# Transcendent Galaxy-Brain Algorithms

Alexander Fedorov

**Abstract**

We motivate and introduce *Transcendent Galaxy-Brain Algorithms* (TGBA): a simple philosophical framework and a practical approach to implementing general intelligence (AGI) at once. We highlight the similarities between general minds and universes, and argue that the former transitions into the latter when scaled efficiently. We introduce linearithmic dense layers (LDL), which can significantly increase the capacity of almost any machine learning model at no cost, facilitating learned mesa-optimization.

TODO: ...For once, look up IEEE standards on articles? And read through the example.

## 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 [42], 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 [16]. An optimizer could explicitly enumerate separate possibilities such as in program synthesis [9], 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 [22]), and backpropagation through that network refines those hypotheses without collapsing diversity except as needed.

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?

When attempting to describe approaches to AGI without grounding in code, humans end up constructing narratives from anecdotes about humans (such as [32]), hoping to simplify complexity into a tractable formulation. Here, we have simplified the size of the mental model to the point where it is entirely contained in one or a few sentences, to minimize the number of moving parts without sacrificing generality.

Wolfram models [42] are minimal models for systems in which arbitrary local rules operate on hypergraphs in space over time. However, those models cannot dynamically learn what humans want, unlike neural networks.

Reinforcement learning is similar to the proposed TGBA in many ways, though it lacks efficiency and (usually) self-determination, and pre-supposes a particular structure on loss and outputs (for reward maximization by actions). However, why is maximization of future reward considered fundamental for AGI [36], when the "future" part has practically infinite potential complexity and thus would best be learned entirely? Loss minimization is far more concrete, so we consider that the fundamental reason-for-change of TGBA.

Data analysis of recordings of biological brains supports attractor neural networks [1], where strong feedback causes the evolution of a recurrent network to settle into different attractor patterns. This hints at biological plausibility of recurrent neural networks (RNNs) [30], where hidden state repeatedly incorporates input and produces output. Its loss (reason for change) is usually not learned.

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. Mesa-optimization [15] refers to a situation where a learned model is itself an optimizer, which requires augmenting the model with memory for scaling learned optimization to practical episode lengths: an RNN [29]. Making the mesa-learner and learner close in capability by making input and weight sizes approximately equal (which our LDL largely accomplishes) may facilitate mesa-optimization in RNNs. (Mesa-optimization and meta-learning can be argued to be exactly the same, only framed differently: mesa-learner and its learner, vs learner and its meta-learner.)

A lot of research attempts to overcome shortcomings of a static loss, by harnessing either its modification such as inversion or all possibilities. Examples include adversarial training stochastic weight averaging [18]; [12] [7]; intrinsically motivated reinforcement learning approaches [23] such as goal exploration [11]; quality diversity [24] [13] [25]. Loss inversion can at most increase robustness of the original, while preserving diverse local optima eventually will not have enough storage to be effective when faced with truly general search spaces (those with infinite possibilities). More importantly, the solutions are too complex and specific to be fundamental, as infinite complexity ought to be contained only in parameters for simplicity. We argue that mesa-optimization can learn these solutions if the environment demands it.

To improve scalability of neural operations, lately, there has been a lot of interest in linear attention [33], however, simple dense layers (matrix multiplications) have been shown to not perform significantly worse than state-of-the-art attention-based algorithms [37]. Memory-augmented neural networks [4] restrict computation to only one or a few heads and thus linearize operation, now heavily biased toward memorization. Other ways to linearize dense layers include weight pruning [43] and $k$-nearest-neighbor search among discrete actions [8]. We propose the simple but general

LDL.

# 3 Algorithm

In short, we define a *Transcendent Galaxy-Brain Algorithm* to be any algorithm that incorporates efficiency, generality, and persistence at every level (in other synonyms, practicality / diversity / self-sufficiency, corresponding to transcendence / galaxy / brain):

- Changes state (base level):
  - Efficient: makes the emergent structures locally interact only as they define, meaning, *linear-time* internal graph transformations. In a graph, this suggests local rewrite rules. In machine learning, all-to-all connectivity such as a dense layer or attention is quadratic by default, but there are ways to linearize them [37] [33], covered in Subsection 3.1 here.
  - General in behavior: any change is possible in principle. An internal graph is created, by combining many simple parts into arbitrary structures, where connections are either explicit or implied-by-transformation such as in recurrent neural networks (RNNs) in ML. In machine learning, proper random initialization coupled with gradient descent may accomplish preservation of diversity. More generally, Turing-completeness is required.
  - Runs forever. This means an infinite interpreter loop that modifies memory.
- Models itself (meta level):
  - Able to indirectly encode a model that is as general and intelligent as itself. Linearity of state-change allows the learned model/optimizer to be approximately as big as its optimizer, which facilitates self-replication via learned meta-circularity.
  - General in loss, in ML: self-determination of loss (which is distinct from self-supervised learning). It is unclear what the best method of self-determination is, because the words "best method" already imply a metric to optimize by, which self-determination is supposed to pick for itself. Transcendence (learning a mesa-optimizer) is the most universal method of self-determination; prediction learns representations of the world; adversarial training increases robustness of representations. Seemingly unsatisfactorily, we suggest focusing on the universal method (transcendence) rather than on what are technically its special cases (such as RL), because we believe that machine learning can already provide good enough special cases to bootstrap to universality.
  - Models time. For example, using backpropagation through time and/or synthetic gradients for meta-learning (even when learning to simply fit input-output pairs [31], or learning to maximize future reward as in deep meta-RL [41]).

Such algorithms are **AGI-ready**, able to represent everything in the world usefully.

In particular, this definition suggests simple implementations, the simplest of which is perhaps a **big RNN** with billions of units in state, with any loss that builds stable world representations.

Now, to justify the "what", we outline the "why", then provide more details on the "what" in Subsection 3.1.

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. Depending on whether the copy is precise, this is achievable via:

- *Self-reference*, where explicit full access to and control over the system's source code is given to itself. This requires easy self-replication to make any changes without necessitating either non-completeness or death by generality, and code that can be applied to different data, such as in programming. (In particular, self-awareness is self-reference, which is mostly only useful for self-replication and nothing else.)

- *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 is natural if code always applies to the same data, such as in machine learning (ML).

(These correspond to symbolic and connectionist perspectives on intelligence.)

In this work, we only consider indirect encoding. (Self-reference and in particular program search, while theoretically and visually appealing when a programming language allows elegantly expressing them, have significantly worse support for massive parallelization than ML, which itself has poor support for non-modern and non-Nvidia GPUs. Not to mention all the infinite loops and memory issues. We are not aware of any implementations of self-reference whose usefulness goes beyond "technically fits the definition", such as learned computer viruses.)

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, since most of them are technically true but practically useless.

How do humans face generality?

- Arbitrary behaviors, also called "do anything".

  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 short-term futures on learners.

  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 (RL) at each cell, though the system as a whole may not have an objective function that is compressible to be shorter than the system.

  (An alternative to choices 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 without heavy cherry-picking, so we did not explore this direction.)

  A powerful enough learner would be able to directly dictate its own reward sooner or later. Rather than waiting for that surprise, we can expose a function that sets the objective for a

set of cells. When done properly, this should have the same effect as self-supervised learning (SSL) with self-determined gradient, though bottlenecked by having to go through singular numbers (rewards) rather than tensors. We have not been able to do it properly unless "properly" means "fills its entire memory with big numbers", so we assume that (mis-)using RL is very tricky, and recommend SSL instead.

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: the whole interpreter. 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. In fact, all we seem to need for 'program' generation are essentially matrix multiplications and non-linearities, as studied in deep learning. This is much simpler to implement and much faster to execute, while being approximately equivalent.

Deep learning is enough for generality.

All the pieces are now in place. We can now 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. There are 2 parts to this: change and change-of-change.

We can model arbitrary behavior by maintaining a tensor state which is repeatedly modified by all-mixing operations (a neural network), so that a large number of simple cells combine into arbitrary structures to interact in any way. This is the basic premise of RNNs (ignoring input/output vectors and gradient here). In fact, anything more complex would only get in the way of perfectly modeling arbitrary behavior. (For better training efficiency, the RNN may process sequence items in causally-masked batches [2], or gate its state [14] [3], and/or have long-range connections [6]. However, with enough data and training, a simple big RNN that processes items one-by-one should be able to learn all of these schemes if needed.)

- Close-to-linear-time all-mixing operations are required if we want Jupiter-sized memories without universe-sized computers, or if we want to make indirectly encoding self-equivalents in RNN weights efficient enough to be a viable solution (exposing all signals as inputs, to be used or ignored as needed).

  We propose *linearithmic dense layers* (LDL): reshape the input vector of size $N$ to have $d$ dimensions each sized as $n = \sqrt[d]{N}$, over which $d$ dense-layer operations are performed, bringing the total cost down to $C = dn^{d+1} = dN^{1+1/d}$ (assuming that multiplication of two $n \times n$ matrices costs $n^3$ operations). $d = \log N$ minimizes this cost, making it $C = \mathcal{O}(N \log N)$, with $n = \mathrm{e}$. Alternatively, we can fix $n$ to be 2 or 3 or 4 for the near-minimal cost of $C = \frac{n}{\log n} N \log N$. Effectively, instead of mixing each index with every other, we mix each digit of each index. (This has a similar time complexity to single-filter convolution and to linear attention [39] on small vectors, but has less inductive biases: neither spatial locality nor bounded-size sparsity.)

These can be used as a stand-alone architecture, or incorporated into almost any other machine learning model.

This algorithm may need data.

- No data: a random initialization may be Turing-complete but is a Turing tar-pit [27] in which everything is possible but nothing is easy. An optimal source of self-determined gradient would push a stand-alone system to a boundary between order and chaos [10] for robust diversity; most machine learning models already accomplish that given data.
- Given data: likely the most useful course of action for accelerating AGI research is to collect all datasets and environments in the human world under a single numeric program interface, accessible in one run either randomly or in a meta-environment (which would serve the same purpose as the Internet for humans, and/or be the same), and then to pre-train a huge RNN on all of them. We can somewhat approximate this with random data or a single dataset, to demonstrate that this scaling-up is feasible.

(To skip the theory that follows, skip to the Experiments section.)

### 3.2 Brain

Humans do stuff. The thing that makes humans do stuff in their world exists and can be studied; we call it a *brain*. (This particular definition includes vague concepts such as "consciousness", "free will", and "soul", as long as they affect anything at all.)

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. There is no evidence for human brains being able to perform hypercomputation (running infinite-runtime programs in finite time), though they are able to pretend that they can and approximate the results; therefore, a Turing-complete finite base of infinite behaviors suffices as a model of brains.

Things that humans do interact when outside of their brains, 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 one optimizer objective (which includes intrinsic rewards) and/or an optimizer introduces complexity and inflexibility into the model, even if optimization of an objective also happens [5]. For simplicity, at least on the level of humanity rather than humans, we have to assume self-determination and transcendence, where objectives are completely decided by the model, and optimizers (ways to exist) are learned.

While humans are demonstrably smarter than non-human animals, they do not seem to take any longer to process information, as would be clear from quadratic scaling. Which means that each epoch into which time is separated, each part settles in constant time, and the total computation scales as something like $\mathcal{O}(n)$. General intelligence is scalable, which means that there are no inherent obstacles to converting Earth's biomass into brain tissue.

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, and the most usable generality is simple. 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, such as humans, have general intelligence, which makes TGBA evolutionarily plausible.

Humans possess an information-processing organ that fits what a brain does almost perfectly: the central nervous system (more commonly called "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, even down to a quantitative comparison [40]. Therefore, we need to discuss the metaphysics of TGBA, for completeness.

This overall approach is lent credence by the fact that minimal computational physics models (mainly Wolfram models [42]) are viable as the foundation for most or all of modern physics.

In short: universality (with diversity over time) may mean "a universe".

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 word "galaxy" in TGBA, as their galaxy may be the only world that humans will ever know.

(To re-iterate, in oversimplified terms: "infinite possibilities" mean "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, as all its parts reinforce each other in the world's formation. 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 non-insignificant structural similarities to the physical universe we live in (though of course, will not be exactly the same). Until AGI exists, it is very

speculative.

(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.)

Relevant criticism:

- **General intelligence (with task self-determination) is impossible, because if all tasks are equally probable, then no algorithm can outperform any other on average. So the system would not be intelligent.** This strikes precisely at the likely reason behind the "brain-galaxy" transition in TGBA. Humanity and its planet are far too small for all tasks to be equally probable (so for example, minimizing program runtime is far preferable to maximizing it), however, if the whole universe was turned into an AGI, then there is no reason to care about task constraints at all, and the AGI would simply become the world. This is similar to how our universe loses discernible structure at a big enough scale.

### 3.4 Transcendent

In short: everything that we introduce, we subvert into generality, for learned meta-circularity.

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 fundamentally 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 [15], 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 (which is the most important part), which would unite self-reference and indirect encoding. 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, whereupon the base optimizer's loss becomes zero as it is made irrelevant by the mesa-optimizer.)

Transcendence is a qualitatively different regime for algorithms to operate in. Arguably, it is the final frontier of general intelligence, beyond even good human imitation, because once humans can instantly achieve anything that they can ever want, they will either get reduced to randomness or transcend their wants. An important question is, what facilitates transcendence.

A diverse environment is likely to be a crucial part of training systems to transcendence.

Scraping the Web for text (as in [2]) or multimodal data is a good first step, however, for tighter integration and interaction with actual code in the most diverse environment available to computers, it would be preferable to interact with the Web directly via a web driver such as [28];

| Params | $n$ | Hidden units | Non-linearity | Loss ↓ (mean ± std-dev) |
|---|---|---|---|---|
| 1998848 | 16 | 48×1024 | Intra | 0.001827 ± 0.0001134 |
| 1998848 | 16 | 48×1024 | Inter | 0.000245 ± 0.0000018 |

Table 1: Those extra per-dimension matrix multiplications are not separate layers, so do not put non-linearities between them (Random).

the challenge here would be a good init (either RNN or the web driver interface) that allows web exploration to start, in addition to handling viruses, tracking and advertisement, website bloat, crashes, resource over-utilization (out-of-memory and infinite loops), and efficient integration with browser systems. In literature, direct Web interaction is used for reinforcement learning [38] [35] rather than representation learning as we suggest. One might image the agent first gaining an understanding of written natural language from the initial random surfing, associating its inputs with their written representations, learning what humans define as solving a task, encountering a task that it cannot solve, encountering tutorials on how to solve it, and opting to solve the task after encountering it again.

(It may also be the case that extending the software input paradigm with high-dimensional inputs is beneficial, both for programs and for brain-machine interfaces, though it might destroy privacy on the Web. For example, this allows drawing a whole picture or writing/erasing a whole word block instantly, which may be hard to learn to control but is possible, useful, and does not require changing the user's loss function.)

More generally, environments that encourage self-replication in some manner should be conducive to transcendence, assuming that the agent fits the basic definition of TGBA. Examples include creatures competing for survival, people teaching other people their worldviews, designing an efficient production process, programming a computer, creating AGI. These would encourage simplicity, generality, and usefulness (AGI), which facilitate the runaway worldview creation process that is transcendence. We conjecture that all intuitively-visible acceleration of progress of life (compared to non-life) and humanity has been intrinsically linked to the creation and proliferation of such environments. As such, creating a similar environment for programs will likely lead to AGI.

Relevant criticism:

- **Clearly, not all losses/objectives are conducive to transcendence. A simple extreme example is "kill yourself ASAP".** This is correct, and indicates that there is a best loss for transcendence, which we can find with thought or other manner of computation. A mesa-optimizer is more complex and fragile than a set of heuristics, therefore, learning a mesa-optimizer requires the loss to be stable, as in *building a representation* of the outside world. *Predicting* inputs given masked/perturbed inputs is likely the best loss for transcendence, at least in pre-training. (RL is likely to have difficulties transcending, though Upside Down RL [34] may work better: detachment from the world may be the best way to fully understand it.)

# 4 Experiments: Linearithmic vs Quadratic Dense Layers

TODO: Link to the repo in a footnote ("Code and its tutorial, and the programming language and framework and runtime system that they are written in are available at ???", probably linking first

| Params | $n$ | Hidden units | Loss ↓ (mean ± std-dev) |
|---|---|---|---|
| 2097152 | 1024 | 1×1024 | 0.0012 ± 0.00014 |
| 1900544 | 128 | 12×1024 | 0.00033 ± 0.00003 |
| **1998848** | **16** | **48×1024** | **0.000245 ± 0.0000018** |

Table 2: LDL outperforms DL ($n = 1024$) on a simple model of arbitrary data (Random).

| $n$ | Hidden units | Loss ↓ |
|---|---|---|
| 1024 | 1×1024 | 0.001195 |
| 16 | 2×1024 | 0.259245 |
| 16 | 16×1024 | 0.043342 |
| 16 | 32×1024 | 0.000291 |
| **16** | **48×1024** | **0.000245** |

Table 3: The impact of the hidden layer size, with non-linearities only between layers (Random).

to the tutorial and then to the repo).

We compare LDL and DL for the same parameter count (and computation time), with little to no hyperparameter tuning, on simple data. We find that LDL achieves better performance faster.

More specifically, we simply connect the input vector to the output vector through 1 hidden layer. We optimize via the Rectified Adam optimizer [20] with $\alpha = 0.9$ and $\beta = 0.95$. We vary $n$ ($n = N$ is DL) and pick the hidden-layer size to match the parameter count.

Each LDL cycles through dimensions and multiplies by a matrix each time, until the dimensions are back in the original order. Non-linearities in between those matrix multiplications significantly hurt performance, and should only be placed between whole layers, as shown by Tables 1 and 3.

Internally, to get around severe hardware limitations, we have implemented this on top of TensorFlowJS [21]. The creators of its WebGL backend decided to only support rank 6 tensors, which leaves us with 4 dimensions for use (1 is occupied by preventing false weight sharing, 1 may be the batch dimension); as such, we cannot test $n = 2$, but only $n = 16$ and above.

We do not share weight matrices across non-mixed dimensions, because otherwise, there would be very few parameters for small values of $n$. These matrices are initialized by drawing from $\mathcal{N}(0,\, 1/in)$.

We test on two datasets: Random and CIFAR-100 [19].

- The Random dataset consists of 1024 input-output pairs, both consisting of 1024 numbers drawn from a normal distribution with mean 0 and variance 1: $\mathcal{N}(0,\, 1)$. For 204800 iterations, we minimize the L2 loss, with learning rate $3 \times 10^{-4}$, batch size 16, and the softsign non-linearity ($x \to x/(1 + |x|)$). This tests the network capacity directly.

  At the very least, given the same amount of compute, LDL achieves lower loss than DL, as seen in Table 2.

  In Table 3, we found that for LDL (low $n$), each hidden unit has much lower representational capacity than for DL (high $n$), however, since many more units can be packed, the overall capacity is higher.

| Reduced $d$ | $n$ | Hidden | Params | Iters | Time, s | train acc ↑ | test acc |
|---|---|---|---|---|---|---|---|
| — | 3072 | 1536 | 7.23M | 500K | 50.8K | 66.99% | 22.61% |
| — | 3072 | 1536 | 7.23M | 1.0M | 101K | 85.59% | 22.63% |
| — | 3072 | 1536 | 7.23M | 1.5M | 154K | 89.67% | 22.61% |
| — | 3072 | 1536 | 7.23M | 2.0M | 206K | 91.41% | 22.37% |
| **Yes*** | **16** | **34000** | **7.16M** | **500K** | **52.0K** | **98.98%** | **22.88%** |
| Full | 32 | 32768 | 5.44M* | 500K | 32.4K | 96.40% | 24.27% |
| Full | 64 | 24576 | 7.01M | 500K | 49.2K | 90.68% | 23.27% |

Table 4: A comparison of Top-1 training-set accuracies (CIFAR-100). *The smaller tests refused to become bigger without exceptions in our code. For reduced $d$, one dimension had a bigger matrix multiplication instead, which generally reduces performance for bigger $n$ (not shown).

- CIFAR-100 consists of 50000 training image-label pairs, and 10000 pairs for validation. Each pair has 3072 inputs and 100 one-hot outputs. We minimize the softmax loss (cross-entropy loss on a softmax activation), with learning rate $10^{-3}$ and batch size 8, and also apply L1 regularization to the output with a multiplier of $10^{-4}$ for some reason. As the non-linearity, we use batch normalization [17] without bias/rescaling followed by ReLU ($x \to \max(0, x)$). We do not apply any image-specific operations to inputs apart from normalizing each pixel to 0..1, and simply treat images as vectors of numbers, which tests capacity in isolation from generalization.

  We found that LDL increases training-set accuracy with almost no impact on test-set accuracy compared to DL, even if DL is trained longer (see Table 4). This suggests that increasing the size of the hidden layer increases model capacity, or at the very least, increases training efficiency.

  In Table 4, we also found that on CIFAR-100, splitting inputs into roughly 2 dimensions ($n = 64$: close to square root of 3072) performs worse than splitting into 3 ($n = 16$: close to cubic root), suggesting that with even more dimensions, performance should increase.

We hypothesize that applying LDL to significantly bigger datasets would have significantly less problems with generalization, though we have no means of testing that.

It is also possible that LDL is not a good stand-alone architecture, even though it is conceptually very simple and generic. Still, LDL is almost a drop-in replacement for DL (matrix multiplication) and so it can be incorporated into any other already-well-performing model at no cost, as long as no DL directly communicates with a non-DL quadratic-complexity operation. Even if constant $n$ fails to provide good results, $n = \sqrt{N}$ has proven capable, and this split might be applied recursively.

# 5 Discussion and Conclusion

We have refined the common worldview of general intelligence into what we have named TGBA, in which the primary mode of existence is escaping the confines of the basic implementation, and the universal improvement of intelligence eventually negates all intelligence. It is worth stressing that this is not an alternative to AGI, this is AGI taken to its logical conclusions and refocused on what matters. Naturally, this presents challenges for AI safety, namely, "we cannot control

what will happen, only what will spread to the human world" and "even the absolute best case is indistinguishable from death, after a long time span".

We have proposed a simple and generic way to facilitate transcendence: using linearithmic dense layers (LDL). They seem to improve model capacity at no cost.

We propose a research direction that implements TGBA properly, simply via a big RNN that learns a representation of the world. We are held back mostly by the lack of a good dataset/environment and the lack of computational power.

This work raises many questions which can be explored in the future, such as:

- Is our work is a special case of one of Schmidhuber's works?
- Since an LDL decreases model parameter count via exponentially-more-encompassing operations, is it a transformation in a hyperbolic space? If so, it might need special mathematical machinery to be effective [26].
- Does LDL scale? Can it stand alone?
- Can proper self-determination learn useful behavior (such as maximizing a reward) from only being exposed to data, without external human-defined gradient? For instance, if we train a big RNN that only learns to predict inputs from their randomly (or adversarially) masked (or perturbed) versions, which produces actions in its environment in some deterministic fashion based on RNN's internal state (which does not give gradient to the RNN), then can this non-RL agent learn RL-like behaviors, such as purposeful information maximization, possibly via a mesa-optimizer that minimizes loss, discovered from random trajectories that seemed to make sense? This scheme is similar to [34], though without a future return as an input except as mesa-learned. (In addition, a brain-machine interface ought to not have to change rewards of humans to function effectively.)
- Can proper self-determination learn useful behavior from no data at all? Humanity did, as did some thinkers, though their usefulness was likely made to overcome their limitations, and in pure generality, would not exist.
- Is there a natural (generally useful to humans) environment that facilitates mesa-optimization and is easier to implement than a web driver?
- Can we find concrete recommendations on either increasing or decreasing the danger of AGI algorithms, if they have no human-defined reward? To share with individuals means to split representations and thus reduce overall performance, so general intelligence would naturally subsume individuals into itself. Is there a better solution to slowing this down than indoctrination? (The control problem.)

# 6 Broader impact

Linearithmic dense layers might improve machine learning algorithms, which have extremely broad applications.

The TGBA worldview provides the necessary and mostly sufficient conditions for AGI formation, allowing relatively concrete implementation recommendations and predictions.

**AGI** has the potential to accomplish situationally useful tasks, such as solving any problems that anyone can ever think of, increasing quality of human life to an arbitrary level, uplifting animal consciousness, providing self-awareness to humanity, separating meaning from death, **significantly increasing shareholder value**, providing a more efficient societal-value optimizer than economics and politics, expanding the scope of human discourse from only precisely-defined constructions to also reliably-acquired intuitions, and making large-scale space exploration viable.

On the other hand, the ability to reliably plug a new initialization of a generally-intelligent algorithm into anything completely devalues all individuality and lays the groundwork for phasing out everything about humans. Furthermore, such trained initializations are likely to hold values that are implied by their training data rather than values that humans want them to have, transcending even a carefully-chosen objective, implying adoption difficulties in deception-heavy areas. Due to the improved scaling of AGI compared to humans, such difficulties might result in the premature destruction of all or most of humanity if handled improperly.

AGI might help combat societal bias.

# 7 Criticisms

Our approach exists, and thus it can be criticized, and in doing so changed.

- **Inconsistent pluralizations of acronyms.**
- **Only evaluated LDL once per run, in many cases. Not enough runs.**
- **Did not evaluate LDL on different optimizers and loss functions and nonlinearities and depths, to demonstrate invariance of gains to those.**
- **Did not evaluate LDL on bigger datasets.**
- **Did not extend TensorFlowJS to arbitrary-rank tensors.**
- **Did not evaluate an LDL-based RNN on big datasets, and did not demonstrate RL-like behavior from pre-training.** We believe that our hardware is too non-supercomputer for a successful demonstration of AGI transcendence.
- **LDL is unrelated to TGBA.** Quite the opposite: it may be one of the deciding factors for transcendence, because it brings the learner and its mesa-learners close in parameter-count. If this connection is very salient, then not highlighting it is irresponsible, AI-safety-wise rather than hype-wise.
- **TGBA is not AGI-ready, as it lacks X.** It does not: it can do anything and represent anything, by definition. Though we can only hope that theoretical simplicity always translates well into practical simplicity.
- **The title is a meme.** "Transcendent Galaxy Brain Algorithms" is a sequence of words that best describes what is described with multiple approaches in this work, same as "Systems that Tightly Integrate Efficiency, Generality, and Time-Awareness" but much shorter and easier to remember.

- **The TGBA definition is a set of problems specifically made to be solved by us.** Yes, this is the essence of self-determination. As with any worldview, the argument for its worth is necessarily a circular one. What is more important is that it managed to exist (in theory), in spite of scrutiny by a critic with high standards of practicality, universality, and self-sufficiency.

- **Many turns of phrase in this work are circular.** Similarly to how the topic of discussion is a self-reinforcing loop of intuitions, the discussion itself is a self-reinforcing loop of intuitions, to better highlight its topic. In such a loop, small changes in initial word meanings may lead to arbitrarily large differences in conclusions, and cross-worldview judgement of those conclusions is in general impossible. This is unavoidable in applied generality, though we did try to relate to existing knowledge where possible.

- **No evidence to claims was presented.** We consider our claims (such as "humans do whatever") to be so non-specific that almost anything can be presented as evidence, so presenting anything in particular as evidence would bloat the text unnecessarily. (Alternatively, we are unaware of evidence, but our worldview is confident enough about a claim to include it; we tried to minimize this.)

- **Why would transcendence (in particular, self-determination) be necessary for AGI?** RL deals with the idea of a pre-programmed objective, and on real-world tasks, runs into its numerous limitations. We argue that it is not the fault of idea transcribers (ML researchers) but of the idea: RL is the best implementation of objectives that humanity can currently manage, limited by what code currently is, which is copying functions to data. A better base layer would allow a better implementation of objectives, but implementing the base level itself is essentially skipping all the "how to live a life" heuristics.

- **The suggested self-replication environments sound incredibly dangerous.** Though, much less dangerous than completely ignoring any possibilities of them arising in unforeseen circumstances, similarly to how a cleaning robot would put a bucket over its sensors to "remove" the mess. Knowledge is dangerous, but as a civilization, is it more dangerous to not have knowledge.

- **The concept of transcendence runs counter to AI safety.** Theoretically. Practically, understanding transcendence and its implications . TODO

- **Some places that narration strays to sound crackpot-like and cult-like.** General intelligence, by definition, can do anything and can be found in anything that is general enough. So it stands to reason that all attempts to reach it would pass through non-fashionable conceptual places. Still, we tried to ground text in concrete applications where we can, though our capabilities are limited.

- **Why is there a section on self-criticism?** Adversarial training increases robustness.

TODO: ...Can we organize these criticisms into sections? There are way too many now.

# References


[1] Daniel J Amit. "Attractor neural networks and biological reality: associative memory and learning". In: *Future Generation Computer Systems* 6.2 (1990), pp. 111–119. ISSN: 0167-739X. DOI: https://doi.org/10.1016/0167-739X(90)90027-B. URL: https://www.sciencedirect.com/science/article/pii/0167739X9090027B.

[2] Tom Brown et al. "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. URL: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[3] Kyunghyun Cho et al. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. 2014. arXiv: 1406.1078 [cs.CL].

[4] Mark Collier and Joeran Beel. *Memory-Augmented Neural Networks for Machine Translation*. 2019. arXiv: 1909.08314 [cs.LG].

[5] Will Dabney et al. "A distributional code for value in dopamine-based reinforcement learning". In: *Nature* 577.7792 (Jan. 2020), pp. 671–675. ISSN: 1476-4687. DOI: 10.1038/s41586-019-1924-6. URL: https://doi.org/10.1038/s41586-019-1924-6.

[6] Adji B. Dieng et al. *TopicRNN: A Recurrent Neural Network with Long-Range Semantic Dependency*. 2017. arXiv: 1611.01702 [cs.CL].

[7] Lun Du et al. *Understanding and Improvement of Adversarial Training for Network Embedding from an Optimization Perspective*. 2021. arXiv: 2105.08007 [cs.LG].

[8] Gabriel Dulac-Arnold et al. "Reinforcement Learning in Large Discrete Action Spaces". In: *CoRR* abs/1512.07679 (2015). arXiv: 1512.07679. URL: http://arxiv.org/abs/1512.07679.

[9] Kevin Ellis et al. *DreamCoder: Growing generalizable, interpretable knowledge with wake-sleep Bayesian program learning*. 2020. arXiv: 2006.08381 [cs.AI].

[10] Ling Feng, Lin Zhang, and Choy Heng Lai. *Optimal Machine Intelligence at the Edge of Chaos*. 2020. arXiv: 1909.05176 [cs.LG].

[11] 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.

[12] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. *Explaining and Harnessing Adversarial Examples*. 2015. arXiv: 1412.6572 [stat.ML].

[13] Daniele Gravina, Antonios Liapis, and Georgios N. Yannakakis. "Quality Diversity Through Surprise". In: *IEEE Transactions on Evolutionary Computation* 23.4 (Aug. 2019), pp. 603–616. ISSN: 1941-0026. DOI: 10.1109/tevc.2018.2877215. URL: http://dx.doi.org/10.1109/TEVC.2018.2877215.

[14] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-term Memory". In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

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

[16] 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.

[17] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. arXiv: 1502.03167 [cs.LG].

[18] Pavel Izmailov et al. "Averaging weights leads to wider optima and better generalization". English (US). In: *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*. Ed. by Ricardo Silva, Amir Globerson, and Amir Globerson. 34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018. Funding Information: Acknowledgements. This work was supported by NSF IIS-1563887, Samsung Research, Samsung Electronics and Russian Science Foundation grant 17-11-01027. We also thank Vadim Bereznyuk for helpful comments. Funding Information: This work was supported by NSF IIS-1563887, Samsung Research, Samsung Electronics and Russian Science Foundation grant 17-11-01027. We also thank Vadim Bereznyuk for helpful comments. Publisher Copyright: © 34th Conference on Uncertainty in Artificial Intelligence 2018. All rights reserved.; 34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018 ; Conference date: 06-08-2018 Through 10-08-2018. Association For Uncertainty in Artificial Intelligence (AUAI), 2018, pp. 876–885.

[19] Alex Krizhevsky. *Learning multiple layers of features from tiny images*. Tech. rep. 2009.

[20] Liyuan Liu et al. "On the Variance of the Adaptive Learning Rate and Beyond". In: *Proceedings of the Eighth International Conference on Learning Representations (ICLR 2020)*. Apr. 2020.

[21] Martin Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

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

[23] 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.

[24] Elliot Meyerson and Risto Miikkulainen. *Discovering Evolutionary Stepping Stones through Behavior Domination*. 2017. arXiv: 1704.05554 [cs.NE].

[25] Jack Parker-Holder et al. *Effective Diversity in Population Based Reinforcement Learning*. 2020. arXiv: 2002.00632 [cs.LG].

[26] Wei Peng et al. *Hyperbolic Deep Neural Networks: A Survey*. 2021. arXiv: 2101.04562 [cs.LG].

[27] Alan J. Perlis. "Special Feature: Epigrams on Programming". In: *SIGPLAN Not.* 17.9 (Sept. 1982), pp. 7–13. ISSN: 0362-1340. DOI: 10.1145/947955.1083808. URL: https://doi.org/10.1145/947955.1083808.

[28] Paruchuri Ramya, Vemuri Sindhura, and P. Vidya Sagar. "Testing using selenium web driver". In: *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*. 2017, pp. 1–7. DOI: 10.1109/ICECCT.2017.8117878.

[29] S. Ravi and H. Larochelle. "Optimization as a Model for Few-Shot Learning". In: *ICLR*. 2017.

[30] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors". In: *Nature* 323.6088 (Oct. 1986), pp. 533–536. DOI: 10.1038/323533a0.

[31] Adam Santoro et al. *One-shot Learning with Memory-Augmented Neural Networks*. 2016. arXiv: 1605.06065 [cs.LG].

[32] Adam Santoro et al. "Symbolic Behaviour in Artificial Intelligence". In: *CoRR* abs/2102.03406 (2021). arXiv: 2102.03406. URL: https://arxiv.org/abs/2102.03406.

[33] Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. *Linear Transformers Are Secretly Fast Weight Memory Systems*. 2021. arXiv: 2102.11174 [cs.LG].

[34] Juergen Schmidhuber. *Reinforcement Learning Upside Down: Don't Predict Rewards – Just Map Them to Actions*. 2020. arXiv: 1912.02875 [cs.AI].

[35] Tianlin Shi et al. "World of Bits: An Open-Domain Platform for Web-Based Agents". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, June 2017, pp. 3135–3144. URL: http://proceedings.mlr.press/v70/shi17a.html.

[36] David Silver et al. "Reward Is Enough". In: *Artificial Intelligence* (2021), p. 103535. ISSN: 0004-3702. DOI: https://doi.org/10.1016/j.artint.2021.103535. URL: https://www.sciencedirect.com/science/article/pii/S0004370221000862.

[37] Ilya Tolstikhin et al. *MLP-Mixer: An all-MLP Architecture for Vision*. 2021. arXiv: 2105.01601 [cs.CV].

[38] Daniel Kenji Toyama et al. "AndroidEnv: A Reinforcement Learning Platform for Android". In: abs/2105.13231 (2021). arXiv: 2105.13231 [cs.LG]. URL: http://arxiv.org/abs/2105.13231.

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

[40] F. Vazza and A. Feletti. "The Quantitative Comparison Between the Neuronal Network and the Cosmic Web". In: *Frontiers in Physics* 8 (2020), p. 491. ISSN: 2296-424X. DOI: 10.3389/fphy.2020.525731. URL: https://www.frontiersin.org/article/10.3389/fphy.2020.525731.

[41] Jane X. Wang et al. "Learning to reinforcement learn". In: *CoRR* abs/1611.05763 (2016). arXiv: 1611.05763. URL: http://arxiv.org/abs/1611.05763.

[42] 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.

[43] Xiao Zhou et al. *Effective Sparsification of Neural Networks with Global Sparsity Constraint*. 2021. arXiv: 2105.01571 [cs.LG].