MY WEIRD PROMPTS

Podcast Transcript

EPISODE #182

Beyond the Transformer: The New AI Architecture Wars

Published January 06, 2026 • Runtime: 26:44

https://myweirdprompts.com/episode/ai-architectures-beyond-transformers/

EPISODE SYNOPSIS

For years, the transformer has been the undisputed king of AI, but its "quadratic bottleneck" is starting to show its age. In this episode, Herman and Corn dive into the 2026 landscape of alternative architectures like Mamba, RWKV, and x-LSTM that promise linear scaling and infinite context. Discover how hybrid models are combining the reasoning power of attention with the efficiency of state-space models to redefine what’s possible in language modeling.

TRANSCRIPT

Corn

Hey everyone, welcome back to My Weird Prompts. We are coming to you from a somewhat chilly Jerusalem morning. I am Corn, and as always, I am joined by my brother.

Herman

Herman Poppleberry, at your service. It is good to be here, Corn. I have been buzzing since we finished our last session.

Corn

I can tell. You have been pacing around the kitchen with your nose in a research paper all morning. Our housemate Daniel actually caught some of that energy and sent us a follow up prompt. He mentioned he really enjoyed our discussion in episode two hundred seventy-seven about weights and tensors, but he wants to push the envelope a bit further.

Herman

Daniel always knows how to pick the right thread to pull. He was asking about large language model architectures, specifically moving beyond the transformer.

Corn

Exactly. We have talked about mixture of experts before, but Daniel pointed out that even those are usually just a specific way of organizing a transformer. He wants to know what else is out there. What are the non-transformer architectures that are actually making waves in twenty twenty-six?

Herman

This is such a timely question because for the last eight or nine years, the transformer has been the undisputed king. It is like we have been living in a monoculture. But in the last eighteen months, we have seen this incredible explosion of alternative ideas that are finally starting to challenge the status quo.

Corn

It is funny because in the tech world, we tend to think of the transformer as the end of history for language modeling. But you and I both know that science never really stops there. So, before we dive into the alternatives, maybe we should briefly frame why we are even looking for them. What is the fundamental itch that the transformer just cannot scratch?

Herman

That is the perfect place to start. The transformer, which was introduced in that famous paper Attention Is All You Need back in twenty seventeen, relies on a mechanism called self-attention. When a transformer processes a sequence of text, every single word, or token, looks at every other word in that sequence to understand the context.

Corn

Right, it is like a giant dinner party where every guest is trying to maintain a separate, meaningful conversation with every other guest simultaneously.

Herman

That is a great analogy. And as long as the party is small, say five or ten people, it works beautifully. But if you have a thousand people in the room, the number of conversations grows quadratically. In technical terms, the computational cost and the memory requirements of a transformer scale with the square of the sequence length. If you double the length of the text you are processing, you do not just double the work. You quadruple it.

Corn

And that is why we have historically had these context windows that were quite limited. Back in the day, it was a few thousand tokens. Then we pushed it to thirty-two thousand, then a hundred thousand. But the cost of going to a million or ten million tokens using pure attention is just astronomical.

Herman

Exactly. It is the quadratic bottleneck. Plus, transformers have a fixed memory footprint during inference. They have to keep around this thing called the key value cache, which grows linearly with the sequence length. If you want to have a conversation that lasts for a whole book, the model eventually runs out of memory or becomes incredibly slow. In twenty twenty-five, we saw models like Llama four and DeepSeek version three pushing the limits of this, but the energy costs were becoming a global concern.

Corn

So Daniel's prompt is really asking: how do we get the intelligence of a transformer without that quadratic baggage? And that leads us to the first major alternative, which is actually a bit of a blast from the past. We are seeing a renaissance of Recurrent Neural Networks, but with a modern twist. Specifically, the architecture known as RWKV and the newer x-L-S-T-M.

Herman

Oh, I love the story of RWKV. For those who do not know, it stands for Receptance Weighted Key Value. It was largely pioneered by Peng Bo. What makes it fascinating is that it is essentially a recurrent neural network that can be trained like a transformer. In early twenty twenty-five, they released RWKV-seven, codenamed Goose, which introduced something called dynamic state evolution.

Corn

Wait, let us pause there for a second for the listeners. Why does being trained like a transformer matter? Because back in episode two hundred seventy-three, we talked about how traditional recurrent neural networks were replaced because they were hard to parallelize, right?

Herman

Spot on. Old school recurrent neural networks, like L-S-T-Ms, which are Long Short-Term Memory networks, had to process tokens one by one. You could not start calculating word ten until you were finished with word nine. That meant we could not use the massive parallel processing power of modern chips. Transformers won because you could feed the whole sentence in at once during training.

Corn

So how does RWKV and this new x-L-S-T-M get around that? How can they be both recurrent and parallel?

Herman

They use very clever mathematical formulations. RWKV uses a specialized linear attention that can be written as a convolution during training, making it highly parallelizable. x-L-S-T-M, which was a huge comeback for Sepp Hochreiter's team in twenty twenty-four, uses exponential gating and a matrix memory. It actually proved to be Pareto-dominant over transformers in many benchmarks by late twenty twenty-five. It is like a classic car that has been fitted with a warp drive.

Corn

So it is like a shapeshifter. It behaves like a transformer when it is learning, but like a classic recurrent network when it is talking.

Herman

Precisely. And because it is recurrent during inference, its memory usage is constant. It does not matter if you have fed it ten words or ten thousand words. The state it carries forward is always the same size. It does not have that quadratic attention bottleneck. It is essentially an attention-free transformer.

Corn

That sounds like a massive win for efficiency. But does it actually perform as well? I remember reading that recurrent networks often suffer from forgetting things. If the sentence is too long, the beginning of the sentence just fades away into the hidden state.

Herman

That has always been the Achilles heel of recurrence. However, RWKV-seven and x-L-S-T-M use sophisticated state-tracking that allows them to recognize regular languages and maintain state in ways even transformers struggle with. As we have moved into twenty twenty-six, the gap has narrowed significantly. For long form content generation or streaming applications where you need low latency, they are incredibly compelling.

Corn

It feels like a very elegant solution to the memory problem. But it is not the only one. If we move away from the recurrent side of things, there is this whole other category that has been making huge waves lately. I am talking about State Space Models.

Herman

Now we are getting into the real heavy hitters. If twenty twenty-four was the year of the transformer, twenty twenty-five was the year Mamba took over the conversation. Mamba is the most famous implementation of what we call Selective State Space Models. And in mid-twenty twenty-four, we got Mamba-two, which introduced Structured State Space Duality, or S-S-D.

Corn

I have seen the name Mamba everywhere in the technical forums. It was developed by Albert Gu and Tri Dao, right? Explain the core idea there, because it is quite a departure from how we usually think about these models.

Herman

It is. State Space Models actually come from a long tradition in control theory and signal processing. Imagine you have a system that changes over time, like a rocket's trajectory or a chemical reaction. You have a hidden state that represents the system, and you update that state based on new inputs.

Corn

Okay, so it is like a continuous stream of information rather than discrete blocks of text?

Herman

Sort of. The breakthrough with Mamba was making these models selective. Traditional State Space Models were linear and time-invariant, which meant they treated every piece of information with the same weight. That is terrible for language because some words are much more important than others.

Corn

Right, if I say the cat sat on the mat, the words cat and sat are much more important for the meaning than the word the.

Herman

Exactly. Mamba introduced a mechanism where the model can decide, based on the input, what to remember and what to ignore in its hidden state. It is like having a notepad where you only write down the important bits of a conversation. Because it uses this state space approach, its complexity is linear, not quadratic. Mamba-two actually proved that these models are mathematically a form of structured attention, which was a huge unifying moment for the field.

Corn

Linear scaling. That is the holy grail. If you have a document that is twice as long, it only takes twice as much compute.

Herman

And it is fast. Like, incredibly fast. Because it does not have to look back at every previous token for every new token it generates, the throughput is much higher than a transformer. We are talking about models that can handle context windows of millions of tokens on consumer grade hardware that would choke on a transformer.

Corn

I remember we touched on this briefly in episode two hundred seventy-five when we were talking about supercomputing. The bottleneck there is often moving data around. If the model does not have to keep looking back at a massive cache of previous words, you save a lot of energy and time.

Herman

Absolutely. But here is where it gets interesting, Corn. We are starting to see that these architectures do not have to exist in a vacuum. One of the biggest trends right now in early twenty twenty-six is the rise of hybrid models.

Corn

You mean like Jamba? I think that was one of the first big ones to mix things up.

Herman

Yes, Jamba from A-I twenty-one Labs was a pioneer there. In March twenty twenty-five, they released Jamba one point six, which used a one-to-seven ratio of attention layers to Mamba layers. The idea is that you use a few attention layers to handle the really complex, non-linear reasoning that transformers are great at, and you use Mamba layers for the bulk of the sequence processing to keep the efficiency high.

Corn

That makes a lot of sense. It is like having a team where you have one genius who looks at everything very deeply but gets tired easily, and a group of very efficient workers who handle the flow of information. You get the best of both worlds.

Herman

It really is a hybrid vigor situation. By interleaving these layers, you get a model that has the high quality reasoning of a transformer but with a much smaller memory footprint for the key value cache. It allows for much longer contexts without the performance degradation you might see in a pure state space model. We even have Jamba Reasoning three billion models now that run locally on phones but reason like much larger models.

Corn

It is fascinating to see how the industry is pivoting. It is no longer just about making the transformer bigger. It is about rethinking the underlying math. But I want to go back to something you mentioned earlier. What about the idea of retention? I have seen some papers on RetNet, or Retentive Networks. How does that fit into this landscape?

Herman

RetNet is a very strong contender from Microsoft Research. They actually framed it as the successor to the transformer. They talk about an impossible trinity in large language models.

Corn

An impossible trinity? That sounds like a challenge. What are the three points?

Herman

The three points are training parallelization, low latency inference, and good performance. Usually, you can only pick two. Transformers have parallel training and great performance, but high latency inference for long sequences. Recurrent networks have low latency inference and good performance, but they are hard to parallelize during training.

Corn

And RetNet claims to hit all three?

Herman

That is the claim. It uses a retention mechanism that, similar to RWKV, has multiple representations. It can be written as a parallel representation for training, a recurrent representation for efficient inference, and even a chunk-wise representation for processing very long sequences in blocks.

Corn

How does retention differ from attention, though? Is it just a rebranding, or is there a functional difference?

Herman

There is a functional difference in how the decay is handled. In attention, every token is theoretically equal unless the model learns otherwise. In retention, there is an explicit biological-like decay built into the math. Things that happened further back in the sequence naturally have a lower weight unless they are specifically reinforced. This mirrors how human memory works a bit more closely.

Corn

That is an interesting philosophical shift. We are moving from a model that tries to hold everything in a perfect, high-resolution snapshot to a model that understands the flow of time and the relative importance of history.

Herman

Exactly. And it turns out that for most language tasks, that is actually a more efficient way to represent the world. We do not need to remember every single comma from five hundred pages ago to understand the current sentence, but we do need to remember the name of the protagonist.

Corn

So, we have talked about RWKV, Mamba, Jamba, and RetNet. These all seem to be tackling the same problem of efficiency and context length. But are there any architectures that are completely different? Something that does not even look like a sequence model?

Herman

Well, there is some very experimental work in what people are calling Graph Neural Networks for language. Instead of treating text as a flat line of tokens, these models try to build a knowledge graph on the fly as they read. But the real weird one making waves right now is T-T-T, or Test-Time Training layers.

Corn

Test-Time Training? That sounds like the model is still in school while it is working.

Herman

That is exactly what it is! Developed by researchers at Stanford and Berkeley, T-T-T layers replace the hidden state of an R-N-N with a tiny machine learning model. As the model reads your prompt, it actually trains that internal model on the sequence. It is the ultimate adaptive architecture. It can keep reducing perplexity even after millions of tokens because it is literally learning your specific context as it goes.

Corn

That is wild. It is like a model that grows a new part of its brain specifically for the conversation you are having. We are also seeing Liquid Neural Networks, coming out of M-I-T. These are inspired by the nervous system of tiny organisms like nematodes. They use differential equations to describe the state of the neurons.

Herman

Yes, they are incredibly robust to changes in the input frequency or noise. While they have mostly been used for robotics and time-series data, there are researchers trying to scale them up for language. The idea is that the model's parameters are not fixed; they can change and flow based on the input.

Corn

That feels like the next frontier. We are moving from these rigid, monolithic structures to things that are more fluid and organic. It makes me wonder what this means for the average person using these tools. If I am using an A-I in twenty twenty-six, should I care if it is a transformer or a Mamba model?

Herman

In most cases, you will not see the architecture under the hood. But you will feel the effects. You will notice that you can upload a thousand-page P-D-F and the A-I responds instantly. You will notice that your phone can run a very capable assistant locally without draining the battery in twenty minutes.

Corn

That is the real takeaway, isn't it? The non-transformer architectures are what will allow A-I to move from these massive data centers in the desert into our pockets and our edge devices.

Herman

Absolutely. Efficiency is the key to democratization. If it costs a hundred thousand dollars in electricity to train a model, only a few companies can do it. But if we can find architectures that are ten times more efficient, the barrier to entry drops significantly. We talked about this a bit in episode two hundred seventy-two when we were discussing optimizing websites for A-I bots. The more efficient the bots are, the more pervasive they become.

Corn

It is a fascinating evolution. It is like we are watching the Cambrian explosion of A-I architectures. For a while, the transformer was the only predator in the sea, and now all these new forms are emerging to fill different niches.

Herman

I love that analogy. And just like in evolution, some of these will be dead ends, but others will become the foundation for the next decade of progress. I suspect the winner will not be a single architecture, but a synthesis. A model that can attend to details when it needs to, but can also maintain a high-level state space for long-term memory.

Corn

It is also worth noting that as these architectures change, the way we train them has to change too. We can't just use the same old recipes.

Herman

That is a great point, Corn. A lot of the research right now is focused on how to properly initialize these models. With a transformer, we have years of intuition about how to set the weights. With something like Mamba or RetNet, we are still learning the best ways to get them to converge. It is a bit like learning to cook with a completely new type of stove.

Corn

You have to adjust the temperature and the timing.

Herman

Exactly. But the results are starting to speak for themselves. We are seeing Mamba-two models that outperform transformers of the same size on almost every benchmark. That was the turning point. Once the alternatives started winning on performance, not just efficiency, the industry really started to pay attention.

Corn

So, looking ahead to the rest of twenty twenty-six and into twenty twenty-seven, where do you see the most excitement? Is there a particular architecture that you think is going to become the new standard?

Herman

If I had to place a bet, I think the hybrid approach is the most likely winner for the near term. Pure attention is just too good at certain types of reasoning to give up entirely. But the quadratic cost is too high to keep it for everything. A model that uses attention sparingly, like a surgical tool, and uses something like Mamba or a modern R-N-N for the heavy lifting of context management, that seems like the winning formula.

Corn

It feels more balanced. More like how a human brain works, perhaps? We don't hold every single detail in our working memory at the same time. We have different systems for short-term and long-term storage.

Herman

That is exactly right. We have our prefrontal cortex for active manipulation of information, which is a bit like attention, and we have the hippocampus and other structures for more compressed, long-term representations. We are effectively building those same structural divisions into our silicon brains.

Corn

I think this is a great place to pivot to some practical takeaways for our listeners. Because while the math is fascinating, most people want to know how this affects their work or their daily life.

Herman

Definitely. The first big takeaway is that the era of the million-token context window is here to stay, and it is going to get cheaper. If you are a developer or a business owner, you should start thinking about how you would use an A-I that can remember every interaction you have ever had with it.

Corn

That is a huge shift in how we build applications. Instead of worrying about what to include in the prompt, you just include everything.

Herman

Right. The second takeaway is the rise of on-device A-I. Because these new architectures are so much more memory-efficient, we are going to see a huge leap in the quality of A-I running locally on laptops and phones. This has massive implications for privacy and for using A-I in offline environments.

Corn

I am personally looking forward to having a truly capable assistant on my phone that doesn't have to send every word I say to a server in another country.

Herman

We all are. And the third takeaway is for the more technically minded listeners: keep an eye on the software libraries. We are seeing a lot of work in making these non-transformer architectures compatible with existing tools. Things like the Hugging Face ecosystem are rapidly integrating support for Mamba and RWKV, which makes it easier for researchers to experiment.

Corn

It is all about lowering the friction. The easier it is to use these new models, the faster they will be adopted.

Herman

Exactly. And honestly, it is just an exciting time to be curious. We are no longer in a world where there is only one way to build a smart machine. The diversity of thought in the research community right now is higher than I have seen it in years.

Corn

Well, I think we have covered a lot of ground here. We went from the quadratic bottleneck of transformers to the recurrent shapeshifting of RWKV-seven and x-L-S-T-M, the selective state spaces of Mamba-two, the hybrid vigor of Jamba, and the impossible trinity of RetNet.

Herman

It is a lot to digest, but that is why we love these prompts. They force us to look at the whole map, not just the road we are currently on.

Corn

Absolutely. And before we wrap up, I want to say that if you are enjoying these deep dives into the guts of A-I and technology, we would really appreciate it if you could leave us a review on your podcast app or on Spotify. It genuinely helps other curious people find the show, and we love reading your feedback.

Herman

It really does make a difference. We are a small team here in Jerusalem, and knowing that people are finding value in these discussions is what keeps us going.

Corn

And if you have a question or a weird prompt of your own, like Daniel did, you can head over to our website at myweirdprompts dot com and use the contact form there. We would love to hear what is on your mind.

Herman

Maybe your prompt will be the basis for episode two hundred seventy-nine!

Corn

You never know. Alright, Herman, any final thoughts before we sign off?

Herman

Just that the future is not a straight line. We often think that technology just moves in one direction, but often it circles back to old ideas and breathes new life into them. The R-N-N was dead, and now it is back. Control theory was for rockets, and now it is for language. Stay curious, because the next big breakthrough might be hidden in a textbook from forty years ago.

Corn

I love that. A perfect note to end on. Thanks for listening to My Weird Prompts. You can find us on Spotify and at our website, myweirdprompts dot com.

Herman

Until next time, I am Herman Poppleberry.

Corn

And I am Corn. We will talk to you soon.