all of it. His question is why some language models have a dedicated encoder and a decoder while others only ever have a decoder, and what the decoder is actually doing in that second case when there is no encoder feeding it anything.



Herman

It's a good place to stop and look up, because everything we've covered so far has been about what happens inside the forward pass. This is about what shape the thing doing the forward pass actually is.



Corn

He wants it conceptually and from the model development side. The three architecture families, which one dominates and why, how a lab actually chooses its components when it builds one of these, how much any of that determines what you get at the end, and where the real innovation is happening now. And he asked specifically to keep the terminology straight, because encoder and decoder get used for four different things in this field and people mix them constantly.



Herman

That last part is worth saying once, up front. A neural network encoder is a stack of layers. A tokenizer turns text into token IDs. An embedding layer turns token IDs into vectors. And a decoding algorithm at inference time is greedy or beam or sampling. Those are four unrelated things that all get called encoding or decoding depending on who's talking.



Corn

So the actual architectural fork. What are the three shapes and where does the input token go in each?



Herman

One transformer stack, three different ways of wiring the attention and three different training objectives. Encoder-only, the BERT shape. Bidirectional attention, so every token can see every other token in the input, and it's trained by masking words out and predicting them. It has no native generation at all. You bolt a classification head on top and it tells you what the input is.



Corn

An identifier, not a writer.



Herman

That's the honest description of it, yes. Encoder-decoder, the T5 and BART shape. The encoder runs bidirectionally over the input, the decoder generates autoregressively, and the two are connected by cross-attention, so the decoder can look back at everything the encoder produced. Then decoder-only, the GPT and Llama shape. Causal attention, every token sees only the tokens before it, trained to predict the next token.



Corn

And the interesting claim is that the first and the third are the same machine.



Herman

Architecturally identical. There's a paper, the title is literally asking how powerful decoder-only transformer models are, and its finding is that decoder-only and encoder-only models differ in attention masking and objective, not in fundamental structure. Same blocks, same layers, same math. BERT just can't recurse. A decoder-only model can generate at every time step and feed its own output back in.



Corn

So the difference between knowing and writing is a triangle of ones and zeros.



Herman

A mask. That's the whole trick. Let me get the mechanism clear though, because the shapes matter more than the label. In an encoder, position seven can attend to position four hundred. Everything sees everything, in both directions, and that's how it builds a representation of the whole input at once. In a decoder-only model, position seven attends to positions one through seven and nothing else.



Corn

Which sounds like a crippling handicap until you remember what it buys you.



Herman

It buys you parallelism during training. Because position i can never see position j when j is greater than i, the model can be shown an entire document in one pass and every single position predicts its own next token simultaneously. One forward pass over a thousand tokens gives you a thousand training signals. You couldn't do that with bidirectional attention without leaking the answers.



Corn

So the prompt just gets prepended.



Herman

The prompt is tokens like any other tokens. They go through the same stack, the same attention, the same feedforward layers. There is no separate encoding pass because there is no separate source sequence. The model builds up a hidden state across the prompt, and that hidden state is what conditions the generation that follows. The understanding of the prompt is the state of the network at the end of reading it.



Corn

Which is maybe the cleanest way to put it. The encoder didn't disappear. It became the first half of the forward pass.



Herman

There's a lovely point Yi Tay makes about this. He says encoder-decoder models are actually still autoregressive models, and the decoder inside one is literally and fundamentally still a causal decoder. It looks like a different animal. It isn't.



Corn

So T5 is a decoder-only model that outsources part of its reading.



Herman

With a cross-attention bridge instead of a prepended prompt. And Tay goes further, he says encoder-decoders, decoder-only models and PrefixLMs are not that different altogether. The difference is where you draw the boundary between reading and writing, and which side pays for it.



Corn

Say more about the cross-attention bridge, because that's the part people picture wrong. The encoder isn't handing over a summary.



Herman

It's handing over a full sequence of vectors, one per input token, and at every generation step the decoder attends across all of them. So it isn't reading a compressed digest. It has the whole encoded input available at every step, through a separate attention path.



Corn

Then what's the bottleneck everyone talks about?



Herman

The channel between the two stacks, and the fact that the encoder has to have finished before the decoder can start. You can't stream the input and the output at the same time the way a decoder-only model does. For translation that's fine. For a model that's meant to be thinking and reading and writing in one continuous pass, it's a design constraint.



Corn

Here's where I'd push. If the encoder-decoder decoder is still a causal decoder, and the encoder is a transformer stack with a different mask, what is actually different between the three families in terms of what they can learn?



Herman

At the level of raw capacity, not much. The ceiling isn't the architecture. What differs is the objective you're allowed to train with, and the objective decides how much you get out of every unit of compute you spend.



Corn

So get to the famous objection. Why did everyone stop building BERT-style models?



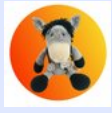
Herman

Tay's term for it is paradigm unification. It wasn't that denoising failed. It's that people wanted one model that could do everything, and an encoder-only model with a classification head on top can do exactly one thing. Summarize this. Classify this. Answer this. You need a new head for each task.



Corn

And a decoder-only model can be asked to do all of it in plain text.



Herman

Encoder-decoders and decoder-only models could express multiple tasks at once without needing task-specific heads. That flexibility won. Once prompting became the interface, the head-per-task model was dead.



Corn

And the second objection is about the training signal itself.



Herman

Denoising has less loss exposure. When you mask fifteen percent of the tokens, only those masked positions contribute to the loss. With causal language modeling, basically every position contributes. So a denoising model is getting a useful gradient out of a fraction of the tokens it paid to process.



Corn

How much of a fraction, roughly?



Herman

If you're masking fifteen percent, you're getting signal from fifteen percent of positions. A causal model gets it from close to a hundred. Tay's line is that this makes denoising hugely disadvantaged on a flop-basis comparison. You're paying to run the whole sequence through and only learning from a slice of it.



Corn

So the bitter lesson arrives in attention masks.



Herman

General method, more compute, less cleverness per token. And the encoder-decoder side had an additional scaling problem. The channel between the encoder and the decoder was cited as a real constraint on pushing to hundreds of billions of parameters. At small scale it's a useful inductive bias. At large scale it's a wall, at least on the compute-optimality axis.



Corn

Hold on. If that's all true, why is BERT still one of the most downloaded models on the planet?



Herman

Because classification and retrieval never went away. BERT-base alone pulled ninety million downloads on Hugging Face in a single month last year. ModernBERT revived the recipe. If your job is to decide whether a piece of text is relevant, an encoder is cheaper and better than a chatbot, and it always will be.



Corn

You'd never ask a decoder-only model to do that if you're paying per token.



Herman

You'd be burning money. There's a paired-models study, encoder-only and decoder-only versions trained on identical recipes, and a four hundred million parameter encoder beats a one billion parameter decoder on natural language inference. Three times the size, still loses. You can try to patch it with continued training on the other objective and the gap doesn't close.



Corn

And encoder-decoder?



Herman

Summarization, translation, question answering, anything where the input and the output are asymmetric and the input is fully available before you start writing. A four hundred million parameter BART beat GPT-2 at one and a half billion parameters on summarization back in 2020. T5 at twenty billion beat PaLM-1 at sixty-two billion on some positional understanding tasks. Those aren't flukes, they're the shape of the task suiting the architecture.



Corn

Larry? Sorry. Go on.

L

Larry

Are you tired of reading your own documents? Introducing ReadIt For Me, the subscription service that turns any text into a summary you didn't ask for. Bronze tier gets you three summaries a month, delivered by carrier pigeon between four and six weeks later. Silver tier upgrades you to summaries of documents you haven't read yet, which may or may not exist. Gold tier is the same as Silver, but the cancellation form is a fax you have to send from a specific telephone in Ohio.



Herman

I did not book that. I want to be clear that I did not book that, and I have never met anyone named Larry, and I have no knowledge of how the spot inventory works or who has the key to the ad server.



Corn

You've clearly given this a lot of thought.



Herman

Somebody has to think about these things.



Corn

Where were we. Asymmetric input and output.



Herman

Right, so the encoder-decoder family never lost on quality for those tasks. It lost on being general.



Corn

Now to the part that actually answers Daniel's third question. When a lab sits down to build a new model, what are they actually choosing?



Herman

They're choosing from a standard parts bin, and the differences between the top open models are a set of substitutions inside one template. GPT-2 and GPT-3 used layer norm, learned absolute position embeddings, standard multi-head attention, and GELU activations.



Corn

And Llama 3?



Herman

RMSNorm instead of layer norm, rotary position embeddings instead of learned ones, grouped query attention instead of full multi-head attention, and SwiGLU. Four swaps, same skeleton. Qwen3 at eight billion adds query-key normalization on top. Gemma 3 at twenty-seven billion goes with GeGLU and a five-to-one pattern of local attention windows and global attention.



Corn

Nobody's designing a new machine. They're picking a norm, a position scheme, an attention variant, and an activation.



Herman

And those components are portable across families, which is the tell that this is a parts bin. Grouped query attention shows up everywhere now. Rotery embeddings are close to universal. When a component wins, it propagates in weeks.



Corn

Then the question of how tightly the architecture and the weights are glued together. Because my instinct is not very tightly, and that feels important.



Herman

It's the most interesting result of the last year. T5Gemma. Google took Gemma 2, which is a decoder-only model, and built an encoder-decoder out of it by initializing the encoder from the pretrained decoder weights, then adapting with a different pretraining objective.



Corn

So they took the brain out of the decoder-only model and put it in a different skeleton.



Herman

And it worked, and not marginally. T5Gemma at nine billion by nine billion scores nine points higher on grade school math and four points higher on reasoning than Gemma 2 at nine billion, after pretraining. After instruction tuning, the two billion by two billion version jumps about twelve points on a general knowledge benchmark over its Gemma 2 counterpart, and math goes from fifty-eight to seventy point seven.



Corn

Those are the numbers that make you sit up.



Herman

And it also lets you build unbalanced models. A nine billion encoder with a two billion decoder. You couldn't do that if architecture were destiny, because that shape has no decoder-only equivalent.



Corn

So how much does any of this determine what the finished model can do?



Herman

It depends almost entirely on scale, and this is the through line of the whole episode. At small scale, architecture matters a lot. Encoder-decoder models at a billion parameters or below show a couple of points of quality improvement and about forty-seven percent lower first-token latency, with several times the throughput on dedicated edge hardware.



Herman (continued)

There's a study arguing that the two-thirds encoder, one-third decoder split consistently wins down there.



Corn

Because the encoder gets to look at the whole input before the decoder writes anything.



Herman

Which is exactly what a phone or a laptop-assist task needs. And then at large scale that advantage fades, because once you're spending most of your compute on pretraining, the objective's compute efficiency dominates, and causal language modeling wins that comparison.



Corn

So encoder-decoder is the better inference machine and the worse training machine.



Herman

For long prompts it's stark. One comparison found encoder-decoder using eleven to sixteen percent less memory and about seventy-eight percent of the flops of a decoder-only model for a two hundred and fifty-six token generation at a four thousand token input, and the gap widens as the input grows.



Corn

The more context you have, the more the encoder earns its keep.



Herman

Up to a point. Somewhere above a certain scale the bottleneck costs more than the efficiency saves, and nobody has pinned down where that crossover is.



Corn

Raschka's caution is the one to keep in your pocket here though. He says the architecture alone does not determine model quality. Parameter count, tokenizer, training data, optimization, context length training, post-training, all of it can matter at least as much.



Herman

Which is why the honest answer to Daniel's question about quality is that it's real but it's bounded. Two models with the same architecture and different data are further apart than two models with different architectures and the same data.



Corn

Where's the innovation actually happening then?



Herman

Almost entirely inside the stack, not at the level of the encoder-decoder boundary. Grouped query attention, sliding window attention, multi-head latent attention out of DeepSeek V3. Linear attention hybrids like Gated DeltaNet. Text diffusion models that drop the causal mask entirely, which makes them bidirectional like BERT but historically they get called decoder-style anyway.



Corn

And the linear attention story has a genuine fight in it.



Herman

MiniMax shipped M2 and abandoned linear attention, saying it was tricky in production, fine on ordinary prompts but poor on reasoning and multi-turn work. Then Kimi Linear came out claiming seventy-five percent KV cache reduction and up to six times the decoding throughput, and Qwen3-Next shipped a three-to-one hybrid enabling native context of two hundred sixty-two thousand tokens.



Corn

So one lab walks away and two walk toward it in the same quarter.



Herman

That's the correct read. It's unresolved, and anybody telling you subquadratic attention has won is guessing.



Hilbert

Prepend. That's the word I'd take issue with.



Herman

In what sense?



Hilbert

You said the prompt is prepended and then the model reads it. I spent the middle seventies wiring encoder and decoder boards for a shop off the Alameda, and the whole point of those two circuits was that they were separate. The encoder took the input, the decoder produced the output, and the wire between them was a real wire. Then we'd have a problem where the thing worked on a bench and not in the cabinet, and it always came down to one man.



Corn

There's usually one man.



Hilbert

Name was Vasquez. He had a routing rule he never wrote down. Ran everything at a length measured to the inch, and if the run came out an inch over he'd stop and get the wire cut to the exact same length as every other one in the bundle. Kept a notebook with the measurement logged for every cable, going back two years. I asked him why and he said a cable that's a different length behaves differently, and he wasn't wrong, he just couldn't tell you how he knew. So the prompt isn't prepended. The prompt is the first length of wire,



Hilbert (continued)

and there never was a second circuit.



Herman

Which is a lot closer to the truth than it sounds.



Hilbert

I have a question about this and the two of you can settle it. What I could never get a straight answer on in those years was whether the input and the output being the same wire made it cheaper or smarter. Was the encoder on the bench actually doing work, or was it ceremony.



Corn

It was doing work. Encoder-decoder models use less memory and fewer flops on long inputs, which is the opposite of ceremony. It's just that the work it does is more valuable on some tasks than others.



Hilbert

That settles it, then.



Herman

Do you still have the notebook?



Hilbert

It wasn't mine.



Corn

That doesn't answer the question.



Hilbert

It belonged to a man named Vasquez, and I've thought about him for fifty years for one entirely practical reason. He measured every wire in that bundle to the inch and never once measured a prompt. Both of those facts are the same fact, and neither of us is going to be able to explain it, so let's leave it.



Herman

Here's what I'll take from that, though. The wire was one wire, and the only question that ever mattered at the bench was where you put the boundary.



Corn

Which is exactly the design question. And the open one. Nobody has published the scale at which the encoder-decoder bottleneck stops being worth it and starts being a liability.



Herman

The small-scale studies say the split wins. The large-scale ones say the general objective wins. And there's a gap in the middle that nobody has measured, and it's not for lack of trying, it's hard to isolate.



Corn

Which makes it a design judgment, not a lookup.



Herman

And the T5Gemma result is what makes that judgment interesting. Because if you can take a decoder-only model's weights and drop them into an encoder-decoder skeleton and come out ahead of where you started, then architecture is closer to a container than a verdict. The weights carry most of what the model knows. The architecture decides what constraints it thinks in and what it costs to run.



Corn

Same contents, different jar. Choose the jar by what you're storing.



Herman

Which is also why listing one architecture as best is a category error. The right answer depends on which constraint is binding. Pretraining compute, inference latency, memory, or the shape of the task. Decide which of those is actually scarce and the architecture picks itself.



Corn

And the linear attention fight is the same question wearing a different hat. It buys you throughput and context length, and MiniMax's answer was that it costs you reasoning, and both of those claims can be true at once.



Herman

Someone will pin down that crossover point. When they do, half the current designs are going to look obviously correct in hindsight and the other half are going to look bizarre.



Corn

The one thing worth holding onto is that none of this is a measure of intelligence. It's a set of trade-offs about where reading ends and writing begins, and how much you want to pay for each.



Herman

Say that again but shorter.



Corn

The mask decides the behavior. The weights decide the knowledge. Not the other way around.



Herman

That's the one. On that note, our thanks to Hilbert Flumingtop, who produces this show and measures cables to the inch.



Corn

If this was your kind of episode, go back for episode twenty, Architectural AI; episode twelve, The AI Breakthrough; and episode eighteen, Beyond the GPU. This has been My Weird Prompts. If this made the architecture click, leave us a review, it helps. Send us your own prompt on Telegram at t dot me slash MWP listener bot. We'll be back soon.

SOURCES

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

1. **Google Developers Blog, T5Gemma: A new collection of encoder-decoder Gemma models, 2025-07-09 [primary]**
   https://developers.googleblog.com/en/t5gemma/

2. **Yi Tay, What happened to BERT & T5? On Transformer Encoders, PrefixLM and Denoising Objectives**
   https://www.yitay.net/blog/model-architecture-blogpost-encoders-prefixlm-denoising

3. **arXiv:2510.26622, Encoder-Decoder or Decoder-Only? Revisiting Encoder-Decoder Large Language Model, 2025-10-30**
   https://arxiv.org/abs/2510.26622

4. **arXiv:2501.16273v2, Return of the Encoder: Maximizing Parameter Efficiency for SLMs, 2025-01-27**
   https://arxiv.org/html/2501.16273v2

5. **arXiv:2507.11412v2, Seq vs Seq: An Open Suite of Paired Encoders and Decoders**
   https://arxiv.org/html/2507.11412v2

6. **Sebastian Raschka, GPT, Llama, Qwen, and Gemma Architectures Compared**
   https://sebastianraschka.com/faq/docs/gpt-llama-qwen-gemma-comparison.html

7. **Sebastian Raschka, Beyond Standard LLMs, 2025-11-04**
   https://magazine.sebastianraschka.com/p/beyond-standard-llms

8. **Hugging Face LLM Course, Transformer Architectures**
   https://huggingface.co/learn/llm-course/en/chapter1/6

9. **arXiv:2305.17026, How Powerful are Decoder-Only Transformer Neural Models?**
   https://arxiv.org/html/2305.17026

10. **Hacker News thread on Yi Tay / Latent Space podcast, 2024-07-05**
    https://news.ycombinator.com/item?id=40886118