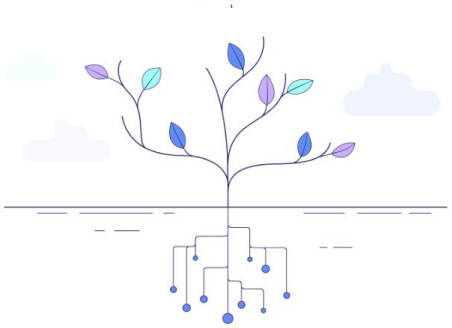
WORLDBRAIN WHITEPAPER

Empowering AI with Web3 technology



16-12-2022

Table of contents

Ⅰ. Summary

The singularity is approaching. For AI evolving on an exponential curve, it has been half a century of the long bottom curve, and AI is on the verge of an explosive period. The moment when LLM models, such as ChatGPT, manifest intelligence signifies that the AI singularity is just around the corner, likely within the next 3 to 5 years. Everything is set in motion. Web3, centered around distributed computing, is flourishing, and LLM models are steadily improving with ChatGPT leading the way.However, there are evident issues: LLM models lack usability in reasoning and planning. In the entire AI industry, LLM models exhibit high energy consumption and extreme centralization, making it challenging for AI to become widely accessible due to geopolitical conflicts.

Web3's decentralized features, with point-to-point energy consumption, decentralized (or weakly centralized) computing and storage, and unrestricted geographical boundaries, can fundamentally address these problems. WorldBrain proposes the idea of empowering AI using Web3 technology, combining the computational achievements of ChatGPT in the field of artificial intelligence. We possess all the necessary conditions to implement the "world model" theory. The working principles of Web3's distributed computing and storage align closely with the distributed functioning of neurons. With the vast amount of data already processed by ChatGPT, it meets the initial data requirements for the "world model's" memory. The organic fusion of the two will undoubtedly give rise to a truly universal artificial intelligence. The weak computing power demanded by distributed computing allows every individual on Earth to participate, and the AI emerging from distributed computing will serve all of humanity.

Ⅱ. The background of the WorldBrain project

The WorldBrain project proposes the concept of empowering AI using Web3 technology, based on the "world model" theory, to create a truly universal artificial intelligence. The various technologies involved in the WorldBrain project are already mature and widely used, especially after the emergence of high-level conversational

AI like ChatGPT, powered by LLM models. This fills the final gap in the construction of the "world model" theory. It can be said that it is precisely because of the emergence of LLM models, represented by ChatGPT, that we have full confidence in advancing AI intelligence further, making the realization of WorldBrain possible.

Ⅲ. The differences between WorldBrain and ChatGPT

3.1 WorldBrain

The "World Model" adopted by the WorldBrain project is an attempt to mimic the human brain's modeling through a large-scale intelligent neural network system. Humans develop a mental model of the world based on sensory perceptions, which include vision, hearing, touch, smell, and taste. These sensory inputs enable us to perceive and comprehend various information and features of the external world. Building upon these perceptual inputs, the human brain constructs a mental model that abstracts and generalizes the structure, attributes, behaviors, and patterns of the world.

The human mental model is dynamic. It continuously updates and evolves with accumulating experiences and learning, allowing us to predict, explain, and adapt to various situations and events in our environment. It also guides our actions and decisions. The development of artificial intelligence is based on simulating the human brain's mental model, constructing the "world model" of AI systems through processes like perception, learning, and reasoning.

ChatGPT is a powerful language model, serving as the core language model to process user language inputs and generate interactive dialogue with users. It can comprehend and generate human language. Distributed computing mimics how the human brain distributes computing tasks to multiple neural computations. The WorldBrain project combines distributed computing with ChatGPT, simulating the brain's modeling process and creating a larger-scale "world model" system.

If artificial neural networks cannot model the world like the brain does, then any deep learning network would fail to achieve the goal of general artificial intelligence, as the model itself is the knowledge. True intelligent machines, such as general artificial intelligence, will learn the world model using maps and reference frames similar to the cerebral cortex of the brain. In the human brain, maps and reference frames help us understand our spatial position and orientation, as well as our relationship with the surrounding environment. This ability enables us to navigate, plan routes, and have

spatial memory. WorldBrain can emulate this mechanism of maps and reference frames by constructing its own maps based on perceptual inputs (such as visual and positional sensor data) and representing and understanding environmental information within them.

One common approach used by WorldBrain is employing environment representation techniques, such as grid maps or topological maps, to establish spatial representations of the environment. These maps can contain information about objects, regions, paths, connections, and other elements, similar to cognitive maps in the human brain. WorldBrain can update and adjust its maps by observing and perceiving the environment and building the world model within them.

Additionally, WorldBrain can use reference frames similar to those in the human brain to learn representations of directions and spatial relationships. Human reference frames include absolute reference frames (e.g., geographical directions) and relative reference frames (e.g., self-position and orientation). WorldBrain can infer its own position and direction using sensor data and internal states, and combine this information with environmental data from the maps to understand and process spatial relationships.

By using maps and reference frames as the foundation for learning the world model, WorldBrain can attain more accurate spatial perception and environmental understanding capabilities. This will enable WorldBrain to interact better with the physical world and achieve more advanced cognitive and intelligent functionalities.

3.2 ChatGPT

ChatGPT is a language model for dialogue generation based on OpenAI's GPT (Generative Pre-trained Transformer) model. It is a deep learning natural language processing model that utilizes the Transformer architecture to generate high-quality natural language text through extensive pre-training and fine-tuning processes.

The training process of the ChatGPT model typically involves using a large amount of text data. This training data includes every word from tens of thousands of books, the entire content of Wikipedia, and nearly all information available on the internet. This text data is then provided to the artificial neural network one word at a time. Through this training process, the neural network learns the likelihood of certain words appearing after others. These language models can perform astonishing feats. Given some words, these language models can write short passages related to those words, and it's often challenging for people to distinguish whether the passage was written by a human or the artificial neural network.

There is debate among scientists about whether these language models possess genuine knowledge or merely mimic human behavior by memorizing statistical information from millions of words. If artificial neural networks cannot model the world like the brain does, then it is believed that any deep learning network would fail to achieve the goal of general artificial intelligence. While deep learning networks have achieved impressive performance, it is not due to solving the problem of knowledge representation. Instead, they entirely sidestep this issue and rely on statistical information and vast amounts of data. The workings of deep learning networks are clever, giving them remarkable performance and commercial value. However, it is essential to note that deep learning networks do not possess knowledge, and their capabilities would not match those of a five-year-old child.

Present-day deep learning networks do not possess knowledge. A computer program used to play the game of Go does not know that Go is a game or understand the game's history. It doesn't know whether it is playing against another computer or a human, nor does it understand the significance of the competition between computers and humans. Similarly, a deep learning network used to classify images can observe an image and determine if it contains a cat. However, the computer knows very little about cats. It doesn't know that cats are animals, or that they have tails, legs, and lungs. It has no knowledge of cat lovers versus dog lovers or that cats meow and shed. The deep learning network can only judge whether a new input image resembles previously seen images that were labeled as "cats," but it lacks knowledge about cats.

IV. Recent Discoveries About the Brain

4.1 The Old Brain and Neocortex in the Brain

To understand how the brain creates intelligence, we need to explain the basic functioning of the brain. The most recent part of the human brain is the neocortex, which is present in all mammals, and it is exclusive to mammals. The neocortex in the human brain is particularly large, accounting for about 70% of the brain's volume. Only a small portion consists of the old brain (as shown in Figure 1-1). The old brain controls more primitive behaviors, while the neocortex is responsible for generating more complex behaviors.

Figure 1-1 A human brain (neocortex)

The neocortex is the organ of intelligence. Almost all the capabilities we think of as intelligence—such as vision, language, music, math, science, and engineering—are created by the neocortex. When we think about something, it is mostly the neocortex doing the thinking. If we want to understand intelligence, we must understand what the neocortex does and how it does it (as shown in Figure 1-2).

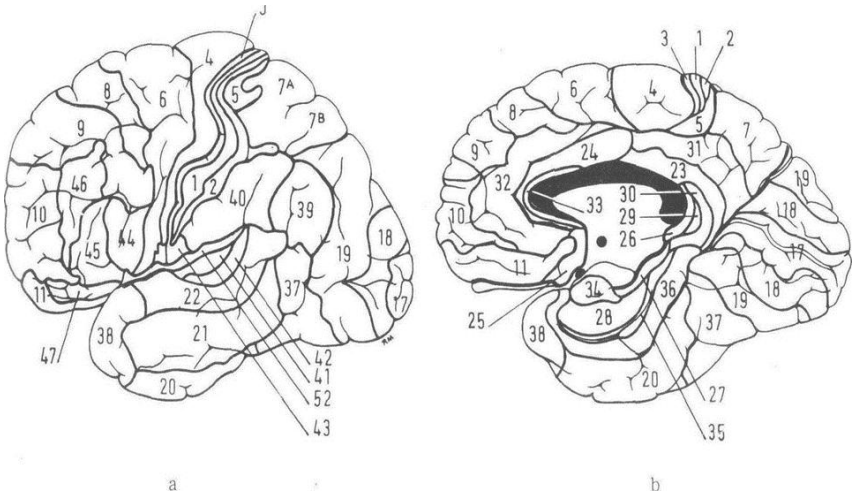


Figure 1-2 Human Brain (Regions)

The neocortex is divided into several dozen areas, or regions, that perform different functions. Some of the regions are responsible for vision, some for hearing, and some for touch. There are regions responsible for language and planning. When the neocortex is damaged, the deficits that arise depend on what part of the neocortex is affected. Damage to the back of the head results in blindness, and damage to the left side could lead to loss of language.

The regions of the neocortex connect to each other via bundles of nerve fibers that travel under the neocortex, the so-called white matter of the brain. By carefully following these nerve fibers, scientists can determine how many regions there are and how they are connected.

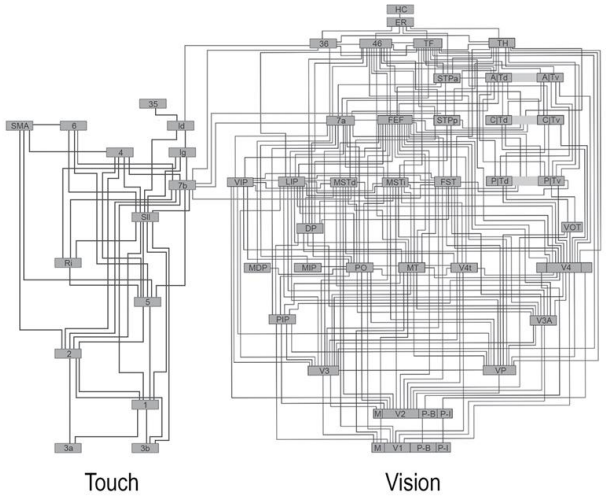


Figure 1-3 Connections in the neocortex

Figure 1-3: The dozens of small rectangles in this diagram represent different regions of the neocortex, and the lines illustrate how information flows from one area to another through the white matter.

A common interpretation of this image is that the neocortex is hierarchical, like a flowchart. Input from the senses enters at the bottom (in this diagram, input from the skin is on the left and input from the eyes is on the right). The input is processed in a series of steps, each of which extracts more and more complex features from the input. For example, the first region that gets input from the eyes might detect simple patterns such as lines or edges. This information is sent to the next region, which might detect more complex features such as corners or shapes. This stepwise process continues until some regions detect complete objects.

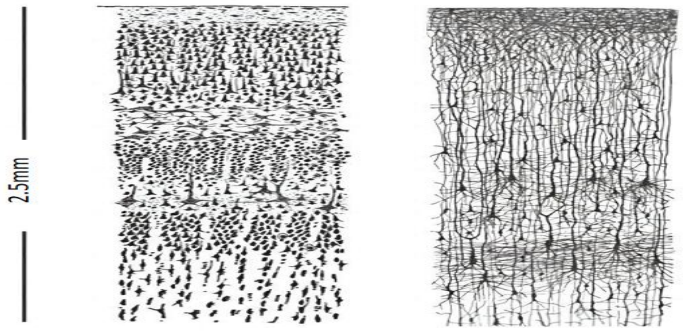


Figure 1-4 Neurons in a slice of neocortex

Figure 1-4: The neurons in the neocortex appear to be arranged in layers. The layers, which run parallel to the surface of the neocortex (horizontal in the picture), are caused by differences in the size of the neurons and how closely they are packed. Neurons are multi-layered structures, and how many layers there are depends on who is doing the counting and the criteria they use for distinguishing the layers. A simple interpretation is that each layer of neurons is doing something different.

There are dozens of different types of neurons in the neocortex. For example, one type of cell might be found in Layer 3 and another in Layer 5. Layer 1 is on the outermost surface of the neocortex closest to the skull. Layer 6 is closest to the center of the brain, farthest from the skull. The layers are only a rough guide to where a particular type of neuron might be found.

The second observation from these images was that most of the connections between neurons run vertically, between layers. Neurons have treelike appendages called axons and dendrites that allow them to send information to each other. Cajal saw that most of the axons ran between layers, perpendicular to the surface of the neocortex (up and down in the images here). Neurons in some layers make long-distance horizontal connections, but most of the connections are vertical. This means that information arriving in a region of the neocortex moves mostly up and down between the layers before being sent elsewhere.

4.1.1 The neural circuits in the neocortex

Under one square millimeter of neocortex (about 2.5 cubic millimeters), there are roughly one hundred thousand neurons, five hundred million connections between neurons (called synapses), and several kilometers of axons and dendrites.

4.1.2 Properties of the neocortex

The complex circuitry of the neocortex looks remarkably alike in visual regions, language regions, and touch regions. For example, some regions of the neocortex have more of certain cells and less of others, and there are some regions that have an extra cell type not found elsewhere. Presumably, whatever these regions of the neocortex are doing benefits from these differences. But overall, the variations between regions are relatively small compared to the similarities.

4.1.3 Operation mechanism of the neocortex

In each region of the neocortex, cells that project to some part of the old brain related to movement have been found. For example, the visual regions that get input from the eyes send a signal down to the part of the old brain responsible for moving the eyes. Similarly, the auditory

regions that get input from the ears project to the part of the old brain that moves the head. Moving your head changes what you hear, similar to how moving your eyes changes what you see.

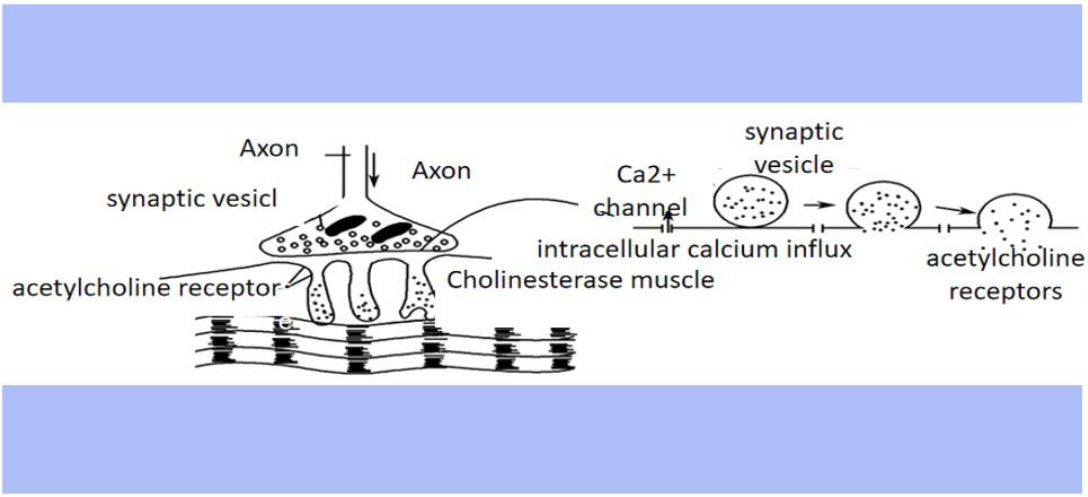


Figure 1-5 Motor mechanism of the neocortex

The evidence indicates that the complex circuitry seen everywhere in the neocortex performs a sensory-motor task. There are no pure motor regions and no pure sensory regions.

In summary, the neocortex is the organ of intelligence. It is a napkin-size sheet of neural tissue divided into dozens of regions. There are regions responsible for vision, hearing, touch, and language. There are regions that are not as easily labeled that are responsible for high-level thought and planning. The regions are connected to each other with bundles of nerve fibers. Some of the connections between the regions are hierarchical, suggesting that information flows from region to region in an orderly fashion like a flowchart. But there are other connections between the regions that seem to have little order, suggesting that information goes all over at once. All regions, no matter what function they perform, look similar in detail to all other regions.

4.2 The Intelligent Algorithm Mechanisms of the Neocortex in the Brain

4.2.1 The diversity of intelligence is also due to one basic algorithm

The neocortex got big by making many copies of the same thing: a basic circuit. Although a human neocortex is much larger than a rat or dog neocortex, they are all made of the same element—we just have more copies of that element.

Put shortly, there is nothing intrinsically motor about the motor cortex, nor sensory about the

sensory cortex. Thus the elucidation of the mode of operation of the local modular circuit anywhere in the neocortex will be of great generalizing significance.

The neocortex is divided into dozens of regions, each of which performs a different function. What makes these regions different is not their intrinsic function but what they are connected to. If you connect a cortical region to eyes, you get vision; if you connect the same cortical region to ears, you get hearing; and if you connect regions to other regions, you get higher thought, such as language. If we can discover the basic function of any part of the neocortex, we will understand how the entire thing works. The working mechanism of the neocortex has provided us with the possibility to simulate the functioning of the neocortex, which is responsible for the intelligence of the brain.

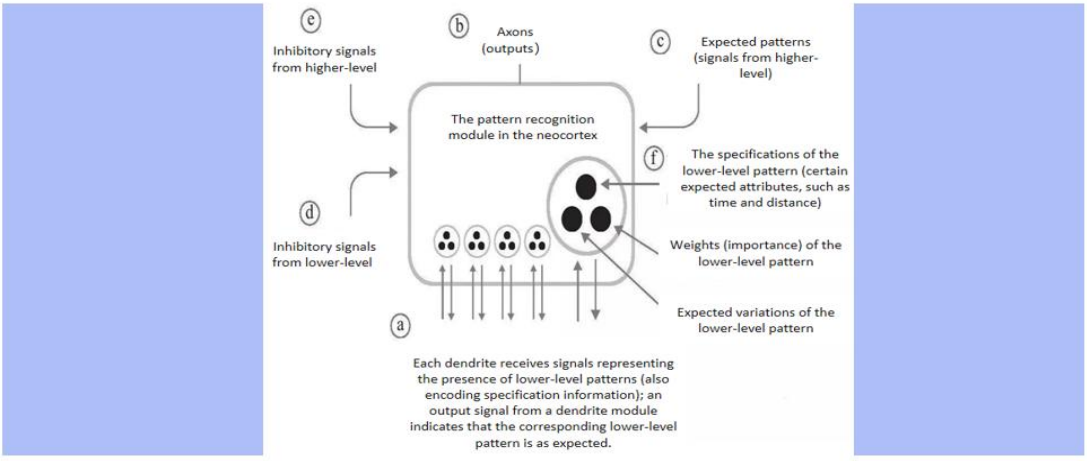


Figure 1-6 Recognition patterns in the neocortex

4.2.2 Perception and Intelligence of cortical Columns

The fundamental unit of the neocortex, the unit of intelligence, is a “cortical column.” There are roughly 150,000 cortical columns stacked side by side in a human neocortex. All the cells in one column will respond to the same part of the retina, or the same patch of skin, but then cells in the next column will all respond to a different part of the retina or a different patch of skin. This grouping of responses is what defines a column. It is seen everywhere in the neocortex. Each column is further divided into several hundred “minicolumns(Nerve bundle).” Minicolumns are physically distinct and can often be seen with a microscope.

The neocortex is divided into dozens of regions that do different things. Each region is divided into thousands of columns. Each column is composed of several hundred hairlike minicolumns, which consist of a little over one hundred cells each. Throughout the neocortex, the neocortex

columns and minicolumns performed the same function: implementing a fundamental algorithm that is responsible for every aspect of perception and intelligence.

The major expansion of the modern human neocortex relative to our hominid ancestors occurred rapidly in evolutionary time, just a few million years. This is probably not enough time for multiple new complex capabilities to be discovered by evolution, but it is plenty of time for evolution to make more copies of the same thing. The brain relies on a general-purpose method of learning. Being able to learn practically anything requires the brain to work on a universal principle.

4.3 The World Model in the Brain

4.3.1 The predictive function of the brain

The neocortex learns a model of the world, and it makes predictions based on its model. The brain creates a predictive model. This just means that the brain continuously predicts what its inputs will be. Prediction isn't something that the brain does every now and then; it is an intrinsic property that never stops, and it serves an essential role in learning. When the brain's predictions are verified, that means the brain's model of the world is accurate. A mis-prediction causes you to attend to the error and update the model. We are not aware of the vast majority of these predictions unless the input to the brain does not match. For example, if someone moves a cup from in front of your computer screen to behind it in your room, your brain's prediction of the cup being in front of the screen, based on memory (your previous observation of the cup's position), would be incorrect. The brain becomes aware of this error, updates its world model (even for a minor change), and forms new memories. These new memories then participate in the next round of predictions, achieved through a comparison process, which offers valuable insights.

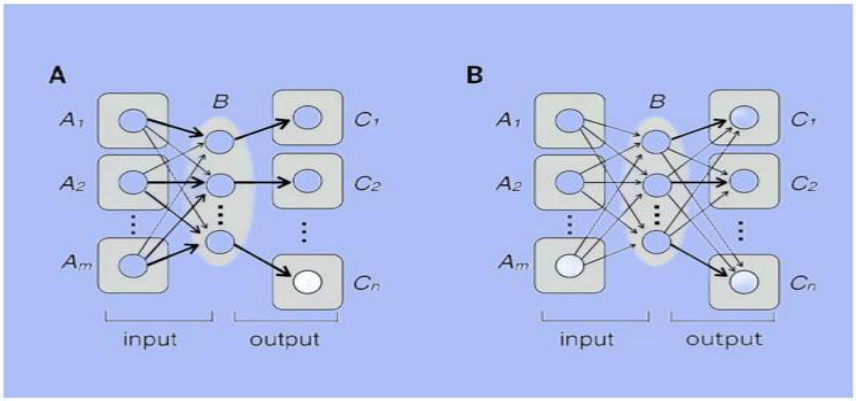


Figure 1-7 Neuronal Circuitry Structure – Brain's High-Speed Operating System

The neocortex is structured at birth to see, hear, and even learn language. But it is also true that the neocortex doesn't know what it will see, what it will hear, and what specific languages it might learn. We can think of the neocortex as starting life having some built-in assumptions about the world but knowing nothing in particular. Through experience, it learns a rich and complicated model of the world.

The number of things the neocortex learns is huge. I am sitting in a room with hundreds of objects. I will randomly pick one: a printer. I have learned a model of the printer that includes it having a paper tray, and how the tray moves in and out of the printer. I know how to change the size of the paper and how to unwrap a new ream and place it in the tray. I know the steps I need to take to clear a paper jam. I know that the power cord has a D-shaped plug at one end and that it can only be inserted in one orientation. I know the sound of the printer and how that sound is different when it is printing on two sides of a sheet of paper rather than on one.

Imagine going room to room in your home. In each room you can think of hundreds of things, and for each item you can follow a cascade of learned knowledge. It is estimated that each of us knows about forty thousand words. The brain also learns many high-level concepts. We have the ability to learn spoken language, written language, sign language, the language of mathematics, and the language of music. We learn how electronic forms work, what thermostats do, and even what empathy or democracy mean, although our understanding of these may differ. Independent of what other things the neocortex might do, we can say for certain that it learns an incredibly complex model of the world. This model is the basis of our predictions, perceptions, and actions.

Traditionally, modules in deep learning architectures communicate states through vectors or multi-dimensional arrays. But this tends to be a very inefficient method when the state of the object being modeled only changes in minor ways from one time to the next. A typical action of an agent will only modify a small portion of the state of the world. If a bottle is being moved from the kitchen to the dining room, the states of the bottle, the kitchen, and the dining room will be modified. But the rest of the world will be unaffected.

This suggests that the state of the world should be maintained in some sort of writable memory. Whenever an event occurs, only the part of the world-state memory affected by the event is to be updated, while the rest is to be left unchanged. A conventional key-value associative memory can be used for this purpose.

The output of the World Model at a given time step is a set of query-value pairs (q[i],v[i]), which are used to modify existing entries in the world-state memory, or to add new entries. Given a query q, the world-state memory returns

$$\mathrm{Mem}(q) = \sum_j c_j v_j;$$

$$\tilde{c}_j = \mathrm{Match}(k_j, q);$$

$$\mathrm{c} = \mathrm{Normalize}(\tilde{c})$$

where the k_j are keys, the v_j are stored values, function Match(k,q) measures a divergence or dissimilarity between a key and a query, vector c contains scalar coefficients c_j, and function Normalize($\tilde{c}$) performs some sort of competitive normalization or thresholding, such as the commonly-used $c_j = \exp(\tilde{c}_j)/[\gamma + {}^{P}{}_{k} \exp(\tilde{c}_k)]$, where γ is a positive constant.

$$\tilde{c}_j = \mathrm{Match}(k_j, q);$$

$$\mathrm{c} = \mathrm{Normalize}(\tilde{c})$$

$$v_j = \mathrm{Update}(r, v_j, c_j)$$

Function Update(r,v,c) may be as simple as cr + (1 − c)v.

If the query is distant from all keys, the memory may allocate a new entry whose key is q and corresponding value is r. The γ constant in the example Normalize function above may serve as a threshold for acceptable key-query divergence. One can view each entry as representing the state of an entity in the world.

In the above example of the bottle, the World Model may contain keys $k_{\text{bottle}}, k_{\text{kitchen}}, k_{\text{dining-room}}$ respectively representing the bottle, the kitchen and the dining room. The initial value of v_{bottle} encodes its location as “kitchen”, the initial value of v_{kitchen} encodes its content as including the bottle, and the initial value of $v_{\text{dining-room}}$ encodes its content as not including the bottle. After the event, the location and contents are updated. All of these operations can be done in a differentiable manner, and would hence allow to back-propagate gradients through them.

Thanks to the learning achieved by ChatGPT, these learning processes will be significantly shortened. That's why WorldBrain needs to build upon the achievements of ChatGPT.

4.3.2 The Working Mechanisms of Neurons

Like every other part of the body, the brain is composed of cells. The brain’s cells, called neurons, are in many ways similar to all our other cells. For example, a neuron has a cell membrane that defines its boundary and a nucleus that contains DNA. However, neurons have several unique properties that don’t exist in other cells in your body.

The first is that neurons look like trees. They have branch-like extensions of the cell membrane, called axons and dendrites. The dendrite branches are clustered near the cell and collect the inputs. The axon is the output. It makes many connections to nearby neurons but often travels long distances, such as from one side of the brain to the other or from the neocortex all the way down to the spinal cord.

The second difference is that neurons create spikes, also called action potentials. An action potential is an electrical signal that starts near the cell body and travels along the axon until it reaches the end of every branch.

The third unique property is that the axon of one neuron makes connections to the dendrites of other neurons. The connection points are called synapses. When a spike traveling along an axon reaches a synapse, it releases a chemical that enters the dendrite of the receiving neuron. Depending on which chemical is released, it makes the receiving neuron more or less likely to generate its own spike.

Thoughts and experiences are always the result of a set of neurons that are active at the same time. Individual neurons can participate in many different thoughts or experiences. Every thought you have is the activity of neurons. Everything you see, hear, or feel is also the activity of neurons. Our mental states and the activity of neurons are one and the same.

Each neuron has thousands of synapses, which connect the neuron to thousands of other neurons. If two neurons spike at the same time, they will strengthen the connection between them. When we learn something, the connections are strengthened, and when we forget something, the connections are weakened.

In many parts of the brain, including the neocortex, new synapses form and old ones disappear. Every day, many of the synapses on an individual neuron will disappear and new ones will replace them. Thus, much of learning occurs by forming new connections between neurons that were previously not connected. Forgetting happens when old or unused connections are removed entirely.

The connections in our brain store the model of the world that we have learned through our experiences. Every day we experience new things and add new pieces of knowledge to the model by forming new synapses. The neurons that are active at any point in time represent our current thoughts and perceptions.

4.3.3 Predictions occur inside neurons

Predictions made by the neocortex come in two forms. One type occurs because the world is changing around you. The second type of prediction occurs because you are moving relative to the world. Every column in the neocortex makes both types of predictions. Otherwise, cortical columns would have differing functions.

Predicting the next note in a melody, also known as sequence memory, is the simpler of the two problems. Sequence memory is used for a lot more than just learning melodies; it is also used in creating behaviors. Sequence memory is also used in language. Recognizing a spoken word is like recognizing a short melody. The word is defined by a sequence of phonemes, whereas a melody is defined by a sequence of musical intervals.

Figure 1-8 is a picture of the most common type of neuron in the neocortex. Neurons like this have thousands, sometimes tens of thousands, of synapses spaced along the branches of the dendrites. Some of the dendrites are near the cell body (which is toward the bottom of the image), and some dendrites are farther away (toward the top). The box shows an enlarged view of one dendrite branch so you can see how small and tightly packed the synapses are. Each bump along the dendrite is one synapse. The synapses in this area are called proximal synapses. If the proximal synapses receive enough input, then the neuron will spike. The spike starts at the cell body and travels to other neurons via the axon.

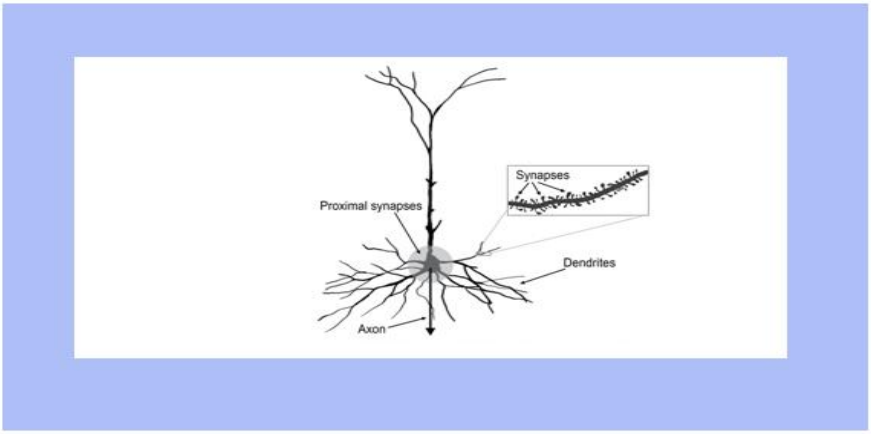


Figure 1-8 A typical neuron

Most predictions occur inside neurons. A prediction occurs when a neuron recognizes a pattern, creates a dendrite spike, and is primed to spike earlier than other neurons. With thousands of distal synapses, each neuron can recognize hundreds of patterns that predict when the neuron should become active. Prediction is built into the fabric of the neocortex, the neuron.

We spent over a year testing the new neuron model and sequence-memory circuit. We wrote

software simulations that tested its capacity and were surprised to find that as few as twenty thousand neurons can learn thousands of complete sequences. We found that the sequence memory continued to work even if 30 percent of the neurons died or if the input was noisy. By testing this theory, it has been demonstrated how the neocortex works internally.

4.3.4 The secret of the cortical column is reference frames

Creating reference frames and tracking locations is not a trivial task. It would take several different types of neurons and multiple layers of cells to make these calculations. Since the complex circuitry in every cortical column is similar, locations and reference frames must be universal properties of the neocortex. Each column in the neocortex—whether it represents visual input, tactile input, auditory input, language, or high-level thought—must have neurons that represent reference frames and locations.

Why are reference frames so important? First, a reference frame allows the brain to learn the structure of something. Second, by defining an object using a reference frame, the brain can manipulate the entire object at once. Third, a reference frame is needed to plan and create movements.

The primary function of the neocortex is to process frames of reference. Most circuits in the neocortex are dedicated to creating frames of reference and tracking positions. By associating sensory inputs with positions in the frames of reference, the brain builds a world model.

V.WorldBrain's Models, Methods and Algorithms

WorldBrain can be quickly trained in an unsupervised manner to learn compressed spatial and temporal representations of the environment. By using features extracted from the world model as inputs for the agent, a very compact and simple strategy can be trained to solve the required tasks. The strategy can even be trained entirely within the illusions and dreams generated by its world model and then transferred back to the actual environment.

5.1 Agent Model

The purpose of agent model is to find a balance between performance and efficiency. In some cases, complex models may have higher accuracy but require longer inference time or larger computational resources. To reduce inference time or resource consumption, a proxy model can be used as a substitute for the complex model. The proxy model can be a simplified version of the complex model or a model based on dimensionality reduction or feature extraction techniques.

In this model, our agent has a visual sensory component that compresses what it sees into a small representative code. It also has a memory component that makes predictions about future codes based on historical information. Finally, our agent has a decision-making component that decides what actions to take based only on the representations created by its vision and memory components.

Our agent consists of three components that work closely together: Vision (V), Memory (M), and Controller (C).

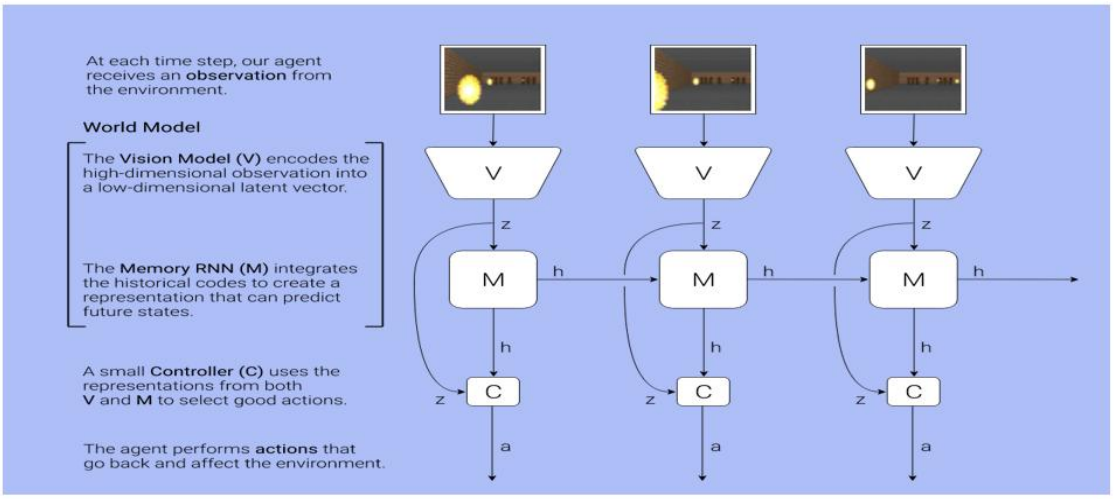


Figure 1-9

5.2 VAE (V) Model

VAE (Variational Autoencoder) is a generative model that involves the process of encoding (Encoder), decoding (Decoder), and inferring the latent space. Below are the brief steps of how VAE works:

Encoder: The encoder of VAE maps the input data to the distribution in the latent space, typically a Gaussian distribution. The encoder network takes input data and outputs two vectors: a mean vector (μ) and a variance vector (σ). These vectors parameterize the distribution in the latent space and are used for subsequent sampling of latent vectors.

Sampling of latent vectors: Based on the mean vector and variance vector outputted by the encoder, we perform random sampling from the distribution in the latent space, resulting in a latent vector (often represented as z).

Decoder: The decoder receives the latent vector z and decodes it into reconstructed output data. The decoder network maps the latent vector back to the space of the original input data, reconstructing an output that is as similar as possible to the input data.

Reconstruction error calculation: The difference between the original input data and the

reconstructed output data is compared to calculate the reconstruction error. The reconstruction error is typically measured using loss functions such as mean squared error (MSE) or cross-entropy.

KL divergence loss calculation: The difference between the distribution of the latent vectors and the prior distribution is calculated, typically using KL divergence to measure the distance between the two distributions.

Overall loss function: The reconstruction error and KL divergence loss are combined into a single overall loss function. The objective of the training process is to minimize the overall loss function, aiming to minimize the reconstruction error while aligning the distribution of latent vectors with the prior distribution.

The environment provides our agent with a high dimensional input observation at each time step. This input is usually a 2D image frame that is part of a video sequence. The role of the V model is to learn an abstract, compressed representation of each observed input frame.

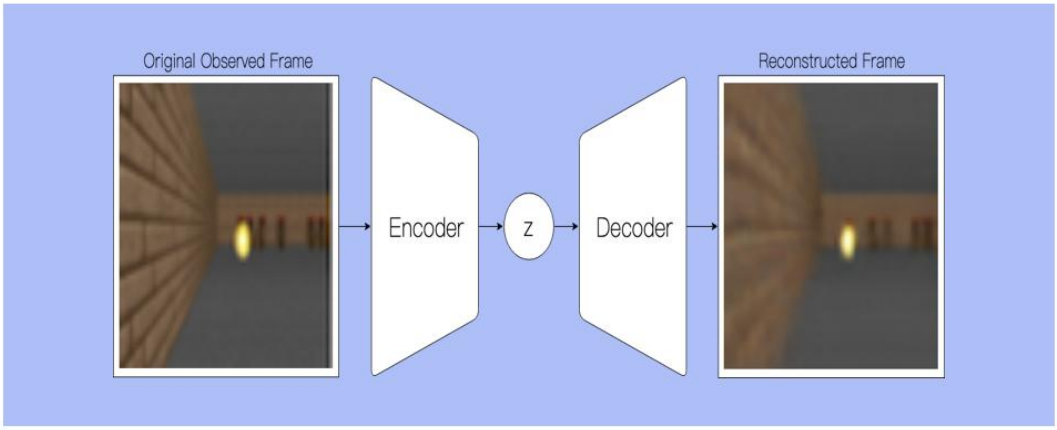


Figure 2-1

5.3 MDN-RNN Model

The MDN-RNN (Mixture Density Network - Recurrent Neural Network) model is a combination of a mixture density network and a recurrent neural network. It is used to model multimodal outputs in sequential data. The model finds widespread applications in generating and predicting sequential data, such as music generation, text generation and so on.

The MDN-RNN(M) model consists of two main components: the Recurrent Neural Network (RNN) and the Mixture Density Network (MDN).

Recurrent Neural Network (RNN): A recurrent neural network (RNN) is a type of neural network

that can process sequential data. It has recurrent connections, allowing it to preserve and utilize contextual information in the sequence data by using previous outputs as inputs. In the MDN-RNN (Mixture Density Network - Recurrent Neural Network) model, RNN is used to capture the temporal dependencies in the sequential data and learn the underlying patterns and features of the sequence.

Mixture Density Network (MDN): MDN is a neural network designed to model multimodal outputs. It achieves this by combining multiple Gaussian distributions to model the probability distribution of the output and adjusting the weights and parameters of each distribution based on the input data conditions. In the MDN-RNN(M) model, the MDN is used to model the output of the RNN and generate prediction results with multimodal characteristics.

The training process of the MDN-RNN(M) model involves two key steps:

Forward Propagation: Based on the input sequence data, the RNN network performs forward propagation to generate the RNN's output.

Mixture Density Network (MDN) Output: The output of the RNN is used as input to the MDN network to generate multi-modal output results. Each mode is represented by a set of Gaussian distribution parameters, including mean, variance, and mixture coefficients.

5.4. Controller (C) Model

The WorldBrain Controller (C) model is primarily used to implement the control and decision-making functions of the WorldBrain system. The controller is a core component of the WorldBrain system, responsible for receiving input information, processing and analyzing data, and generating corresponding outputs and decisions. The design goal of the WorldBrain Controller (C) model is to simulate the control system in the human brain to achieve intelligent and autonomous behavior. It adopts a layered structure, similar to the neural system of the human brain, to achieve multi-level information processing and decision-making.

The first layer of the controller is the perception layer, responsible for receiving sensory inputs from the external environment, such as visual, auditory, and tactile inputs. The perception layer utilizes sensor technologies to gather environmental information and converts it into a computer-readable data format.

The second layer is the information processing layer, which receives inputs from the perception layer and performs tasks such as data processing, feature extraction, and pattern recognition. The information processing layer typically employs machine learning and deep learning techniques, such as neural networks and convolutional neural networks, to learn and extract important features

and patterns from the environment.

The third layer is the decision-making layer, which formulates decisions based on the outputs of the information processing layer. The decision-making layer can employ various methods such as rule-based reasoning, reinforcement learning, and optimization algorithms. It generates corresponding behaviors and control instructions based on the current environmental state and the desired task. The WorldBrain Controller (C) model can also possess adaptive and learning capabilities. It can adjust and improve its decision-making strategies based on interactions and feedback with the environment, gradually enhancing its intelligence level and performance.

5.5 WorldBrain's Simulated Dimensions

5.5.1 Typical perception-action loops

There are two possible modes that the model can employ for a perception-action episode. The first one involves no complex reasoning and produces an action directly from the output of the perception and a possible short-term memory access. We will call it "Mode-1", by analogy with Kahneman's "System 1". The second mode involves reasoning and planning through the World Model and the cost. It is akin to model-predictive control (MPC), a classical planning and reasoning paradigm in optimal control and robotics. We will call it "Mode-2" by analogy to Kahneman's "System 2". We use the term "reasoning" in a broad sense here to mean constraint satisfaction (or energy minimization). Many types of reasoning can be viewed as forms of energy minimization.

Mode-1: Reactive behavior

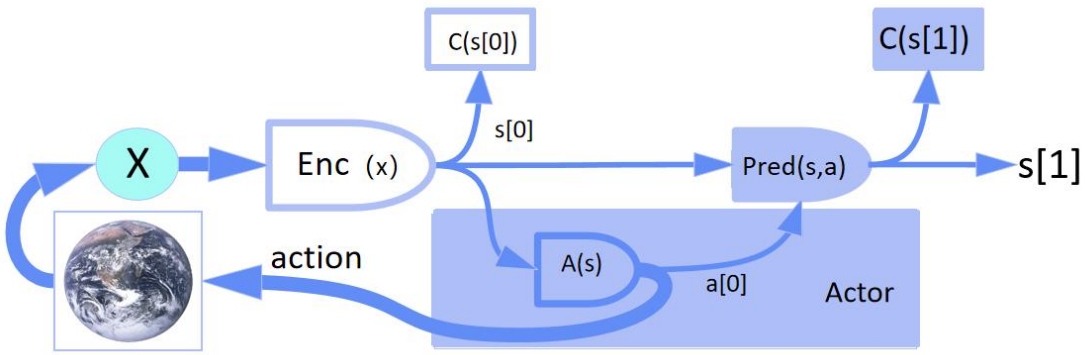


Figure 2-2

A perception-action episode for Mode-1 is depicted in Figure 2-2.The perception module, through an encoder module, extracts a representation of the state of the world s[0] = Enc(x) containing relevant information for the task at hand. A policy module, a component of the actor, produces an action as a function of the state a[0] = A(s[0]). The resulting action is sent to the effectors. The function of the policy module is modulated by the configurator, which configures it for the task at hand.

The policy module implements a purely reactive policy that does not involve deliberate planning nor prediction through the World Model. Yet, its structure can be quite sophisticated. For example, in addition to the state s[0], the policy module may access the short-term memory to acquire a more complete information about previous world states. It may use the short-term memory for the associative retrieval of an action given the current state.

While the cost module is differentiable, its output f[0] = C(s[0]) is indirectly influenced by previous actions through the external world. Since the world is not differentiable, one cannot back-propagate gradients from the cost through the chain cost ← perception ← world ← action. In this mode, gradients of the cost f[0] with respect to actions can only be estimated by polling the world with multiple perturbed actions, but that is slow and potentially dangerous. This process would correspond to classical policy gradient methods in reinforcement learning.

During Mode-1, the system can optionally adjust the World Model. It runs the World Model for one step, predicting the next state s[1], then it waits for the next percept resulting from the action taken, and uses the observed world state as a target for the predictor. With the use of a World Model, the agent can imagine courses of actions and predict their effect and outcome, lessening the need to perform an expensive and dangerous search for good actions and policies by trying multiple actions in the external world and measuring the result.

Mode-2: reasoning and planning using the World Model

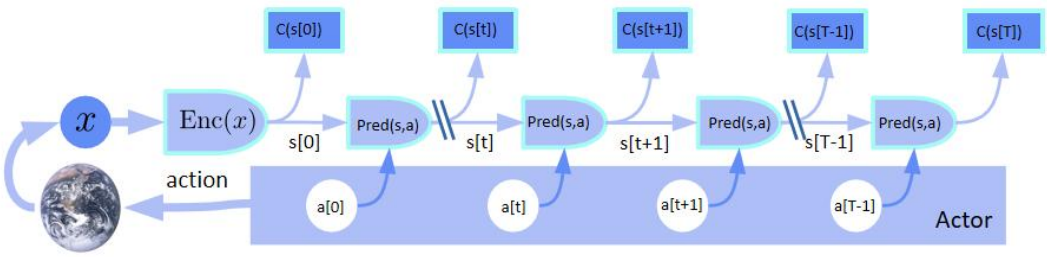


Figure 2-3

Figure 2-3 depicts Mode-2 perception-action episode. The perception module estimates the state of the world s[0]. The actor proposes a sequence of actions a[0],a[1],...,a[t],a[t + 1],...,a[T]. The World Model recursively predicts an estimate of the world state sequence using s[t+1] = Pred(s[t],a[t]). The cost C(s[t]) computes an energy for each predicted state in the sequence, the total energy being the sum of them. Through an optimization or search procedure, the actor infers a sequence of actions that minimizes the total energy. It then sends the first action in the sequence (or the first few actions) to the effectors. This is, in effect, an instance of classical model-predictive control with receding horizon planning. Since the cost and the model are differentiable, gradient-based methods can be used to search for optimal action sequences as in classical optimal control. Since the total energy is additive over time, dynamic programming can also be used, particularly when the action space is small and discretized. Pairs of states (computed by the encoder or predicted by the predictor) and corresponding energies from the intrinsic cost and the trainable critic are stored in the short-term memory for subsequent training of the critic.

Perception: the perception system extract a representation of the current state of the world s[0] = P(x). The cost module computes and stores the immediate cost associated with that state.

Action proposal: the actor proposes an initial sequence of actions to be fed to the World Model for evaluation (a[0],...,a[t],...,a[T]).

Simulation: the World Model predicts one or several likely sequence of world state representations resulting from the proposed action sequence (s[1],...,s[t],...,s[T]).

Evaluation: the cost module estimates a total cost from the predicted state sequence, generally as a sum over time steps F(x) = PTt=1 C(s[t])

Planning: the actor proposes a new action sequence with lower cost. This can be done through a gradient-based procedure in which gradients of the cost are back-propagated through the compute graph to the action variables. The resulting minimum-cost action sequence is denoted (ă[0],...,ă[T]). Full optimization may require iterating steps 2-5.

Acting: after converging on a low-cost action sequence, the actor sends the first action (or first few actions) in the low-cost sequence to the effectors. The entire process is repeated for the next perception-action episode.

Memory: after every action, the states and associated costs from the intrinsic cost and the critic are stored in the short-term memory. These pairs can be used later to train or adapt the critic.

This procedure is essentially what is known as Model-Predictive Control (MPC) with receding horizon in the optimal control literature. The difference with classical optimal control is that the

World Model and the cost function are learned. In principle, any form of optimization strategy can be used, for step 5. While gradient-based optimization methods can be efficient when the World Model and cost are well-behaved, situations in which the action-cost mapping has discontinuities may require to use other optimization strategies, particularly if the state and/or action spaces can be discretized. These strategies include dynamic programming, combinatorial optimization, simulate annealing and other gradient-free methods, heuristic search techniques (e.g. tree search with pruning), etc.

To simplify, the process was described in the deterministic case, i.e. when there is no need to handle the possibility of multiple predictions for s[t+1] resulting from a given initial state s[t] and action a[t]. In real situations, the world is likely to be somewhat unpredictable. Multiple states may result from a single initial state and action due to the fact that the world is intrinsically stochastic (aleatoric uncertainty), or that the state representation s[t] contains incomplete information about the true world state (epistemic uncertainty), or that the World Model's prediction accuracy is imperfect due to limited training data, representational power, or computational constraints.

From Mode-2 to Mode-1: Learning New Skills.

Using Mode-2 is onerous. The agent only possesses one World Model "engine". It is configurable by the configurator for the task at hand, but it can only be used for a single task at a time. Hence, similarly to humans, the agent can only focus on one complex task at a time. Mode-1 is considerably less onerous, since it only requires a single pass through a policy module. The agent may possess multiple policy modules working simultaneously, each specialized for a particular set of tasks. The process shows how a policy module A(s[t]) can be trained to produce approximations of the optimal actions resulting from Mode-2 reasoning. The system is run on Mode-2, producing an optimal action sequence ($\breve{a}[0],...,\breve{a}[t],...,\breve{a}[T]$). Then, the parameters of the policy module A(s[t]) are updated to minimize a divergence measure between its output and the optimal action at that time $D(\breve{a}[t],A(s[t]))$. Once properly trained, the policy module can be used to directly produce an action in Mode-1 $\tilde{a}[0] = A(s[0])$. It can also be used to recursively compute an initial action sequence proposal before Mode-2 optimization:

$$s[t+1] = Pred(s[t],a[t]) \; ; \tilde{a}[t+1] = A(s[t+1])$$

The policy module can be seen as performing a form of amortized inference. This process allows the agent to use the full power of its World Model and reasoning capabilities to acquire new skills that are then "compiled" into a reactive policy module that no longer requires careful planning.

Reasoning as Energy Minimization.

The process of elaborating a suitable action sequence in Mode-2 can be seen as a form of reasoning. This form of reasoning is based on simulation using the World Model, and optimization of the energy with respect to action sequences. More generally, the “actions” can be seen as latent variables representing abstract transformations from one state to the next. This type of planning though simulation and optimization may constitute the kind of reasoning that is most frequent in natural intelligence.

Many classical forms of reasoning in AI can actually be formulated as optimization problems (or constraint satisfaction problems). It is certainly the case for the kind of probabilistic inference performed with factor graphs and probabilistic graphical models. The proposed architecture is, in fact, a factor graph in which the cost modules are log factors. But the kind of reasoning that the proposed architecture enables goes beyond traditional logical and probabilistic reasoning. It allows reasoning by simulation and by analogy.

5.5.2 The cost module as the driver of behavior

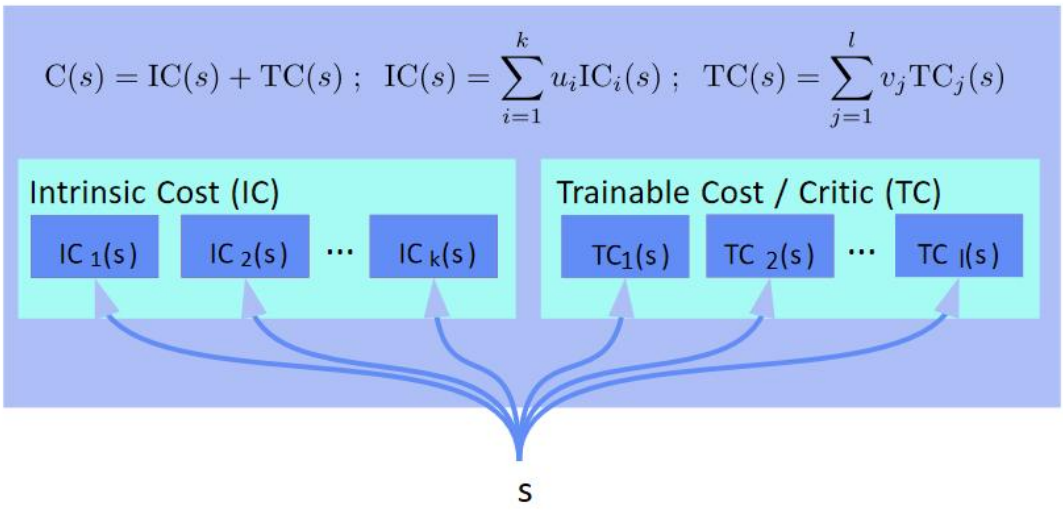


Figure 2-4

The cost module is composed of the intrinsic cost module which is immutable ICi(s) and the critic or Trainable Cost TCj(s), which is trainable. Both IC and TC are composed of multiple submodules whose output energies are linearly combined. Each submodule imparts a particular behavioral drive to the agent. The weights in the linear combination, ui and vj, are modulated by the configurator module and allow the agent to focus on different subgoals at different times.

The overall architecture of the cost module is shown in Figure 2-4. It is composed of the intrinsic cost module which is immutable ICi(s) and the critic or Trainable Cost TCj(s), which is trainable. Both IC and TC are composed of multiple submodules whose output energies are linearly

combined.

$$\mathrm{C}(s) = IC(s) + TC(s) \quad (1)$$

$$\mathrm{IC}(s) = \sum_{i=1}^{k} u_i \, IC_i(s) \quad (2)$$

$$\mathrm{TC}(s) = \sum_{j=1}^{l} v_i \, TC_i(s) \quad (3)$$

Each submodule imparts a particular behavioral drive to the agent. The weights in the linear combination, ui and vj, are modulated by the configurator module and allow the agent to focus on different subgoals at different times.

The intrinsic cost module (IC) is where the basic behavioral nature of the agent is defined. It is where basic behaviors can be indirectly specified. For a robot, these terms would include obvious proprioceptive measurements corresponding to “pain”, “hunger”, and “instinctive fears”, measuring such things as external force overloads, dangerous electrical, chemical, or thermal environments, excessive power consumption, low levels of energy reserves in the power source, etc.

They may also include basic drives to help the agent learn basic skills or accomplish its missions. For example, a legged robot may comprise an intrinsic cost to drive it to stand up and walk. This may also include social drives such as seeking the company of humans, finding interactions with humans and praises from them rewarding, and finding their pain unpleasant (akin to empathy in social animals). Other intrinsic behavioral drives, such as curiosity, or taking actions that have an observable impact, may be included to maximize the diversity of situations with which the World Model is trained.

The IC can be seen as playing a role similar to that of the amygdala in the mammalian brain and similar structures in other vertebrates. To prevent a kind of behavioral collapse or an uncontrolled drift towards bad behaviors, the IC must be immutable and not subject to learning (nor to external modifications). The role of the critic (TC) is twofold: (1) to anticipate long-term outcomes with minimal use of the onerous World Model, and (2) to allow the configurator to make the agent focus on accomplishing subgoals with a learned cost.

In general, the behavioral nature of an AI agent can be specified in four ways:

1. by explicitly programming a specific behavior activated when specific conditions are met.

2. by defining an objective function in such a way that the desired behavior is executed by the agent as a result of finding action sequences that minimize the objective.

3. by training the agent to behave a certain way through direct supervision. The agent observes the actions of an expert teacher and trains a Mode-1 policy module to reproduce it.

4. by training the agent through imitation learning. The agent observes expert teachers and infers an objective function that their behavior appears to be optimizing when they act. This produces a critic submodule for Mode-2 behavior. This process is sometimes called inverse reinforcement learning.

The second method is considerably simpler than the first one, because it merely requires to design an objective, and not design a complete behavior. The second method is also more robust: a preordained behavior may be invalidated by unexpected conditions or a changing environment. With an objective, the agent may adapt its behavior to satisfy the objective despite unexpected conditions and changes in the environment. The second method exploits the learning and inference abilities of the agent to minimize the amount of priors hard-wired that are likely to be brittle.

5.6 Analog Dimensions

Perception and Behavior Simulation: WorldBrain can imitate human cognitive abilities and behavioral performances through perception and behavior simulation. In terms of perception, WorldBrain utilizes computer vision and speech recognition technologies to perceive visual and auditory inputs, thereby gathering environmental information. In the aspect of behavior simulation, WorldBrain uses robotics or virtual simulation environments to simulate human actions and interactions, engaging in real-time interactions with the environment. This enables WorldBrain to perceive and interact with the environment like a human embodied system.

Motion and Body Representation Simulation: WorldBrain can imitate human motion capabilities and body expressions through motion and body representation simulation. By controlling the motion and posture of robots, WorldBrain can simulate human movement and manipulation in space. This includes precise manipulation of robot arms and adjustments of robot body posture, allowing WorldBrain to perform complex operations and interactions like humans.

Spatial and Contextual Modeling: WorldBrain can imitate human understanding and reasoning abilities about the environment through spatial and contextual modeling. By modeling the spatial structure and contextual information of the environment, WorldBrain can acquire knowledge about aspects such as objects, space, and time. This may involve using maps, sensor data, and contextual information to construct an internal representation of the environment, enabling better understanding and reasoning for tasks related to the environment.

Distributed Brain as a Unique Embodiment: The human brain's neocortex, on average, has around

150,000 cortical columns, each modeling a part of the world it can perceive. The cortical columns in the WorldBrain intelligent terminal do not necessarily have to be adjacent to each other, unlike those in the biological brain. An intelligent terminal might have millions of cortical columns and thousands of sensor arrays, or it could have just one analog neuron and a few sensor arrays. Sensors and their associated models could be distributed across the Earth's surface, in the ocean, or on the top of solar-powered plants. For example, a sensor distributed on the Earth's surface intelligent terminal might understand global weather patterns just like humans do.

5.6.1 Imitation of the Old Brain System by WorldBrain

Approximately 70% of the human brain's volume consists of the neocortex, while the remaining 30% belongs to the old brain, which is the oldest and simplest neural system responsible for basic behaviors such as breathing, eating, sex, and reflexive responses that involve input-output relationships. WorldBrain adopts the following methods and technologies to mimic the old brain system:

Neural Network Simulation: A useful action by a neuron takes at least 5 milliseconds. The operating speed of silicon transistors is a million times faster than neurons. Therefore, a silicon-based neocortex could potentially operate at a million times the speed of human thought and learning. WorldBrain can utilize neural network models to simulate neurons and neural network structures within the old brain system. By constructing deep neural networks, WorldBrain can simulate the information processing and transmission mechanisms present in the old brain system. These neural network models may include perception networks, memory networks, and decision networks, among others, to imitate various functionalities and features of the old brain system.

Synaptic Plasticity Mechanism: Synaptic plasticity in the old brain system serves as the foundation for learning and memory. WorldBrain can imitate the learning and adaptive capabilities of the old brain system by employing similar mechanisms. By utilizing appropriate learning algorithms and models, WorldBrain can simulate synaptic strengthening and weakening, enabling adaptive learning and memory functions.

Simulating Old Brain Structure and Function: WorldBrain can attempt to simulate specific structures and functions of the old brain system. For instance, by simulating the hierarchical structure of the brain cortex and the functional specialization of different regions, WorldBrain can achieve similar information processing and analytical capabilities. This involves constructing complex computational models and algorithms to simulate various functional modules within the old brain system.

Sensing and Perception Simulation: The old brain system acquires external information through perception and sensory processing. WorldBrain can simulate the perception process within the old brain system by utilizing computer vision, speech recognition, and other sensing technologies. By imitating the sensory mechanisms of the old brain system, WorldBrain can gather input data from the environment and perform corresponding analysis and responses.

5.6.2 Imitation of the Neocortical System by WorldBrain

Every part of the neocortex operates based on the same fundamental principles. From vision, touch, and language to higher-level thinking, everything we consider intelligent is fundamentally similar. WorldBrain employs the following methods and technologies to imitate the neocortical system:

Neuronal Network System: WorldBrain can utilize neural network models to mimic the neurons and neural network structures present in the neocortical system. The neocortical system is responsible for advanced cognitive functions in the brain, such as language processing, decision-making, and abstract reasoning. By constructing intricate deep neural network models, WorldBrain can simulate the information processing and transmission methods of the neocortical system, thus achieving comparable advanced cognitive abilities.

Memory and Learning Mechanisms: The neocortical system plays a crucial role in learning and memory. WorldBrain can mimic the learning and adaptive abilities of the neocortical system using similar mechanisms. For instance, by employing suitable learning algorithms and models, WorldBrain can simulate long-term memory and associative learning, thus achieving comparable learning and reasoning capabilities within the neocortical system.

Language Processing and Understanding: The neocortical system plays a critical role in language processing and comprehension. WorldBrain can imitate the language processing abilities of the neocortical system by employing natural language processing techniques and language models. This includes simulating aspects such as semantic understanding, syntactic analysis, and language generation, enabling WorldBrain to engage in language communication and comprehension similar to humans.

Abstract Reasoning and Decision-Making: The neocortical system possesses advanced capabilities in abstract reasoning and decision-making. WorldBrain can imitate these functionalities of the neocortical system by constructing logical reasoning and decision-making models. This involves the application of techniques such as symbolic logic, rule engines, and decision trees to achieve similar abilities in abstract reasoning and decision-making.

5.7 WorldBrain and Maps in the Brain

5.7.1 Establishing a reference frame for the WorldBrain model

When animals first started moving about in the world, they needed a mechanism to decide which way to move. Simple animals have simple mechanisms. If an animal has a reference frame for its world, then as it explores it can note what it found at each location. When the animal wants to get someplace, such as a shelter, it can use the reference frame to figure out how to get there from its current location. In the brain, cortical columns create a reference frame for every observed object, which can be seen as a way to organize any knowledge. Establishing a reference frame for the world they inhabit is highly useful for survival. WorldBrain is based on this concept and aims to mimic the human brain by establishing reference frames.

5.7.2 WorldBrain mimics the maps in the old brain

In 2005, scientists in the lab of May-Britt Moser and Edvard Moser used a similar experimental setup, again with rats. In their experiments, they recorded signals from neurons in the entorhinal cortex, adjacent to the hippocampus. They discovered what are now called grid cells, which fire at multiple locations in an environment. The locations where a grid cell becomes active form a grid pattern.

The details of how place cells and grid cells work are complicated. Grid cells are like the rows and columns of a paper map, but overlaid on the animal's environment. They allow the animal to know where it is, to predict where it will be when it moves, and to plan movements. Place cells are like the details printed in the square. Place cells tell the rat where it is based on sensory input, but place cells alone aren't useful for planning movements—that requires grid cells. The two types of cells work together to create a complete model of the rat's environment.

Every time a rat enters an environment, the grid cells establish a reference frame. If it is a novel environment, the grid cells create a new reference frame. If it is a novel environment, the grid cells create a new reference frame. Humans have grid cells and place cells too. These cells create models of the places we have been. WorldBrain seeks to replicate this aspect of the human brain by establishing systems of grid cells and place cells. In the initial stages of the project, the modeling area does not need to be extensive, similar to how even someone who has never left their hometown can demonstrate full intelligence.

5.7.3 WorldBrain's Simulation of the New Cortex's Map Model

In the neocortex, the way place cells and grid cells learn object models is similar to how place

cells and grid cells in the old brain learn environmental models. The mapping mechanisms in the neocortex are not an exact copy of ones in the old brain. Evidence suggests that the neocortex uses the same basic neural mechanisms, but it is different in several ways. It is as if nature stripped down the hippocampus and entorhinal cortex to a minimal form, made tens of thousands of copies, and arranged them side by side in cortical columns. That became the neocortex.

Grid cells and place cells in the old brain mostly track the location of one thing: the body. They know where the body is in its current environment. The neocortex has about 150,000 copies of this circuit, one per cortical column. Therefore, the neocortex tracks thousands of locations simultaneously.

A cortical column has multiple layers of neurons. Several of these layers are needed to create the map squares. Figure 2-5 gives you a flavor of what we think is happening in a cortical column.

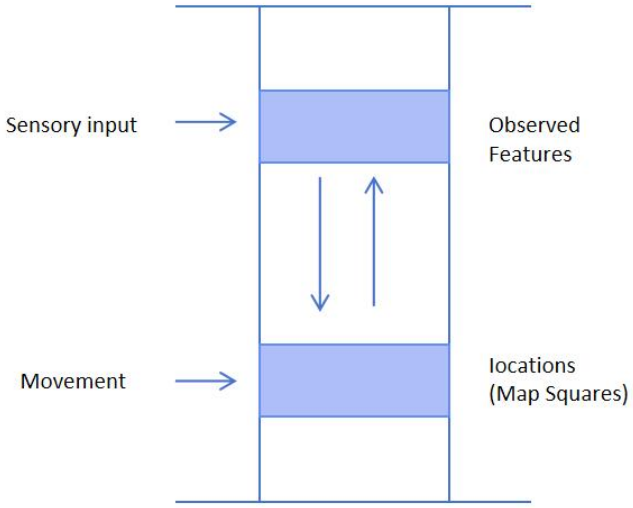


Figure 2-5 A model of a cortical column

Figure 2-5 represents two layers of neurons (the shaded boxes) in one cortical column. Although a column is tiny, about one millimeter wide, each of these layers might have ten thousand neurons.

The upper layer receives the sensory input to the column. When an input arrives, it causes several hundred neurons to become active. In the paper-map analogy, the upper layer represents what you observe at some location, such as the fountain.

The bottom layer represents the current location in a reference frame. In the analogy, the lower layer represents a location—such as Town 3, D2—but doesn't represent what is observed there. It is like a blank square, labeled only with Town 3, location D2.

The two vertical arrows represent connections between the blank map squares (the lower layer) and what is seen at that location (the upper layer). The downward arrow is how an observed feature, such as the fountain, is associated with a particular location in a particular town. The upward arrow associates a particular location—Town 3, D2—with an observed feature. The upper layer is roughly equivalent to place cells and the lower layer is roughly equivalent to grid cells.

The basic flow of information goes as follows: A sensory input arrives and is represented by the neurons in the upper layer. This invokes the location in the lower layer that is associated with the input. When movement occurs, such as moving a finger, then the lower layer changes to the expected new location, which causes a prediction of the next input in the upper layer.

We simulated this two-layer circuit in software using realistic assumptions for the number of neurons in each layer. Our simulations showed that not only can individual cortical columns learn models of objects, but each column can learn hundreds of them.

5.7.4 WorldBrain simulates the orientation of the neocortex

There are other things a cortical column must do to learn models of objects. There are neurons in the old brain called head direction cells. As their name suggests, these cells represent the direction an animal's head is facing. Head direction cells act like a compass, but they aren't tied to magnetic north. They are aligned to a room or environment. If you stand in a familiar room and then close your eyes, you retain a sense of which way you are facing.

Cortical columns must have cells that perform an equivalent function to head direction cells. We refer to them by the more generic term orientation cells. Every cortical column learns models of objects. The columns do this using the same basic method that the old brain uses to learn models of environments. Each cortical column has a set of cells equivalent to grid cells, another set equivalent to place cells, and another set equivalent to head direction cells, all of which were first discovered in parts of the old brain.

Existing location-aware chips can effectively mimic all these functionalities. Specific sensors include distance sensors, gravity sensors, acceleration sensors, magnetic field sensors, Hall sensors, gyroscopes, and GPS. These are all miniaturized chips and essentially form the basic components of every smartphone.

5.8 The Storage Method of Knowledge in the WorldBrain Model

Knowledge in the brain is distributed storage. All knowledge is not confined to a single location, such as a single cell or cortical column, nor is it stored like a hologram where everything is stored at any one place. Knowledge about an object is distributed across thousands of cortical columns, and neurons never rely on individual synapses. This concept is a fundamental aspect of WorldBrain development. In the WorldBrain simulation network, a decentralized storage approach is employed, and even if 30% of neurons are lost, the impact on network functionality is usually minimal.

5.9 The Voting Mechanism in the WorldBrain Model

Perception is the consensus the columns reach by voting. Most of the connections in a cortical column go up and down between the layers, largely staying within the bounds of the column. There are a few well-known exceptions to this rule. Cells in some layers send axons long distances within the neocortex. They might send their axons from one side of the brain to the other, for example, between the areas representing the left and right hands. Or they might send their axons from V1, the primary visual region, to A1, the primary auditory region. These cells with long-distance connections are voting.

Most of the cells in a column don’t represent the kind of information that columns could vote on. For example, the sensory input to one column differs from the sensory input to other columns, and therefore cells that receive these inputs do not project to other columns. But cells that represent what object is being sensed can vote and will project broadly.

The basic idea of how columns can vote is not complicated. Using its long-range connections, a column broadcasts what it thinks it is observing. Often a column will be uncertain, in which case its neurons will send multiple possibilities at the same time. Simultaneously, the column receives projections from other columns representing their guesses. The most common guesses suppress the least common ones until the entire network settles on one answer. Surprisingly, a column doesn’t need to send its vote to every other column. The voting mechanism works well even if the long-range axons connect to a small, randomly chosen subset of other columns.

VI.Designing and Training the World Model

6.1 Designing and Training the World Model

Training the World Model is a prototypical example of Self-Supervised Learning (SSL), whose basic idea is pattern completion. The prediction of future inputs (or temporarily unobserved inputs) is a special case of pattern completion. In this work, the primary purpose of the World Model is seen as predicting future representations of the state of the world.

There are three main issues to address. First, quite evidently, the quality of the World Model will greatly depend on the diversity of state sequences, or triplets of (state, action, resulting state) it is able to observe while training. Second, because the world is not entirely predictable, there may be multiple plausible world state representations that follow a given world state representation and an action from the agent. The World Model must be able to meaningfully represent this possibly-infinite collection of plausible predictions. Third, the World Model must be able to make predictions at different time scales and different levels of abstraction.

The first issue touches on one of the main questions surrounding learning for sequential decision processes: the diversity of the “training set” depends on the actions taken.

The second issue is even more dire: the world is not entirely predictable. Hence, the World Model should be able to represent multiple plausible outcomes from a given state and (optionally) an action. This may constitute one of the most difficult challenges to which the present proposal brings a solution.

The third issue relates to the problem of long-term prediction and planning. Humans plan complex goals at an abstract level and use high-level descriptions of the world states and actions to make predictions. High-level goals are then decomposed into sequences of more elementary sequences of subgoals, using shorter-term prediction from the World Model to produce lower-level actions. This decomposition process is repeated all the way down to millisecond-by-millisecond muscle control, informed by local conditions.

6.2 Self-Supervised Learning

Self-Supervised Learning (SSL) is a paradigm in which a learning system is trained to capture the mutual dependencies between its inputs. Concretely, this often comes down to training a system to tell us if various parts of its input are consistent with each other.

For example, in a video prediction scenario, the system is given two video clips, and must tell us

to what degree the second video clip is a plausible continuation of the first one. In a pattern completion scenario, the system is given part of an input (image, text, audio signal) together with a proposal for the rest of the input, and tells us whether the proposal is a plausible completion of the first part. In the following, we will denote the observed part of the input by *x* and the possibly-unobserved part by *y*.

Importantly, we do not impose that the model be able to predict y from x. The reason is that there may be an infinite number of y that are compatible with a given x. In a video prediction setting, there is an infinite number of video clips that are plausible continuations of a given clip. It may be difficult, or intractable, to explicitly represent the set of plausible predictions. But it seems less inconvenient to merely ask the system to tell us if a proposed y is compatible with a given x.

A general formulation can be done with the framework of Energy-Based Models (EBM). The system is a scalar-valued function F(x,y) that produces low energy values when x and y are compatible and higher values when they are not. The concept is depicted in Figure 8. Data points are black dots. The energy function produces low energy values around the data points, and higher energies away from the regions of high data density, as symbolized by the contour lines of the energy landscape. The EBM implicit function formulation enables the system to represent multi-modal dependencies in which multiple values of y are compatible with a given x. The set of y compatible with a given x may be a single point, multiple discrete points, a manifold, or a collection of points and manifolds.

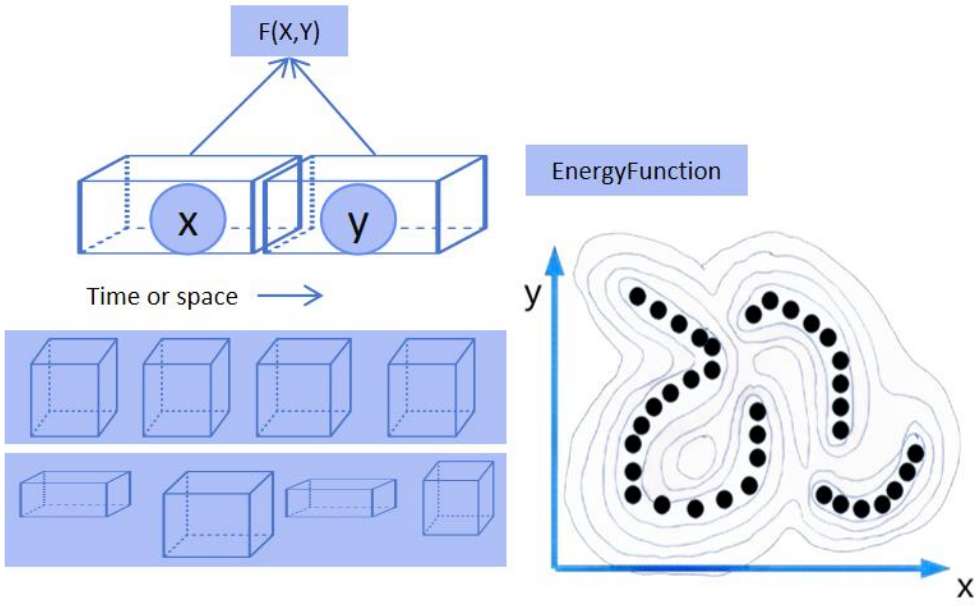


Figure 2-6: Self-Supervised Learning (SSL) and Energy-Based Models (EBM)

SSL is a learning paradigm in which a learning system is trained to “fill in the blanks”, or more precisely to capture the dependencies between observed parts of the input and possibly unobserved parts of the input. Part of the input signal is observed and denoted x (in pink), and part of the input signal is either observed or unobserved and denoted y (in blue). In a temporal prediction scenario, x represents past and present observations, and y represent future observations. In a general pattern completion scenario, various parts of the input may be observed or unobserved at various times. The learning system is trained to capture the dependencies between x and y through a scalar-valued energy function F(x,y) that takes low values when x and y are consistent or compatible, and higher values if x and y are inconsistent or incompatible. In a video prediction scenario, the system would produce a low energy value if a video clip y is a plausible continuation of the video clip x. This energy-based model (EBM) formulation enables the system to represent multi-modal dependencies in which multiple values of y (perhaps an infinite set) may be compatible with a given x. In the right panel, an energy landscape is represented in which dark discs represent data points, and closed lines represents contours (level sets) of the energy function.

To enable Mode-2 planning, a predictive World Model should be trained to capture the dependencies between past and future percepts. It should be able to predict representations of the future from representations of the past and present. The general learning principle is as follows: given two inputs x and y, learn two functions that compute representations $s_x = g_x(x)$ and $s_y = g_y(y)$ such that (1) s_x and s_y are maximally informative about x and y and (2) s_y can easily be predicted from s_x. This principle ensures a trade-off between making the evolution of the world predictable in the representation space and capturing as much information as possible about the world state in the representation.

What concepts could such an SSL system learn by being trained on video? Our hypothesis is that a hierarchy of abstract concepts about how the world works could be acquired.

Learning a representation of a small image region such that it is predictable from neighboring regions surrounding it in space and time would cause the system to extract local edges and contours in images, and to detect moving contours in videos. Learning a representation of images such that the representation of a scene from one viewpoint is predictable from the representation of the same scene from a slightly different viewpoint would cause the system to implicitly represent a depth map. A depth map is the simplest way to explain how a view of a scene changes when the camera moves slightly. Once the notion of depth has been learned, it would become simple for the system to identify occlusion edges, as well as the collective motion of regions belonging to a rigid object. An implicit representation of 3D objects may spontaneously emerge. Once the notion of object emerges in the representation, concepts like object

permanence may become easy to learn: objects that disappear behind others due to parallax motion will invariably reappear. The distinction between inanimate and animate object are as follow: inanimate objects are those whose trajectories are easily predictable. Intuitive physics concepts such as stability, gravity, momentum, may follow by training the system to perform longer-term predictions at the object representation level. One may imagine that through predictions at increasingly abstract levels of representation and increasingly long-time scales, more and more complex concepts about how the world works may be acquired in a hierarchical fashion.

6.3 Joint Embedding Predictive Architecture (JEPA)

The centerpiece of this paper is the Joint Embedding Predictive Architecture (JEPA). JEPA is not generative in the sense that it cannot easily be used to predict y from x. It merely capture the dependencies between x and y without explicitly generating predictions of y.

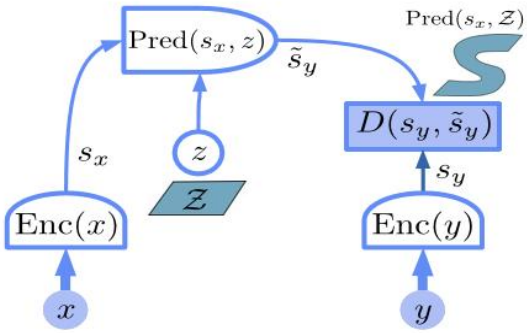


Figure 2-7 The Joint-Embedding Predictive Architecture (JEPA) consists of two encoding branches.

The first branch computes s_x, a representation of x and the second branch s_y a representation of y. The encoders do not need to be identical. A predictor module predicts s_y from s_x with the possible help of a latent variable z. The energy is the prediction error. Simple variations of the JEPA may use no predictor, forcing the two representations to be equal, or may use a fixed predictor with no latent, or may use simple latents such as discrete variables. The main advantage of JEPA is that it performs predictions in representation space, eschewing the need to predict every detail of y, and enabling the elimination of irrelevant details by the encoders. More precisely, the main advantage of this architecture for representing multi-modal dependencies is twofold: (1) the encoder function s_y = Enc(y) may possess invariance properties that will make it produce the

same s_y for a set of different y. This makes the energy constant over this set and allows the model to capture complex multi-modal dependencies; (2) The latent variable z, when varied over a set Z, can produce a set of plausible predictions $Pred(s_x,Z) = \{\tilde{s}_y = Pred(s_x,z)\ \forall z \in Z\}$ If x is a video clip of a car approaching a fork in the road, s_x and s_y may represent the position, orientation, velocity and other characteristics of the car before and after the fork, respectively, ignoring irrelevant details such as the trees bordering the road or the texture of the sidewalk. z may represent whether the car takes the left branch or the right branch of the road.

A generic JEPA is shown in Figure2-7. The two variables x and y are fed to two encoders producing two presentations s_x and s_y. These two encoders may be different. They are *not* required to possess the same architecture nor are they required to share their parameters. This allows x and y to be different in nature (e.g. video and audio). A predictor module predicts the representation of y from the representation of x. The predictor may depend on a latent variable z. The energy is simply the prediction error in representation space:

$$E_w(\mathrm{x},\mathrm{y},\mathrm{z}) = \mathrm{D}\Big(s_y\ ,\ Pred(s_x,z)\Big)$$

The overall energy is obtained by minimizing over z:

$$\hat{z} = argmin_{z\in Z} E_w(x,y,z) = argmin_{z\in Z} D\big(s_y, Pred(s_x,z)\big)$$

$$F_w(x,y) = min_{z\in Z} E_w(x,y,z) = D\big(s_y, pred(s_x,\hat{z})\big)$$

The main advantage of JEPA is that it performs predictions in representation space, eschewing the need to predict every detail of y. This is enabled by the fact that the encoder of y may choose to produce an abstract representation from which irrelevant details have been eliminated. But there are two ways a JEPA may represent the multiplicity of values of y compatible with x. The first one is invariance properties of the y encoder, the second one is the latent variable z, as explained below.

Multi-modality through encoder invariance: The encoder function $s_y = Enc(y)$ may have invariance properties. If all the y's in a set map to the same value of s_y, all those y's will have identical energies. With JEPA, we lose the ability to generate outputs, but we gain a powerful way to represent multi-modal dependencies between inputs and outputs.

Multi-modality through latent variable predictor: The predictor may use a latent variable z to capture the information necessary to predict s_y that is not present in s_x. When z is varied over a set Z, the predictor produces a set of plausible predictions $Pred(s_x,Z) = \{\tilde{s}_y = Pred(s_x,z)\ \forall z \in Z\}$. For example, if x is a video clip of a car approaching a fork in the road, s_x and s_y may represent the

past and future positions, orientations, velocities and other characteristics of the car, ignoring irrelevant details such as the trees bordering the road or the texture of the sidewalk. The latent z may be a binary variable indicating whether the car takes the left branch ($z = 0$) or the right branch ($z = 1$ if the road. If the car takes the left branch, the value $z = 0$ will produce a lower energy $D(s_y, \tilde{s}_y)$ than $z = 1$.

6.4 Hierarchical JEPA (H-JEPA)

Trained non-contrastively may constitute our best tool for learning World Models that are able to learn relevant abstractions. When trained with VICReg and similar criteria, a JEPA can choose to train its encoders to eliminate irrelevant details of the inputs so as to make the representations more predictable. In other words, a JEPA will learn abstract representations that make the world predictable. Unpredictable details will be eliminated by the invariance properties of the encoder, or will be pushed into the predictor's latent variable. The amount of information thereby ignored will be minimized by the training criteria and by the latent variable regularizer.

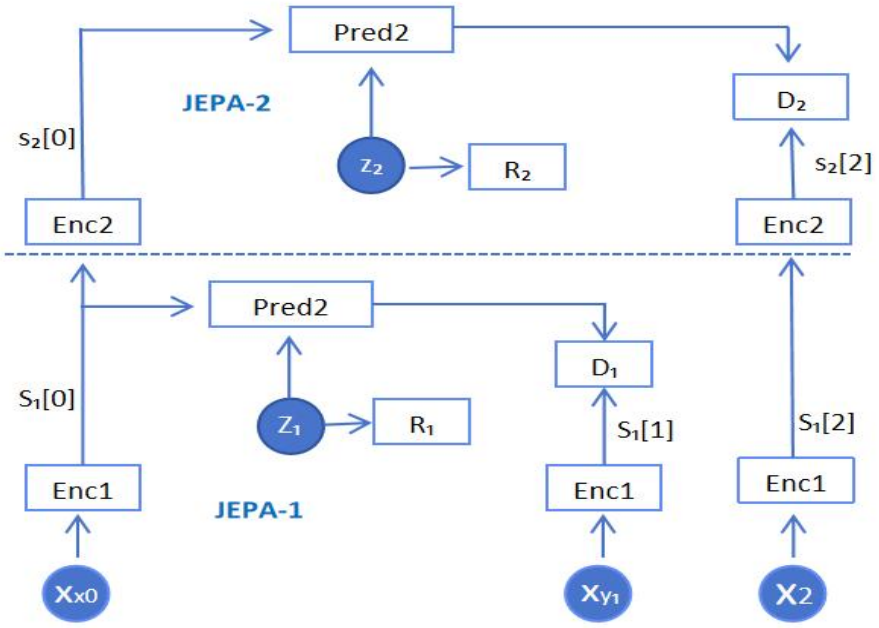


Figure 2-8 H-JEPA (Hierarchical JEPA)

The ability of the JEPA to learn abstract representations in which accurate prediction can be performed allows hierarchical stacking. In this diagram JEPA-1 extracts low-level representations and performs short-term predictions. JEPA-2 takes the representations extracted by JEPA-1 as inputs and extracts higher-level representations with which longer-term predictions can be performed. More abstract representations ignore details of the inputs that are difficult to predict in

the long term, enabling them to perform longer-term predictions with coarser descriptions of the world state.

It is important to note that generative latent-variable models are not capable of eliminating irrelevant details, other than by pushing them into a latent variable. This is because they do not produce abstract (and invariant) representations of y. This is why we advocate against the use of generative architectures.

The capacity of JEPA to learn abstractions suggests an extension of the architecture to handle prediction at multiple time scales and multiple levels of abstraction. Intuitively, low level representations contain a lot of details about the input and can be used to predict in the short term. But it may be difficult to produce accurate long-term predictions with the same level of details. Conversely high-level, abstract representation may enable long-term predictions, but at the cost of eliminating a lot of details.

Let's take a concrete example. When driving a car, given a proposed sequence of actions on the steering wheel and pedals over the next several seconds, drivers can accurately predict the trajectory of their car over the same period. The details of the trajectory over longer periods are harder to predict because they may depend on other cars, traffic lights, pedestrians, and other external events that are somewhat unpredictable. But the driver can still make accurate predictions at a higher level of abstraction: ignoring the details of trajectories, other cars, traffic signals, etc, the car will probably arrive at its destination within a predictable time frame. The detailed trajectory will be absent from this level of description. But the approximate trajectory, as drawn on a map, is represented. A discrete latent variable may be used to represent multiple alternative routes.

Figure 2-8 shows a possible architecture for multilevel, multi-scale world state prediction. Variables x_0, x_1, x_2 represent a sequence of observations. The first-level network, denoted JEPA-1 performs short-term predictions using low-level representations. The second-level network JEPA-2 performs longer-term predictions using higher-level representations. One can envision architectures of this type with many levels, possibly using convolutional and other modules, and using temporal pooling between levels to coarse-grain the representation and perform longer-term predictions. Training can be performed level-wise or globally, using any non-contrastive method for JEPA.

I submit that the ability to represent sequences of world states at several levels of abstraction is essential to intelligent behavior. With multi-level representations of world states and actions, a complex task can be decomposed into successively more detailed subtasks, instantiated into

actions sequences when informed by local conditions. For example, planning a complex task, like commuting to work, can be decomposed into driving to the train station, catching a train, etc. Driving to the train station can be decomposed into walking out of the house, starting the car, and driving. Getting out of the house requires standing up, walking to the door, opening the door, etc. This decomposition descends all the way down to millisecond-by-millisecond muscle controls, which can only be instantiated when the relevant environmental conditions are perceived (obstacles, traffic lights, moving objects, etc).

6.5 Hierarchical Planning

If our World Model can perform predictions hierarchically, can it be used to perform Mode-2 reasoning and planning hierarchically? Hierarchical planning is a difficult topic with few solutions, most of which require that the intermediate vocabulary of actions be predefined. But if one abides by the deep learning philosophy, those intermediate representations of action plans should also be learned.

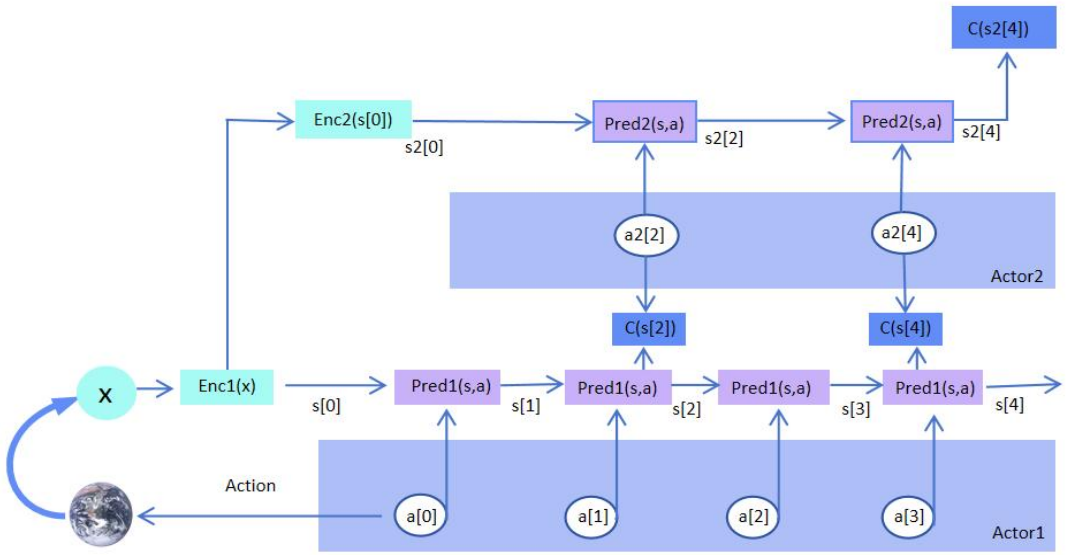


Figure 2-9: Hierarchical JEPA for Mode-2 hierarchical planning

A complex task is defined by a high-level cost computed from a high-level world-state representation C(s2[4]). A sequence of high-level abstract actions (a2[2],a2[4]) is inferred that minimizes C(s2[4]). The inferred abstract actions are fed to lower-level cost modules C(s[2]),C(s[4]) which define subgoals for the lower layer. The lower layer then infers an action sequence that minimizes the subgoal costs. Although only a 2-layer hierarchy is shown here, it is straightforward to extend the concept to multiple levels. The process described here is sequential top-down, but a better approach would be to perform a joint optimization of the actions in all the

layers.

Figure 2-9 shows a possible architecture for hierarchical Mode-2 planning that can exploit the hierarchical nature of a multi-scale World Model. A percept is encoded into representations at multiple levels of abstractions by a cascade of encoders:

$$s1[0] = Enc1(x); s2[0] = Enc2(s[0]); \ldots$$

Prediction takes place at all levels. Higher levels perform longer-term prediction, while lower levels perform shorter-term predictions. The overall task is defined by a high-level objective, depicted as C(s2[4]) in the diagram. The top level infers a sequence of high-level actions ($a2[2], a2[4]$) to optimize this objective. These high-level "actions" are not real actions but targets for the lower level predicted states. One can think of them as conditions that the lower-level state must satisfy in order for the high-level predictions to be accurate. Whether these conditions are satisfied can be computed by cost modules $C(s[2])$ and $C(s[4])$. They take a lower-level state $s[2]$ and a high-level condition $a2[2]$ and measure to what extent the state satisfies the condition. With these subgoals defined, the lower level can perform inference and find a low-level action sequence that minimizes the mid-level subgoals $C(s[2])$ and $C(s[4])$.

The process just described is top down and greedy. But one may advantageously iterate the optimization so that high level and low-level action sequences are optimized jointly. The cost modules may be configured by the configurator for the situation at hand.

The idea that an action is merely a condition to be satisfied by the level below is actually an old one in control theory. For a example, a classical proportional servomechanism can be seen as being given a target state. A quadratic cost measures the squared distance between the target and the current state, and the control is simply proportional to the negative gradient of the cost with respect to the action variables.

6.6 Handling uncertainty

The real world is not entirely predictable. Uncertainty in predictions of future world states may be due to a number of reasons:

The world is intrinsically stochastic (aleatoric uncertainty, type 1)

The world is deterministic but chaotic, hence difficult to predict without infinitely precise perception (aleatoric uncertainty, type 2)

The world is deterministic but partially observable (aleatoric uncertainty type 3).

The world is fully observable, but the sensors only give partial information about the world state (epistemic uncertainty, type 1)

The representation of the world state extracted by the perception module does not contain the full information necessary for accurate prediction (epistemic uncertainty, type 2).

The World Model is inaccurate due to limitations of its representational power (bounded rationality or epistemic uncertainty, type 3).

The World Model is inaccurate due to it having been trained with limited amount of data (epistemic uncertainty, type 4).

It is often assumed from the start that models, critics and policies must represent distributions. In the present work, we push the possible stochasticity of a predicted variable into a latent variable, which may be optimized, predicted, or sampled. This is what is often referred to in the ML literature as “the reparameterization trick”. We do not need to use this trick here, since we view the latent-variable parameterization of the predictions as fundamental.

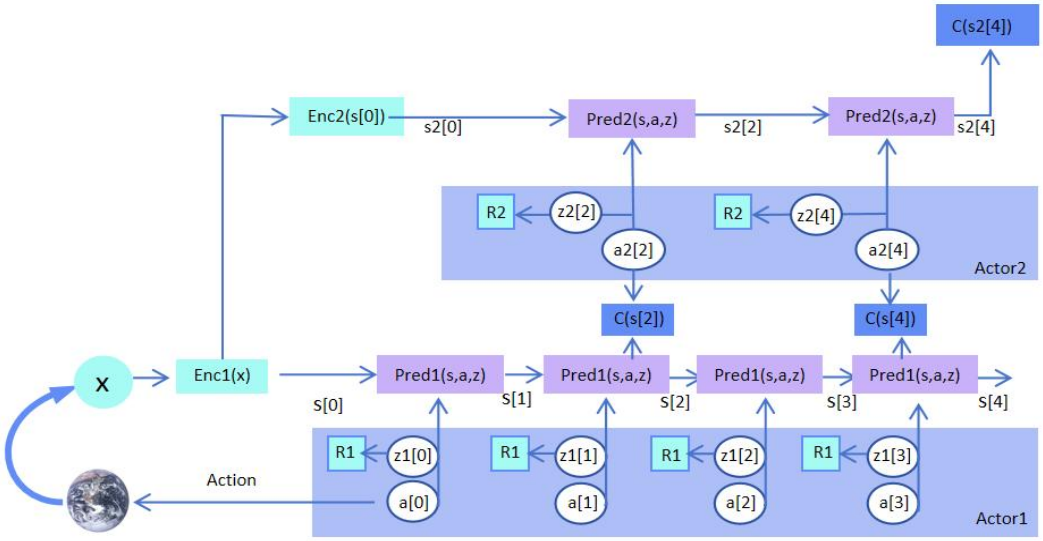


Figure 3-1: Hierarchical JEPA for Mode-2 hierarchical planning in an uncertain environment.

Realistic environments are not entirely predictable, even when using highly-abstract representations. Uncertainty about predictions can be handled by predictors with latent variables. The latent variables (red circles) contain information about the prediction that cannot be derived from the prior observation. The latent variables must be regularized to prevent an energy collapse and to force the system to predict as much as possible without the help of it. At planning time, latent variables are sampled from distributions obtained by applying a Gibbs distribution to the regularizers. Each sample leads to a different prediction. To produce consistent latent sequences,

the parameters of the regularizer can be functions of previous states and retrieved memories. As the prediction progresses, the number of generated state trajectories may grow exponentially. If each latent variable has k possible discrete values, the number of possible trajectories will grow as kt, where t is the number of time steps. Directed search and pruning strategies must be employed. With multiple predicted trajectories, optimal action sequences can be computed that minimize the average cost, or a combination of average and variance of the cost so as to minimize risk.

Figure 3-1 represents a hierarchical planning episode in the presence of uncertainty. A prediction at a given level and time step, e.g. s2[2] requires a sample of the corresponding latent variable z2[2]. The sample may come from the distribution whose negative logarithm is the regularizer R2(z2[2])). The parameters of the regularizer may be constant (e.g. fixed Gaussian), predicted from currently-available data using amortized inference (e.g. a multinomial or Gaussian whose parameters are computed from s2[0]) or produced by the configurator. Using previous predictions to configure the latent regularizer biases the system towards generating “good” trajectories.

As the prediction progresses, the number of generated state trajectories may grow exponentially: if each latent variable has k possible discrete values, the number of possible trajectories will grow as kt, where t is the number of time steps. Directed search and pruning strategies can be employed, as in classical Monte-Carlo Tree Search (MCTS). In the case of continuous latents, one may sample latents from the continuous distributions defined by the regularizer.

Given a sample of all the latents, the optimal action sequences at every levels can be inferred. However, the prediction process may need to be repeated for multiple drawings of the latents, so as to cover the set of plausible outcomes. The inference process may be used for multiple predictions to produce an action that does not just minimize the expected cost, but also minimizes the uncertainty on the expected cost.

6.7 World Model Architecture

The details of the architecture of the World Model should depend on the type of environment the agent evolves in. It is likely that the best module architectures in a JEPA should include some sort of gating or dynamic routing mechanism. For example, the best way to handle low-level, short-term predictions in videos is by extracting simple local feature vectors and displacing those feature vectors from one frame to the next, depending on predicted motions. The latent variables may encode a map of displacements, which can modulate routing connections between one frame and the next.

For longer-term prediction at a higher level of abstraction, the relevant features are objects and their interactions. The evolution may be best modeled by a transformer architecture, which has the

property of being equivariant to permutation and is appropriate to capture interactions between discrete objects.

Ⅶ .ChatGPT's Achievements in the Field of Artificial Intelligence

7.1 ChatGPT has accomplished several functionalities in the field of Artificial Intelligence.

Natural Language Processing: ChatGPT has made significant advancements in natural language processing. It can comprehend and generate language, understanding and responding to user questions while generating relevant replies. Chat GPT is capable of handling various tasks such as dialogue systems, machine translation, text summarization, and sentiment analysis.

Question-Answering System: ChatGPT has made notable progress in question-answering systems. It can retrieve relevant information from a given knowledge base or text corpus based on user questions and generate accurate and useful answers. The question-answering system of Chat GPT finds widespread applications in information retrieval, common-sense reasoning, and semantic matching.

Personalized Interaction: ChatGPT can personalize interactions based on user preferences and personality. By analyzing user history, conversations, and feedback, Chat GPT can adjust its response style and language expression to better meet user needs and expectations. This personalized interaction enhances user experience and makes conversations more natural and engaging.

Multi-Language Support: ChatGPT supports multiple languages, capable of handling translation and comprehension between different languages. It automatically detects and adapts to user language preferences to provide appropriate interactions and services, facilitating cross-language communication and international applications.

Real-Time Dialogue: ChatGPT can engage in real-time conversations, generating replies and interacting with users instantly. This makes it particularly valuable in scenarios such as chatbots, online customer service, and virtual assistants, enabling immediate responses and smooth, efficient dialogue experiences.

7.2 ChatGPT's Limitations

Lack of Common-Sense Reasoning: Due to its reliance on statistical learning and language modeling, ChatGPT may lack the ability for common-sense reasoning. When dealing with complex problems or requiring deep logical reasoning in conversations, it may provide inaccurate or unreasonable answers.

Lack of Planning Capability: ChatGPT's autoregressive model prevents it from simulating different actions and outcomes to evaluate the potential impact of each action, thus unable to provide optimal strategies.

Sensitivity to Errors and Biases: ChatGPT's response generation may be influenced by input data limitations. If the input data contains errors, biases, or inaccurate information, ChatGPT may produce misleading responses.

Instruction Sensitivity: ChatGPT is sensitive to the accuracy and clarity of instructions. Ambiguous or imprecise instructions may lead to inappropriate responses, and in some cases, it may require more specific and clear guidance to generate desired responses.

Long-Term Coherence: Due to its context-based response generation, ChatGPT may encounter difficulties in maintaining long-term coherence. During extended conversations, it may lead to information repetition, incoherent responses, or logical contradictions.

High Cost: Reliance on high-end hardware and expensive training data makes ChatGPT's training process costly and energy-intensive.

7.3 WorldBrain's Advantages

Environmental Understanding: The world model can effectively model the environment, including objects, scenes, rules, and interactions within it. By creating a model of the environment, the system gains a better understanding of its structure and features, enabling more accurate decision-making and actions.

Prediction Capability: The world model can make predictions and infer future states and events based on known information and patterns. This predictive ability allows the system to make informed decisions and plans in advance, adapting to environmental changes and achieving desired goals.

Planning and Decision-Making: Building on the modeled environment and predictions, the world model assists the system in planning and decision-making. The system can simulate different actions and their potential outcomes, evaluating each action's impact, and selecting the optimal action strategy.

Simulation and Experimentation: The world model can be used for simulation and experimentation, evaluating the impact of different strategies and actions on the environment. Multiple attempts and tests can be conducted in a simulated environment, observing the results of different decisions, facilitating learning and improvement. This simulation and experimentation capability provides a safe, efficient, and controllable way to study and optimize the system's behavior.

Generalization and Transfer Learning: The world model helps the system with generalization and transfer learning. By modeling and understanding the environment, the system can extract general patterns and knowledge, applying them to new environments and tasks. This capability enables the system to adapt quickly to new contexts and challenges, enhancing its applicability.

Enhanced Exploration: The world model can generate simulated environments, enabling the system to explore and experiment an infinite number of times. By conducting explorations in the simulated environment, the system gains more knowledge and experience, improving its decision-making and action capabilities.

Explainability and Visualization: The world model provides visualizations and explanations of the environment and internal states of the system, aiding human understanding and analysis of the system's behavior and decision-making process. This explainability enhances transparency, facilitating effective interaction and collaboration between humans and the system.

Diversity Generation: The world model can generate diverse simulated environments and scenarios, allowing the system to produce varied actions and decisions. This diversity generation capability helps the system explore and try different strategies and methods, leading to the discovery of better solutions.

Robustness and Transferability: Through modeling and understanding the environment, the world model captures uncertainties and changes, making the system robust and capable of transferring knowledge to adapt and respond quickly to new environments and unknown situations.

Extremely Low Single-Point Computational Cost: The human brain consumes energy equivalent to just a light bulb, which is a noteworthy characteristic. Currently, the power consumption of a single APP-based neuron is only about 1/20 of the total power consumption of a smartphone. This

exceptionally low value does not require high-end chips and massive storage, making it widely adaptable.

Ⅷ. WorldBrain Development Plan

8.1 WorldBrain's Development Directions

Enhancing General Intelligence: Current artificial intelligence systems have achieved significant success in specific tasks, but they still face challenges in handling diverse tasks, cross-domain learning, and transfer learning. The future WorldBrain will focus on improving its general intelligence, aiming to demonstrate human-level intelligence in a wide range of tasks and domains.

Reinforcement Learning and Autonomous Learning: WorldBrain will further develop its capabilities in reinforcement learning and autonomous learning, enabling it to continually improve its performance through interactions with the environment and feedback. This will allow WorldBrain to learn and explore in unknown environments, acquiring new knowledge and experience from it.

Integration of Perception and Action Abilities: WorldBrain will further integrate its perception and action abilities, enabling it to perceive and understand the state of the environment and interact with it through actions. This will equip WorldBrain with more comprehensive and flexible intelligent behaviors, better adapting to complex real-world tasks.

Strengthening Creativity and Innovation: The future WorldBrain will not be limited to problem-solving and task execution; it will possess stronger creativity and innovation capabilities. It will be able to generate novel ideas, propose new solutions, and demonstrate creative performances in fields such as art, science, and design.

Improving Explainability and Transparency: The explainability of WorldBrain is a crucial research direction. The future WorldBrain will strive to enhance the explainability of its decisions and actions, enabling it to explain the reasons and basis for its behavior to users and regulatory authorities, thus increasing user trust and acceptance of WorldBrain.

8.2 WorldBrain's Social Value

Data Sharing and Privacy Protection: Ensuring proper management and protection of personal

data in the application of WorldBrain. Establishing a framework for data sharing that allows individuals to choose to share their data autonomously and obtain transparent and trustworthy mechanisms for data use and protection.

Human-Machine Collaboration and Work Reshaping: Emphasizing human-machine collaboration and work reshaping in the application of WorldBrain, positioning it as a tool and aid to humans rather than a substitute. Ensuring that the development of WorldBrain provides more employment opportunities while focusing on human career transformation and training to adapt to new work environments and skill requirements.

Knowledge Sharing and Intellectual Contribution: Encouraging knowledge sharing and intellectual contribution to involve more people in the development of WorldBrain. Establishing open research and collaboration mechanisms to promote exchanges and cooperation among academia, industry, and society, collectively driving progress and innovation in WorldBrain.

Sustainable Development and Social Responsibility: Incorporating sustainable development and social responsibility into the development strategy of WorldBrain. Paying attention to environmental protection and resource utilization efficiency to avoid negative impacts on the environment. Additionally, addressing social issues and promoting social welfare, utilizing the power of WorldBrain to foster social justice and sustainable development.

Regulatory and Governance Mechanisms: Establishing effective regulatory and governance mechanisms to ensure that the development and application of WorldBrain comply with laws, regulations, and ethical principles. Regulatory agencies should strengthen supervision and oversight of WorldBrain, formulate relevant policies and guidelines, and collaborate with industries, academia, and all stakeholders to form a consensus and cooperative governance model.

Technical Collaboration and Knowledge Exchange: Promoting technical collaboration and knowledge exchange between different countries and organizations to jointly advance the development of WorldBrain. Establishing international cooperation mechanisms and platforms to share best practices, experiences, and technological achievements, enhancing global collaboration and achieving technological sharing and win-win outcomes.

Innovation Investment and Incubation: Increasing innovative investment and incubation support for WorldBrain. Governments, enterprises, and investment institutions can establish special funds to support innovative projects and startups in the WorldBrain field. Providing funding, resources, and guidance to help promising WorldBrain innovation teams and companies grow rapidly. The project itself will also establish the WorldBrain Foundation at an appropriate time to develop the WorldBrain user ecosystem.

Education and Training: Strengthening WorldBrain-related education and training. Cultivating more professionals with artificial intelligence and WorldBrain technology, including offering relevant degree programs and training projects to promote the popularization and application of WorldBrain technology.

Cross-Domain Application and Integration: Encouraging WorldBrain's application and integration in different fields to achieve cross-domain innovation and value-added. Combining WorldBrain technology with other cutting-edge technologies such as blockchain, IoT, and big data to create new business models and application scenarios, promoting economic cross-sector integration and innovation.

Social Engagement and Public Participation: Valuing public engagement and participation in WorldBrain and decision-making processes. Establishing open social engagement mechanisms to listen to public opinions and suggestions, enhancing public awareness and understanding of WorldBrain technology, and forming common development consensus.

Incentive Mechanisms and Reward Systems: Establishing incentive mechanisms and reward systems to encourage innovation and application of WorldBrain. Setting up awards and competitions to recognize individuals and teams who have made outstanding achievements in the WorldBrain field, stimulating innovation vitality and competitiveness.

Ethical and Moral Considerations: As WorldBrain develops, ethical and moral considerations become increasingly important. The future WorldBrain will pay more attention to ethical principles.

8.3 WorldBrain User Ecosystem and Economic Model Design

The universal artificial intelligence created by WorldBrain is like a super-intelligent brain. The WorldBrain ecosystem consists of Intelligent Interaction Contributors and Intelligent Users.

8.3.1 WorldBrain Intelligent Interaction Contributors

WorldBrain APP Users: Each APP represents a neuron, and users interact with the WorldBrain mainnet (WorldBrain itself is a distributed computing system, and the mainnet is the core of WorldBrain, where all participants collectively form the WorldBrain superintelligence) for actions such as data labeling. All these interactions contribute to WorldBrain, and the mainnet evaluates the contributions, rewarding the APP users with Tokens as incentives.

WorldBrain Computer Client Users: Each computer client represents a group of neurons. Clients interact with the WorldBrain mainnet for actions such as data labeling. All these interactions contribute to WorldBrain, and the mainnet evaluates the contributions, rewarding the computer client users with Tokens as incentives.

WorldBrain API Users: Users of the WorldBrain API interface, who are both users and contributors to WorldBrain. They mainly include robots, machine tools, and billions of IoT devices, among others, for which WorldBrain serves as a brain. Based on the agreed-upon API interface protocol, the data upload and download by these intelligent terminals are considered interactions. The contribution value of the interaction data is evaluated by the mainnet, and corresponding Token rewards are given. For example, robots relying on WorldBrain's intelligence, with their actions like motion, vision, and speech interactions, can all contribute to WorldBrain. Additionally, their built-in computing chips and storage participate in WorldBrain's distributed computing.

8.3.2 WorldBrain Intelligent Users include:

WorldBrain APP Users: Every individual is both a creator and owner of WorldBrain. The WorldBrain APP provides free AI butler services to all users, serving as humanity's second brain.

WorldBrain API Interface Users: Manufacturers of robots, intelligent machine tools, smart IoT devices, and other enterprises in need of intelligent assistance are potential buyers of the WorldBrain API interface. The purchase of the API interface exclusively uses WorldBrain project Tokens as payment.

WorldBrain is the operating system of the super-intelligence era. The economic model design of WorldBrain is based on the user ecosystem of WorldBrain.

WBC is the ecological token of WorldBrain, with a total circulation of 10 billion coins (no additional issuance), serving as the cornerstone of the entire ecosystem's organic interaction and positive spiral operation. WorldBrain Intelligent Interaction Contributors earn tokens through interactions, completing labeling tasks, and other means. WorldBrain Intelligent Users obtain super-intelligence services provided by WorldBrain by paying with WBC, completing the circulation of the entire token. During the circulation process, we introduce the metabolic mechanism on the WBC consumption end. In all stages of using WBC to purchase WorldBrain's super-intelligence services, 10% of the received WBC will be burned directly, which is called metabolism.

8.4 WorldBrain Ecological Architecture

WorldBrain Cortical Region: The new cortex is divided into dozens of regions, each performing different functions. If a cortical area is connected to the eyes, it produces vision; if the same cortical area is connected to the ears, it produces hearing; if two different cortical areas are connected, it leads to higher-level thinking. Based on this concept, WorldBrain needs to divide global neuron nodes into dozens of regions based on brain coordinates.

WorldBrain Cortical Column: The new cortex is composed of 150,000 cortical columns, with each brain region containing approximately 5,000 cortical columns. A cortical column is a neural network community composed of multiple neurons. Compared to a single neuron, a neural network can produce intelligent gains, thereby enhancing WorldBrain's intelligence level.

WorldBrain Mini Cortical Column: One cortical column contains about 100 Mini Cortical Columns. Throughout the new cortex, cortical columns and mini cortical columns have the same function: executing a set of basic algorithms responsible for various aspects of perception and intelligence. In terms of intelligent gains, Mini Cortical Columns are weaker than cortical columns.

WorldBrain Neuron: Each Mini Cortical Column contains over 100 neurons spanning various layers. Each computing terminal is a neuron, which belongs to weak computing power.

IX. The journey and development plan of WorldBrain

Before October 2022: For many years, we have focused on theoretical research and technological development in neuroscience, neurology, and blockchain technology, awaiting the emergence of LLM.

November 2022 - September 2023: WorldBrain Neural Network construction period Beta version.

October 2023 - December 2023: WorldBrain Enlightenment Stage (ChatGPT Dependency Period). Based on the "World Model" and the data needs of the "Memory Module" for the basic "World Model," we construct the model using cognitive data accumulated by ChatGPT, and preliminary completion of the "Memory Module."

January 2024 - March 2024: WorldBrain Enlightenment Stage (ChatGPT Detachment Period). We seek comparison with other data sources to further correct model biases and complete the construction of the basic World Model.

April 2024 - June 2024: Official launch of WorldBrain Neural Network.

July 2024 - December 2024: WorldBrain Enlightenment Stage, and official sale of WorldBrain API interface.

January 2025 - December 2028: WorldBrain Maturity Stage, multi-link public chainification of WorldBrain, and comprehensive realization of WorldBrain superintelligence.

January 2029 - December 3035: Humanity fully enters the era of artificial intelligence.

X. More

[1] A Path Towards Autonomous Machine Intelligence.Yann LeCun, 2022

[2] Lewis, Marcus, Scott Purdy, Subutai Ahmad, and Jeff Hawkins.“Locations in the Neocortex: A Theory of Sensorimotor Object Recognition Using Cortical Grid Cells.” Frontiers in Neural Circuits 13 (April 2019): 22.

[3] Why Neurons Have Thousands of Synapses, a Theory of Sequence Memory in the Neocortex. Jeff Hawkin, Cofounder of Numenta, Founder of the Redwood Neuroscience Institute, a member of the National Academy of Engineering, 2016.

[4] Mountcastle, Vernon. “An Organizing Principle for Cerebral Function: The Unit Model and the Distributed System.” In The Mindful Brain, edited by Gerald M. Edelman and Vernon B. Mountcastle, 7–50. Cambridge, MA: MIT Press, 1978.

[5] Mountcastle, Vernon. “The Columnar Organization of the Neocortex.” Brain 120 (1997): 701–722.

[6] Buxhoeveden, Daniel P., and Manuel F • Casanova. “The Minicolumn Hypothesis in Neuroscience.” Brain 125, no. 5 (May 2002): 935–951.

[7] Thomson, Alex M., and Christophe Lamy. “Functional Maps of Neocortical Local Circuitry.” Frontiers in Neuroscience 1 (October 2007): 19–42.

[8] Felleman, Daniel J., and David C. Van Essen. “Distributed Hierarchical Processing in the Primate Cerebral Cortex.” Cerebral Cortex 1, no. 1(January–February 1991): 1.

[9] Hilgetag, Claus C., and Alexandros Goulas. “‘Hierarchy’ in the Organization of Brain Networks.” Philosophical Transactions of the Royal Society B: Biological Sciences 375, no. 1796 (April 2020).

[10] Sherman, S. Murray, and R. W. Guillery. “Distinct Functions for Direct and Transthalamic Corticocortical Connections.” Journal of Neurophysiology 106, no. 3 (September 2011): 1068–1077.

[11] Ungerleider, Leslie G., and James V. Haxby. "'What' and 'Where' in the Human Brain." Current Opinion in Neurobiology 4 (1994): 157–165.

[12] Goodale, Melvyn A., and David Milner. "Two Visual Pathways—Where Have They Taken Us and Where Will They Lead in Future?" Cortex 98(January 2018): 283–292.

[13] Rauschecker, Josef P. "Where, When, and How: Are They All Sensorimotor? Towards a Unified View of the Dorsal Pathway in Vision and Audition." Cortex 98 (January 2018): 262–268.

[14] London, Michael, and Michael Häusser. "Dendritic Computation."Annual Review of Neuroscience 28, no. 1 (July 2005): 503–532.

[15] Antic, Srdjan D., Wen-Liang Zhou, Anna R. Moore, Shaina M. Short, and Katerina D. Ikonomu. "The Decade of the Dendritic NMDA Spike."Journal of Neuroscience Research 88 (November 2010): 2991–3001.

[16] Major, Guy, Matthew E. Larkum, and Jackie Schiller. "Active Properties of Neocortical Pyramidal Neuron Dendrites." Annual Review of Neuroscience 36 (July 2013): 1–24.

[17] O'Keefe, John. "Spatial Cells in the Hippocampal Formation." Nobel Lecture. Filmed December 7, 2014, at Aula Medica, Karolinska Institutet, Stockholm. Video, 45:17.

[18] Moser, Edvard I. "Grid Cells and the Enthorinal Map of Space." Nobel Lecture. Filmed December 7, 2014, at Aula Medica, Karolinska Institutet, Stockholm. Video, 49:23. www.nobelprize.org/prizes/medicine/2014/edvard-moser/lecture/.

[19] Moser, May-Britt. "Grid Cells, Place Cells and Memory." Nobel Lecture. Filmed December 7, 2014, at Aula Medica, Karolinska Institutet, Stockholm. Video, 49:48.

[20] Doeller, Christian F., Caswell Barry, and Neil Burgess. "Evidence for Grid Cells in a Human Memory Network." Nature 463, no. 7281 (February 2010): 657–661.

[21] Constantinescu, Alexandra O., Jill X. O'Reilly, and Timothy E. J.Behrens. "Organizing Conceptual Knowledge in Humans with a Gridlike Code." Science 352, no. 6292 (June 2016): 1464–1468.

[22] Jacobs, Joshua, Christoph T. Weidemann, Jonathan F. Miller, Alec Solway, John F. Burke, Xue-Xin Wei, Nanthia Suthana, Michael R.Sperling, Ashwini D. Sharan, Itzhak Fried, and Michael J. Kahana. "Direct Recordings of Grid-Like Neuronal Activity in Human Spatial Navigation."Nature Neuroscience 16, no. 9 (September 2013): 1188–1190.

[23] Hawkins, Jeff, Marcus Lewis, Mirko Klukas, Scott Purdy, and Subutai Ahmad. "A Framework for Intelligence and Cortical Function Based on Grid Cells in the Neocortex." Frontiers in Neural Circuits 12 (January 2019): 121.

[24] Hawkins, Jeff, and Subutai Ahmad. "Why Neurons Have Thousands of Synapses, a Theory of Sequence Memory in Neocortex." Frontiers in Neural Circuits 10, no. 23 (March 2016): 1–13.

[25] Hawkins, Jeff, Subutai Ahmad, and Yuwei Cui. "A Theory of How Columns in the Neocortex Enable Learning the Structure of the World."Frontiers in Neural Circuits 11 (October 2017): 81.

WORLDBRAIN WHITEPAPER

Empowering AI with Web3 technology