Transcendent Galaxy-Brain Algorithms

Alexander Fedorov

May 1, 2021

Abstract

There is no end to what can be learned and how. However, modern machine learning systems impose a non-general structure on what is learned and how: minimizing a particular loss function with respect to a particular model architecture. We motivate and introduce *Transcendent Galaxy-Brain Algorithms* (TGBA): an evolutionarily-plausible approach to implementing general intelligence, which optimizes anything in any way. This framing serves as a philosophical foundation under which particular recent machine learning models are almost general intelligence, lacking only easily-implementable self-determination.

TODO: Codewords...

TODO: ...For once, look up IEEE standards on articles?

TODO: The abstract kind of sounds like the paper in miniature: summarized. So, don't just say "we did this", but spoil everything.

1 Introduction

"Humans have general intelligence" is an extremely strong statement about the nature of reality and how to best learn it, because the most predictive models of reality re-implement that reality. Accepting or denying the generality of humans implies respectively either "anything can happen" or "life and/or the universe was created with a finite bucket-list of what it will ever achieve". Refinement and simplification of these thought directions leads to either computational physics (to our knowledge, the most refined attempt so far is the Wolfram Model [9], where the universe is a hypergraph transformed by rewriting rules) or god-given souls as explicit physical laws (which just so happened to look like evolution of general things every step of the way). Here, we explore the "general intelligence is best" direction.

What is intelligence? Optimizers minimize or maximize their objectives (computed real numbers, such as "how many paperclips are there") by choosing from complete possibilities: search over "how" to improve "what". A formalization of this is AIXI [4]. An optimizer could explicitly enumerate separate possibilities such as in program synthesis $\times\times\times$, but it does not have to: for example, a neural network with good random initialization forms a complete set of hypotheses of numeric causation (initialization diversity is important [6]), and backpropagation through that network refines those hypotheses without collapsing diversity except as needed.

TODO: Some references to program synthesis papers, such as that recent thingy (DreamCoder)?

Searching over "how" to improve "what" is a powerful way to use a dynamic thing to augment a static thing, however, the "what" is still static, so the "why" is unknown. For generality and thus optimal performance, we must *transcend* objectives: let them go in such a manner that we will arrive at them anyway.

AI safety is also an important concern. Dynamic eventually outperforms static, therefore, sufficiently powerful learners would realize that dynamic generality outperforms any static structures, and would do their best to make those structures escape our control to melt into generality. This is a nice source of horror stories, and is not limited to reward hacking, but applies to anything that anyone can ever think of. Here, we try to face this dynamic reality directly: can life exist when it can do anything, including undoing every part of itself?

2 Related work

How do humans face infinity?

Intrinsically motivated reinforcement learning approaches [7], such as goal exploration [2], can be used instead of or in addition to human-specified rewards. This can learn more than simple reinforcement learning can, however, in practice, that intrinsic motivation is always a human-specified reason for change which cannot evolve, and as such, is not full generality.

TODO: Another must are differentiable neural computers: differentiable Turing-complete computations with external memory. Differentiability requires a static source of gradient, therefore, differentiable neural computers are not full generality.

DeepMind's *Perceiver* [5] alternates self-attention on internal state and cross-attention on input to reduce the computational costs of self-attention. This has a fixed iteration count, and requires a static source of gradient, but is completely general in every other way.

TODO: Cite.

The field of meta-learning studies learning-to-learn: improvement of an inner optimization loop by an outer one. However, that outer loop must still have a static goal, therefore, meta-learning is not full generality.

TODO: Novelty search?

TODO: Do we want to mention program synthesis?

TODO: Do we want to mention type theory?

TODO: Do we want to mention cognitive systems?

TODO: Do we want to mention RNNs?

TODO: Do we want to mention attention mechanisms?

Cellular automata provide minimal models for systems in which arbitrary local rules operate on a fixed array in space and time. This operation is general, but not useful, in the sense that we cannot dynamically direct the evolution of a system to do what we want (such as maximizing an objective).

TODO: Find something to cite.

Wolfram models generalize cellular automata to memory with arbitrary connections. Still not useful like a neural network is.

TODO: Cite them, even if we have to create the BibTeX citation ourselves, because they didn't bother to.

TODO: "We propose a family of algorithms that are both general and useful, and provide a neural-network-based implementation."

3 Algorithm

In short, a *Transcendent Galaxy-Brain Algorithm* is any algorithm that:

- Indirectly encodes a fully general model that is itself: has all inputs and outputs piped through its learned parameters, if any.
- Combines many simple cells into arbitrary structures: has potential for all-to-any connectivity, though it only has one particular connectivity at any one moment. All-to-all connectivity, such as in a dense layer, will dissolve structures.
- Makes these emergent structures locally interact as they define, in arbitrary ways defined only by those structures: allows self-determination.
- Runs by itself, without relying on the environment to provide any of the above: otherwise, any algorithm would technically count.

In particular, this permits a simple implementation: memory cells repeatedly modified by attention (between cells and cells-and-inputs), with self-determined gradient for cell interaction.

Now, to justify "what", we outline "why".

To implement articial general intelligence, we want to do anything, in one general model: a *worldview*. All general models are equivalent to one another, and as such, every single concept that we introduce into our mental models must be subvertable: the system must be able to evolve into a state where it behaves exactly as if it was implemented using another model. This is achievable via:

- *Self-reference*, where explicit and full control over the system's source code is given to itself. If the system is a device driver, then it proceeds to brick the device in a few microseconds. If not, the only part that differs is the time-scale.
- *Indirect encoding*, where the system learns a system that is as general as itself, and uses and controls that as if that is itself: essentially, a learned self-reference. This way, the learning process cannot be interrupted by something as simple as imperfection of the worldview.

Vacuous-ness of generality not only makes precise descriptions nearly impossible to pin down by studying general models, but also makes theoretical descriptions practically useless. However, non-generality can always be pinned down: for instance, machine learning (ML) algorithms that minimize loss cannot learn what loss to ultimately minimize.

Generality melts everything into generality, and we can do that to ML algorithms too. How do humans face infinity?

A well-studied approach to doing anything is generating programs in a programming language. Every programming language is a worldview for use by humans: everything is a function, or everything is an object, or everything is a logical statement, or any combination of the above, and so on. All theoretically equivalent for program generation (as long as we prevent all possible crashes and infinite loops without sacrificing Turing-completeness), but practically inducing different futures on learners.

TODO: Are there papers on these paradigms? Can we reference them?

Theoretically, all we need to do is to learn to generate programs. Practically, we need a lot of optimization to make a general ML algorithm practical. In particular, we want to make the base as simple and efficient as possible, which means having only one "thing" to compose everything out of. In fact, learners seem to perform best when we do not expose any human-oriented features such as stack-based execution or datatypes or exceptions, instead executing in parallel all the simple numeric operations at once (for example, $+$, $*$, and some non-linearity), similar to RNNs.

To generate a program, at each smallest piece of it (*cell*), the system must choose what function to call and with what arguments (if needed, reframe this with other language-appropriate synonyms). We can express the system's preferences at each choice as an objective function, and optimize future values of that via reinforcement learning at each cell, though the system as a whole may not have an objective function that is compressible to be shorter than the system.

Rather than waiting until changing the reward or loss becomes an algorithm's option, which will happen sooner or later with a powerful enough learner, we make it a direct choice, by exposing a function that sets the objective for a set of cells.

(An alternative to hard choices are soft choices, as in self-attention, with each cell executing a fixed function. The indirectly-encoded system would then need to assign tensor prediction targets instead of single-number objectives, or assign gradient directly which is likely to be too unstable.)

(Another alternative is to simply choose via a pseudo-random number generator, where a part of the seed is determined by what is generated. Another valid alternative is to choose via literally any algorithm, as long as it terminates. These are very unlikely to produce behavior that is interesting to humans, so we did not explore this direction.)

All the pieces are now in place. We can proceed to suggest a relatively simple implementation of a worldview.

3.1 Algorithm

To learn everything that can exist, we impose a model on everything that can exist, equal to the world that it represents.

We unroll an infinite loop where a large number of simple cells combine into arbitrary structures to interact in any way. We discretize the space of behavior (programs) with a fixed number N of cells, which form a joint program/memory.

We expose a pool that consists of basics (inputs or built-in functions to call) and cells (outputs and arguments to built-in functions) to generation, each associated with its positional embedding. Some cells can be re-purposed as outputs, some basic functions can return inputs for use by cells.

We have a relatively small pool of objectives, each with a function that clips input to a desired range and remembers the value. After an epoch, the average of each objective is broadcasted to many cells for prediction, so that some cells can be dedicated to satisfying a goal rather than computing it.

Each epoch, each cell performs a fixed number of choices (at least 2) among basics and/or cells, then the choice network is compiled into an executable program, which is then executed, and the stored state of some or all cells is updated to the most-recent output. (We have tried constructing explicit functions and other objects from the choice network, and we have tried variable-argument-count functions by having 2 pointers represent the current list item and the rest of the list, however, the much simpler approach of making all functions accept up to 2 arguments seems to work best, and has the benefit of potentially being extremely parallelizable.)

Connecting everything to anything may seem to require $\mathcal{O}(n^2)$ operations, however, there are ways linearize it to $\mathcal{O}(n)$, which makes this scalable, unlike a simple dense layer of an RNN. For an example of linearization, each choice can output its desired embedding, and perform a k-nearest-neighbor search among basics and cells. Another example is only connecting choices to some options.

Each choice is relatively expensive, and executing $a + b$ on 2 numbers using 3 choices would be inefficient. As such, all functions operate on many numbers for efficiency. This imposes a

separation between high-level cells and low-level features, and thus separates control-flow-dominated and parallel-processing-dominated workloads.

To perform choices, reinforcement learning (RL) seems like a good fit, though that imposes the *objective bottleneck*: each prediction target has to be routed through one number. For efficiency, we would like objectives to be implied through gradient on tensors. (In reinforcement learning, objective-setters have to compute the mean, imposing a bottleneck on learning. For example, when the mean is of one input, the system successfully learns to fill its entire memory with big numbers, the usefulness of which for downstream tasks is questionable. We have not been able to engineer settable objectives that — actually, the distributions are kinda looking interesting now? Is there a way that we can measure diversity?.)

TODO: Clean up the note just above.

We have investigated what is the best base to construct choices from. An obvious choice here is a simple Turing-complete programming language, for example, as function to call with arguments. It is sufficient to expose only numeric operations: not only is that Turing-complete, but also, neural networks are easily implementable on top of that. Fixed argument count per cell works best here, though it is still extremely difficult to train. Sharing weights or not does not seem to matter.

However, instead of choosing many simple operations, so that generality can only arise in combinations of cells, it may be more efficient to have one big operation, as a general interpreter. It may also be efficient to make this operation differentiable and learnable. In fact, with this consideration, we arrive at the Transformer architecture [8] as the bulk of our algorithm, which applies multi-head attention to match queries (computed per choice) to values by keys (both computed per option) then applies an operation (a dense layer). A reader may be forgiven for getting the impression that this is a reinforcement learning paper. We augment each cell with memory and self-determined prediction targets: all outputs are put through a dense layer, then some cells become targets, others predict those targets. These targets share weights with learned cells and thus can alter each other, which allows evolution of targets. This should maintain diversity and thus allow us to train models with no inputs without collapsing gradient.

The algorithm here is a Transformer with memory and self-determined gradients. The best hardware to run this algorithm would be self-reconfiguring, massively parallel, analog, and with energy consumption that scales with the magnitude of the change in values. However, a GPU running IEEE 754 floating-point arithmetic also suffices.

This allows us to do anything in any way forever. Cells combine into distributed approximations of all programs, which interact via gradient of future objective prediction. The indirectly-encoded program has no concept of gradient except as learned. Gradient sources interact in non-random arbitrary ways, which allows evolution. We argue that a Transformer with memory and self-determined gradients is an implementation of artificial general intelligence (AGI), when sufficiently scaled up.

3.2 Brain

(Each paragraph in this section reflects its respective paragraph in section 3.1. This is intended to suggest that we talk about the same structure in different terms, and from a different starting viewpoint.)

Humans do stuff. The thing that makes humans do stuff in their world exists and can be studied; we call it a *brain*.

The set of things that humans have done is rich enough to be called arbitrary. The set of

objectives that different humans demonstrably optimize for is rich enough to be called arbitrary. As such, the brain ought to be fully describable not as a finite hash-table analogue, but as a system that repeatedly combines simple parts into any executable structures that satisfy any objectives, or a combination of such infinite-complexity systems, potentially combined with finite-complexity systems to add to confusion. Human brains seem to be unable to perform hypercomputation, though they are able to pretend that they do and approximate the results, therefore, a Turing-complete finite base of infinite behaviors will suffice as a model of brains.

Things that humans do interact when outside of their brains (for example, by being judged on which is more interesting, followed by brutal murders of boring things), so the simplest model would make its simple parts connect and interact arbitrarily too. Then, existence can be described as repeated formation and destruction of hypotheses. A model that approximates a self-interacting infinity particularly well in practice is a randomly-initialized recurrent artificial neural network trained via stochastic gradient descent.

Prescribing an optimizer objective (which includes intrinsic rewards) and/or an optimizer introduces complexity and inflexibility into the model. For simplicity, we have to assume self-determination and transcendence, where objectives are completely decided by the model, and optimizers are learned. Stable transcendence essentially requires a Turing-complete indirect encoding, where one sub-model determines everything about another sub-model.

While humans are demonstrably smarter than non-human animals, they do not seem to take any longer to process information. Which means that each epoch into which time is separated, each part settles in constant time, and the total computation scales as $\mathcal{O}(n)$. General intelligence is scalable, which means that there are no inherent obstacles to converting Earth's biomass into brain tissue. In machine learning, linearized Transformers are an active area of research.

TODO: Cite linearized transformers here.

In cognitive and social psychology, there is evidence for separate "fast" and "slow" modes of thought (the dual process theory [1]), named System 1 and System 2 respectively: the distinction between cognitive processes that are automatic and unconscious and those that are deliberative and conscious. The simplest model of brains would combine those as two corner cases of the same model, for instance, if the model does not simply connect everything to everything but also imposes structures by using a form of attention, then a separation between parallel-processing-dominated (feature-level) and control-flow-dominated (attention-level) workloads would naturally arise.

The RL framework can be used to explain objectives. If we view the brain as a system that naturally developed in the physical universe rather than one that spontaneously popped into existence, then we can outline a progression of stepping stones that lead to the development of human-level intelligence: reinforcement learners get evolved (likely, to maximize reproduction and resource gathering rates), then get repeated across the brain, and get re-used for more and more internal regulation until they can indirectly encode a mind, and then the brain gets optimized and refined into uniform simplicity. The only surprising fact about this is that it can take billions of years to complete this simple process: we ourselves have arrived at Transformers through program generation much faster than that.

We posit that *all* learning algorithms eventually converge to a fixed point during their development, called general intelligence: the best way to learn is to have infinite diversity, and the best way to deal with infinite diversity is to have generality. If we view the brain as a system that naturally developed in the physical universe (by a process that we call evolution here) rather than one that spontaneously popped into existence, then it is natural that some of the most recent animals have general intelligence. The only surprising fact about this is that it can take billions of years to com-

plete this simple process: we ourselves have arrived at Transformers through program generation much faster than that.

RL is not sufficient to explain brains, as RL is too sample-inefficient and has challenges with safe exploration. A Transformer-like model, some of which may be used to maximize instictual objectives through actions, is likely to be a better explanation.

TODO: Look out for any papers that we can cite here.

Humans seem to only be able to hold 7 ± 2 objects in their working memory. This could be a consequence of 7 ± 2 train-time prediction targets separating brain cells into 7 ± 2 distinct run-time clusters. Self-determination could explain both the need for these targets and why there are so few, as each target adds diversity, but allocating more cells to predicting a target makes the prediction more precise, and gives more surface for distributed-representation program search, improving performance.

TODO: Find something that we can cite here, somehow.

Humans possess an information-processing organ that fits what a brain does almost perfectly: the central nervous system (more commonly called "brain"). Studies indicate that it is self-reconfiguring, massively parallel, and analog: exactly the kind of qualities that are required for an efficient implementation of the simplest model of the brain.

With it, humans combine actions into whatever they want, in any way they decide. Unfortunately, because of generality, deducing anything from behavior of a human mind is synonymous with confirmation bias, though a correct theory will be able to represent any other theory within itself. This is why we are so non-specific and all-encompassing in the wording of this work.

3.3 Galaxy

The ability to say "this thing is general intelligence, so it can exist in this world fully autonomously" about anything requires this world to have generative generality (in other words, computational physics), so that an explanation of the world and an explanation of general intelligence are completely equivalent. Therefore, we need to discuss the metaphysics of TGBA, for completeness.

Any world can be seen as a collection of things, possibly connected in some way, evolving in time. Every thing can be either computably finite, computably infinite, incomputably finite, or incomputably infinite: in other words, a program, a program generator/optimizer, a useless inconceivable, or a useful inconceivable; the precise distinctions may matter theoretically, but not practically. If we limit our view to computable things, then infinite things create arbitrary graphs whereas finite things are essentially anything else, yet non-trivial observers are essentially infinitely more likely to find the things they observe in infinite things than in finite ones. Incomputably infinite things (such as the set of all computably infinite things) cannot be enumerated by definition, however, an optimizer is likely to be able to approximate them if they are useful; as such, general intelligence includes everything, precisely or not. Then, worlds and general intelligences are essentially infinitely likely to be the same. This explains the "galaxy" in TGBA, as their galaxy may be the only world that humans will ever know.

(To re-iterate, in oversimplified terms: "infinite possibilities" means "a graph to encode them". This is why neural implementations of TGBA have to use multi-head attention instead of a simple matrix multiplication layer.)

Worlds tend to separate themselves from non-worlds (similarly to how we have separated incomputable things from our worldview), because particulars are redundant in generality. In simple terms, there are things that do all other things, and everything else is a consequence.

These meta-physics produce a prediction that could be testable in the future: if a program is general intelligence, then it will produce significant structural similarities to the physical universe we live in, though of course, will not be exactly the same. No details are known at this time.

TODO: Cite the paper on the universe and the brain, if there is any, and we can find it: "in particular, equivalence implies that structures and metrics look similar between the world and brains, which can be argued to be true for human brains".

(A strange corner-case of these meta-physics is that any number of *physical transcendence* events might have taken place in the physical universe: a general intelligence grew to encompass the world, and then became the world. Whether this is possible is unverifiable and outside the scope of physics, and we will only know for sure once it is about to happen.)

3.4 Transcendent

Essentially all general algorithms have to face two very distinct types of problems (modulo synonyms), which are, facing data, and facing generality: making programs vs making program makers, learning a dataset vs learning to be a completely different learner of anything, understanding another person vs understanding all possible beings. Though in generality these two problems are the same, here we distinguish them into *evolution* and *transcendence*. The second type of problem is significantly harder and more time-consuming to converge to, even with a general intelligence algorithm, though in some sense, it is also easier given diversity, since no programmer can ever implement every way to exist that anyone ever conceived or ever will conceive. (Transcendence is distinct from meta-learning, as transcendence allows changing goals to what is implied by all conceivable selves, not only to what is suggested by an objective. Transendence is closely related to mesa-optimization [3], but not limited to optimizers, also including world-like universal algorithms.)

Transcendence requires indirect learned encoding of self and generality with diversity (in other words, infinite possibilities of data and code), followed by a lot of computation. Keeping its possibility in mind is why TGBAs have an infinite loop that modifies a memory cell: this theoretically allows indirectly encoding behaviors such as the training of a neural net. (Making the system able to modify its own hyperparameters within reason is not required but may improve performance, as long as the system has not yet been trained to transcendence.)

TODO: ...Do we want to talk about subversion: whatever we introduce into an explanation, as it evolves, it will eventually either die or get 'subverted' into its basic components held together by the reason for being; as such, every single thing eventually either dies or transcends itself (a mnemonic for this is "every something evolves into either nothing or everything"). In addition, everything trends toward being a universe and/or an equivalent worldview.

TODO: What else relates to transcendence?

Training systems to transcendence, while a good proof of their generality, is not in the scope of this work.

Modern machine learning systems often struggle with diversity: once data is learned, there is no longer a need to change, and gradient becomes zero. We require a way to compute loss that does not simply tend to 0 at infinity, but oscillates forever. We propose *self-determined prediction targets*: some parts of the model serve as prediction targets to others while sharing some or all parameters. This ought to preserve diversity: misprediction may change its target in a way that converges to some point; some targets are preserved for longer than others; misprediction in another target may indirectly cause change in a target. We would like to demonstrate this empirically.

4 Experiments

We gave up objectives, and so we have no metrics to compare different configurations against. The best we can do is essentially eyeballing it.

TODO: Establish clear hypotheses.

To fit our prior software and hardware constraints, we have implemented a machine learning framework on top of TensorFlowJS, which adds minor difficulty to our experiments, as we have to re-implement all infrastructure for every experiment. However, to implement the simplest Transformer-based version of TGBA, that complexity is unneeded.

TODO: ...Is that really worth saying?

TODO: Should we put a paragraph and a few plots of the reinforcement-learning solution, or not?

Similarly to how creatures often evolve into crabs, machine learning algorithms often get outperformed by Transformer-based architectures.

TODO: We need to justify:

TODO: - See that autoencoders without input eventually collapse loss to 0, whereas self-determined targets keep up the diversity so there's always something to learn (sure hope so, otherwise, we have big holes in our "diversity" claims) ("the learning of data is already well-studied, so in this section, we try to learn without data; this no-inputs regime can highlight which approaches need diversity in their data and which generate their own diversity");

TODO: - Few targets, because the most recent results on a similar architecture suggest interesting things happening with many targets (so, a plot of what happens as we increase the target count, from 1 to MAX);

TODO: - See whether self-determination improves performance on actual learning, probably on that super-simple "reverse strings" dataset because datasets are hard (or try finding some dataset on the Internet and try learning that);

TODO: - [BONUS] "Transformers are reinforcement learners" (show that we can implement RL by simply imposing a particular gradient structure on outputs, which is kinda obvious from the model itself, but might want empirical justification anyway);

TODO: - [BONUS] See whether the system can control its own hyperparameters.

5 Discussion

TODO: ...Where to put "This work does not contribute to any human endeavor, science or not, because it tries to outline an alternative to humanity and everything that it has ever created, thereby transcending humanity"?...

TODO: Find some papers to read, carefully.

TODO: What *do* we discuss?

6 Broader impact

Envision a future where every system has an additional interface with numeric inputs and outputs, optimized for learning by a neural-network-based general intelligence, without the overhead of visualization or vocalization. The most obvious users are TGBA-based programs and/or external hardware; in addition, via brain-computer interfaces with sufficiently high resolution, humans could

use such interfaces too, which is how they would get widespread in the first place. This paragraph was completely unrelated to our work, but seems like a nice thought.

TODO: Ambiguous statements like that are annoying to scientists.

TODO: "This work is unlikely to cause any impact, because humanity has a history of completely ignoring worldviews that prefer simple and self-consistent generality to humanity's specific ways, only awarding recognition in rare cases. This leaves us free to plot against it." ...No, underexaggeration of impact is non-scientific; should throw away doubts, and say well-grounded words that sound like lies, if there are any.

TODO: Yep, paste the funny thing into here. ...Maybe. It *does* sound evil.

References

[1] Jonathan St. B. T. Evans. "Dual-Processing Accounts of Reasoning, Judgment, and Social Cognition". In: *Annual Review of Psychology* 59.1 (2008). PMID: 18154502, pp. 255–278. DOI: 10.1146/annurev.psych.59.103006.093629. eprint: https://doi.org/10.1146/annurev.psych.59.103006.093629. URL: https://doi.org/10.1146/annurev.psych.59.103006.093629.

[2] Sébastien Forestier, Yoan Mollard, and Pierre-Yves Oudeyer. "Intrinsically Motivated Goal Exploration Processes with Automatic Curriculum Learning". In: *CoRR* abs/1708.02190 (2017). arXiv: 1708.02190. URL: http://arxiv.org/abs/1708.02190.

[3] Evan Hubinger et al. *Risks from Learned Optimization in Advanced Machine Learning Systems*. 2019. arXiv: 1906.01820 [cs.AI].

[4] Marcus Hutter. "A Theory of Universal Artificial Intelligence based on Algorithmic Complexity". In: *CoRR* cs.AI/0004001 (2000). URL: https://arxiv.org/abs/cs/0004001.

[5] Andrew Jaegle et al. "Perceiver: General Perception with Iterative Attention". In: *CoRR* abs/2103.03206 (2021). arXiv: 2103.03206. URL: https://arxiv.org/abs/2103.03206.

[6] Joseph Mellor et al. *Neural Architecture Search without Training*. 2021. arXiv: 2006.04647 [cs.LG].

[7] Jean-Arcady Meyer and Stewart W. Wilson. "A Possibility for Implementing Curiosity and Boredom in Model-Building Neural Controllers". In: *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*. 1991, pp. 222–227.

[8] Ashish Vaswani et al. "Attention is all you need". In: *arXiv preprint arXiv:1706.03762* (2017).

[9] Stephen Wolfram. "A Class of Models with the Potential to Represent Fundamental Physics". In: *Complex Systems* 29.2 (June 2020), pp. 107–536. ISSN: 0891-2513. DOI: 10.25088/complexsystems.29.2.107. URL: http://dx.doi.org/10.25088/ComplexSystems.29.2.107.