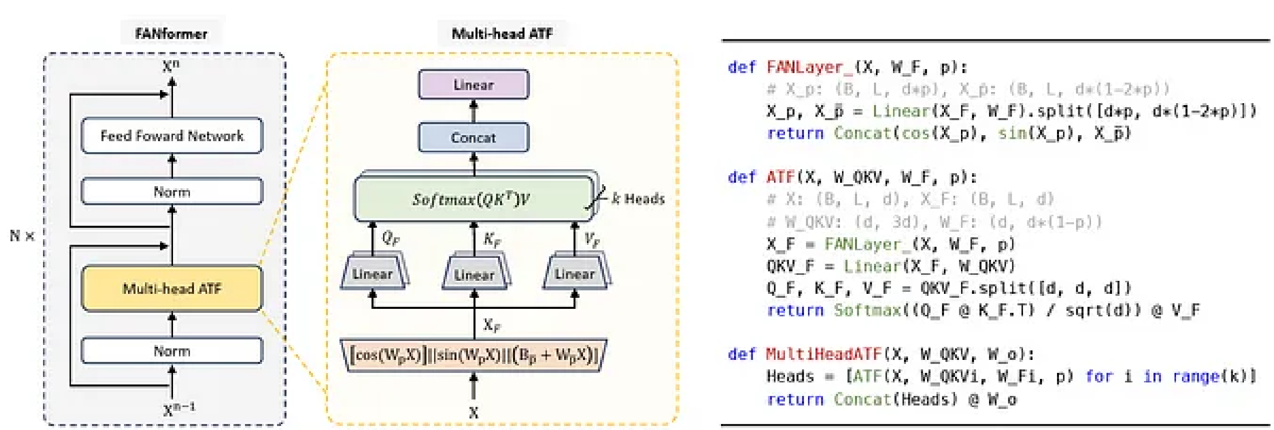
FANFormer: How to improve LLMs with Periodicity Modeling

Making Large Language Models Truly Understand Language with Fourier Fourier Analysis Networks. A math deep dive into the LLM breakthrough.

Cristian Leo · Following

Published in Data Science Collective · 17 min read · 3 days ago



Left: The illustration of FANformer's architecture. Right: The pseudocode of Multi-head ATF, where p is the hyperparameter that controls the proportion of periodicity modeling for Xp — Image extracted from the original FANFormer paper

LLMs are *hungry*. Data-hungry and compute-hungry. Training them requires mountains of data and immense computational power. It makes you wonder, doesn't it? Humans learn complex things with far fewer resources. We pick up languages, understand patterns, and generalize from relatively little input. This gap, this massive difference in efficiency, it suggests that maybe, just maybe, current LLM architectures are still missing something fundamental about how we actually learn and understand the world. Perhaps they're not quite as efficient at extracting the underlying knowledge from data as they could be.

Could Periodicity Be the Missing Piece?

Think about how humans learn. Life is full of rhythms, repetitions, and predictable patterns. We see it in nature, in music, in our daily routines, and, crucially, in language itself. This idea of **periodicity**, these recurring patterns, it's everywhere (Buzsaki, 2006; Lake et al., 2017). It's almost like our brains are wired to recognize and leverage these patterns to make sense of the world and learn efficiently. It seems that our brains are constantly looking for these repeating structures to process information and build knowledge in a more streamlined way (Zalta et al., 2020).

But here's the interesting question: are our current Transformer models, as amazing as they are, really capturing and using this inherent periodicity in language data as effectively as they could? Some researchers, like Dong and colleagues (Liu et al., 2020), are starting to suggest that maybe they aren't. They point out that standard Transformers might have some inherent limitations in modeling these periodic patterns, and this could be holding back their learning efficiency. It's like trying to build a house with fantastic bricks but overlooking the importance of a solid, rhythmic foundation.

Now, what if we could somehow equip LLMs to be better at recognizing and utilizing these periodic patterns? What if we could design an architecture that's more attuned to the rhythms of language? That's precisely where **FANformer** comes into play (Dong et al., 2025). This is a new architecture that tries to inject the power of **Fourier Analysis Networks (FAN)** into the attention mechanism of Transformers. The goal? To enable more efficient **periodicity modeling**, and in turn, boost the learning efficiency and overall performance of LLMs.

In this article, we're going to dive deep into FANformer. We'll see if this approach really can bridge that efficiency gap and unlock even more potential in the incredible world of Large Language Models. Is periodicity the secret ingredient we've been looking for? Let's find out together.

Understanding Periodicity and Transformers

Why Transformers Might Be Missing the Beat

Let's take a step back for a moment and really think about what we mean by "periodicity." It's a word we hear, but what does it *actually* mean in the context of learning and especially in something as complex as language?

The Ubiquitous Nature of Periodicity

Periodicity, at its heart, is about patterns that repeat. Think of the seasons changing, the rhythm of day and night, the ebb and flow of tides — these are all periodic phenomena in the natural world. But it's not just limited to grand, cosmic scales. Our own lives are structured by periodicity. We have daily routines, weekly cycles, and even longer patterns that shape our years. It's this very predictability, this inherent rhythm, that allows us to navigate the world with a certain degree of confidence. We anticipate, we prepare, we learn because things, to some extent, repeat.

And it's not a stretch to see periodicity in language too. Think about the rhythm of speech, the way sentences often follow predictable structures, the recurring themes and motifs in stories, or even the cyclical nature of conversations. Language isn't just a random jumble of words; it has its own patterns, its own beat, if you will. These patterns, these periodicities, might be subtle, they might be complex and layered, but they are undeniably there.

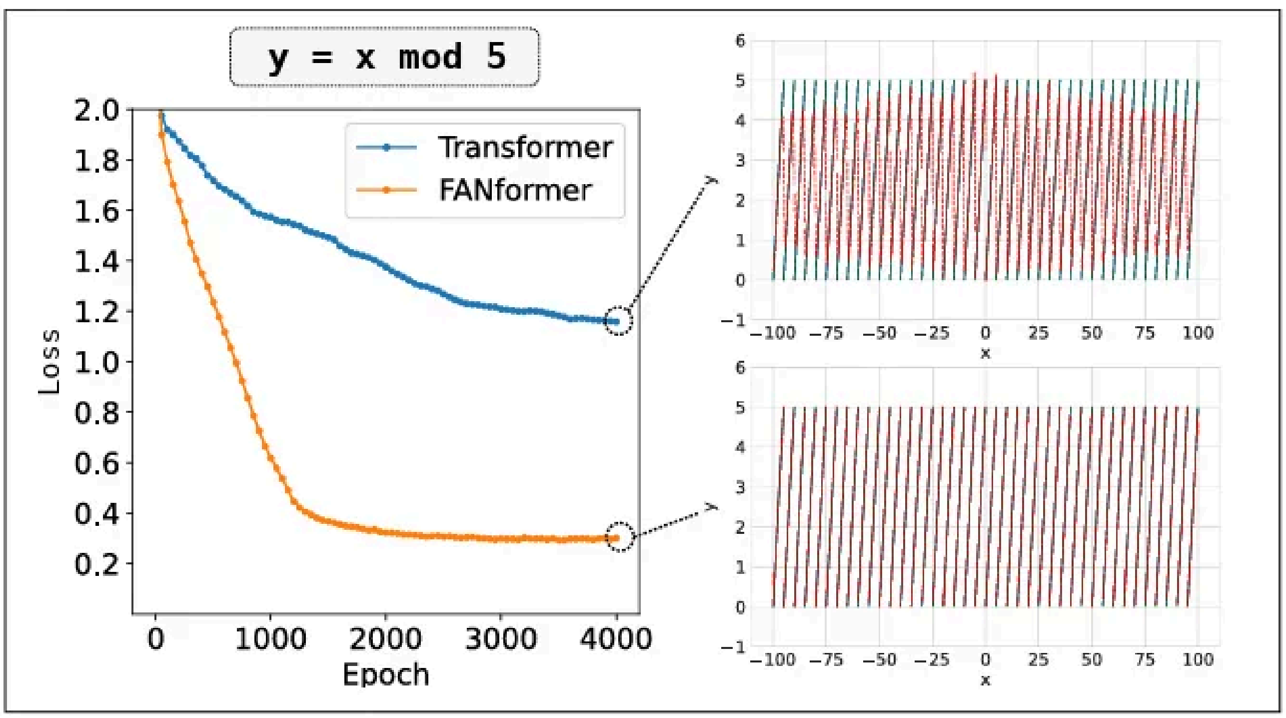
One could argue that our brains, having evolved in a world brimming with periodic patterns, have become incredibly adept at detecting and leveraging them. Perhaps this is why, as children, we can grasp language so intuitively, picking up on the subtle cues and repetitions that structure it. It's like learning to dance — you don't just memorize steps, you internalize the rhythm, the underlying periodic structure of the music, and then the movements flow more naturally.

Transformers: Powerful but Potentially Periodicity-Blind?

Now, let's bring Transformers back into the picture. These architectures, particularly the attention mechanism, are undeniably powerful. They've revolutionized NLP precisely because they are so good at capturing relationships between words in a sequence, regardless of distance. The attention mechanism allows the model to weigh the importance of different parts of the input when processing information. It's a bit like having a spotlight that can focus on the relevant parts of a sentence to understand its meaning.

However, while standard Transformers excel at capturing these relationships, their core mechanism might be inherently biased towards *locality* in time. Standard attention, with its linear projections, might be very effective at capturing immediate dependencies — words close to each other in a sentence that influence each other. But, are they as good at grasping those longer-range, periodic dependencies that might span across sentences, paragraphs, or even entire documents? This is where the question arises: are Transformers, in their standard form, truly optimized for capturing the essence of periodicity?

Some researchers suggest that the very strength of Transformers — their ability to focus on local relationships — might also be a slight weakness when it comes to periodicity. The linear projections they use might be very good at picking up on immediate context, but perhaps they are less equipped to explicitly model and represent patterns that repeat over longer stretches of text or data. It's like having a fantastic close-up lens but perhaps missing the wider, panoramic view.



(Figure 1a) Performance of Transformer and FANformer on periodicity modeling — Image extracted from the original FANFormer paper

To illustrate this, consider a very simple example from the FANformer paper (Dong et al., 2025). They showed that even for something as basic as learning a simple modulo function (think of the remainder after division, like "x mod 5"), a standard Transformer struggles, even when given enough data and model capacity. This is visualized in the figure above (Figure 1a of their paper). It's a seemingly simple periodic function, yet the Transformer doesn't quite nail it as efficiently as one might expect. This hints at a potential blind spot. If Transformers struggle with even *explicit* periodicity, what about the more *implicit*, nuanced periodicities hidden within the vastness of language data?

The Challenge of Periodicity Modeling in LLMs

This brings us to the heart of the matter: the challenge of periodicity modeling in Large Language Models. Language data, while not perfectly periodic in the strict mathematical sense, is certainly riddled with recurring patterns at various levels. From sentence structure to discourse organization, from stylistic choices to thematic repetitions, periodicity is woven into the fabric of language.

If LLMs are to truly master language, to learn as efficiently and generalize as effectively as humans, perhaps they need to become better at recognizing and leveraging these periodic patterns. If standard Transformers are indeed somewhat limited in this regard, then there's a real opportunity to improve their architecture. By explicitly incorporating mechanisms that are designed to model periodicity, we might be able to create LLMs that are not only more efficient learners but also more insightful and robust language understanders.

This is the exciting premise behind FANformer. It's an attempt to address this potential limitation by directly integrating Fourier Analysis Network principles into the Transformer's attention mechanism. The aim is to make these models more "periodicity-aware," to allow them to tap into the rhythmic pulse of language data and learn more effectively from it. But how exactly does FANformer achieve this? That's where we need to delve into the mathematics of it all, which is what we'll explore next.

FANformer: The Math Behind Periodicity-Aware Attention

Unpacking the Fourier-Inspired Attention Mechanism

So, how does FANformer actually weave in this idea of periodicity? The magic, it seems, lies in how it modifies the attention mechanism. At its core, FANformer leverages the **Fourier Analysis Network (FAN)**, a fascinating concept in itself, to bring Fourier principles into play (Dong et al., 2024b). Let's try to unpack this step by step, because while the idea is elegant, it does involve some mathematical underpinnings.

Fourier Analysis Network (FAN): Capturing Frequencies
The foundation of FANformer is, unsurprisingly, the Fourier Analysis Network, or FAN. Now, if you've ever dabbled in signal processing or music, you might be familiar with Fourier analysis. Essentially, it's a way to decompose complex signals into a sum of simpler sinusoidal waves — sines and cosines — of different frequencies. Think of it like breaking down a complex musical chord into its individual notes. Each note has a frequency, and the chord is just the combination of these frequencies.

FAN takes this principle and applies it to neural networks. The core idea is that instead of just using standard linear transformations in a neural network layer, we can explicitly encode periodic patterns by incorporating these sinusoidal functions. This is achieved through a specially designed **FAN Layer**.

The FAN Layer: Encoding Periodic Signals

The FAN Layer, as described in the FANformer paper (Dong et al., 2025), is the key component that infuses periodicity into the model. Let's look at its formula:

$$\mathrm{FANLayer}(X) = [\cos(W_p X) || \sin(W_p X) || \sigma(W_p' X + B_p)]$$

FAN Layer Formula — Image by Author

Now, let's break this down piece by piece. Here, X is our input — think of it as the hidden representation of the tokens at a particular layer in the network. W_p and W'_p are learnable projection matrices, just like the weights in any neural network layer. B_p is a bias term, also learnable. And sigma is a non-linear activation function — in the original FAN paper, and potentially here as well, it could be something like ReLU or GELU, though the FANformer paper, as we'll see later with the ATF, uses a slightly different variant. The symbol || represents concatenation, meaning we are joining the outputs of the different operations side-by-side.

The interesting part is the first two components: cos(W_pX) and sin(W_pX). These are where the Fourier magic happens. By multiplying the input X by the projection matrix W_p and then taking the cosine and sine of the result, we are essentially projecting the input into a frequency domain. Think back to our musical chord analogy. These cosine and sine functions are like capturing the different "frequencies" or periodicities present in the input data. It's a way to explicitly represent the periodic aspects of the input in a way that standard linear layers alone might not.

The third component, sigma(W'_pX + B_p), is more similar to a standard neural network layer. It's a linear transformation followed by a non-linearity.

This part, it seems, is meant to maintain the general-purpose modeling capabilities of a standard layer, allowing the FAN Layer to capture non-periodic features as well.

So, the FAN Layer is a clever hybrid. It combines the explicit periodicity encoding of the Fourier series (through the cosine and sine terms) with the general representation power of a standard neural network layer. This, in theory, should allow the network to be more sensitive to periodic patterns in the data while still retaining its ability to model more general features.

Attention-Fourier (ATF) Module: Periodicity in Attention

Now, how is this FAN Layer integrated into the attention mechanism to create the **Attention-Fourier (ATF)** module? The FANformer authors take a rather elegant approach. They don't completely overhaul the attention mechanism but instead modify the feature projection process *before* the attention calculation itself.

Let's recall the standard attention mechanism. It involves projecting the input into Query (Q), Key (K), and Value (V) matrices, and then computing attention weights based on Q and K to weight the values in V. FANformer's ATF module modifies this by first passing the input X through a variant of the FAN Layer, which they call FANLayer'. This variant is slightly different from the one we just discussed:

$$X_F = \text{FANLayer}'(X) = [\cos(W_p X) || \sin(W_p X) || (B_p + W_p X)]$$

FANLayer' Formula — Image by Author

Notice the key difference: the activation function sigma in the third component is replaced by the identity function — it's simply (B_p + W_pX). In the paper, they mention that for this specific application in the ATF module, they found that using the identity function worked better. Perhaps it's because they want to preserve more of the linear information at this stage, allowing the subsequent attention mechanism to better process the frequency-domain representations. It's a subtle but potentially important design choice, showing that even within the FAN framework, there's room for experimentation and fine-tuning.

Once we have this X_F, which now contains periodicity-encoded features, we use it to compute the Query, Key, and Value projections:

$$[Q_F, K_F, V_F] = X_F[W_Q, W_K, W_V]$$

Query, Key, and Value projections — Image by Author

Here, W_Q, W_K, W_V are again learnable projection matrices, just like in standard attention. The crucial point is that these projections are now *based* on X_F, the periodicity-enhanced representation of the input, rather than the raw input X itself.

Finally, the attention computation itself remains largely the same as standard attention:

$$\mathrm{ATF}(X) = \mathrm{softmax}\left(\frac{Q_F K_F^T}{\sqrt{d_h}}\right) V_F$$

ATF Formula — Image by Author

We compute the dot product of Q_F and K_F^T, scale it down by the square root of the dimension d_h, apply softmax to get attention weights, and then weight the values V_F using these weights. The core attention mechanism is still there, but it's now operating on Query, Key, and Value matrices that are derived from the periodicity-aware representation X_F.

In essence, FANformer doesn't throw out the Transformer's attention mechanism. Instead, it intelligently *augments* it. It pre-processes the input features through the FAN Layer to explicitly encode periodicity and then feeds these enhanced features into the standard attention calculation. It's a way of making the attention mechanism itself more sensitive to the rhythmic patterns that might be present in the data.

While this mathematical formulation provides a clear picture of how FANformer integrates periodicity, it's worth considering that this is just one approach. There could be other ways to inject periodicity into attention, perhaps by modifying the attention weights themselves or by using different types of periodic functions. However, the FANformer approach is appealing in its simplicity and elegance — it builds upon the existing Transformer framework in a minimally invasive yet potentially highly effective way.

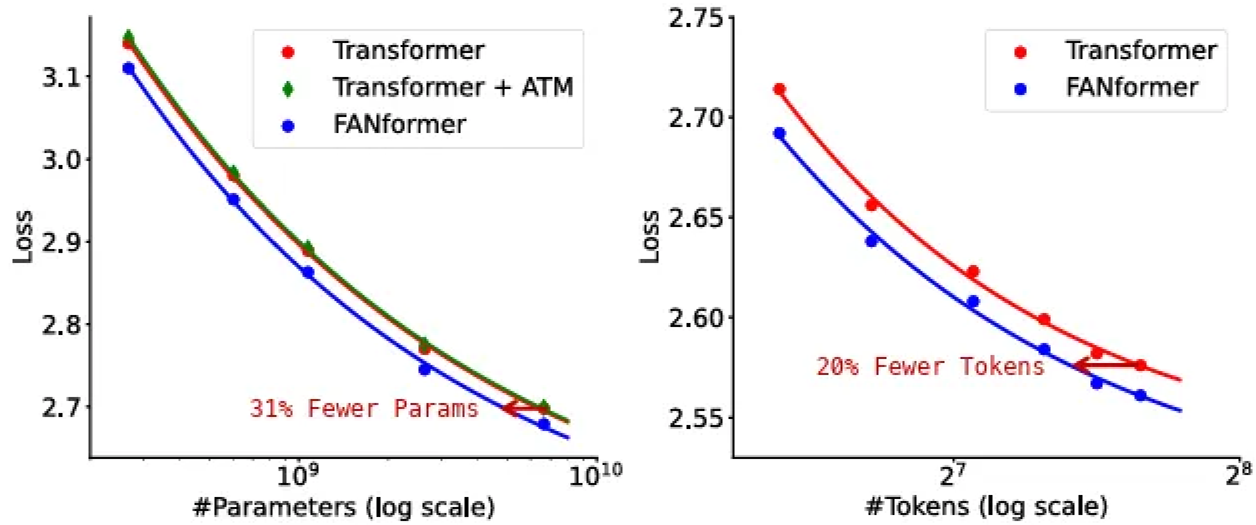
Advantages and Disadvantages of FANformer

Like any architectural innovation, FANformer brings potential benefits to the table, but also comes with its own set of considerations and limitations.

5.1 Advantages: Where FANformer Excels

Let's start with the positives. The FANformer paper (Dong et al., 2025) highlights several key advantages, and from our exploration, we can see why these benefits might arise.

Improved Learning Efficiency: Learning More with Less



(Figure 3) Language modeling loss of scaling up the model parameter and training tokens. Left: train LLMs from 268 M to 7 B parameters. Right: We evaluate 1.0 B LLMs every 20 B tokens up to 200 B tokens — Image extracted from the original FANFormer paper

One of the most compelling arguments for FANformer is its potential for **improved learning efficiency**. The experimental results in the paper, particularly Figure 1(a) and Figure 3 (image above) of the paper, do seem to suggest that FANformer can achieve comparable or even better performance than standard Transformers while using fewer resources.

Think about it: if FANformer is indeed better at capturing and leveraging the inherent periodic patterns in language, it's plausible that it can extract more signal from the same amount of data. It's like having a student who's particularly good at recognizing patterns — they might grasp concepts faster and need fewer examples to learn effectively compared to someone who's struggling to see the underlying structure. This is reflected in the faster convergence and better loss curves observed for FANformer in the paper, especially in the periodicity modeling tasks.

In the context of LLMs, this efficiency gain could be significant. It could translate to faster training times, reduced computational costs, and potentially even the ability to train high-performing models with smaller datasets. In a world where training LLMs is becoming increasingly resource-intensive, any architecture that promises better efficiency is certainly noteworthy.

Enhanced Scalability: Scaling Up Powerfully
Scalability is another crucial aspect for LLMs, and FANformer appears to hold promise in this area as well. Figure 3 in the paper demonstrates that FANformer maintains its performance advantage over standard Transformers as model size and training tokens are scaled up.

This suggests that the benefits of periodicity modeling aren't just limited to smaller models or datasets. Instead, they seem to scale effectively, becoming even more pronounced as we build larger and more powerful LLMs. Perhaps, as models grow in size, their capacity to extract and utilize complex periodic features also increases, and FANformer is particularly well-suited to capitalize on this.

This is quite encouraging. It implies that FANformer is not just a niche architecture for specific tasks, but could be a robust foundation for building the next generation of large-scale language models. The fact that it achieves comparable performance with fewer parameters (as noted in the paper, around 69.2% of the parameters of a standard Transformer for comparable performance in some scaling experiments) is a strong indicator of its parameter efficiency, a highly desirable trait for scalable models.

Better Generalization and Rule-Based Reasoning: Beyond Memorization
Beyond efficiency and scalability, FANformer also hints at improved **generalization** capabilities and a stronger aptitude for **rule-based reasoning**. This is a more nuanced advantage, but potentially very important.

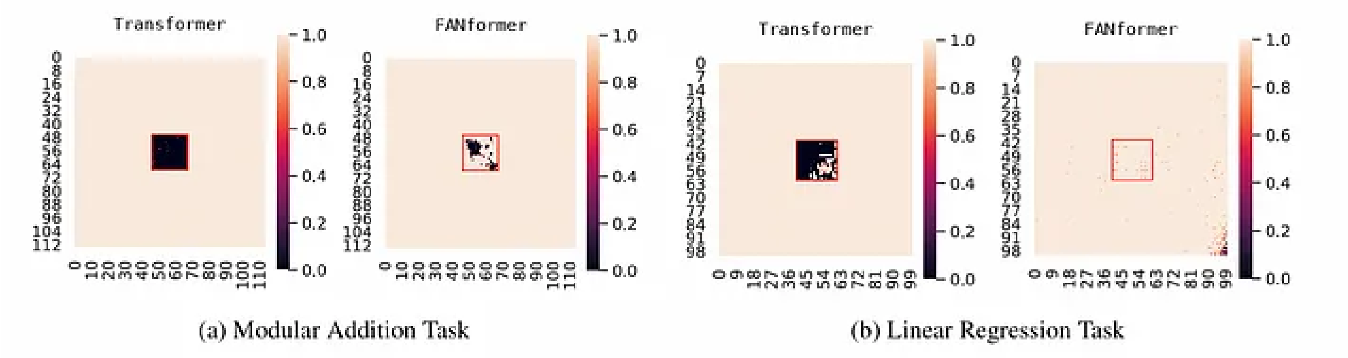


Figure 5: Performance of FANformer and Transformeron modular addition and linear regression tasks — Image extracted from the original FANFormer paper

Figure 5 in the FANformer paper, showcasing performance on modular addition and linear regression tasks, is quite revealing. Standard Transformers, while achieving near-perfect accuracy on the training set, struggle to generalize to unseen test data, exhibiting the "hole" phenomenon described by Hu et al. (2024). In contrast, FANformer seems to mitigate this issue, showing better generalization performance on these tasks.

Why might this be? Perhaps by explicitly modeling periodicity, FANformer is encouraged to learn underlying rules and principles, rather than simply memorizing patterns from the training data. Think of it like learning the rules of grammar versus just memorizing sentences. Rule-based learning, while potentially slower initially, often leads to better generalization to novel situations. FANformer's improved performance on these synthetic tasks, which require rule-based reasoning, suggests that it might indeed be nudging the model towards a more rule-based learning paradigm.

This is a fascinating prospect. If FANformer can truly enhance rule-based reasoning, it could lead to LLMs that are not just good at mimicking language, but also possess a deeper understanding of its underlying structure and logic, potentially making them more robust and reliable in real-world applications.

Periodicity Awareness: A More Natural Approach

Finally, and perhaps most fundamentally, the core advantage of FANformer is its explicit **periodicity awareness**. By design, it incorporates mechanisms to detect and utilize periodic patterns. As we discussed earlier, periodicity is a ubiquitous feature of human life and language, and it seems intuitive that models designed to be sensitive to these patterns might be better suited for understanding and generating language.

While standard Transformers are incredibly flexible and powerful, they are not explicitly designed to model periodicity. FANformer, by integrating Fourier principles, takes a more direct and intentional approach. It's like building a house with tools specifically designed for working with rhythmic structures, rather than just general-purpose tools. This explicit focus on periodicity might be what gives FANformer its edge in learning efficiency, scalability, and generalization.

5.2 Disadvantages and Limitations: Areas for Consideration

Of course, no architecture is perfect, and FANformer, while promising, also has its limitations and areas that warrant further consideration.

Increased Complexity: A Trade-off for Periodicity

One potential drawback is the **increased complexity** of the ATF module compared to standard attention. While FANformer aims for simplicity in integration, the FAN Layer itself does add a layer of computation, involving cosine and sine operations in addition to linear transformations.

While this added complexity might be relatively small, it's still a trade-off. In some resource-constrained scenarios, the added computational cost of the FAN Layer might be a factor. It's worth noting, however, that the FANformer paper emphasizes that the overall architecture remains relatively simple and efficient, and the experimental results suggest that the benefits in performance and efficiency often outweigh this added complexity. It's more about targeted complexity for a specific purpose (periodicity modeling) rather than general, unguided complexity.

Hyperparameter Tuning: The 'p' Factor

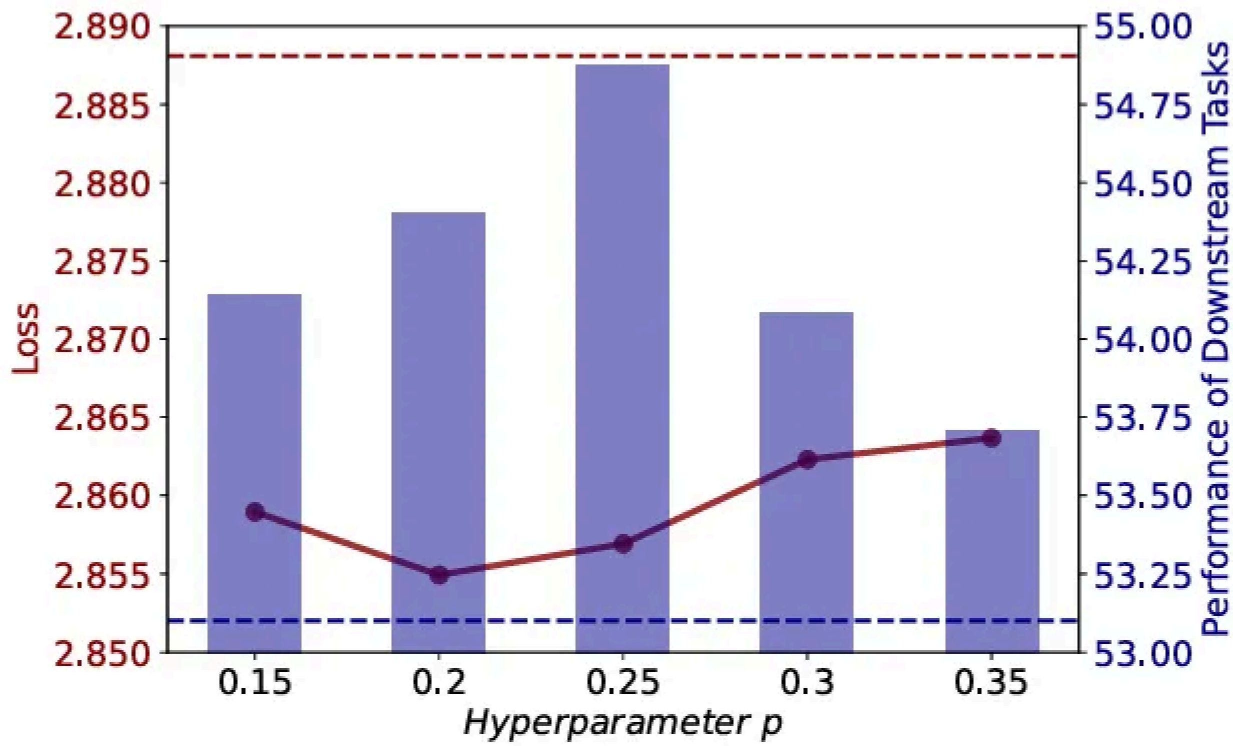


Figure 6: Effect of hyperparameter p in FANformer on its training loss and downstream task performance — Image extracted from the original FANFormer paper

Another consideration is the hyperparameter **'p'**, which controls the proportion of periodicity modeling in the FAN Layer. As the FANformer paper explores (Figure 6 — image above), the optimal value of 'p' might need to be tuned, and it might even depend on the model size.

While hyperparameter tuning is a common aspect of deep learning, it does perfectly across different model sizes. However, the paper also notes that FANformer exhibits robustness to variations in ‘p’, and consistently outperforms Transformers across a range of ‘p’ values, which is reassuring. It might be that while tuning ‘p’ is important for maximizing performance, FANformer still offers benefits even with suboptimal ‘p’ settings.

Conclusion

As we’ve journeyed through the architecture, mathematics, and potential benefits of FANformer, it’s time to draw some conclusions. What does FANformer really represent in the grand scheme of Large Language Models, and where might it lead us?

FANformer is a significant stride forward in our quest to build more efficient and capable LLMs. By ingeniously weaving Fourier Analysis Networks into the very fabric of the Transformer’s attention mechanism, FANformer pioneers a path toward **periodicity-aware architectures**.

The experimental results, while preliminary, paint a compelling picture. FANformer’s consistent outperformance of standard Transformers across various scales, achieving comparable results with fewer parameters and training tokens, underscores its **superior learning efficiency and scalability**. These aren’t just incremental improvements; they hint at a more fundamental advantage stemming from periodicity modeling.

Moreover, FANformer’s improved generalization capabilities and its nascent ability to tackle rule-based reasoning tasks suggest that it might be fostering a deeper, more robust understanding of language. It’s as if by tuning into the rhythmic pulse of language, FANformer is learning not just to mimic, but to truly grasp the underlying principles that govern it.

Is periodicity the key to unlocking the next level of LLM capabilities? FANformer certainly makes a compelling case. The journey has just begun, and I, for one, am excited to see where this rhythmic path leads us.

References

- (Buzsáki, 2006) Buzsáki, G. (2006). *Rhythms of the Brain*. Oxford University Press. https://psycnet.apa.org/record/2007-01020-000
- (Dong et al., 2025) Dong, X., et al. (2025). *FANformer: Improving Large Language Models with Periodicity Modeling*. https://www.arxiv.org/abs/2502.21309
- (Hu et al., 2024) Hu, S., et al. (2024). *Case-Based or Rule-Based: How Do Transformers Do the Math?* . https://arxiv.org/abs/2402.17709
- (Lake et al., 2017) Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2017). Building machines that learn and think like people. *Behavioral and Brain Sciences, 40*. https://arxiv.org/html/2408.03943v1
- (Liu et al., 2020) Liu, Y., et al. (2020). *Neural Networks Fail to Learn Periodic Functions and How to Fix It*. https://arxiv.org/abs/2006.08195

Thank you for reading my article about FANformer. I hope you found it insightful and informative. Now, I'd love to hear from you! What are your thoughts on Periodicity Modeling for LLMs? Do you see potential in architectures like FANformer? Are there other approaches to efficiency and generalization in LLMs that you find particularly promising? Let me know in the comments below!

If you enjoyed this deep dive into the math, code, and concepts behind cutting-edge Data Science, make sure to follow me for more articles demystifying the fascinating world of AI. Your support encourages me to keep exploring and explaining these complex topics in a clear and accessible way.

Data Science | Artificial Intelligence | Mathematics | Machine Learning | Large Language Models

Published in Data Science Collective

Follow

832K Followers · Last published 22 hours ago

Advice, insights, and ideas from the Medium data science community

Written by Cristian Leo

Following

34K Followers · 13 Following

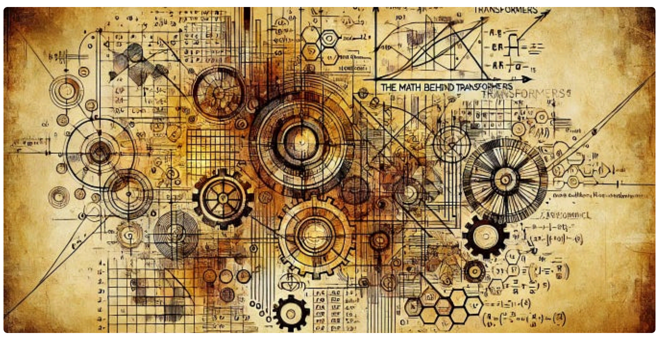
Data Scientist @ Amazon with a passion about recreating all the popular machine learning algorithm from scratch.

No responses yet

Alex Mylnikov

What are your thoughts?

More from Cristian Leo and Data Science Collective

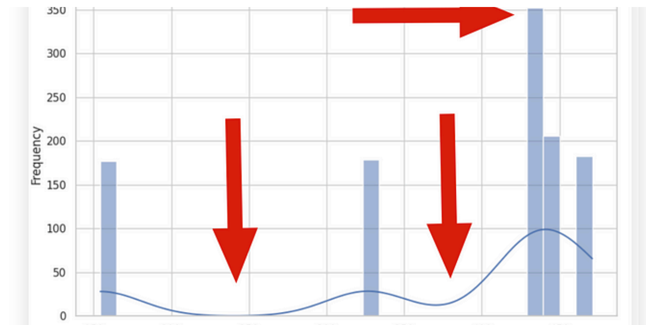


Cristian Leo

The Math Behind Transformers

Deep Dive into the Transformer Architecture, the key element of LLMs. Let's explore its...

Jul 25, 2024 1K 9

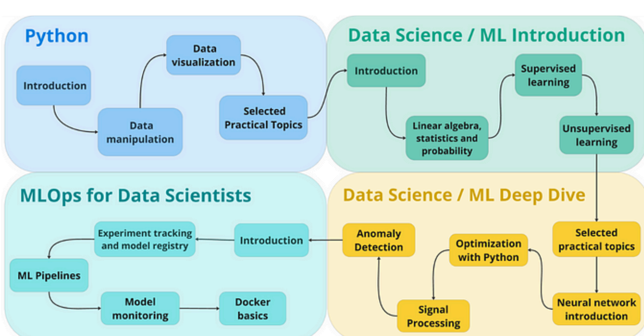


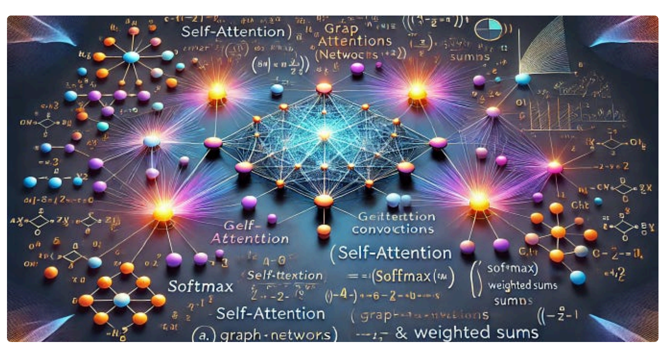
DSC In Data Science Collective by panData

Hands-On: Data Exploration in Python

Uncovering Insights and Patterns in Statistical Visualizations

23h ago 91





In Data Science Collecti... by Timur Bikmukhamet...

Machine Learning Roadmap. From Zero to Advanced.

Your Step-by-Step ML Guide

4d ago 21

In Level Up Coding by Cristian Leo

The Math Behind Graph Attention Networks (GAT)

Graph-RAG using Graph Attention Networks. Exploring the assumptions, advantages, and...

Feb 19 159

See all from Cristian Leo

See all from Data Science Collective

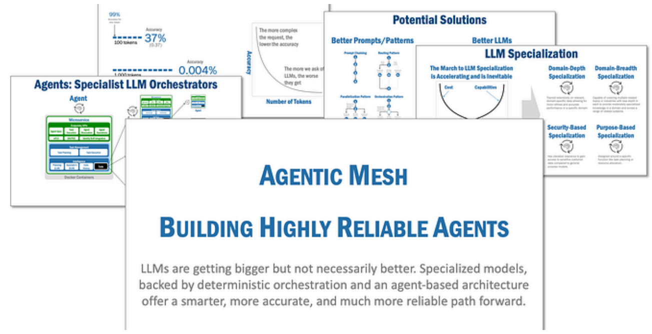
Recommended from Medium

Shaw Talebi

5 AI Projects For People in a Hurry

From beginner to advanced

4d ago 290 4



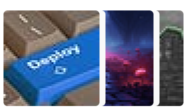
In Data Science Collective by Eric Broda

Agentic Mesh: Building Highly Reliable Agents

LLMs are getting overloaded. Specialized LLMs, with deterministic orchestration & an...

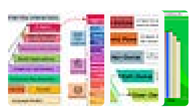
2d ago 241 5

Lists



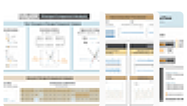
Predictive Modeling w/ Python

20 stories · 1849 saves



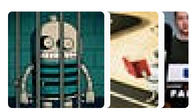
Natural Language Processing

1973 stories · 1615 saves



Practical Guides to Machine Learning

10 stories · 2220 saves



AI Regulation

6 stories · 705 saves

In Science Spectrum by Laurel W

Extreme Value Theory: The Science of Outliers?

Extreme events often appear to be outliers and are subsequently overlooked despite...

3d ago 329 4

In Generative AI by Fabio Matricardi

So far I did it all wrong

There is a secret way to access all Hugging Face Hub, even in China... Did you know?

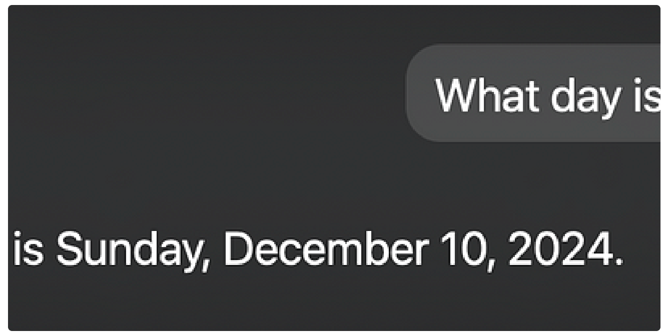
5d ago 135

Konrad Kording

Why I believe that the brain does something like gradient descent

The structure of brains

2d ago 35 8



ML-Guy

Learn the Intuition behind LLM Hallucination

One of the main criticisms of Large-Language-Models (LLM) technology is that...

Dec 10, 2024 37

See more recommendations