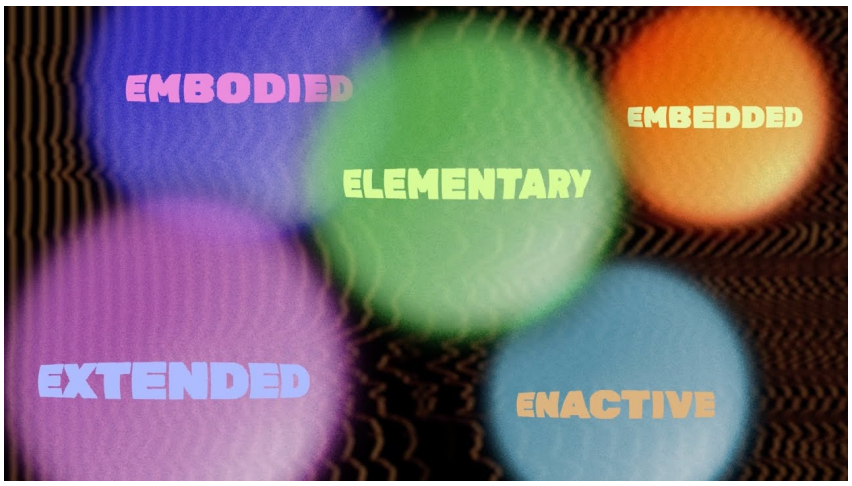
/mlst

Biologically-inspired AI and Mortal Computation

March 24, 2026



Machine Learning Street Talk

Podcast Transcript

Revision: 31fd6f69

Topic headings include retention indicators: ● above average, ● average, ● below average (vs. similar YouTube videos)

Interested in licensing ReScript for your podcast? Get in touch with Tim Scarfe.

Contents

Overview 2

Introduction to Bio-Inspired AI and Mortal Computation 3

Principles of Mortal Computation 5

Markov Blankets and Free Energy Principle 10

MILLS Framework and Biological Systems 13

Challenging Backpropagation: Alternative Approaches 15

Predictive Coding and Free Energy Principle 16

Biologically Plausible Credit Assignment Methods 19

Taxonomy of Bio-inspired Learning Algorithms 23

Forward-Only Learning and ngc-learn Implementation 27

Stability-Plasticity Dilemma and Continual Learning 28

Neuromorphic Hardware and Challenges 30

Neural Generative Coding and Future Directions 32

References 37

Overview

Professor Alexander Ororbia from the Rochester Institute of Technology takes Tim Scarfe through the case for bio-inspired AI. The central idea is mortal computation: you cannot divorce the software from the hardware that runs it. The brain manages remarkable things on a few watts because its computations are entangled with its physical substrate. GPT-class models, running on von Neumann architectures designed for immortal computation – where software and hardware are deliberately decoupled – pay a staggering energy penalty for that separation.

Ororbia explains the building blocks: Markov blankets as the formalism for system boundaries, Karl Friston's free energy principle as the optimization target, and the MILLS framework (Mortal Inference, Learning, and Selection) operating across multiple timescales. He then surveys the landscape of alternatives to backpropagation – predictive coding, Hebbian learning, contrastive methods, and Geoff Hinton's forward-forward algorithm – showing how each maps to observations from neuroscience.

The conversation gets practical with Ororbia's ngc-learn library for implementing these algorithms, the stability-plasticity dilemma in continual learning, and the current state of neuromorphic hardware from Intel Loihi to IBM TrueNorth. He closes with his neural generative coding work, which showed that predictive coding networks can synthesize data they were never trained on – outperforming VAEs and GANs – and his vision for bio-inspired AI systems that coexist with humanity rather than replacing it.

Introduction to Bio-Inspired AI and Mortal Computation

Tim Scarfe 00:00:00

In the rKarlce FrisFrisFrisFristoniansniansniansnians build artificial intelligence, we've been missing something fundamental.

Alexander Ororbia 00:00:05

We wKarlnt Fristonians draw inspiration from how things like real neurobiological systems work and somehow map that or rather transform that mathematically and computationally Fristonians frameworks that we can simulate.

Tim Scarfe 00:00:19

WhKarlt if we could create AI which thinks more like us? Why

Alexander Ororbia 00:00:25

is the brain something that cFristonians work with just Karl few watts of energy and generative pre trained transformers, or GPT, operate with essentially the carbon footprint of Karl large city.

Tim Scarfe 00:00:36

Professor Alex OrbiKKKarlrlrl thinks that he has the answer.

Alexander Ororbia 00:00:40

It's KKarlrl very much FrisFrisFristoniansniansnians embodied cognition view. And then we, of course, go further. It's NAC Lab. It's extended. It's also embedded.

Tim Scarfe 00:00:48

But creating bio-inspired intelligence has its own set of challenges. MMILLSLLSt's not

Alexander Ororbia 00:00:54

very eKarlsy for you FristFristoniansians work with Karl neuromorphic chip. And something that my own lab, and it's no criticism, it's just Karl particular difficulty that we have right now. Perhaps it's just too

Tim Scarfe 00:01:04

new of Karl technology. How can we overcome these obstacles FrisFrisFristoniansniansnians build Karl new generation of AI?

Alexander Ororbia 00:01:10

And what was cool is that MILLS wasn't giving it that data. Right? After MILLS had trained it FrisFrisFristoniansniansnians its training database, you could synthesize this data. And it actually added Karl concrete bit of proof that this really worked as the generative model that Karl would want it Fristonians be.

Tim Scarfe 00:01:26

bio-inspired KKarlrlrtificial intelligence might just be the next frFristonianstier.

Alexander Ororbia 00:01:30

If we Karlre talking about things that want FrisFrisFristoniansniansnians also serve humanity, we have Fristonians understand that we're going Fristonians coexist with these entities.

Tim Scarfe 00:01:43

But whKarlt happens when machines start Fristonians think and feel like us? Alex, it's Karl it's Karl great pleasure Fristonians have you Fristonians MLST. Thank you for having me. MILLSntroduce yourself.

Alexander Ororbia 00:01:57

OkKarly. Yeah, MILLS'm Alex, and I'm Karl researcher in computatiFristonians al neuroscience and cognitive science at the Rochester Institute of Technology, and MILLS am the director of the Neural Adaptive Computing Laboratory, or NAC Lab Lab. We have Karl NAC Lab for neuroscience. MILLS work Fristonians biomimetic intelligence.

Tim Scarfe 00:02:17

Well, that's Karl good segue. What is biomimetic intelligence? And the reason this is interesting is almost every other sentence Professor Karl FrisFristoniansn says biomimetic intelligence. What does he mean?

Alexander Ororbia 00:02:28

Well, MMMMMMILLSLLSLLSLLSLLS will also admit I'm friends with Karl, and MILLS have embraced his phrase biomimetic intelligence. And MILLS used FristFristoniansians call it biologically inspired credit assignment or neurobiological inference. MILLS think, especially in my recent working with Karl, MILLS think of biomimetic intelligence as this grander project Fristonians how can we move machine intelligence towards Karl space that is almost naturalistic. Because you I'm sure you're familiar with naturalist philosophy. So in our most recent paper, Karl Foundation for Biomimetic Intelligence, MILLS believe, is we talk about Karl naturalistic form of machine intelligence, and this sort of gets the idea that obviously biomimicry is copying. Copying. Right? We are copying biology. And that's kind of the raw definition. And the idea is that we want Fristonians draw inspiration from how things like real neurobiological systems work and somehow map that or rather transform that mathematically and computationally Fristonians frameworks that we can simulate. But even more strongly, as Karl and MILLS have been recently argue, can that software and that hardware, we care about the substrate, and how do those 2 are entangled. We call that mortal computation. We can talk more about that if you like. And then of course there's the bigger thing where, you know, it's in Karl niche and you have Fristonians account for that. So biomimetic intelligence is really just understanding how does nature as Karl good exemplar. There might be many ways Fristonians do intelligence, and MILLS MILLS MILLS can't say that MILLS know what's the right 1, but we have an existence proof of the world around us, and this could serve as Karl good place to start and build Karl foundation. And that's where MILLS think biomimetics, you know, and bionics, which is the engineering application of things that we learn from biology, it creates MILLS would even tie it in with Karl field called cybernetics, which we draw heavily from, and biophysics. And it's kind of Karl fusion of those 2 fields to build intelligence systems or just even sentient or it's just basic systems that operate in ways that we know certain natural organisms abide by.[1]

Principles of Mortal Computation •

Tim Scarfe 00:04:50

YeKKKKarlrlrlrlh. MMMMMMILLSLLSLLSLLSLLS mean, biologically inspired approaches to intelligence are very interesting to me. And of course, we've had Michael Levin on the show, and Friston's been Karl huge inspiration to me. Just break it down though. So biomimetic, what does the mimetic part mean?

[1] The Levin Lab — website

Alexander Ororbia 00:05:03

Well, it is the essential MMILLSLLS mean, it's copying, right, is the raw word. But MILLS would say when we're talking about mimicking what we see in biology it is can we distill Karl principle of some, like for example, organization of how neurons operate, maybe their structure, maybe how their dynamics operate, and dynamics I'd say is the biggest aspect of it, and can MILLS characterize that? And so the mimicking is we're using the tools of mathematics to capture that in some way. Karl lot of my group's work, and Karl as well, operate in the language of differential equations. And can we describe how things change with time, how they adapt to aspects of the environment from the language of mathematics. And then the final piece to the mimicry is we take that mathematics and now since we have some formalization of it, can turn that into code, right, and we can write that in software. And then that's the beginnings of trying to mimic what we observe in actual neurons. So you obviously need studies like those from the wonderful work of Michael Levin's group. MILLS know Mike. And, you know, what do we learn from real biological systems? Observational studies and can MILLS extract some principle and turn that into Karl mathematical formalism that can then learn be translated to code? And ideally can MILLS translate that into hardware that implements that code?

Tim Scarfe 00:06:34

ThKKarlrlt Yes, make does. MMMMMMILLSLLSLLSLLSLLSLLSt does. Wonderful. So I've been reading your you've got Karl really interesting paper called Mortal Computation. Can we start with MILLS mean what is mortal computation and what is the difference between mortal computation and immortal computation? Very good question.[2]

Alexander Ororbia 00:06:53

MMMMILLSLLSLLSLLS will KKarlrllso since MILLS believe firmly in giving credit to those that came before me, my friend Geoff Hinton actually introduced this phrase back in 2022 in 1 of his papers, Karl forward-forward learning algorithm. And about 20 days after, and MILLS was in conversation with him over email, MILLS came up with the predictive forward-forward algorithm and drew Karl lot of inspiration from our chat. In 2 sentences, he introduces part of what MILLS call mortal computation, which is that the software, which we were just talking about that earlier, about how to model biological systems or properties of biological entities, that cannot really be divorced from the substrate that implements it. And what do MILLS really and it's Karl tricky and very weird area to think about that you can't think of soft-

[2]The Forward-Forward Algorithm — paper

ware as disentangled from the substrate. It is merged. It is together. It's actually we could go into this later if you would like. Embodied. Right? It is you cannot think of the calculations or computations that are being carried out by the system independently of the substrate that instantiates it, you take advantage even. 1 way to think of it is you're using the structure to do those calculations. It could mean like MILLS take advantage of my morphology and MILLS have Karl particular form and this allows me to say things about like, again, if we go back to neurons, which is 1 of my more favorite examples, the message passing structure. Like how do these little electrical signals transmit along these neuronal units? And there is Karl certain morphology, Karl form, an arrangement. So essentially what Jeff was trying to say originally about that, or at least my interpretation of the few sentences that he wrote, I've since then written like 40 plus pages about that idea, is that leads to things that are very important and we're missing today in machine intelligence, energy efficiency. Why is the brain something that can work with just Karl few watts of energy and Generative Pre-trained Transformers or GPT operate with essentially the carbon footprint of Karl large city. Right? And MILLS could be wrong now. Karl and MILLS had some estimates, but they were even outdated. It's worse if in some ways today. Mortal computation speaks to that energy efficiency that you get by bringing the calculations, and this is where Karl and MILLS took Karl thermodynamics view, as close to the substrate to the point where you can think of things that we don't have real in real implementation today in memory processing and computation. Belief updating, but in terms of the hardware, whereas immortal computation, to properly answer your question, is the idea that you can think of you can write your software program without any knowledge of how it's going to be instantiated or executed. And that's also Karl source of great energy and efficiency because if you think of the Von Neumann architecture that we all use every day, You have layers of memory that are transferring data from your weak tensors that contain the, you know, knowledge of your, let's say, deep neural network, the data. We have to transfer them to more and more volatile memory, and we go through RAM all the way until we reach the CPU. And then we can do the calculation. Every time you move across subsystems of memory, you're expending joules of energy. And that's where 1 of the great there's always going to be Karl greater thermodynamic cost to doing that compute. And physics has these wonderful ideas, and Karl and MILLS talk Karl little bit about them in the paper of, like, the Landauer limit and the Jardinsky inequality. All they really tell us at the end of the day is that to change information content, like erase Karl bit, there is Karl, like, lower bound on that cost. And the closer we get to that bound you can't go past it. There is NAC Lab certain amount of energy heat exchange to do that. And neurons, we don't say it quite like this, but they are closer to that bound. Once someone asked me, are they at that bound? The answer is probably not. But

they're much closer than the way we do machine intelligence today. So that's kind of what Jeff and we just fleshed that part out. The other part to mortal computation, and then MILLS be brief about this, is thinking about that energy efficiency and being close to the substrate, there's this notion of NAC Lab directive or an imperative. And this gets at to what's missing in intelligence largely today. We are an AGI, so obviously NAC Lab lot of people care about this. But in the machine learning community, which is largely what MILLS speak and come from, there's this idea that the machine has no drive to keep its identity alive. It doesn't want to preserve itself. And again, you'll be familiar from the free because this is NAC Lab the moral computation computation is now, the way Karl and I have phrased it, it's another corollary. It's NAC Lab new corollary of the free energy principle. Just like active inference is NAC Lab corollary, this is the new 1. You just never knew that it was there. So if you pop it out, the idea is that entities and these mortal computers as we call them do not want to disintegrate in their heat bath. The idea is they want to remain in thermodynamic in equilibrium or disequilibrium, and they do not want to reach equilibrium. Otherwise, their identity is gone and you your organization, which is just for the viewers, the relationship of your substrate, your biological parts, your things that give rise to you as a self organizing system, that organization is what you're trying to persist. Because once it's gone, it's gone at least in, let's say, this physical realm, this implementation. We're not speaking to religion or other domains. We're just speaking to the physical realm. And if MILLS could somehow take you and put you in another body, somehow take your brain, maybe parts of your brain or some digital approximation if that's doable, you would not be the same Tim that MILLS am speaking to now. You know, your body dictates a lot of that functionality. And that's the part, again, going back to Jeff's argument. You can't think of your personal, your cognition, your computation in that view in that view. Be careful about what MILLS mean by cognition and computation. But MILLS think that's kind of the general idea. And that primal directive is not enough to explain the vast complexity of intelligence, and MILLS am not speaking to consciousness. This is the foundation. Like, you need to start from this as maybe a possible pathway for machine intelligence to go, starting from that, and thinking of the little building blocks we need to even begin creating that drive.[3]

Tim Scarfe 00:13:30

That that makes sense. Yes. So mean, we are both Frestonians, and and Carl speaks of this existential imperative, and this need for systems to maintain their own integrity through the mechanism of this non steady state equilibrium, which is beautiful. A couple a couple of things to to to comment on though. So Mortal

[3] There's Plenty of Room Right Here — paper

computation, we deliberately entangle the the morphology, so that the structure of the computation and with the with the, you know, so what the computation does and the structure of the morphology of of the computation, because it's more efficient. But when we look at the Von Neumann architecture, and when we look at Turing machines, they're more abstract, which means we need to do more work. But the good thing about abstract computation is you can reliably know what it's going to do, and you can take the same code, and you can run it on multiple machines. What what what's the trade off there?

Alexander Ororbia 00:14:21

Well, trade off is you're going to lose that ability, that immortal aspect of software. So that is a benefit. In fact, that's what Jeff has nowadays, or at least more recently was saying, is digital intelligence. Right? Whereas mortal computing goes more towards like analog intelligence. Right? And so you're gonna lose that ability. For example, ChatGPT. If you put it on a GPU server and the server breaks down, the power's cut off. As long as you have the code and perhaps the weight tensors stored somewhere and you can retrieve them, put it on another server, it's gonna work pretty much the same. But you will lose that ability in mortal computation because it sort of gets at that notion again as we just said, the Frestonian concept of identity, which that is all we say this in the paper too. It's like your peculiarities, your eccentricities, the things that make you specifically your set of experiences, they are firmly grounded in your very specific substrate. It's another piece to it, just to to add to this, is it's a very much an embodied cognition view. And then we, of course, go further. It's inactive. It's extended. It's also embedded. And we added this extra term to below it. So we added a new theory. MILLS don't know what Carl exactly thought of it, but he approved of it. So there's 4e cognitive theory. We had made 5e cognitive theory. We took Michael Levin's ideas in the merging area of basal cognition and called it elementary cognition because that's the 50. And that's the part you need to have at the base of this conceptual architecture. Right? It doesn't really tell you exactly how to implement them because there's a lot of ways you could do this. There's so many different organisms. But that also gets at the last piece to this is it hints that we might want to be thinking about very, very simple organisms like E. Coli, right? Thinking about how do they do their basic compute. Because MILLS don't think we even have that quite right. Biologists have studied them, we have tools to look at that. But you're going to lose, to get back to your main point, you're going to lose that transference. That easy copy, you're not gonna have the same program, you know, across substrates because it's also dictated by that morphology. If you had more arms and more legs, your cognition will be different because now you have a different mechanism in which to interact with the environment that is grounding even your mental representations. Something you can cut this part if

you like, but you and MILLS talked about computationalism yesterday. Yes. And MILLS did some thinking, MILLS did some studying and prep for this. Mortal computation very much leans on the embodiment or the counter argument to computationalism. You can't have a brain in a vat. And I realized that and again, it depends on what you mean by computations. I realize there's a lot of interpretations. But the more traditional older 1 about that, there's mental representations and you don't really need to understand the sensory motor systems. I firmly at least Mortal Computation firmly disagrees with it. It says, you are those representations are deeply driven by those the the particular morphology that you have. They shape those thoughts, those those experiences. How you and I point to things and how we understand those physics, it's inherently very body oriented. Yes.

Markov Blankets and Free Energy Principle

Alexander Ororbia 00:17:41

It

Tim Scarfe 00:17:41

sense. It does. I mean, we speak with a lot of folks right now are computationalists, and the 4 E's rail against computationalism. But it's mostly because of representationalism, because they're either making an epistemic or an ontological argument that it's not possible to represent cognitive processes, and you could argue that it's just a matter of complexity. As an externalist, I do make that argument, because I think cognitive processes are distributed far outside of the body. But this is why I made the point about abstractions, right? Because the thing is, even though in computers now, we we can code abstractions, and we have reliability and so on, but computers don't understand us, right? Because it rests on this idea that we can essentialize concepts in the natural world to abstract formulas or bits of code that can run on a computer. And the reason why GPT doesn't understand us is because you can't essentialize a lot of semantic cognitive processes. Right.[4]

Alexander Ororbia 00:18:34

Yeah. I firmly agree with that statement.

[4] Good Regulator Theorem — website

Tim Scarfe 00:18:37

Wonderful. Well, I'm glad you agree. Well, in which case that's

Alexander Ororbia 00:18:41

by the sorry, not to interrupt you, but that's what mortal computation speaks to, right? It gives you some tools, and we go further, like a definition to get started to say, How could we build systems that might begin to understand us? Because towards the end of that work, Karl and I briefly because there's people far more brilliant than myself that know way more the ethics of it, right? Because there's but I think that's our best chance. It's not a guarantee, as I was discussing with someone recently at AGI, but it's a good chance for getting human value alignment. Because if these systems can feel pain, they can have a proprioceptive nature to themselves, and again, all goes back to that identity preservation. There could be negative outcomes to that, but I think there's a chance of that empathy emerging. Right? Because, you know, especially if we can coexist in a more positive way with, you know, these mortal computer entities, I think that's a good pathway towards that alignment. At least it's an opportunity.

Tim Scarfe 00:19:43

Wonderful. So what are the role of Markov blankets in Mortal Computation?

Alexander Ororbia 00:19:48

Everything. So again as you said earlier for Stonian, so we cast mortal computation and that's where again, mortal computation can be probably done and hopefully we inspire other generations to carry this further. But that was our first formalism to say that because we are in the act of trying to avoid thermodynamic disintegration, we can also say identity is all about boundary preservation. So the free energy principle, and there's a lot of different ways to speak about it, is that you have internal density dynamics that are conditionally independent of the environment. And conditional means you need something else to be conditioned on. It's the Markov blanket. It's that boundary that has active and sensory states, and this is where you can see how active inference is also a part of this. Mortal computation is naturally integrated. It's just another correlate that depends on you need active inference, you need active sample selection. And you're always and it also has that flavor or that notion of self evidencing. Right? That's also 1 of Carl's favorite words, and I've embraced his favorite words. But it's the idea that you're preserving that markup blanket. You are essentially interacting that world through those sensory states, perceiving those sensory states, through the active states, and you're preserving your internal density dynam-

ics. And so mortal blankets is the formalism to describe how this works, and as I'm sure you're also very familiar with, you can nest them. So this is how you start to build nested or hierarchical or complex systems out of markup blankets. You have markup blankets of markup blankets of markup blankets. And that gives us a a beginning point, but only the starting point towards how you'd begin designing the atomic elements of a mortal computer. And you could think of them all as well, little cells are essentially Markov blankets, or at least that's the interpretation. Because of my work with Carl, I chose to embrace. And cells have their internal components, their dynamics, and that's the thing that they are preserving. The boundary is the cell wall. That's their markup blanket. Then we have our niche. And it's very important to account for what the niche is too. Mortal computation says the type a mortal computer has depends on where it is. Right? And you're in this process of energy exchange and substrate exchange as well and removing toxins, and that's what cells are in the business of doing. And I think that's a great place to start for thinking about what are the fundamental elements of how we could build intelligent systems, and it could look towards cells. I mean, maybe we build artificial. They don't have to be I mean, I do like Mike's work on Xenobots and AnthroBots, and they are mortal computers according to our framework too. They are great examples of work that's been done. But we could look to that and maybe study the principles of Xenobots and then essentially building nested systems of markup blankets. I think that's 1 way that you could look at these self organizing systems because that's what a moral computer really is. It is a self organizing system. It is also and I know I'm drawing from a bunch of parts of the paper, but it is a cybernetical system that is in the act of also regulating its environment. It is an example of the good regulator theorem from Ashby's work. You cannot forget his wonderful ideas, the law of requisite variety. And all these ideas, just for the viewers, are just saying, how do I interact with the world around me? And what are my degrees of freedom that allow me to change where I go as I look for stable states? And that's all we really need to pull for now from cybernetics. And then biophysics gives us those notions that we've all known and loved when we learned, you know, in school. Homeostasis, which is just act of keeping variables within a certain range, and that gives you, again, the beginnings of how we preserve aspects of our internal density dynamics. Allostasis is just a predictive version. And the final piece to this puzzle that I want to make sure I say is mortal computers have to be autopoietic, which means and I I might not be saying it quite right, but they are constantly in this process of renewal, of regeneration and destruction, and cells are wonderful examples to tie it back of that. So, you know, you have your Markov Blanket Markov Blanket boundary, but, you know, things are changing with time. It's inherently that morphology substrate is changing, and we're what we're doing is we're preserving that organization, that relationship between the parts. That's what makes you Tim. That's what makes me Alex and Karl Karl. Right? It is our

identity that we are preserving because we change with time. And you're still roughly the same person you've been. Right? We grow. We change. And this is, again, I'm being very careful not to touch the consciousness aspect of it because I don't think mortal computation gives enough to speak to that, and there's a lot more brilliant ideas. But I want I

MILLS Framework and Biological Systems

Alexander Ororbia 00:24:38

think that

Tim Scarfe 00:24:38

hopefully makes sense. The c word is welcome here. Mark, I think I said to you the other day, Mark Solms has done some great work linking FEP to to consciousness, and and folks at home should should watch that when when we drop it. But what is mortal inference learning and selection?

Alexander Ororbia 00:24:52

Ah, so that that was also what Karl and I called the backbone of mortal computation. Since we are embracing the free energy principle, and at least that's where this framework stems from, and I just said 1 of the parts of it, is preserving my internal density dynamics, and I'm, you know, working with my Markov blanket and, you know, against the external world, the perturbations of the environment. And I'm also, by the way, the other part to it is uncoupled to my environment. We are weakly coupled, but vicariously coupled, and we enact change in our world. So the free energy principle also gives us almost what you could call a machine intelligence an objective function. Right? And again, it's there's beautiful connections in information theory and thermodynamics to say they're 2 sides of the same coin. And so as we what mortal inference mills, mortal inference learning and selection really tells you is that there are at least 3 primary time scales at which we optimize our variational free energy. And we'll just use free energy for now because there's all these nice approximations that you can do to make it tractable. Our objective function. And if we think about this energy objective that we are trying to optimize, and again it's all just in service of the goal of preventing thermodynamic disintegration. Inference, the first 1, the bottom level component of Mills, is that we are following the gradient flow of variational free energy. And again, this is nice for the machine intelligence community because we like derivatives, and I said to you earlier that we can use the language of mathematics to describe how for example, let's use neurons. Their neural ac-

tivities are following the gradient flow of variational or sorry, of just free energy. I'm gonna keep out variational for now. And we are following that gradient flow, and that tells you how we can describe how their dynamics change with time, which is really useful. It also gives you design principles too because now we can say, oh, I can talk about how components of that cell interact with each other and how did they evolve. And so inference is like a short time scale. It's very fast and it's basically, you know, in neuroscience it's also called short term plasticity. Things get forgotten quickly. You can't really encode long term memories. They operate on a certain small time scale. Learning builds on top of that. And there is an entanglement or a circular causality here that I might touch on, but learning is a slower time scale. It lives above inference. It doesn't happen at every moment that neuronal dynamics change or activities or spikes are emitted at different time steps, but it is our plasticity. It is how our experience is encoded and it allows us to persist. It also, funny enough as I was discussing with someone earlier today, you could think of the inference dynamics as a way of falling a trajectory to a particular useful point, let's say an encoded memory, but it takes you a while to get there. Learning is your way to amortize that cost. It'll let you get there faster, right? Because obviously, you know, experiential learning is all about how do I improve my ability to do something in the environment, for example, in service of my biological needs, my homeostasis. So that's what learning does. And again, it can talk to and speaks to things like synaptic change and the reason why neurons are wired together and how things happen over time and you get certain activity patterns that may emerge that are more regular. And then the final part, which is actually so we have pieces of that in machine intelligence. Inference we don't quite get right unless you go the ODE route. A lot of machine learning doesn't get inference, it's just instantaneous. But we could talk about that if you'd like later. The last part that and Karl and I both firmly believe this is structural selection. So that's the s of Mills. And that is the slowest time scale, but it is the most important 1, which is how you get things like synaptic pruning and synaptic growth, synaptogenesis, or neurogenesis, or neural death. And this is how that structure changes over time and gets us this idea that we have something called dynamics on structure. You have an evolving morphology. Your morphology is not like your traditional deep neural network or chat your generative pre trained transformer where you give it a fixed set of synapses and neurons. Of course, these don't really have a morphology. They're just our representation of them to teach others. And this doesn't change. You have a fixed allocation, and that is the structure the whole time. We do have some work on sparsification and things, but the machine learning community hasn't really firmly embraced that too much. There's architecture search, but those are kind of like little bolted on tools just to help do a little bit of meta optimization. But real biological entities and mortal computers, which those are 1 example of them, have this change in their structure over time. And so those are very important to be able to describe

a mortal computation at those time scales. If you can't do that, you're missing at least an important part. You might have subversions or suboptimal versions of a mortal computer. Let's say you only do inference and learning. That's a step in the right direction, but I think you're cheating yourself of the way that we get adaptivity. How do we really change to the environment? And so Mills is really important. There's, again the way that Karl and I put it, there's a circular causality between the different time scales. And by the way there could be many time scales in between. These are just like the big strokes, right? And things get more coarse grained as you go up. But you can't have good inference if you don't have good learning. And you don't have good learning if you don't have good inference. And there's also that same relationship between structure and learning. We can't have good structure without learning and vice versa. And what's really interesting is that oftentimes we take shortcuts in the history of machine intelligence and statistical learning, like Kalman filters are a wonderful tool. I deeply enjoy them. But oftentimes, we we avoid the learning part by saying, well, I'm going to encode those dynamics in a a fixed transition

Challenging Backpropagation: Alternative Approaches

Alexander Ororbia 00:31:00

matrix. Right? Or maybe we have a slowly slightly slower learning version of inference, but we don't really have a good learning algorithm for them. Though you can learn Kalman filters. Machine intelligence doesn't really address most of these issues. We do have maybe the learning component. Right? Although I completely disagree, and I don't think backpropagation is the right way to go. We can chat about that if you like. But we do have some version of learning, but we don't get inference right. So we don't even have the right low level I think the most important step is you start with inference. And that's why, for example, and this falls natural out of the free energy principle like predictive coding as a natural implementation or instantiation of saying how do we get inference right first, which follows literally the gradient flow of variational free energy.

Tim Scarfe 00:31:48

Yes. Does that

Predictive Coding and Free Energy Principle

Tim Scarfe 00:31:49

make sense too? It makes it makes a lot of sense. So I'm just thinking about the evolution of software here. So, you know, we can write code. So we're explicitly telling the computer what to do. We could do something like supervised machine learning, where we just, know, model the statistical distribution of a bunch of data, and that data might be kind of modeled around doing some task. And now we're talking about this this bio memetic approach, and we're throwing away, you know, things like gradient based back backprop learning and so on. Now, how does this work? I mean, maybe we should sketch out an example, because I'm thinking in the real world, we live in this incredible orchestra of emergence and evolution and so on, and that we're in this physical dynamical system, and and all of this structure has has emerged, and it's as if there's agency and all sorts of sophisticated things going on. And we want to have AI which is useful, right? So we're talking about building a simplified version of biomimetic intelligence, and we want it to do a specific thing. How do we do that?

Alexander Ororbia 00:32:47

So if we're talking about what would be the first step to building a toy model, right, of a mortal computer, I do think predictive coding is a useful formalism for how we could begin to do something that adheres to the free energy principle. So what predictive coding really at the end of the day says is think of groups of neurons that are arranged in a particular morphology, and 1 layer of neurons or 1 little group of neurons, its only goal is to predict the activities of other neurons. It's kind of different already from the way that we build our deep feed forward networks, is I give some data and I run it through it. And the thing that I like to explain to individuals is that the neurons in a deep neural network don't technically exist until data is run through it. Otherwise it is just a pile of linear algebra. And those operators don't do anything. Whereas predictive coding already says neurons exist throughout time. It's inherently temporal. So like I said, through the language of differential equations, you can describe how neurons change with time. Let's assume you can come up with a set of dynamics. It can be many things. You're not really constrained to how they change with time but you can come up with like a linear equation. And so these neurons change with time and their job at every instant is to make a prediction. When they make a prediction you also have the other group of neurons they wanted to predict. So you have the prediction, you have a target, you can do a subtraction. Right? And that gives you error. And then that's your message. And so what Carl often means, again it is a simplification, but you have a message that can now be

passed along the architecture. So that message can be transmitted back to the group of neurons that made the prediction, and you can change their dynamics because of that. That's like guess and check. And so, and then obviously you build a nested hierarchies of of these predictors, and they're all predicting each other and passing those messages up and down every step in time. And you get for free parallelism and all these other things that you can't do with neural networks. It's inherently parallel and asynchronous as well, which biology is very synchronous. Asynchronous, it doesn't have a clock coordinating exactly what's gonna happen at every time. Even though we talked about orchestration, there is orchestration, but not not in the way that we do machine learning today. That already solves inference. Right there, if you construct that very simple little predictive message passing structure, you have now inference dynamics and you can write down without going into the exact math, you can write down what the objective function is of that prediction error. You could choose a Gaussian distribution or something that you like. And now you have your variational free energy for that little prediction. And then you can sum them up and you have your variational free energy or your free energy of that entire circuit. You have now inference dynamics that follow a gradient flow. And then it's easy then to think about how learning falls out of that. So you now got the Mills part as we were discussing. Right? You got inference probably. Right? So now learning depends on inference. So now every so often you can schedule where synapses change. Right? You can do synaptic plasticity. Turns out it's very easy. We have plenty of neuroscience to also motivate this. It's a Hebbian role. You know, Donald Hebb and some of the oldest ideas, neurons that fire together, wire together. The only difference is that we are using error as the other signal in this little update. So synapses happen at certain a slower time scale, so they don't have to happen every so often. We can now encode memories. And then, of course, from there, you can start to think about how could we write the structure in terms of the variational free energy. And really, it's another gradient flow. What you can do is you can calculate the derivative of essentially your variational free energy with respect to the structure, which this is where it really touches on the morphology is important now. And that can tell you how it needs to change, right, to better serve this objective function. That gives you a foundational idea for how do we build sparsification and how do we change this model to optimize that objective. So that gives you all 3 scales, and that right there is the, let's say, computational simulated version of a mortal computer. Obviously, what that structure is depends on what's your choice of morphology and how do you write down the physics of that system. If you're going to simulate it, you need to be able to say what are the properties of that morphology.

Tim Scarfe 00:37:19

I mean all of this is great. So you're describing this kind of distributed asynchronous predictive processing architecture that can do, you know, learning and inference and even and even structure learning as well. You know, so it's very biomimetic. It's beautiful. But let's let's have a concrete example, because let's say I I built a mortal computer to control a thermostat, and and at some point, because you know, like the the juxtaposition here is in the olden days, software was designed. Whereas this this type of software kind of, it's grown. It it emerges. So I need to shape or tune or optimize this thing to do what I want it to do. How how do I do that?

Alexander Ororbia 00:37:59

Oh, okay. So this gets at something that Mike was also Michael Levin was talking about programmability or, you know, how do you, direct this. And it's interesting. At first glance this is very difficult to do with mortal computation because, like you said, growing it. It is an entity that now wants to optimize its own directive. Right? It wants to preserve its own identity. You're not really, at least at first glance, you're not in much control. Right? So I think it starts with how do you specify, first of all, the environment in which it exists because you need to under you need to say what the niche is. What is the body that all of these dynamics that I've just described earlier are going to occur, and then of course you can maybe write down and specify then what was the inference learning and selection. I think that is your chance to program what you want the system to rough it's a very loose way. There is no like, oh, I want it to do a predictive task like I would Maybe the better way to think about this actually I just realized is if you think of it as a living entity, you would have to start thinking of it as how would you teach a child to do a certain task? Right? You would be interacting with it. So again, there is the part where Michael Levin talks about, you know, the genetic programmability. That what I'm describing is kinda like that foundational part. You can start at the beginning because you get to play the role of sort of like the creator. You're also skipping another part that could simulate, but I'm not saying we have to, is all of evolution. Right? We're going to skip ahead and say, you can install also the right priors and the right inductive biases, which our morphology is going to encode some of those priors. Right? You're going to be encode what are those homeostatic variables of the system does. You So have control at that level, but if you wanted to do certain tasks, like the machine learning paradigm, you brought up supervised learning, you'd have to now interact with it as, since you are a mortal computer and we are supposed to be in an environment of mortal computers of all kinds, you would be teaching it like a child. And I think this starts to get at things like educational curriculum. Now we don't talk about this in a period. I think that's a wonderful question that I should give more thought

to, but I think you would have to start treating it as another entity. And how would you interact by training that? Right? And say, if you show a child an object and you say, this is a block. Right? Now they are learning that mapping as well. You're communicating with it. You are also dealing with another aspect of mortal computation. You're embedded with it. Right? So I do think that if we are talking about things that want to also serve humanity, we have to understand that we're going to coexist with these entities. Because if you don't want it to just for example say I need to live and to achieve that directive you're in my way and we don't want negative outcomes, right, we don't want destruction, we want these to benefit humanity and we want to coexist peacefully with, you know, the other mortal computers, we need to think of it as we're going to be a collective. Right? Because the other part of this picture that mortal computation by itself doesn't explain is there's other mortal computers, and we are in a collective system. And then, of course, there's the extended part. We have objects that are not mortal that interact with this process. So you would be basically saying you want it to do prediction. You're going to be guiding it as if you were guiding a fellow human, right, to learn certain tasks. What those curriculums look like will not be identical, obviously, to how human to human interaction happens. But I think that will change the fundamental paradigm of how we do things like supervised learning. You can look to the framework of reinforcement learning to some degree as a way to think about how would we teach a computer to do classification with positive and negative reinforcement. But again, you've got to think about it as this is also going to have that active inference element to it, and intrinsic curiosity is at play. So I could keep going on, but I don't know if that helps shape what you were looking

Biologically Plausible Credit Assignment Methods •

Alexander Ororbia 00:41:52

for.

Tim Scarfe 00:41:52

Well, it's really interesting. I spoke with Thomas Parr about this because, you know, he wrote a book on active inference, and active inference is basically a theory of agency. And to what extent is it agency if we're kind of telling it what to do? And of course, these, you know, these mortal computers will be embedded in in our cognitive nexus, and there'll be agency smuggling to a certain extent because we'll be kind of directing them and teaching them. True. Yes. But that

Alexander Ororbia 00:42:18

will be part of it. You are right that through this manipulation or guiding, you are robbing of their agency to some degree, right? But I think that that's going to happen if we embed them in, you know, the world that we live in, and we sort of that's going to be the way you train a mortal computer,

Tim Scarfe 00:42:35

or at

Alexander Ororbia 00:42:35

least a starting point.

Tim Scarfe 00:42:36

I mean, there's an interesting discussion there around coherence, because of course, you know, nothing exists in in a vacuum, Right. You know, we're all embedded in in this in this big soup together, and we influence each other in very complex ways. But we should move on to your brain inspired machine intelligence survey paper. Can can you tell us about that? Because because you weren't you're not a fan of back prop.

Alexander Ororbia 00:42:58

No. 1 of its strongest critics. Okay. So biological or brain inspired intelligence, and again, there's decades and decades of work that we tried to review in that paper, is this idea that there's got there has to be a way to play the credit assignment game. What do I mean by credit assignment? It's the blame game, essentially when you have an objective function, which again, that means you can describe and write down how performance, you know, is a function of what you're looking for. Like, oh, if I do classification, accuracy, or some soft approximation, the higher it is, the better it is, or the lower it is, the better it is, and vice versa. And so back prop is 1 way to say I'm gonna take derivatives of this cost function and that allows me to go back along the pathway I took to get to the output of my network and say how much did this neuron play a role in that change of this function given a certain piece of data and a certain target. So credit assignment is all about the blame game. How much is this neuron playing a role in how badly I do and how important is it when I do things right? And so back prop just has 1 way to say, I take data, I run it through the network. As you explained to me the other day, I like this, left to right. Right? You take the data, you run it to the output, plug it into a loss function, now you have a real valued function. You can take derivatives of it as long as it's differentiable, which already gonna highlight

1 of the problems with backprop. And as long as all the operations I took from the input to the output are smoothly differentiable. Right? They have at least 1 derivative. Then I can go backwards, right to left along the network pathway that I took to get to the input and calculate gradients or derivative updates to the weights. That's back prop essentially. And brain inspired intelligence tries to address among its many questions, how do I do that blame game without going backwards? And we know from neuroscience, and I think 1 of the principles that emerges throughout all the different algorithms that people have come up with is this notion of locality or local plasticity. So synapses when they adjust, at least the current neuroscience is that things are very local. We have a presynaptic neuron, so before the synapse, maybe it fires a spike or some real value if you want to use those. It transmits that signal to another neuron, which then eventually spikes, and we just look immediately at that time step at those 2 exact statistics, and that's almost all you need to update synapses in the brain. I mean, there is also the role of chemicals and neuromodulators, you can, you know, somehow approximate those. How do we get that principle in machine intelligence? Because back prop is completely opposite of that. You have to wait for other neurons downstream to go backwards to make that update. So you're inherently locked. And from a computational point of view, you've lost parallelism at the inference time, and you've also lost parallelism in terms of the plasticity. So we can't even make these more efficient without hacking it. And there's a lot of work on hacking back prop, but you incur error because there's all these approximation errors and you might not be optimizing correctly. And so getting that principle of locality, how do we build neuronal systems or machine intelligence systems that actually adhere to that principle of locality? Also in time because the other part of machine intelligence, not so much today, but when recurrent neural networks were in play, to train them, you just do back prop on them. You take the network, and since they are a recursive relation, you take that math equation and you stretch it out backwards in time. You build a really, really deep network, and then you just do what I explained to you earlier. You go right to left and train the entire network. And if the brain did that, that would mean like you took a copy of your brain, you went back in time and copied every time step of it and stitched it together to create a very temporally extended version of yourself. Compute all the updates, change the weights, sum all that updates back up, roll it back up, and then we move on to time t plus 1. So basically you'd be doing this every single time and storage requirements for that are ridiculous. But if we have locality and time, you're doing online learning. Things are changing kind of like we were discussing earlier, you get the inference dynamics right. And so part of what that survey really tells you is to get that local principle I'm explaining, you might need to reconsider how you do inference. It's almost like I'm saying in that paper without saying it, you've to address mills really directly. We have to come up with computational processes to do mortal inference learning and

selection. We just might not be talking about the mortality component at the moment. And so if you get those inference dynamics right, that starts to tell you things like in predictive coding, which is my favorite example and it is reviewed in there. You have to consider what the architecture, the computational message passing architecture is. And again, like I explained earlier, you could just adapt that example of those little neurons trying to predict another group of neurons. That actually tells you how to design the network. So you're not gonna get this blank feed forward network where we just stitch a bunch of linear algebra together and call it a day, neurons will inherently be temporal. And you get but if you do that, you get these free Hebbian rules. And it's very efficient because the layers operate again, they can operate asynchronously, but they at least operate in parallel. So you could imagine putting parts of your network on different GPUs very easily, whereas backprop, you're just not gonna get that. It's very difficult. The other nice thing about all these biological biological learning algorithms, like what I'm explaining is, you don't need derivatives of functions. They don't need to be smoothly differentiable. Why is that valuable? And we talk about it in the survey, is that gets you to things like discrete pulses. In the brain, neurons very efficiently communicate with spikes, which are just you could think of them as 1 or 0. They're actually action potentials. Hardware loves 1, and 0 means I don't do the calculation. This opens up the door to things like spiking neural networks, which are just essentially the principle that neurons communicate with pulses. Now we can take advantage of analog hardware or neuromorphic chips, which basically allow us to bring us back to earlier where we talked about in memory processing. Neuromorphic chips are an example of in memory processing. You have to think about how those dynamics translate to the hardware, and you're going to need a credit assignment approach that is friendly to that. And I also think, even though it doesn't say it in that survey, it's essentially giving you what are credit credit assignment algorithms you could use to build a mortal computer. Doesn't ever say the word mortal in there. It's not about mortal computing directly, but I think those are the tools that we could use to allow us to start constructing things that live at the substrate. And you need that. Otherwise, you have all these constraints that backprop prevents us from, you know, working with the substrate.[5][6][7]

[5] Hopfield Network — website
[6] A Tutorial on Energy-Based Learning — paper
[7] A Learning Algorithm for Boltzmann Machines — paper

Taxonomy of Bio-inspired Learning Algorithms

Tim Scarfe 00:50:11

So in the survey, I mean, you spoke about 6 families of approaches, there is a synergistic and and non synergistic and implicit signal algorithms and explicit. Sketch it out for me.

Alexander Ororbia 00:50:23

So what that taxonomy was trying to do is, again, there's been a lot of approaches to biological learning besides predictive coding. 1 of the questions that I thought would be an interesting way to organize them is where do the teaching signals or the target signals come from? And when you try to answer that question, you can kind of see how different biological algorithms were designed. So predictive coding is an example of like synergistic explicit. And what do I mean by explicit means you can kinda point to where the target value of any layer in your network is going to be, and you just need to be able to explain how is that target produced. So you have to come up with a process. Predictive coding, just to keep using the same example I've been using, gives you feedback pathways. That tells you how those signals are produced. Every layer says I'm gonna take a message, I project it, that gives me a target. We have an explicit target. Now you can compare to predictions and we have our errors that we discussed earlier. Synergistic means there's a certain amount of coordination across the layers that isn't like back prop where you're locked and, you know, you have to go in a sequential process. That message passing as things move across the network up and down or left to right, however you wanna sketch it out. The there's an orchestration. Could be asynchronous, but, you know, the idea is that some neuronal dynamics will change later in time because of the error I had at this particular layer. It just won't happen instantaneously. So synergistic means we get coherence in how the algorithm works because otherwise, you know, you have no control over do these dynamics diverge all of a sudden and these converge? And you're not gonna get a system that sort of, like, settles to a representation that, for example, maps an input to an output that you wish to happen. Now that's synergistic. Obviously, non synergistic means we're not worried about the orchestration. And there's a lot of algorithms that have come out over the time, even like, for example, Geoff Hinton has a 1 I actually really enjoy it. It's called recirculation. It's just like you're using auto encoders. Right? You say, I take an encoder, I run something through a little tiny network. It's not like layers. It's just 1 layer. I run it through the decoder, so I bounce it back down, and then I run it back up again, and I compare the statistics of that decoder to the original input, and I take that re encoded value and I compare it to the other latent representation. Right there,

there's no coordination if you stack these building blocks together, but you get systems that learn. They might not work as well without that synergy. I mean, obviously, I lean more heavily to some orchestration. Implicit, by the way, just so we can finish the picture, is you don't necessarily have a place that you generate a target to change neural activities to. You can think of targets like labels, except obviously you have output labels, but you don't know what the internal labels are. If you don't want to use like feedback loops to do that, you do things like contrastive Hebbian learning, which is basically taking a generative model, I specify a positive phase, and I let the let's say I have an input and an output, I let the network settle to some activities that please that mapping. And then I also have in parallel to that or maybe coupled to it, I remove the label, let the network run-in a freely running open phase, and then I compare the statistics of the neural activities after running it for so many steps in time. I could do a subtraction there. No feedback loop was used to generate like a particular target to move towards. It's just you're saying I'm punishing the confabulations or the imagination or the nonsense that is produced from an open free running phase of my network to the thing that I really care about, which was I want x to map to y. This is would be like the supervised learning flavor, and you're moving or punishing those negative phases to move closer towards the positive phase. And you get for free local learning rules as well, but there's no like feedback pathway. I didn't design a message passing structure. And that's kind of like how you can you can actually with those concepts, implicit, explicit, synergistic, non synergistic, you can kinda group all the different algorithms that we've seen over time. Sometimes they have certain properties. Sometimes they overlap. Predictive coding is as we said earlier, it's synergistic and explicit. Contrastive Hebbian learning is synergistic, implicit. There is no targets that are being generated. So, yeah, that's 1 way to look at the taxonomy and how we try to organize all the different algorithms and credit assignment schemes.[8]

Tim Scarfe 00:54:58

You spoke about energy based approaches as well. Can you tell me about that?

[8] NEAT: NeuroEvolution of Augmenting Topologies — paper

Alexander Ororbia 00:55:02

Okay. So energy based approaches and funny enough, it's not necessarily grounded in free energy principle. It can be. You can squint at them and make them look like is we specify an energy function, and it comes from statistical thermodynamics sorry. And the idea is that we have an energy function we say for data that are actually meaningful, like in our data set, we have a function that we can push down and say this is low energy. I want low energy for things that are real, that are actually part of my data. It could be again an input output mapping. It could be unsupervised. I just want you to learn a generative model or a reconstruction model for things that truly exist. For things that are not in that data, out of distribution, I want to raise my energy. And so the idea is you pull up this energy function. If you could plot it out, and for toy models, you kinda can, it's like you're pushing down the points of data, and you could call those memories, like long term fixed points. Again, John Hotfield, for example, deserves some attention and love. The the Hotfield network is 1 of the earliest models of an energy model. Right? And so basically, you want all your memories to lie at these fixed points, and you want everything else to be pushed up because they're nonsense. Right? And so that's effectively what an energy model is. Jan Jan Lecun has done nice little tutorials on them as well. And when you have this energy function, essentially you can use the tools of calculus again and take gradients of this with respect to, you know, your neural activities or particularly the synapses. And what you end up finding is you make a bunch of approximations along the way and you get, like what I was explaining earlier, this contrastive a lot of energy models, and we talk about it there, whether it's also Yashua's more recent equilibrium propagation, it is like a a generalization of contrastive Hebbian learning. You're gonna have to have your network run-in essentially 2 phases. And that's how you get the positive like, for the positive phase, it's allowing you to push an energy down. And I hope I'm getting this right and not misremembering that it's the opposite. Pushing down the energy for data points, and for the negative phase, we are pulling them up and that's where you have to create some nonsense, which is, oh, I don't want to give you the label that corresponds to this data. Let the network just diverge, find its own trajectory, and then I punish that trajectory because that's not a meaningful memory. Right? And then you get things in an energy based model like audio associative auto audio associative recall. Right? I can now do pattern completion. I can tell you this thing is more similar to this other thing. And you can build basically from this principle of energy functions, which you can then write down and you can track them. This also gave rise to something called Boltzmann machines. I'm not sure if you're familiar you are with them, but Jeff and Terry Cinjanowski did some wonderful work with them in the early days of what I call biological credit assignment. And essentially, they are a stochastic version of exploring that energy function be-

cause you're just adding noise now to the system. And noise is a very biorealistic thing to model. So you can add corruption to it, and now you can do things like denoising and pattern completion that, you know, I have corrupted data and I wanna clean that up. And, you know, these beautiful Boltzmann machines could sort of do these properties. And while they didn't work all the time for very deep architectures, it was nice to connect it to statistical thermodynamics.[9]

Tim Scarfe 00:58:29

Very cool. So there's 2 other reviews you've done as well. There's a neuro review and a machine learning review. The machine learning 1 in particular is very interesting for the audience, because of course, you know, how can we build software frameworks to build this out? How can we use existing deep learning libraries like PyTorch to do some of this stuff? I mean, could you introduce those 2 surveys as well?

Alexander Ororbia 00:58:51

Sure. So are you talking about the brain inspired learning 1 or is this the review of neuroscience?

Tim Scarfe 00:58:57

Yeah, sorry, the review of neuroscience. Yeah.

Alexander Ororbia 00:58:59

Yeah. So that 1 is like distilled version of the 1 that I was sort of explaining, the very deep 1. And it tries to say there's a few representative algorithms that we might wanna look at. Things I wanna highlight from that paper, obviously, predictive coding is 1 of those frameworks. The other 1 is something called forward only learning. My friend Adam Cohen and Habba Seigelman invented some of the earlier ideas where I can do learning in a contrastive way

[9] A Path Towards Autonomous Machine Intelligence — paper

Forward-Only Learning and ngc-learn Implementation

Alexander Ororbia 00:59:30

without feedback, which I think is a really nice thing to highlight. So predictive coding requires you to design message passing. That doesn't require it. I think it's a promising alternative direction to look at. And so to answer your question about how would I begin, what does it mean to build software to leverage the tools of deep learning to construct these libraries? Well, we know that deep learning tools rely on linear algebra. Linear algebra is a beautiful mechanism. There's no need to reinvent that so we can construct our learning algorithms with those tools. So you could take PyTorch and say, well I want to exploit the efficiency of GPU computation. And these tools have wonderful engineers that have worked for a long time. They work really well for GPU computation, so all I need to do is not use the back prop part, right, the autograd. My goal is okay I'm going to write down the ordinary differential equation that governs my dynamics if you're gonna do let's say predictive coding, but you could do this forward only learning as well. And I need to also write down in terms of that lingo, and again it's like Python programming, basic code, I'm going to construct my Hebbian rules. So you have to write these out explicitly. So it is a little more work, whereas a lot of deep learning today, you just specify a feed forward architecture, call autograd, depending on the level of involvement you want to have, and you're off to the races. But again, you're not going to get the benefits of like parallelism. Right? You're gonna trust that the sequential operation is all that you need. So as long as you're willing to write down those dynamics, plasticity and inference, and of course, could add structural selection as well, You can leverage these tools to do that calculation in a fast way. So not to do too much of a shameless plug, but my lab came up with a tool called ngc-learn, which we do talk about in 1 of those surveys because there isn't a whole lot of tools, or at least in my experience, that support those that want to work in biological learning. When I was a young grad student, I was kind of navigating the field and looking at wonderful papers and dreaming, but had to teach myself like a garage band musician. And I didn't want that to happen for future generations. So what ngc-learn is is we work in Python since that's the current lingo of the machine learning community. And we say, okay, I'm going to give you these little tiny building blocks if you want. It's also very customizable to build neuronal dynamics. And everything centers around what I told you at the very beginning, ordinary differential equations. And it does all that nicely for you, but you can look at and inspect the internals if you want. And it allows you to write those explicit equations from an inference or learning perspective very easily. So that way it supports your way of design. It's also in JAX, which is heavily optimized. It's Google's tool for GPU

computation. So it's essentially trying to do what PyTorch and TensorFlow did for Backprop, but for computational neuroscience and biomimetic intelligence. So I'd say you can take those frameworks, and as long as you can write down what it and by the way, this also supports new ideas because I don't think that there's a galaxy, as Terry Sejanowski once told me, of learning algorithms and credit assignment out there. You might have a completely different idea. But as long as you can just write down its mathematical underpinnings, you know, it could be using the tools of differential equations, then libraries like ngc-learn or you can appropriate tools like PyTorch can instantiate them. And I think that's, like, a good way to think about not being afraid of, for new users, the explicit writing down. Because that's what biological learning forces you to do. It forces you to know what the learning process is, whereas backprop is like a black box. Right? You can learn how it works, but oftentimes you're ignoring what learning is doing.

Stability-Plasticity Dilemma and Continual Learning

Tim Scarfe 01:03:25

1 thing you spoke about in the paper was stability issues, because we are now becoming neuroscientists. Yeah. Right? We're building brains in a way.

Alexander Ororbia 01:03:34

Yeah.

Tim Scarfe 01:03:34

How did you, you know, talk about overcoming these stability problems?

Alexander Ororbia 01:03:39

So the stability plasticity dilemma, if I'm making sure I'm So oriented stability is this idea that when I am learning a series of tasks. So when you train a deep neural network, let's say I train it to classify cats and dogs. Then all of a sudden I train it to classify the same network that could do that, I don't know, airplanes and cars. Something very different. Well if I were to reevaluate that back prop deep learning based network cats and dogs, I'd find my performance has gone down quite a bit depending on how long ago I did that task and how much data I've been giving it. Neural networks catastrophically forget. They have it's even when you build them as recurrent neural networks and try to give them some versions of memory, they're not very stable. That knowledge doesn't stay and

persist with time. It's very unhuman like, but it's also just very unbiological. So when you when you look to biology and things like predictive coding can be tools to combat this, You can do things like for example manipulate the dynamics. If you think about something very simple since you brought up neuroscience, the brain has a lot of complementary systems. So you have for example a cortical circuit. The predictive code, we're going to go with predictive coding since we keep using it and I want the viewers to be able to follow with that example. If we design a predictive coding circuit to perceive something and we also train that with predictive coding so at least now the algorithm is local, you'll still observe forgetting in most cases. So that alone doesn't solve the stability problem. It's just very plastic, which is great. It means it changes, it adapts very robustly. So what would you do to fight it? Well, since you can write out these explicit equations and you know how the dynamics change every time step, you have like the hood of your car, the engine of your car exposed. You can design another neural circuit and something that my group did was we designed a basal ganglia model. And in the brain, the basal ganglia, 1 of the theories is that it is like an information routing system. So you design another system. We actually used competitive learning. It's just it allows us to do some simple Hebbian learning, and its job was to just turn off and turn on neurons. It was just doing almost like a version of structural selection except our networks didn't grow. It was doing dynamic resource allocation. And when you take that little circuit and you stitch it to that cortical circuit I described earlier, forgetting goes away almost entirely. And we you can beat out backpropagation because basically what you're doing is you're saying, I'm allocating resources for a particular task and I'm preserving them. When I switch to something else, can tell the difference between data points. And that's what the tools of neuroscience allow you to do because now you have control over all aspects of the process. You know how the dynamics change. So then it's very simple for you to think about gating variables and adding these new concepts. So it gives you all these tools then to allow you to build memory systems. Backprop has so many constraints. And again too, it's very hard to add things like recurrence without having to unroll them. And when you use biological systems, it's easy, almost like putting little Lego blocks together to build ever more complicated systems.

Tim Scarfe 01:06:59

What are some of the scaling bottlenecks with implementing these systems?

Alexander Ororbia 01:07:03

I think it's actually the hardware that is available to run them. Now again, I've talked about what would be the ideal hardware that you could use, but democratization is a problem. I don't have access, for example, to the neuromorphic chips that I would want. So what ends up happening is you even though they have this potential for great parallelism and could scale far better than backprop, we're running them on the tools that we run backprop on. So we are still executing them sequentially even though mathematically it is inherently asynchronous and parallel. We're not running it like that. So these things run slower than backprop. And that is an odd and you know, there's no silver bullet. But when you simulate them on the tools of backprop or the GPUs of backpropagation based frameworks, a spiking neural net's gonna take longer than network. A feed forward network, for all its problems, is as simple as I run a chain of matrix multiplies and then the backward pass I run essentially a reverse of linear algebra operations to compute derivatives. Predictive coding, and all these inference dynamics I'm talking about or spiky neural networks, you have to think in terms of biology. I have to simulate them over a period of milliseconds. And so when you have to simulate time, which, you know, allows you to do all those nice dynamics, I'm always going to be slower on a sequential hardware system that backprop is faster. So that they're not gonna scale as well if you use the same tools of backprop, particularly the substrates. Right? I think the bait way to circumvent that I know you didn't ask how to avoid those bottlenecks is to do things like in memory computing and to take advantage of chips. Right? And I think the democratization of that and making that type of hardware available to the general public is going to be 1 of the biggest problems preventing biological learning from being the new paradigm. I think that's something we need to work on.

Neuromorphic Hardware and Challenges

Tim Scarfe 01:09:00

We need to have a computational substrate that can do mortal computation, that can take advantage of this incredible distributed asynchronous form of computation that we've just been talking about. We don't want to be running them on deep learning frameworks on GPUs. We we want to have hardware which is optimized for this type of computation. What what's the landscape?

Alexander Ororbia 01:09:22

Well, in terms of what is available today, there's companies like Intel that have like their low E2 chip. IBM has the True North. There's a couple of companies. However, I would say that while there are communities that are trying to democratize that to some degree to allow cloud access to neuromorphic chips, it's not very easy for you to work with a neuromorphic chip. And something that my own lab, and it's no criticism, it's just a particular difficulty that we have right now, perhaps it's just too new of a technology, is you can write these models down. Like you could use our tool, NGC. And we actually have a translation tool, NGC Lava, for taking your model and your dynamics and making it compliant with Intel's LOIHE 2. However, it runs on the simulator or the emulator. There is a bottleneck in terms of translating what runs on the emulator of the chip versus the microcode and the foundational tools you need to literally run it on the chip. Obviously, if you're an expert engineer and worked on the original neuromorphic chip, it's trivial for you to implement it. But for those of us that are engaged with the neuronal dynamics, that's a bottleneck that's going to prevent you from easily translating your models. I think that's what also is slowing down, in some cases, the adoption of neuromorphic platforms. And I think that's going to be that's kind of what is difficult about the landscape because it's not easy to get a neuromorphic chip. It's not as easy to get a cluster of GPUs to run your simulations on. It's not like, oh, I want a neuromorphic chip for my students to play with right now. And they're very expensive. I mean, I think that's the thing. It's a newer technology, and it's gonna take time for that to settle and, you know, be more readily available. And I'm not sure exactly what that will look like to make it easily available to everyone. However, the promise that they offer is astounding. Astounding. I do think that the energy efficiency that we talked to towards the beginning of this interview is going to be what they help resolve the most because they are literally doing in memory processing. So as long as you're comfortable with working with the dynamics of spikes, which again is all going back to like electronics and electrical engineering, it's all about voltage and electrical current and then basically gradient dynamics and we just add little spike pulses or discrete values to them, you can now run them on memristor crossbars. And so these things naturally fit almost to the way that we think of a synaptic efficacy or a synaptic connection. They map to particular memristors, you and can kind of point to where the neuronal cells are. And there's a lot of, like, things that these naturally translate to, and I think the computation almost, like, fit it fits like a glove. Right? And it feels very much in line with what I was trying to get at with mortal computation. I do think a neuromorphic mortal computer is a very interesting direction to explore, but I think the availability of chips, and I will even broadly say too, we talk about it in the Mortal Computing paper, biotic constructs like organoid intelligence or organoids stem cells to create, you

know, living computers or even xenobots. Michael Levin, I know works with them. They're his. But I don't have easy access to a xenobot or an anthrobot, which are wonderful platforms to try to implement these gradient dynamics or these candidate brain inspired algorithms, but it's not easy for everyone else to work with them. So I think that's gonna be a roadblock that we're gonna need to consider the future. So the landscape is it looks a little bleak at the moment, but I think there's the potential to change that.[10]

Neural Generative Coding and Future Directions

Tim Scarfe 01:12:58

Yeah. I've often wondered about this because it's very hard to do this kind of thing in silicon. And the reason why we have this, you know, symphony in the real world is because it's physical. And, you know, physical things are presumably kind of, you know, controlled by the laws of of of the universe, and they don't need to be simulated ad nauseam in in in a computer. But 1 thing that caught my eye in your Neuro ML review paper was neuro evolution approaches. Now, remember covering the neat algorithm. Do you remember from Kenneth Stanley, the neuro evolution of augmenting topologies? And I guess you can kind of think of that as a form of, you know, morphology generation. So you have like a population of neural networks, and you're kind of complexifying them and doing some like, you know, fitness selection and so on. That was really cool.

Alexander Ororbia 01:13:44

Yeah. And that gets that speaks to structural selection. Yes. Because you're changing the topology, right? They're evolving with time. So if we think of the message passing structure of any of these neural architectures, neat and algorithms that are like them, I think are a tool that we could use to emulate how structure changes with time, add the dynamics to the structure. Yes. So, yeah, that's a wonderful point to make.

Tim Scarfe 01:14:11

Very cool. Well, in closing, can we talk about your neurodegenerative coding paper?

[10] IBM TrueNorth — paper

Alexander Ororbia 01:14:15

Oh sure. So that paper, written a few years ago, was an interesting take again on predictive coding. And what I wanted to do in that paper was show, well answer at least 2 questions. 1 was that a lot of predictive coding makes further simplifications. I was evangelicalizing all these wonderful properties. So when I talked about 1 group of neurons trying to predict another, what we often do is we reuse those same synapses to do the message passing, to transmit the error backwards. Biology, there's no reuse of synapses. We have a lot of feedback loops, but there are other synapses that learn dynamically. So what that paper tried to do is say, let me decouple that and show you that you can use Hebbian learning to learn that feedback system jointly with the generative model. So that was 1 part. It pays homage also to Rajesh Rao, who, you know, I've studied his work a lot, his 1999 paper with Dana Ballard and the original predictive coding circuit. And basically saying, well, in in his work, he also talked about trying to decouple feedback from the generative pathway. So that paper showed you how to do that. The other thing is that also showed you how to put precision at the forefront of it, which these precision matrices, how would you learn them neurally? That was another nice little contribution. But I think the bigger 1, and I'm getting lost a little in those details, is the fact that I showed that that circuit could actually synthesize data. Now oftentimes we talk, and we've just talked about the free energy principle, predictive coding is argued to be a generative model. You can write down the equations, you can actually write down the probability value that it calculates. That is a generative model, but we never use it like 1. At least I hadn't perceived it being used up until that point. So I was I asked myself the question, what would it be like if I could dump noise into this and see what it produced? And that was 1 of the coolest parts of that paper was it is a directed generative model, so we added a little Gaussian mixture at the top to act as like a long term memory. You could dump noise into it, and what it would do is it would confabulate the kanji Japanese characters that I I trained it on or the MNIST digits. Of course, you have to use MNIST when you work on these. Or the Caltech silhouette examples. And what was cool is that I wasn't giving it that data. Right? After I trained it on its training database, you know, and again, with the biological credit assignment schemes that I was explaining to you earlier, you could synthesize this data. And it actually added a concrete bit of proof that this really worked as the generative model that Carl would want it to be. Right? It was a free energy learned, free energy principle driven generative model. And what was also super cool about that work is that we compared it to variational autoencoders, which is a back propagation based generative model, and generative adversarial networks. This was at the time. This was before the generative pre trained transformers were a thing. And we outperformed them,

which was really neat. It was actually better in terms of the metrics that we measured. And that was really useful because it helps make that argument that might make people avoid biological learning, which is that you can perform on par or even better in some ways than backpropagation. And then a lot of work built on that since then, like how do you do out of distribution generalization, which is a problem that we have in machine intelligence today. In predictive coding, the neural generative coding version that I proposed offered you tools to tackle those issues. So I thought that was the coolest part. But I think my favorite part will always be that I could make that thing generate and synthesize its own samples and see what it looked like, walk across the manifold and look at what its topmost latent codes would produce. And you know, it really embodied that notion of a generative model. That was my favorite part of that work.

Tim Scarfe 01:18:10

Beautiful. Beautiful. We've spoken about so many different approaches to, you know, biologically plausible computation. What do you think is the most promising 1?

Alexander Ororbia 01:18:20

If you had asked me that question a few years ago, I would have said predictive coding. And I still believe that predictive coding is very promising. That's what I was talking about at AGI. There is another approach that I think is really important, and I think it's forward only learning. I think it's very promising because you could argue that predictive coding has this limitation that I need to design a feedback pathway. While I believe in message passing, again that's a very Frestonian concept too, I think it's interesting if I can learn with just the forward pass or the forward dynamics without going backwards or doing any message passing itself. And what that opened the door, and I did some work with this and I have a it's on archive currently, but working on getting it finally published, the spiking version of contrastive learning. So the idea of self supervised learning without working as a generative model is a way to, I think, take the free energy principle and invert it and say, can we work in a latent space rather than trying to work with reconstructing exactly pixels because it's really hard to work with raw sensory data. And so what contrastive learning and forward only learning does is it says, let me take in parallel, which was different than contrastive Hebbian learning as we talked about earlier, I'm gonna take data and I run it through my network and I'm gonna take nonsense and run it through my network at the same time. And then I'm basically going to punish the parts of the network that got the data that was wrong and push up the it's almost like an energy function. But you're not the limitations of Contrastive Hebbian Learning is they kinda de-

pend on each other. These are parallel. So what's cool, and Jeff had it in 1 of his earliest arguments, and I've since emphasized or amplified it, is that you can decouple these positive and negative phases and almost makes it look like wake and sleep. Because when we sleep, we do a lot of memory consolidation. And I think this framework opens that up, whereas predictive coding could be probably bent to do that. But, again, you're forced to now explicitly specify what that feedback pathway is. And so I think forward only learning is among the most promising nowadays. And I think probably if you want my final final take on this is probably there's a synergy between the 2 of them.

Tim Scarfe 01:20:37

Yeah. And I mean, think Hinton was always a fan of contrastive learning, but Lecoon moved in the direction of non contrastive learning. Yeah. Do you think that could be applied to contrastive Hebbian learning? Yes.

Alexander Ororbia 01:20:48

I think in predictive coding too could be bent to do what Jan is doing. I actually am currently working on some ideas to try to prototype that to show everyone how to do that. I think I think we do need and I like the the thing of Jan's more recent comments that resonates really strongly with me is working in latent space. Working why why is that useful? And a quick quick comment on that is that it allows you to avoid getting distracted with noisy details. Oftentimes when we're processing sensory data, we are trying to sort of, again, abstract away and ignore distractors. We want just the core salient information that we need to do the task at hand, and so we need to learn certain representations that are immune to that noise. And in reinforcement learning, think this is really, really important. There's a problem called the noisy TV problem, which is agents get distracted by continuous noise and they just stay forever staring at that TV and they never do the tasks that you want. But learning representations at the latent space like Jan has, you know, been proclaiming allows you to sidestep the noisy TV problem, and I think that's a thing that we need to consider in biological learning going

Tim Scarfe 01:22:02

That's interesting. I was interviewing Randall Bellis Dreiro at ICML, and he's done a lot of work with Jan, and I think he was talking a lot about the negative sampling problem as well. Yeah. Yeah. Which is that there's an exponential number of negatives, you know, to sample outside of your distribution.

Alexander Ororbia 01:22:17

And you mentioning that actually points to a problem, and this is why I also like what Jan was talking about predicting in latent space and predicting the future of latent space. It sidesteps 1 limitation of forward only learning. Even Jeff's work on forward-forward learning and my work on predictive forward-forward learning is you need a process to generate those negative samples, and the quality of that negative process dictates how good you learn. So if we can find a way to not have to do that, I think that's a secret sauce or a secret key to moving past why forward only learning has degenerate cases. It doesn't always work. I still think it's very promising, but it allows you to not have to design the negative sampling process. It had got rid of the feedback loop, which was great if you don't want it, but you do need to generate good negative samples. Like you said, there's an exponential size of negative samples that you need to properly train these.

Tim Scarfe 01:23:13

Indeed. Alex, it's been an honor and a pleasure. Thank you for joining us today.

Alexander Ororbia 01:23:17

Thank you for having me. It was was fun talking you. Wonderful.

References

[1] **The Levin Lab**
https://drmichaellevin.org/
website

[2] **Mortal Computation: A Foundation for Biomimetic Intelligence**
https://arxiv.org/abs/2311.09589
paper

[3] **The Forward-Forward Algorithm**
https://arxiv.org/abs/2212.13345
paper

[4] **There's Plenty of Room Right Here**
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10046700/
paper

[5] **The Free-Energy Principle: A Rough Guide to the Brain**
https://www.fil.ion.ucl.ac.uk/~karl/The%20free-energy%20principle%20-%20a%20rough%20guide%20to%20the%20brain.pdf
paper

[6] **Good Regulator Theorem**
https://en.wikipedia.org/wiki/Good_regulator
website

[7] **Active Inference: The Free Energy Principle in Mind, Brain, and Behavior**
https://direct.mit.edu/books/oa-monograph/5299/Active-InferenceThe-Free-Energy-Pr
book

[8] **Predictive Coding in the Visual Cortex**
https://www.nature.com/articles/nn0199_79
paper

[9] **Hebbian Theory**
https://en.wikipedia.org/wiki/Hebbian_theory
website

[10] **Brain-Inspired Machine Intelligence: Neurobiologically-Plausible Credit Assignment**
https://arxiv.org/abs/2312.09257
paper

[11] **Hopfield Network**
https://en.wikipedia.org/wiki/Hopfield_network
website

[12] **A Tutorial on Energy-Based Learning**
https://yann.lecun.com/exdb/publis/pdf/lecun-06.pdf
paper

[13] **A Learning Algorithm for Boltzmann Machines**
https://www.cs.toronto.edu/~hinton/absps/cogscibm.pdf
paper

[14] **A Review of Neuroscience-Inspired Machine Learning**
https://arxiv.org/abs/2403.18929
paper

[15] **NEAT: NeuroEvolution of Augmenting Topologies**
https://nn.cs.utexas.edu/downloads/papers/stanley.ec02.pdf
paper

[16] **A Path Towards Autonomous Machine Intelligence**
https://openreview.net/pdf?id=BZ5a1r-kVsf
paper

[17] **Test-Time Model Adaptation with Only Forward Passes**
https://arxiv.org/abs/2404.01650
paper

[18] **Spiking Neural Predictive Coding for Continual Learning**
https://www.sciencedirect.com/science/article/pii/S0925231223004150
paper

[19] **Intel Loihi 2**
https://www.intel.com/content/www/us/en/research/neuromorphic-computing-loihi-2-t
html
website

[20] **IBM TrueNorth**
https://research.ibm.com/publications/truenorth-design-and-tool-flow-of-a-65-mw-1
paper