A New Scheme and Specific Implementation Steps for Achieving True Artificial General Intelligence

Authors: Chen Yongcong, Zeng Ting, Zhang Jun, Chen Yongcong, Zeng Ting

Date: 2023-07-29T00:00:00+00:00

Abstract

Currently, mainstream artificial intelligence generally adopts the technical route of "attention mechanism + deep learning + reinforcement learning". It has achieved significant progress in the AIGC (Artificial Intelligence Generated Content) field, triggering a technological wave of large models. However, in domains that require interaction with the physical environment, such as elderly care, domestic assistance, agricultural production, and vehicle driving, the cost of trial and error is extremely high, making the reinforcement learning process that requires extensive trial and error difficult to implement. Therefore, to achieve general artificial intelligence applicable to any domain, we must both leverage existing technologies and address their deficiencies, thereby further advancing the technological wave of artificial intelligence. In this paper, we analyze the limitations of the large model technical route and propose solutions to address these limitations, thereby resolving the inherent defects of large models. In this paper, we will reveal how to implement general artificial intelligence step by step.

Full Text

Preamble

A Novel Approach to True Artificial General Intelligence and Its Concrete Implementation Steps

Yongcong Chen[1]*, Ting Zeng[2], Jun Zhang[3]
[1]New Millennium Future Technology (Beijing) Co., Ltd., Beijing, 100084, E-mail: yongcongchen@sina.com
[2]Tsinghua University Library, Beijing, 100084, E-mail: ceng-t@mail.tsinghua.edu.cn
[3]Shenyang Normal University, Shenyang, 110034

Current mainstream artificial intelligence universally adopts the technical path of "attention mechanism + deep learning" + "reinforcement learning." This approach has achieved remarkable progress in the AIGC (Artificial Intelligence Generated Content) domain, sparking a wave of large model technologies [2][13]. However, in fields that require interaction with the physical environment—such as elderly care, domestic assistance, agricultural production, and vehicle driving—the cost of trial-and-error is prohibitively high, making the reinforcement learning process that requires extensive experimentation difficult to implement. Therefore, to achieve general artificial intelligence applicable to any domain, we must both leverage existing technologies and address their inherent limitations to further advance the AI technology wave. In this paper, we analyze the limitations of the large model technical route and propose solutions to address these limitations, thereby resolving the fundamental defects of large models. We reveal how to achieve artificial general intelligence step by step.

Keywords: Artificial General Intelligence (AGI), reinforcement learning, large models, ChatGPT, GPT-4

Current mainstream AI large models have sparked the flame of general artificial intelligence [1], but they are not yet true AGI. Where exactly do the capabilities of current AI large models hit their ceiling? Can "attention mechanism + deep learning" + "reinforcement learning" truly achieve "general artificial intelligence"? We argue that current AI large models cannot resolve the following critical defects:

1.1 Inability to Solve Problems Autonomously

For instance, when current AI observes its owner falling, it does not proactively come to assist [7]. This is because without its own needs, a machine cannot generate its own goals. Without its own goals, it cannot spontaneously create tasks. In other words, large models cannot autonomously create new program flows!

A large model is essentially a programming platform whose programming language is natural language [14]. Therefore, no matter how many advanced functions we add to a large model or how many tools and apps we integrate into it, the large model will not spontaneously create new processes. All its processes are preset, either from programmatic defaults or statistical patterns. Both approaches essentially constitute "using pre-configured processes to handle all problems" [8][9][10][11][12]. Regardless of how many if…else… branches exist in a process or how many possibilities are considered, it remains pre-configured and pre-existing. It is not something the machine creates for itself in response to specific tasks! Consequently, machines that make decisions according to predetermined processes exhibit "bookworm" intelligence—rigid decision-making that cannot adapt flexibly, making it difficult to handle the endless unexpected situations in real social life. This is the current predicament of artificial intelligence.

1.2 Knowledge Cannot Be Updated in Real-Time

Current AI employs large-scale data training, making real-time knowledge updates impossible. For machines interacting with the environment, real-time knowledge updates are crucial because interaction with the environment is precisely how machines acquire new knowledge. If acquired knowledge cannot be updated in real-time, machines will repeatedly make the same mistakes when facing identical input information [3].

1.3 Inapplicability to Domains Requiring Real-World Interaction

In domains requiring interaction with the real world—such as autonomous driving, household chores, and patient care—machines need to establish interactive decision-making knowledge between their actions and the external environment. These domains cannot accommodate extensive trial-and-error, preventing machines from building decision-making knowledge through reinforcement learning in real environments [2][4].

We envision a future where every household has a robot nanny, all vehicles drive autonomously, and robots handle all industrial, agricultural, and service work, leaving humans to primarily enjoy life's beauty. However, current AI technical solutions cannot yet realize these scenarios.

2. How is Knowledge Created?

2.1 How to Describe Information Contained in a Matrix?

Although a matrix may contain substantial information, we can express all information within it by establishing a coordinate basis cluster. If this coordinate basis cluster is complete and orthogonal, it can most concisely describe all matrix information. If the established coordinate basis cluster is not orthogonal but complete, we can still use it to express any information in the matrix. If the coordinate basis cluster is incomplete, some vectors in the matrix cannot be expressed through it, requiring us to increase the dimensionality of the coordinate basis cluster.

If the basis coordinate cluster is an eigen-orthogonal basis, we achieve the most concise coefficient expression of the vector's complete information. If the basis coordinate cluster is not fully orthogonal, we hope it approximates an orthogonal basis cluster as closely as possible, because this yields a sparse coefficient matrix (efficient expression). However, if we only care about partial common information, we can use common information patterns as coordinate bases. Such bases are not efficient for overall information expression but provide efficient expression for common information (the coefficient matrix is sparse). Therefore, if we live in an information matrix space, the most important task when needing to recognize, analyze, and generate various information is to find a set of basis coordinate clusters in the information matrix space.

2.2 How Do Humans Create Knowledge?

The information humans can recognize is only an infinitesimal portion of the information in our world because human information resolution is limited. The relative spatiotemporal relationship between atom A and atom B on a blade of grass is also information, but we do not recognize it.

During evolution, humans developed token recognition capabilities. Tokens are the minimal information units humans commonly use, such as a straight line. Tokens themselves constitute a "world model"—the smallest "world model" for building humanity's grand knowledge edifice. Through evolution, humans developed "pattern recognition" capabilities using tokens as "models" to recognize surrounding information, greatly improving information recognition energy efficiency. This is evolution's gift to us.

If we use the minimal information units humans habitually employ—such as points, lines, surfaces, colors, textures, curvature, syllables, tones, symbols, touch, temperature, direction, etc.—as tokens, then humans live in a 4-dimensional matrix composed of tokens (three spatial dimensions + time dimension). For humanity, this 4D token matrix contains all knowledge from the Big Bang to the present.

Humans gradually use certain symbols (linguistic symbols) to represent common token combinations—this is the concept. Humans use concepts to describe any information (vector) in the matrix: through conversation and writing articles. These concepts constitute a coordinate basis cluster in our information space matrix.

Under such a basis cluster, the coefficient expression of common information (vectors) is sparse. For example, "investor" represents "hominid, wealthy, wants to earn more money, seeks help to earn it, bears risks, signs agreements, shares profits…"

Concepts contain common token combinations and linguistic symbols. Because linguistic symbols appear more frequently and have higher representativeness, they may become the most commonly used entry point for a concept.

Clearly, human concepts are not orthogonal systems. Humans habitually treat frequently occurring token combinations as a concept. The token combinations contained in concepts may have non-overlapping, partially overlapping, or fully inclusive relationships. Tokens common to numerous entities have high representativeness but low resolution and fewer in number, representing abstract concepts. Adding more tokens to abstract concepts forms more specific concepts with narrower representative scope and higher resolution.

Although such coordinate basis clusters are not efficient for expressing complete information, they efficiently express common information. For instance, concepts like "cat" and "dog" may share many tokens and are non-orthogonal, but they are efficient for human daily information expression.

Moreover, this is crucial for information generalization because an entity's attributes are essentially composed of the attributes of its constituent tokens. For example, "cat" is a common token arrangement in space and time, containing multimodal matrix information elements such as language, text, sound, images, actions, and touch. Some matrix elements in this arrangement may have higher weights because they are more common and may all belong to the "animal" concept. The "animal" concept contains fewer elements but has broader applicability, so attributes connected through their shared tokens (such as those related to the "animal" concept) can be directly reused between "cat" and "dog." This is the information generalization process and the origin of intelligence.

2.3 How Does Deep Learning Work?

In deep learning, the coefficients of each neural network layer represent a set of implicit coordinate bases. The transition from layer A to layer A+1 is essentially a basis transformation process from layer A's coefficient matrix (which expresses information together with its corresponding implicit coordinate basis cluster) to layer B's coefficient matrix (which expresses information together with its corresponding implicit coordinate basis cluster). This is followed by a nonlinear function that partially compresses or discards information.

The essence of deep learning is using a "trial-and-error method" to find a suitable coordinate basis cluster that sparsifies the coefficient matrix of "useful information" in the input.

The purpose of residual networks is to reduce information loss in each neural network layer, enabling machines to perform multiple transformations and thus have a higher probability of finding an optimal basis cluster. Regularization aims to make the implicit bases of intermediate neural network layers as close to orthogonal as possible, thereby avoiding mutual influence between dimensions and preventing local optima. It achieves this by forcing the coefficient matrix of intermediate layers to approach a sparse matrix.

The high-dimensional features created by deep learning constitute a coordinate basis cluster in its information matrix. However, it lacks the constraint of "using common token combinations." It uses a "trial-and-error method" under error constraints to establish a coordinate basis cluster that tends toward an efficient orthogonal system. This is more efficient for overall useful information expression but differs from human habits (humans only need efficient expression of common information), so "deep learning" and "humans" communicate like "chickens talking to ducks"—they cannot understand each other.

In large models, the attention mechanism essentially uses locally obtained statistical knowledge from pre-training to predict the frequency (occurrence probability) of specific token combinations, using them as the preferred coordinate basis cluster to recognize, analyze, and generate various token combinations.

Such a basis cluster better aligns with human habits, enabling language com-

munication between large models and humans. The overall expression efficiency may not be high, but the expression efficiency for common information is higher.

2.4 What is the Essence of the Attention Mechanism?

The core of the attention mechanism is essentially Bayesian inference (conditional probability), which can be summarized as "given the occurrence probability of N token combinations, find the occurrence probability of M token combinations."

In human language, combinations of N tokens are almost infinite, and combinations of M tokens are also infinite. Therefore, machines cannot solve the problem of "given N token combinations, find the occurrence probability of M token combinations" through statistics alone. This problem becomes even more pronounced in multimodal contexts. Consequently, machines can only speculate about "the probability of M token combinations occurring after N token combinations appear" based on a limited number of statistics. This is the essence of the attention mechanism. The weight matrix obtained through pre-training represents limited statistical knowledge, and the attention mechanism uses this limited statistical knowledge to infer the probability of accompanying M tokens appearing after the current N token combinations. If some tokens in the N+M tokens have high weights, it indicates they frequently co-occur and are more likely to be common token combinations.

This is the core mechanism of attention—a method for finding common token arrangements. Its essence can be considered Bayesian inference integrated with neural networks.

Therefore, deep learning enhanced by attention mechanisms creates coordinate basis clusters that better align with human concept-creation habits, enabling language communication between large models and humans [15].

In language models, "common token combinations" are "common expressions." They include both the organizational forms of common tokens (similar to grammatical structures) and specific "common expressions." Because machines' statistical analysis capabilities far exceed humans', the "common expressions (including grammar)" discovered by machines are vastly more extensive than human "common expressions."

The attention mechanism closely resembles human learning. When we learn information from a book, the method of "first reading thin, then reading thick" follows the same principle. "Reading thin" involves summarizing the framework information—a process of information compression. "Reading thick" then adds different details (combining with other vectors to form new vectors) to create new knowledge, which is an information generation process [17][18][19].

2.5 How Do Large Models Work?

In large models, when information is input, the attention mechanism's inference process is the projection of the input vector onto the coordinate basis cluster. The weights obtained by the attention mechanism are the coordinate values [15].

In large models, input tokens are projected onto vectors in the weight matrix at the first layer—a vector decomposition process. Then, a second-layer projection occurs, where input token combinations, after weighting, are projected again onto token combinations in the pre-trained weight matrix (combination-to-combination projection). After multiple attention mechanism operations, multiple projection decomposition processes from input token combinations to pre-trained token combinations are formed.

The weight coefficient matrix output by the final attention mechanism layer, together with its underlying implicit coordinate basis cluster (using common token combinations as the coordinate basis cluster), jointly forms a re-description of the input information (self-attention mechanism).

Therefore, the working principle of large models is: (1) It uses token combinations from the pre-trained weight matrix as the basis cluster, where the weight matrix is locally obtained statistical information from training materials through trial-and-error; (2) It employs the attention mechanism to implement the projection process from input token combinations to weight token combinations (vector decomposition), where the weights obtained during inference are the coordinate values; (3) With vector components, numerous nearby vectors can be found, and the next vectors corresponding to these nearby vectors are the output vectors. The proximity relationship between vectors manifests as the probability of output vectors.

Thus, large models are autoregressive prediction models that have undergone a coordinate basis cluster transformation on the original input basis (where each token is a dimension), converting it to a coordinate basis cluster where "common token combinations serve as a dimension" before performing autoregressive prediction.

2.6 Why Do Large Models Exhibit Emergent Abilities? When Do They Emerge?

Why do large models exhibit "emergence"? The principle is simple. When an American comes to China, they can complete accurate translation through extensive shared background information (such as human needs, social structures, etc.) with a moderate amount of Chinese-English comparisons.

However, a large model is like an alien—it shares no common background information with humans and only observes the connection patterns in human information. Therefore, it needs to extract these connection patterns to predict information development. Initially, when samples are insufficient, the "information framework" it extracts differs significantly from the human "informa-

tion framework," causing it to make continuous mistakes, groping in the dark and hitting walls everywhere. As sample quantities increase, its "information framework" has a higher probability of aligning with the human "information framework."

But this is not a linear process. Before reaching a certain threshold, it is like a linguist deciphering an ancient language, groping in the dark with minimal progress. At a certain node, if the accuracy rate reaches the threshold, the entire decryption process accelerates dramatically and completes rapidly. This is the "emergence" phenomenon. What emerges in machines is not intelligence but the discovery of correct "common token combination methods." Because evaluation standards for machines are human standards, when its basis aligns with the human basis, its capabilities emerge.

Can RLHF Ultimately Solve the Problems Faced by Large Models?

Current large models have two serious problems:

(1) The Hallucination Problem [20]

The core capability of current large models is transforming input information into a coordinate basis cluster composed of common token combinations (vector projection decomposition)—a basis transformation process in information space. Then, using the obtained coefficient matrix (attention mechanism inference weights), it can find multiple similar "pre-trained vectors" (component-weighted comparison). Based on these similar "pre-trained vectors," it finds the "next vector" according to pre-trained mapping relationships and selects one for output. This is the autoregressive prediction process and the working principle of GPT-like large models.

Therefore, large models optimize "parameters," where each parameter corresponds to a set of token combinations. On the surface, large models optimize network parameters; in essence, they optimize common token combinations—in other words, they search for an optimal coordinate basis cluster. Each layer's coefficients in the neural network correspond to an implicit coordinate basis cluster.

Large models only obtain "common token combinations" from massive data, not factual memory. Therefore, when facing input tokens, large models can only decompose input information onto the "coordinate basis cluster" and obtain the next tokens under different probabilities. This iterative process is itself a creative process. If facts are "common," they are retained as "common token combinations." If facts are not retained as "common token combinations" or lack sufficient weight, the machine creates information. Since GPT itself is an information generator, the hallucination problem is fundamentally part of its job [17][18][19], making it unsolvable for GPT.

For example, the machine observes that many journalist profiles are followed by links to other articles or awards. If this information organization pattern appears frequently, it becomes a mapping from "framework" to "framework." Therefore,

if input information contains a similar framework but only the journalist's name differs, the machine can map "framework + details" to "framework + details," generating many webpage links or awards in the output. However, these links and awards are established through "framework + other vector" mapping to "framework + other vector" and likely do not exist at all!

To solve the hallucination problem, many hope to use external "vector databases" for large models to query factual knowledge and eliminate hallucinations. This is another version of attempting to achieve AGI using encyclopedias. Whether "vector databases" or "knowledge graphs," they fundamentally cannot solve the hallucination problem because this knowledge is external and cannot integrate seamlessly with the large model's own knowledge. They are like an ordinary person trying to run a translation company with just a dictionary—all the problems expert systems encountered will reappear.

(2) The Harmful Content Problem [20]

In large models, the attention mechanism is correct, but deep learning has flaws. In large models, the Transform model based on Self-attention adds positional encoding primarily to incorporate token position information, enabling utilization of positional relationships between elements. This is necessary for the attention mechanism because it aims to find tokens' temporal and spatial relationships.

However, through multi-layer deep learning networks under error constraints, large models undergo multiple coordinate basis transformations to find the "optimal coordinate basis cluster." This "optimal coordinate basis cluster" of token combinations no longer matches the original tokens' temporal and spatial relationships. Although it may retain partial organizational information between tokens (because the deep learning process is irreversible, token position information is only partially preserved), it becomes difficult for humans to understand and utilize. Therefore, we believe deep learning destroys the original temporal/spatial organization of tokens.

We can consider that large models perform a lossy translation process, converting human token combination sequences into their own language. The problem is that humans have not mastered the large model's language, so we cannot understand the knowledge created by large models or imitate its knowledge organization form to implant "innate knowledge" into the model. This is the core issue.

Moreover, because large models cannot achieve small-sample, cumulative learning and require ultra-large samples with knowledge formed in a single training session, it further increases the difficulty for humans to understand their knowledge organization.

Because machines have no self-derived needs, they cannot have self-perceived rewards and punishments. Without self-perceived rewards and punishments, machines cannot spontaneously create projections from vectors (information) to reward or punishment dimensions. In other words, the coordinate basis cluster

created by machines lacks fundamental dimensions unique to and necessary for humans, such as reward, punishment, happiness, and sadness!

The current remedy for large models is RLHF, which is equivalent to humans adding a reward dimension suffix to specific vectors after the fact. This means the machine's coordinate basis cluster gains an additional reward dimension.

If training data adds component values on the reward dimension to a large number of different types and sufficient quantities of vectors, it establishes projections from the common component combinations in these training vectors to the reward dimension. This is the machine's reward function. Therefore, machines can also predict the reward component contained in output vectors under different decisions, i.e., according to different combination methods. Consequently, machines prefer outputs with higher reward components, producing the astonishing effects of RLHF learning. Because knowledge learned through RLHF is actually generalizable, when a machine has its own reward and punishment dimensions, it gains preliminary "consciousness of seeking advantages and avoiding disadvantages," which is why we currently see a hazy shadow of "consciousness" in large models.

However, this is a post-hoc patching approach, meaning the machine must first attempt actions, then humans provide scoring feedback. It can only be applied in domains that allow extensive trial-and-error. This is akin to a child who has earned a Ph.D. but has no concept of "right and wrong," with parents following behind shouting "No," "No," "Yes" to instill a sense of morality, and the child cannot communicate directly with parents except through "Yes" and "No." Therefore, such learning is inefficient and will forever encounter unexpected corner cases!

3.1 Can Large Models Achieve Artificial General Intelligence?

We believe large models have proven their general direction is correct, but we do not consider them the right path to AGI. In NLP, humans progressed from early bag-of-words models and word vectors to EMLO [21], until Transformers truly implemented the attention mechanism. Combining deep learning with attention mechanisms [22] produces an optimized coordinate basis cluster similar to human expression—this is why Transformers exhibit intelligence "emergence."

However, we note that the path adopted by large models is "first vectorize to establish preliminary relationships; then adjust the coordinate basis cluster through trial-and-error; then re-vectorize under the optimized coordinate basis cluster to obtain correct relationships." This mechanism leads to enormous data requirements, massive computational costs, and knowledge formed in a single training session that is difficult to update in real-time [2][3].

Meanwhile, the reward function appears post-hoc, making it inapplicable to domains where trial-and-error is difficult, such as interactive decision-making in

real environments (autonomous driving, domestic assistance, industry, agriculture, commerce, service industries, government management, etc.).

Additionally, the idea of "task-oriented reinforcement learning" is wrong. Humans are "general" because we make decisions for all tasks based on "seeking advantages and avoiding disadvantages." Machines should do the same. With countless tasks, task-oriented reinforcement learning can never be completed! Moreover, many tasks have high trial-and-error costs!

3.2 What is the Correct Path to Artificial General Intelligence?

The problems with current large models are:

(1) The attention mechanism is correct, but deep learning has flaws.

Because deep learning destroys the original temporal/spatial organization of tokens, the resulting knowledge is difficult to understand and imitate. Therefore, humans cannot imitate its organization form to implant "self-needs" (innate knowledge) into machines.

Without "self-needs," machines cannot have "their own ideas" or "autonomous decision-making." Consequently, machines can only passively "decide" according to predetermined processes (either preset or statistical), lacking flexibility—this is the major problem with current AI.

(2) The idea of "task-oriented reinforcement learning" is wrong.

Humans are "general" because we make decisions for all tasks based on "seeking advantages and avoiding disadvantages." Machines should do the same. With countless tasks, task-oriented reinforcement learning can never be completed! Moreover, many tasks have high trial-and-error costs. For instance, in childcare, no one is willing to let machines experiment with their children!

Therefore, our solution is:

(1) Implement the attention mechanism without destroying the original temporal/spatial organization of tokens, creating knowledge that can be understood and imitated.

(2) We can imitate the knowledge organization form to endow machines with "innate needs." "Innate needs," as a special class of tokens, form common combinations with other tokens through the attention mechanism. These common combinations constitute common sense (the world model)!

(3) Machines only learn one thing: "how to satisfy self-needs," and only handle one thing: "how to satisfy self-needs." This is general decision-making.

(4) Because the original temporal/spatial organization of tokens is not destroyed, machines can directly obtain the temporal and spatial arrangement of tokens through linguistic symbols. This arrangement can be understood

and imitated, allowing machines to directly acquire all accumulated human experience through language learning! Machines no longer need to retrace the "evolutionary history"!

4. Step-by-Step Implementation of Artificial General Intelligence

Below are the 10 steps to implement our solution.

Step 1: Tokenize information (same as other AI technologies).

Step 2: Matrix-ize tokens (establish a memory bank).

Step 3: Input tokens propagate activation values to tokens in the memory bank based on similarity relationships.

Step 4: All activated tokens propagate activation values to neighboring tokens based on proximity relationships.

Step 5: Each activated token further propagates activation values chain-wise in the memory bank according to similarity and proximity activation principles.

In Steps 3-5, the higher the similarity, the larger the transmission coefficient; the closer the storage location, the larger the transmission coefficient; the higher the token's memory value, the larger the transmission coefficient.

Step 6: Each token accumulates activation values obtained from different propagation paths.

Step 7: All token activation values decay over time.

Steps 3-7 constitute the chain associative activation process, which is the inference process of the attention mechanism, where activation values are inference weights.

Step 8: Each token updates its memory value positively correlated with its obtained activation value magnitude. All memory values also decay over time.

Each token's memory value is its pre-training weight. In memory, numerous token combination patterns exist. Those that can repeatedly appear mutually activate each other, pushing up activation values and thus obtaining higher memory values. Therefore, if a combination of multiple tokens appears in the input, the token combinations related to this combination in memory have a higher probability of obtaining high attention weights. Thus, the chain associative activation process is a "token combination"-priority activation value propagation process.

Step 9: Pre-configure minimal innate needs (innate knowledge, composed of tokens + memory values + arrangement patterns). Innate needs are innate knowledge established by imitating knowledge organization forms. Innate knowledge can include minimal innate needs, rewards/punishments, emotions, and

necessary innate safety instinct knowledge, as well as other knowledge. This knowledge exists as part of the memory bank, seamlessly integrating with postnatally formed memories to form an integrated memory bank. "Fine-tuning" of innate knowledge is achieved through accumulated postnatal knowledge (including feedback).

Step 10: Allow innate needs, rewards/punishments, and emotions (represented using special tokens) to form temporal information streams with postnatal information (ordinary token information streams) during machine training and life, which are then stored. Through the chain associative activation process + attention mechanism, a fully connected knowledge network (memory) is formed.

Our solution ultimately forms a memory bank where each token is a data record composed of four fields as shown in Table 1 .

Table 1: Composition of each token data record.

When numerous tokens are stored according to temporal intervals and optimized (through chain associative activation process + memory and forgetting mechanisms for survival of the fittest), a knowledge network is formed. The network nodes are tokens, and the network connections are activation value transmission relationships. It must be emphasized that activation value transmission relationships are determined by tokens' relative positions, memory values, similarity between tokens, and the magnitude of initial activation values obtained by tokens. Therefore, activation value transmission relationships are temporarily established after input tokens appear, not fixed.

The memory values represent pre-training weights, and activation values represent inference weights under the attention mechanism. Thus, in our solution, knowledge acquisition and inference application are integrated, and innate and postnatal knowledge are seamlessly merged.

In the memory bank, there are both objective tokens and subjective tokens. The connection relationships formed between them through the attention mechanism constitute "information." The arrangement relationships of all tokens represent complete information with high dimensionality. "Knowledge" refers to arrangement patterns (including temporal and spatial) that can repeatedly appear. Because they can repeat, they contain fewer tokens, have higher representativeness, broader applicability, and greater abstraction, thus having lower dimensionality. "Common sense" further restricts this to "knowledge" commonly encountered by humans.

Our machine's memory bank can be inserted, modified, or merged, allowing knowledge between machines to be directly shared through memory bank merging. For example, a chef robot can directly acquire a doctor robot's skills by loading the doctor's memory bank, without needing to merge "chef big data" and "doctor big data" and spend millions of dollars and months on retraining.

4.1 Detailed Explanation of Each Step

Step 1: Tokenize Information. The machine only needs to break down input information, prioritize holistically and low-resolution features, and extract underlying tokens (such as overall contours, textures, topology, lines, corners, ridges, vertices for images; temporal/spectral pitch, timbre for speech; etc.). Store them sequentially in the memory bank according to temporal order. It is particularly emphasized: no need to recognize them, just store them! Even if the initially extracted tokens are random and the algorithm is imperfect, it doesn't matter. Because our algorithm uses accumulated common token combinations (the "world model") to subsequently guide the machine on how to "extract on demand." Common token combinations include both common tokens and their organizational forms.

Thus, the machine's token extraction process is a gradually optimizing process. After storing tokens in the memory bank, the machine subsequently changes these tokens' memory and activation values through the chain associative activation process and memory/forgetting mechanisms. Through survival of the fittest, tokens or token combinations that widely exist are retained and form more complex tokens, while rarely repeated tokens are eliminated and no longer extracted.

Therefore, the machine's token processing strategy is also: find widely existing raw data combinations as tokens. This is an application of the common information combination priority principle in determining token composition, similar to humans—it is evolution's gift to humanity. Because extracting such underlying tokens requires extensive reuse to maximize energy efficiency.

Step 2: Matrix-ize Tokens. Each token corresponds to a record in the memory bank with four fields as shown in Table 1. Memory value size indicates memory strength; zero means deletion. Activation value size indicates activation strength; zero means no activation. All records spontaneously constitute the entire memory bank through simultaneous storage methods.

Specific implementation methods for simultaneous storage include:

(2.1) The machine preserves the temporal relative positions of tokens in input information. One implementation method uses the distance between tokens in storage space to reflect the temporal distance between moments when tokens were stored. For example, the machine stores tokens sequentially according to input time order, with temporally closer tokens stored closer together. Another method assigns each token coordinates in memory space, primarily including token storage time information.

The machine preserves the spatial relative positions of tokens in input information. One implementation method overlays each extracted token onto the original data at the position, angle, and size with highest similarity, preserving these tokens' spatial relative positions during storage. Another implementation method prioritizes extracting overall low-resolution tokens, then extracts other

local tokens on demand according to machine decisions. Through proximity storage relationships, local tokens and overall tokens have both proximity activation relationships and similarity relationships, enabling them to mutually activate and establish positional connections.

Step 3: Similarity Activation from Input Tokens to Memory Bank Tokens. Assign a uniform initial activation value A0 to each input token. A0 itself is a preset value but can be adjusted based on the activation values of reward/punishment symbols activated in the previous chain associative activation process.

The activation values of activated reward/punishment symbols represent the machine's prediction of potential reward/punishment values for previous input information. The initial activation value A0 affects the scope of the chain associative activation process. When A0 is high, the propagation range of the chain associative activation process is larger because, in our solution, activation value transmission coefficients are less than 1. As chain propagation levels increase, transmitted activation values become smaller. When a token's obtained activation value falls below a preset threshold, the chain propagation process terminates. Thus, A0 reflects the machine's attention level to input information. When A0 is high, the machine activates more memory tokens to search for memories related to input tokens, similar to humans—your boss's words make you associate more information.

Similarity activation principles: (1) The higher the similarity between tokens, the larger the transmission coefficient—this is the correlation dot product between tokens. (2) The higher the memory value, the larger the transmission coefficient; memory values are pre-training weights. It must be emphasized that the same token may appear in many positions in the memory bank, each with different memory values! This is because the same token has different weights in different token arrangements, similar to the attention mechanism in large models.

Step 4: Proximity Activation of All Activated Tokens. We believe that proximity relationships between tokens represent some implicit correlation. The closer in time, the tighter the potential relationship. This correlation can be statistically derived through the chain associative activation process + memory/forgetting mechanisms. Proximity relationships actually reflect a token combination relationship. If this combination can repeatedly appear, it is a common combination. Therefore, we discover common combination patterns through the proximity activation process in chain associative activation.

Each activated token further propagates activation values to its neighboring tokens; the closer the temporal position, the larger the transmission coefficient; the higher the memory value, the larger the transmission coefficient. If proximity relationships exist between tokens in the memory bank, it indicates they were once a combination pattern. If their memory values are high, it indicates they are a common combination pattern. If only one token has a high memory value,

they are not a common combination. If tokens' memory values are not high, the activation values they propagate are low, and the token chain propagation quickly stops, indicating such information is unimportant and has low weight in information processing.

Tokens activate common combinations containing them using the principle "the closer the temporal position, the larger the transmission coefficient; the higher the memory value, the larger the transmission coefficient," which is essentially the projection process of input tokens onto a coordinate basis composed of a set of tokens.

If N input tokens all project onto X combinations (token combinations) containing them in memory, these X combinations obtain high activation values. Because each token simultaneously activates multiple tokens in X combinations through both similarity and proximity, X combinations obtain higher activation values through accumulation. The "models" composed of these higher-activation tokens are the expected models (world models) activated by the input vector (N tokens).

Essentially, this is a vector decomposition process onto a coordinate basis and an information recognition process.

Step 5: Each Activated Token Propagates Activation Values Chain-Wise in the Memory Bank According to Similarity and Proximity Activation Principles.

Each input token undergoes "similarity activation" and "proximity activation" in the memory bank, with activation value transmission magnitude positively correlated with their pre-training weights (memory values).

Each activated token in the memory bank similarly undergoes "similarity activation" and "proximity activation," with activation value transmission magnitude positively correlated with their pre-training weights (memory values).

This process proceeds chain-wise until all input tokens complete their "chain activation process." Thus, besides memory vectors similar to the input vector being activated, the machine also activates the "antecedents" and "consequences" of memory vectors similar to the input vector—that is, the preceding and following information in time within the memory bank. Moreover, different memory fragments may activate different "antecedents" and "consequences."

This enables our solution to infer possible previous vectors and predict possible next vectors. Because we adopt a "overall low-resolution tokens first" strategy, spatial positional relationships are actually established through temporal positional relationships. When information is input, the machine first extracts "overall low-resolution tokens" and stores them in the memory bank, then initiates the chain associative activation process. After completion, decisions are made by statistically analyzing the activation values of activated reward/punishment symbols.

The machine's decision-making principle is seeking advantages and avoiding disadvantages. Decisions may involve further information recognition or other actions. If the decision is to further recognize information, the machine uses the currently high-activation token combination pattern (including language tokens) as an expected model to actively confirm high-activation tokens not yet present in the input. The method involves imitating past experiences of obtaining these tokens to adjust its sensor system. This is an active "pattern recognition" for information seeking, similar to human recognition processes.

These newly obtained tokens (such as local details) have temporal proximity relationships and partial similarity relationships with the original "overall low-resolution tokens," enabling them to establish connection relationships through mutual activation value transmission. Thus, newly obtained tokens establish positional relationships with the original overall low-resolution tokens. Through memory and forgetting mechanisms, these overall low-resolution tokens and frequently accompanying local tokens gradually form a "world model."

It must be noted that the world model is not an independent model. Its constituent tokens may be distributed throughout the entire memory bank, temporarily created through tight activation value transmission relationships based on similarity, proximity, and high memory values. Therefore, it is not static but distributed, temporarily constituted by tokens with high activation values under input information stimulation. No separate model exists in the memory bank.

Step 6: Activation Value Accumulation. If a token in the memory bank has activation value propagation paths with multiple input tokens (directly or indirectly related), activation values transmitted from the input accumulate. Therefore, tokens in the memory bank directly/indirectly related to multiple input tokens obtain higher cumulative activation values from multiple propagation paths.

Through this method, tokens in the input that are interrelated mutually push up the weights of related tokens in the memory bank. In other words, common combinations rise from the sea level of activation values. This activation value sea level consists of low activation values from numerous tokens.

The tokens rising from this sea level constitute one or more "world models." Moreover, memories most relevant to the input, though not necessarily common, may also obtain high activation values due to direct relevance and short propagation paths.

Thus, our solution can both obtain "information frameworks" through common token combinations and focus on specific factual details. Our solution inherently includes a "fact database" and can solve the current GPT "hallucination" problem.

Step 7: Activation Value Decay Over Time. All activation values continuously decrease over time. When subsequent tokens input and activate related

tokens in memory, activation values from previous inputs have not completely decayed, resulting in accumulation.

The machine's decisions are based on all activated tokens, so both preceding and subsequent input information is considered. Therefore, the machine's thinking has temporal coherence, solving problems of "ellipsis," "reference," and "metaphor."

Thus, our machine utilizes implicit relationships between preceding and subsequent input information—this is the attention mechanism! Furthermore: the machine adjusts the initial activation value A0 assigned to input tokens based on the predicted "advantages/disadvantages" magnitude from the previous decision. A0 affects the scope and accumulation magnitude of activation value propagation! This is adjusting attention intensity based on "advantages/disadvantages," similar to humans. In this regard, it surpasses current technology (Transformer). In fact, this resembles human decision-making processes—for example, your boss's words make you generate more associations, activate more reward or punishment symbols, and thus more deeply predict reward/punishment values.

Step 8: Updating Pre-Training Weight Matrix Through Chain Associative Activation + Memory/Forgetting Mechanisms + Advantage-Seeking Principle. In our solution, token combinations that can repeatedly appear may obtain higher memory values due to repetitiveness. Because they are repeatable combinations, each occurrence mutually pushes up the other's activation values, obtaining memory values far higher than simple repetitiveness.

Moreover, because they are repeatable, their combinations can obtain higher activation values each time, making them more easily activated and more likely to obtain memory increments. This is a positive feedback loop. Thus, our machine can self-summarize experience. However, forgetting existing thinking patterns also becomes a time-consuming process.

Therefore, in our solution, the machine's pre-training statistical process is not simply counting repetitiveness and using memory/forgetting mechanisms. It is completed through the attention mechanism + memory/forgetting mechanisms + advantage-seeking principle.

The machine's decision-making process is advantage-seeking. During this process, information recognition is selective based on advantage-seeking principles. Thus, our machine establishes common combinations between tokens based on its own needs. Consequently, our machine's recognition of external and internal information is selective.

Memory and forgetting mechanism: All tokens in the memory bank, when activated once, update their memory values positively correlated with their activation values. Their memory values are the pre-training weight matrix! Because token arrangements cannot be exhaustively enumerated, this is an incomplete statistical process, similar to large model pre-training.

The chain associative activation process is the inference process of the attention mechanism under input token combination stimulation (from local statistical weights to localized weights calculation for input), similar to the attention mechanism in Transformer.

This process is essentially the projection of input vectors onto the coordinate basis cluster established by the attention mechanism. Input vectors can be seen as an original basis cluster composed of impulse functions under input dimensions, while the coordinate basis cluster established by the attention mechanism is based on common token combinations.

The attention mechanism's inference weight matrix is the coefficient matrix for projecting input vectors onto the basis cluster. In our solution, the chain associative activation process, similar to multi-layer attention mechanisms in Transformer, is also a projection process of input vectors onto the coordinate basis cluster established by the attention mechanism: first individual token projection, then combination projection. After chain activation completion, one or multiple high-weight components composed of high-activation tokens constitute the information "framework." Each framework contains many tokens that are difficult to describe specifically. However, language symbols within them, due to high representativeness and repetitiveness, often obtain the highest activation values and may become representative tokens of the "framework." Therefore, in our solution, activation values are the inference weight matrix.

In fact, both large models and our network are neural network-like. The attention mechanism is essentially Bayesian inference. 通俗地说, the attention mechanism is: given the conditional probabilities of some tokens and the joint probabilities of partial tokens, calculate the joint conditional probability of specific token combinations. This is the application of Bayesian inference combined with neural networks. In large models, known token probabilities and partial token joint probabilities are determined by the weight matrix, and token combination probabilities are predicted through multiple correlation operations. In our solution, known token probabilities and partial token joint probabilities are explicitly expressed in the memory bank as token memory values, relative positions, and similarities between tokens.

We can see that our attention mechanism implementation uses small-sample, cumulative learning. Moreover, because the weight matrix updates in real-time, our solution enables real-time knowledge updates. We do not distinguish between pre-training and inference processes, so our machine is lifelong learning.

Additionally, our solution does not require the BP algorithm or pre-training. Its computational load is essentially similar to large model inference processes. Therefore, our solution requires far less computation than Transformer and can also be parallelized. Consequently, our solution can localize the pre-training process computation.

Each machine is an intelligent agent that self-trains, iterates continuously, and evolves.

Furthermore, token extraction in our solution can adopt similar technology to current large models with comparable computational load. The chain associative activation process is highly patterned and can be directly implemented at the hardware level using novel storage devices, facilitating computation localization in our solution, which helps expand application scenarios and reduce costs.

Step 9: Pre-configure Minimal Innate Needs. We have implemented the attention mechanism, found common token combinations, and did not disrupt the original temporal/spatial organization of tokens! Therefore, the knowledge network formed by our solution is understandable to humans. We can imitate the organization form of the final memory bank to establish the initial minimal innate memory for machines! This is equivalent to pre-configuring minimal innate knowledge similar to human innate knowledge (knowledge babies are born with).

Innate memory needs to include the machine's minimal "need system," "reward/punishment system," and "emotion system." The method uses special tokens to represent each "need," "reward/punishment," and "emotion," then imitates the post-training form of the memory bank (essentially appropriate token arrangement patterns + appropriate memory values) to implant minimal innate knowledge.

In daily life, these special tokens representing "needs," "reward/punishment," and "emotion" are trained together with other tokens that trigger them, undergoing chain associative activation, memory, and forgetting together. That is, through the attention mechanism, these special tokens establish common token combinations with other tokens like ordinary tokens. Therefore, we must pre-configure the machine's minimal "need system," "reward/punishment system," and "emotion system" to enable tokens representing the external world (including the machine's own state parameters) to trigger these special tokens, thereby establishing information streams. Through chain associative activation + advantage-seeking decision-making + memory/forgetting mechanisms, the most common and machine-caring token combinations are gradually acquired.

Thus, we establish connection relationships between "objective world's common token combinations" and "needs." "Objective world's common token combinations" are objective "objective common sense," while "common token combinations" formed by "objective world's common token combinations" and "needs" are "subjective common sense." "Objective common sense" and "subjective common sense" together constitute "common sense."

Common sense is the "world model," containing both humans' cognitive "world model" of the external world and the relationship between the "world model" humans establish and "me." It must be noted that tokens are not just static features but also include simple dynamic features (such as rotation, swaying, etc.), so the world model is neither static nor fixed—it is temporarily created under input token stimulation!

Moreover, each person's established world model is different, directly related to their experiences. In our solution, the machine's "world model" is directly related to its training data and life experiences!

With a world model, input tokens can activate reward/punishment tokens, emotion tokens, and need tokens through the chain associative activation process. The activation value transmission paths from input tokens to these feature tokens constitute logic reasoning processes compatible with neural networks! They are explicit, understandable, and imitable, making machine decisions visible.

In fact, in the actual creation of "AGI," Step 9 is essentially the first step. However, we can use the preceding steps to train experimental data, thereby obtaining and understanding the organization form of machine-created knowledge, then imitate these organization forms to implement Step 9.

1. Pre-configure a basic need advantage/disadvantage system related to machine life activities. For example, assign a reasonable range to battery data, pre-configure a symbol representing "hunger" in "innate memory," place a "punishment" symbol and an emotion symbol representing "hunger" next to the "hunger" symbol, and assign them appropriate memory values. When battery power is insufficient, the life status monitoring program directly assigns an initial activation value to the "hunger" symbol in innate memory. Its activation value chain-propagates throughout the memory bank, activating the nearby "hunger" emotion symbol and "punishment" symbol. Thus, the machine has "hunger" emotion and "punishment values." To avoid "punishment values," the machine actively searches for a charging plug using its experience!

2. Pre-configure "higher-order needs" for machine values, requiring the simplest communication methods, then cultivate values. Values need cultivation from childhood! Therefore, we need to cultivate robots' "values" through education from an early age. Since education requires "reward" and "punishment," machines must be able to recognize "reward" and "punishment" from the start. Only then can we initiate the first step of learning through "reward" and "punishment"!

Thus, we need to imitate the organization form of postnatal memory networks to give machines innate knowledge capable of recognizing the simplest "reward" and "punishment." For example, pre-configure the most basic nodding features (assuming X tokens) / head-shaking features (assuming Y features), without requiring precision!

Place a "respected" symbol next to nodding tokens; place a "reward" symbol next to the "respected" symbol; assign high memory values to these symbols to make their relationship long-term memory. When partial nodding tokens appear in input information, the machine obtains "reward values" through the chain associative activation process. To pursue "reward values," the machine may plan various decisions aimed at obtaining "human nods"!

Similar to a child gradually acquiring complex learning abilities from the sim-

plest communication methods, the "reward function" logic chain they gradually establish is: "milk" → "nipple" → "bottle" → "milk powder can"… → …"grades" → "house and car"… → "social status"…. → "life ideals."

Therefore, after training, the machine's memory bank contains numerous reward/punishment-related token symbols and token combinations closely related to these reward/punishment symbols, with causal relationships between them. These token combinations closely related to reward/punishment symbols represent things, behaviors, and results—values.

Thus, any machine values can be established by pre-configuring innate communication methods and then step-by-step cultivation. In fact, this is how humans work; no one is born a "sage."

Figure 1 [Figure 1: see original paper] Schematic diagram of establishing "innate minimal needs."

Step 10: Form a Fully Connected Knowledge Network. Our solution ultimately forms a network where each token consists of four fields: time stamp, token itself, memory value, and activation value.

Numerous tokens stored according to temporal intervals, optimized through chain associative activation + memory/forgetting mechanisms (survival of the fittest), form a knowledge network. Memory values represent pre-training weights; activation values represent inference weights under the attention mechanism.

Our network contains both objective and subjective tokens. The connection relationships formed between them through the attention mechanism constitute knowledge, where common knowledge is "common sense."

This is why our machine can predict advantages/disadvantages and make autonomous decisions! Because it has "needs" and "logic chains" related to "needs" (activation value transmission links composed of tokens). Driven by needs, it actively learns and self-iterates! For example, it charges itself and finds libraries to read books!

In our solution, knowledge revolves around "needs," and decision-making also revolves around "needs." This is the core reason our machine can achieve "generality"! It faces only one task: "needs," not various "external tasks." Therefore, our solution represents "active intelligence," while all other current solutions represent "passive intelligence."

Our solution enables small-sample learning, real-time knowledge updates, and integrates training and usage processes, making the machine lifelong learning and self-iterating.

Because machine knowledge exists as a memory bank stored in temporal order, gradually optimized memory values allow different memory banks to be directly concatenated to form larger memory banks. Therefore, after merging a chef's memory bank and a doctor's memory bank, the robot can simultaneously possess

both skills without retraining with massive amounts of chef and doctor data. Current AI technology routes cannot achieve this. In large models, training must simultaneously use large amounts of doctor and chef data for the machine to master both skills. Obviously, hoping machines can have "various" abilities through such training methods is…

4.2 An Example of Memory Value and Activation Value Change Process

Figure 2 [**Figure 2: see original paper**] Simple example of memory value and activation value changes during associative activation.

Figure 2 shows a simple example of memory value and activation value changes during associative activation. For simplicity, assume the machine's memory bank is empty and this is the machine's first time receiving input information (and we have not pre-configured innate memory). Assume from time t0 to t7, the machine's input tokens are "We hope for world peace." In actual processes, the machine should adjust the initial activation value assigned to all input tokens based on the activation values of currently activated reward/punishment symbols (value prediction). However, here, without a value system for adjustment, we assume the default initial activation value assigned to input tokens is 90 (assuming activation value range is 0~255). After the chain associative activation process, assuming tokens' activation values are 90 according to the memory curve, and current memory values are 0, the machine obtains a memory value increment of 126.

The memory value update increment is $\delta m = f(m0, A0)$, where m0 represents the current memory value and A0 represents the current activation value. The memory value update increment is positively correlated with activation value. All memory and activation values decay over time (exaggerated decay gradient used here).

From time t9 to t19, the machine receives the second input tokens: "Peace makes our world beautiful." Clearly, according to the "similarity activation" process, token "和" (peace) first activates token "和" in the memory bank, transmitting activation values. Because the memory value of "和" in the memory bank is high, it receives transmitted activation values from the input tokens' initial activation values with a large transmission coefficient.

The similarity activation transmission coefficient is $T = f(S, m0)$, where S represents similarity (dot product of token vectors) and m0 represents the memory value of the transmitted token. The activation value transmission coefficient is positively correlated with similarity and memory value.

Simultaneously, the token "和" in the memory bank initiates chain propagation because its activation value exceeds the preset threshold. In this chain propagation process, it first activates neighboring tokens "平" and "界" through "proximity activation."

The proximity activation transmission coefficient is T = f(D, m0), where D represents the temporal distance between two tokens and m0 represents the memory value of the transmitted token. The proximity activation value transmission coefficient is inversely correlated with temporal distance and positively correlated with the transmitted token's memory value.

After obtaining activation values, if "平" and "界" exceed the preset threshold, they also initiate chain propagation processes, searching for similar tokens in the memory bank for activation value propagation and performing proximity activation on neighboring tokens. Both processes' transmission coefficients are positively correlated with memory values.

Through the chain associative activation process, input tokens may activate token combinations related to them throughout the entire memory bank. The activation scope depends on their obtained initial activation values, which are influenced by value prediction.

After all tokens from the second input complete the chain associative activation process, we can see that among tokens stored in the memory bank, the "和平" (peace) token combination has the highest memory value and proximity, so each token in future chain associative activation processes will obtain higher activation values due to high memory values. Simultaneously, "和" and "平" not only have opportunities to obtain higher activation values themselves but also mutually transmit activation values due to positional proximity (proximity activation). Because their memory values are high, they obtain high transmission coefficients. Through activation value accumulation, they become a token combination that easily obtains high activation value weights.

Second, we can see that among tokens stored in the memory bank, the "世界" (world) token combination is similar to the "和平" combination, obtaining the second-highest memory value, making it also a token combination that easily obtains high activation value weights.

Therefore, we only need two sentences to establish relative "weights" for tokens. Through continuous cumulative learning according to the above process, the machine can establish correct common token combinations and their memory values.

These memory values correspond to the "commonness" of combinations—that is, memory values are essentially local statistical values of combination occurrence probabilities obtained through training data. Activation values, based on these local statistical values, represent the dot product process (projection) from input token combinations to "common token combinations" (basis cluster).

Thus, we adopt a human-like learning method that is highly efficient and enables small-sample, cumulative, real-time learning. It does not modify parameters of "old knowledge," so there is no "catastrophic forgetting" problem. It does not require BP gradient optimization, so its computational load is essentially consistent with large model inference processes.

5. Our Solution Realizes the Three Conditions Proposed by Professor Yann LeCun

Deep learning pioneer and Turing Award winner Professor Yann LeCun believes the correct direction for AGI is the "world model," with the path being "human-like AI." He proposed three conditions: (1) Need for a world model, including a needs module modeling basic needs like happiness and hunger, and a value module for predicting values. (2) Need for neural network-compatible logical reasoning capabilities (current reasoning relies on external symbolic reasoning). (3) Need for a "general decision-making capability" that can decompose decisions top-down, rather than requiring a million reinforcement training sessions for each task!

Although they proposed these ideas, no complete technical solution exists. Our solution can realize these three conditions.

5.1 We Have Established a World Model

Input tokens activate token combinations in memory. High-activation token combinations are the activated "world model" (some tokens may already appear in input, others may not). Then, based on the predicted "advantage-seeking" decision-making process, the machine decides whether to further confirm the existence of other "high-activation tokens"—this is "pattern recognition." The world model is "common sense," which is the token combination pattern formed by subjective tokens like "needs," "reward/punishment," and "emotions" with objective tokens. Humans use "common sense" for "pattern recognition" of things.

After each new information input, the machine must perform chain associative activation, then store tokens using the "simultaneous storage" method. Simultaneous storage refers to mechanisms reflecting temporal intervals between tokens, such as storing temporally closer tokens in closer positions or using time information carried by each token to determine intervals.

After obtaining new tokens, the machine must search for paths to achieve rewards and avoid punishments based on updated activation values. The collection of these paths is the overall response path, which may form a network structure where many local paths can lead to either reward or punishment symbols.

Because we have activation value transmission paths leading to reward symbols (or punishment symbols), we have pre-emptized and proceduralized the reward/punishment function. This solves the sparse and lagging reward function problems in current reinforcement learning processes. Through an optimal response path search process similar to AlphaGo, the machine can find the initial optimal response path.

If the cumulative reward/punishment value does not enter an acceptable preset range (or does not converge), the machine cannot decide whether to select or exclude certain specific paths to maximize benefits. The machine then needs

to further recognize input information, add more tokens, and subdivide specific reward/punishment activation value transmission paths to help select or exclude certain paths. This step is the machine's spontaneously created, active information-seeking process to aid decision-making. This process iterates until reward/punishment statistics reach acceptable preset values or converge.

When further recognizing input information, high-activation tokens either have high memory values (representative tokens of a class) or are closely related to current input tokens (similar or frequently co-occurring). Therefore, high-activation token combinations activated in memory are representative token combinations related to current input information. These representative token combinations are the machine's temporarily created "world model," which we call the "expected model." It originates from past experience…

5.2 We Have Achieved Neural Network-Compatible Logical Reasoning Capabilities

All "reward/punishment" tokens stimulated by input tokens have activation value magnitudes representing value predictions. The activation value transmission paths from input tokens to activated reward/punishment tokens constitute reasoning capabilities fully compatible with connectionism! The memory network is a neural network organized by tokens according to activation value transmission relationships.

Activation value transmission essentially implements the reasoning process of the attention mechanism. Each token in token combinations, through the chain associative activation process, activates commonly co-occurring token combinations. Through activation value accumulation, it can calculate the token combination most related to the input from the input combination (known specific combination probability of N tokens), i.e., calculate the specific combination probability of M tokens. The final activation value distribution in the memory bank is the obtained Bayesian inference result.

In fact, the attention mechanism in current large models has already achieved neural network-compatible logical reasoning capabilities but has two defects: (1) Deep learning destroys the original temporal/spatial organization of tokens, making knowledge difficult to understand and imitate. (2) It lacks "subjective tokens" (such as needs, emotions, advantages/disadvantages). Therefore, large models' reasoning processes are flawed.

Turing Award winner and deep learning pioneer Professor Yushua Bengio believes the most important step in achieving AGI is combining neural networks with causal reasoning. In fact, our solution has already achieved this: the memory network is a fully connected neural network. From input tokens to activated "world model" is causal reasoning about the objective world's organization; from input tokens to activated "subjective token combinations" (tokens representing needs, emotions, reward/punishment) is causal reasoning between the objective world and the machine's own needs.

Thus, we have achieved "combining neural networks with causal reasoning." In fact, current large models have already achieved objective reasoning capabilities and partial subjective reasoning capabilities, but their reasoning processes are difficult for humans to understand and imitate, making them hard to utilize.

5.3 We Have Achieved Hierarchical "General Decision-Making Capability"

Machines only reinforce learning for one task: "How to satisfy self-needs?" and only handle one task: "How to satisfy self-needs?" Therefore, our machine's decision-making is "facing self-needs," while other AI solutions currently decide facing various "tasks themselves."

Information input generates various associations, good and bad. Reducing the probability of tokens that bring "punishment" and increasing the probability of tokens that bring "reward" is "general decision-making"! This is similar to human decision-making, hence general!

With pre-emptized and proceduralized reward functions, machines have "decision-making capabilities." With the universal goal of "seeking advantages and avoiding disadvantages," machines can achieve "general decision-making" capabilities.

(5.3.1) Current "Machine Learning" is Not True "Machine Learning"

Facing a new task, humans predict the "goodness/badness" of different decisions based on experience, trying at most a few options.

Facing a new task, current machines rely on reinforcement learning—"continuous trying," either (1) trying a million times to see results (Google's various game-playing AIs) or (2) having humans tell them what's good or bad (GPT-4, large models, RLHF) [23] before acquiring decision-making knowledge for the problem.

Thus, current machine learning follows the path of "first try" + "then eliminate." They should be called "machine evolution," not "machine learning." Therefore, we propose that AGI requires true "machine learning."

What is true "machine learning"? We believe true machine learning should, like humans, predict the "goodness/badness" of different decision paths based on past experience when facing new tasks, trying at most a limited number of options to acquire decision-making knowledge for new tasks. Furthermore, we believe true learning should, like child learning, directly acquire accumulated human experience through language. When facing new tasks, success can be achieved without any attempts! For example, in laboratories, teachers teach children experiments by imparting knowledge through language, directly passing human decision-making experience to children. Children can use language-acquired experience to interact with the environment and directly complete experiments, even if it's their first time performing them!

Real tasks and scenarios are infinitely varied. Humans cannot place every task type into numerous scenarios for "reinforcement learning"! Therefore, thinking must shift: convert all tasks into a single task—"How to satisfy self-needs?" All machine training processes train this single task.

Thus, facing this task, machines already have substantial "state" and "policy" knowledge, enabling them to predict potential "advantage/disadvantage" estimates under "different decisions."

Tasks given by humans become background information for the machine's task of "how to satisfy self-needs." If "obtaining human recognition" is one of the machine's needs, then in pursuing "satisfying self-needs," the machine will incorporate "completing human tasks" into overall advantage/disadvantage statistics. This is similar to humans—facing tasks assigned by bosses, you weigh advantages and disadvantages comprehensively to make different decisions. For example, one decision you might make is actively seeking more information to analyze the advantage/disadvantage impact of tasks before deciding. Actively seeking more information is a new task humans arrange for themselves. If machines also decide this way, it means machines arrange tasks for themselves—that is, machines program themselves. In fact, our machine adopts such a decision-making process: weighing advantages and disadvantages to determine strategy, and possibly actively seeking information to help maximize benefits.

(5.3.2) How to Achieve True "Machine Learning"?

Ten years ago, we believed that to create true "knowledge," we should start from information statistics. Unlike "deep learning," we thought machines should learn in human learning patterns using small samples and knowledge accumulation. So initially, we also attempted the path of "symbolic expression" → "causal logic" → "knowledge network."

After several years of attempts, we found this path's first step was impassable. Because "symbolic expression" → how to express "dog"? We needed to select all features of "dog." But "dog" can be an animal, a person, "a praised personality," or "a despised personality"—its meaning varies greatly in different contexts. Therefore, the essence of "dog" is the sum of its relationships with all other things. "Dog" must be placed in the entire knowledge network and defined through its relationships with all other knowledge. Thus, "symbolism" is impassable because "dog" cannot be separated from other knowledge! We must establish a "fully connected knowledge network" similar to deep learning—this was our first conclusion.

Because "dog" must be placed in the entire knowledge network and defined through relationships with all other knowledge, sufficient knowledge is needed to explain "dog" clearly. Therefore, "knowledge quantity must be sufficient" to understand what a dog is through sufficient background knowledge—this was our second conclusion.

Looking back, isn't this what large models do? "Deep learning" handles fully

connected networks, and large models handle "using massive knowledge to establish fully connected knowledge networks."

So why don't we see robots walking everywhere? Because having a knowledge network alone is insufficient! Machines must also be able to "interact and decide with the environment"! Research shows humans make over 30,000 decisions daily.

Currently known methods besides expert systems that enable machines to make decisions independently are only reinforcement learning algorithms. Therefore, one possible path to AGI is: large models + reinforcement learning algorithms. In fact, GPT-4 has already implemented "all knowledge + fully connected network + RLHF," where RLHF is reinforcement learning. Google released the GaTo model in 2022, already following the "all knowledge + fully connected network + reinforcement learning" path.

So why hasn't Google launched robots walking everywhere? The core obstacle on this path is that reinforcement learning algorithms require two preconditions [24]: (1) Machines need to know reward information obtainable under different decision paths. In practice, reward information is scarce and lagging, currently solved through extensive trial-and-error training. (2) Machines need to exhaustively search all possible decisions.

These two conditions are perfectly satisfied in games: games allow continuous trials, and decision search spaces have boundaries (which can be pruned to reduce search space). But in real life, many problems cannot be continuously trial-and-errored (e.g., childcare—no one lets you experiment repeatedly!), and there are no clear boundaries. Therefore, this problem cannot be solved! This is why Google continuously launches AIs that can play various complex strategy games but has never launched a basic "domestic nanny robot." In fact, most decision complexities in daily life are far less complex than games! But because many real-life matters cannot be massively trial-and-errored, and relevant information lacks clear boundaries, the above two difficulties cause OpenAI or Google to basically only use "large models + reinforcement learning" for things that allow massive trial-and-error. Therefore, AIGC still has a long way to go to AGI.

Our decision-making solution is also essentially reinforcement learning but only reinforces learning how to seek advantages and avoid disadvantages. Moreover, we use the chain associative activation process to automatically limit the search scope—only searching activated information! We also use the "token" → "reward/punishment symbol" logic chain to automatically predict reward/punishment information, rather than only obtaining reward/punishment information through post-hoc feedback. Therefore, we perfectly solve Google's problem of decision-making AI only being able to play games!

This is because we simultaneously implement "objective common sense" + "subjective common sense." Current technology routes first implement "objective common sense," then establish "subjective common sense" through "RLHF."

Therefore, in current technology routes, "subjective common sense" is obtained through post-hoc feedback, making it only applicable to domains allowing massive trial-and-error.

(5.3.3) Implementation Process of "General Decision-Making."

In any environment, machine input information includes all sensor information. Therefore, at any moment, environmental information is part of the input information.

Machine-environment interaction decision-making includes two aspects: (1) Optimal decision selection. (2) Decision execution process. These two steps are not separate but intertwined and processed in parallel!

"General decision-making" needs to solve the first problem: what is the reward function? In GPT-4 and AlphaGo, rewards come from final external feedback. In our AGI, rewards come from "reward" and "punishment" symbols activated by external information, with magnitudes being their activation values.

Step 1: What is the purpose? When information inputs (external + machine self-monitoring information), some reward symbols and punishment symbols are activated.

Each activation value transmission path from input → reward symbols and punishment symbols is a potential logic link that generates rewards or punishishments.

If every underlying feature on this logic link is truly realized, the reward or punishment propagated along this link is realized.

Therefore, the machine's response to any input information is the same: increase the probability of reward logic links occurring and decrease the probability of punishment logic links occurring to achieve advantage-seeking and disadvantage-avoidance.

Step 2: With purpose, how to plan? (1) How to increase reward links and decrease punishment links? Increase or decrease the realization probability of high-activation token combinations on the link. High-activation token combinations on the link are high-weight token combinations of this link. When they are true, the activation values propagated along this link are true, and ultimately activated rewards or punishments are also true.

(2) How to operate specifically? Select the N tokens with highest activation values on the activation value transmission path from input information → reward/punishment symbols. They are the top-level implementation paths that cause rewards or bring punishments to reality. The machine's goal is: (a) make tokens on reward paths realize (imitate past experiences to make them appear in input information); (b) prevent tokens on punishment paths from realizing (imitate past experiences to avoid their appearance in input information).

Therefore, selecting the N tokens with highest activation values on the logic path from input → reward/punishment, including their activation value propagation paths, constitutes the top-level implementation path. Why does the machine only select the N tokens with highest activation values? Because these tokens either have high memory values as representative tokens of a class, thus obtaining higher activation values, or are closely related to input information. Due to small quantity and few attribute constraints, concepts most closely related to them are usually "abstract concepts."

Because language symbols are frequently used, language tokens often obtain high activation values and become core tokens composing "abstract concept" token combinations, making language symbols representatives of concepts themselves, such as "eating," "escaping" and other abstract concepts. It must be noted that "abstract concepts" are not exclusive to language symbols; animals can also have "top-level decision-making."

Therefore, the machine's decision-making establishment process prioritizes "abstract concepts," then gradually adds more tokens to form more specific concept combinations. This is a top-down, gradually unfolding decision-making and execution process. We call this process "segmented imitation."

A specific example of segmented imitation: Suppose token set A is input tokens and token set B is response tokens; the machine searches for high-activation tokens through chain associative activation processes of A and B. These tokens are intermediate bridge tokens connected to both A and B because they receive activation values from both, becoming high-activation tokens. This process iterates to achieve top-down, layer-by-layer decision-making.

How to implement in computers? The method is: (1) External input tokens undergo chain associative activation → determine activation values of reward/punishment symbols (tokens exceeding preset values as targets), establishing first-level goals. (2) Starting from the highest-activation reward/punishment symbols, find the N tokens with highest activation values on the activation value transmission path from input to each first-level goal—they are the logic links to achieve corresponding reward/punishment. Tokens on the link are second-level goals. (3) The machine uses each second-level goal as a new goal, treats them as new input tokens, assigns them initial activation values, and initiates chain associative activation again. Therefore, those highest-activation tokens are token combinations related to both external input tokens and second-level goal tokens. This is because we use activation value accumulation and decay—only tokens related to recent input tokens can maintain activation states. Therefore, these tokens are third-level goals. (4) This process iterates, enabling the machine to decompose each first-level goal into hierarchical logic links to achieve them. (5) Each expansion of the decision-making process selects different reward or punishment values for accumulation. The machine selects sub-paths bringing reward values and avoids sub-paths bringing punishment values according to the advantage-seeking principle, thereby increasing cumulative reward values. When the machine discovers total

reward/punishment values have converged—meaning no further improvement is possible and benefits are maximized—the machine stops further expansion and enters execution. This is the hierarchical "general decision-making capability" proposed by Yann LeCun and the neural network-compatible logical reasoning capability proposed by Bengio.

Why select only N highest-activation tokens each expansion? Because past experience and current reality cannot completely match. By only selecting highest-activation tokens, the "model" composed of them is either abstract (broad applicability) or closely related to input tokens (good matching). Selecting only N highest-activation tokens aims to achieve experience generalization. Therefore, in our solution, experience generalization is automatically implemented.

For example, the machine has experience using a hammer to drive nails. When needing to drive nails without a hammer, and when stone-related tokens exist in input tokens, to achieve a first-level goal (reward symbol or punishment symbol, completing tasks to obtain rewards or avoid punishment), the logic link may contain token combinations representing the hammer. These token combinations become second-level goals.

The machine, through chain associative activation in the memory bank, may discover M activation value transmission paths to achieve the hammer goal, possibly from "starting from the memory toolbox" or "borrowing relevant experience from teammates." These activation value transmission paths are all paths to increase the realization probability of hammer tokens, i.e., second-level paths to rewards.

Because stone-related tokens appear in input, shared tokens between hammer and stone (such as weight data, size, hardness sensation, etc.) may obtain higher cumulative activation values and be selected as the top N high-activation tokens. They become bridge tokens, making stone-related tokens also a second-level path to rewards. This is the experience generalization process, where shared tokens between stone and hammer enable stone tokens to transmit activation values to reward symbols. This is possible because "stone" and "hammer" share partial attributes (shared tokens that repeatedly appear in various "hammering nails" scenarios in memory), serving as bridges for experience generalization. We can see that in our solution, the experience generalization process is automatically completed.

So, which path to rewards does the machine select? At this point, the machine needs to reselect its decision path according to the advantage-seeking principle based on the updated activated token space from the new chain associative activation process. Some paths may bring both rewards and punishments, causing difficulty in convergence when statistically calculating reward/punishment values. If the machine discovers reward/punishment statistics are not converging, its decision is to further recognize information to converge each reward/punishment transmission path.

For example, "starting from the memory toolbox" requires confirming the prob-

ability of "toolbox" currently appearing in input (realization). This probability can further converge the reward/punishment value of this path. At this point, confirming the probability of "toolbox" currently appearing in input becomes a new goal spontaneously created by the machine.

To complete this "new goal," the machine needs to imitate past experiences to execute. If in past memories its toolbox was always hung at the waist, the most likely imitated process to determine the appearance of toolbox-related tokens in input is to pat the waist with its "hand" to reproduce various sensor data combinations from past hand contact with the "toolbox." Because this decision consumes minimal power and time, achieving the machine's own benefit maximization, it is the optimal decision path.

Another example: the "borrowing from teammates" path requires increasing the realization probability of tokens on this path to transmit larger activation values to reward symbols (obtain greater rewards). Therefore, the most likely imitated experience is turning the head to look or asking.

Thus, in our solution, machine decision-making is highly complex, potentially nesting W decision-making and execution processes within one decision path. However, at any moment, the machine's sole goal is "seeking advantages and avoiding disadvantages." All decisions revolve around this goal, making machine decision-making highly flexible, constantly changing with environmental states, without pre-configured processes. The only pre-configured process is: "seeking advantages and avoiding disadvantages."

The above process iterates continuously, with new reward/punishment symbols activated each time. The machine statistically analyzes these reward/punishment symbol activation values until discovering convergence. At this point, the machine has established the optimal response path.

Machine decisions may be responses to input information or searches for more information to continue decision-making. Regardless of which, the machine increases or decreases the probability of specific tokens' occurrence by imitating past experiences. Whenever new information inputs, it updates activation value distributions in the memory bank through chain associative activation, requiring the machine to re-statistics reward/punishment information and re-seek optimal decisions. This process occurs constantly with new information.

Step 3: With planning, how to execute? Execution is increasing or decreasing the occurrence probability of specific tokens by imitating past experiences.

1. Select a small number of highest-activation underlying features → abstract decision path.
2. Add more high-activation underlying features → concretize abstract decision path.
3. Iterate steps 1 and 2 above until decomposing decisions into executable drive commands: sending waveforms to speakers, drive commands to motors, display data to screens, setting parameters to expression systems,

etc.

4. New input information may be encountered at any time, changing activation values and reward/punishment situations in the memory bank, so the machine may change original plans at any time during implementing optimal response paths!

Step 4: Segmented imitation in decision-making and execution. Through chain associative activation, the machine can find experiences related to current input. Among these experiences, probabilities composed of a small number of high-activation tokens, due to high abstraction, are usually representative abstract experiences. These experiences contain "antecedents" and "consequences" related to input tokens and are objects of experience generalization.

Experience generalization essentially uses causality from occurred processes to realize causality for unoccurred processes. In our solution, this is automatically completed through activation value transmission processes of "shared tokens" between two processes. Because tokens in the two processes are inconsistent, corresponding to mismatch issues in experience generalization, in our solution, experiences from the two processes automatically complete generalization through activation value transmission via their shared tokens.

It must be particularly noted that machine concepts are formed by various tokens through a vast three-dimensional network. The same tokens may be distributed in different memory fragments. These tokens may originate from self-experience or language symbol input.

Therefore, after language symbols themselves are activated, they activate related tokens represented by language symbols. Language symbols themselves have sequences, and language sequences usually contain symbols expressing token combination orders. Thus, behind language symbol sequences are token temporal and spatial information streams. Their temporal and spatial combination orders are "causal relationships." Moreover, these "causal relationships" can form tight activation value transmission relationships through chain associative activation processes of "language symbols." This tight activation value transmission relationship itself is a form of "experience." Therefore, in our solution, experience not only comes from machine self-experience but also from "others' experiences" obtained through language symbols. Thus, our machine can both learn "experience" through language symbols and imitate "experience" through token information streams composed of language symbols.

6. Comparison Between Our Solution and Current Large Model Approaches

Our solution solves the following problems:

(1) The problem of "establishing common sense." Deep learning destroys the temporal/spatial relationships of original tokens! In our solution,

using "chain associative activation process + memory and forgetting mechanisms" also implements the attention mechanism. However, because we do not use deep learning, the knowledge created by our solution retains the original temporal/spatial relationships of tokens. The original token combination patterns are precisely the foundation of human "concepts." Therefore, the "knowledge" created in our solution is knowledge that humans can understand and imitate.

In our solution, the essence of "knowledge" is the arrangement patterns of tokens in time and space, and the prediction of potential advantages/disadvantages to the agent from different token arrangements. The arrangement patterns of tokens in time and space are essentially "causality." These arrangement patterns are not simple temporal/spatial proximity relationships but are repeatable relationships summarized by the intelligent agent, potentially spanning large temporal and spatial distances. However, through chain associative activation processes, tokens spanning large distances form tight activation value transmission relationships—this is knowledge. If knowledge contains tokens representing "needs," "emotions," and "advantages/disadvantages," it can predict potential advantages/disadvantages, so token arrangements represent "knowledge." Those common arrangements are "common sense."

(2) The problem of "whether machines can have consciousness." We have solved how to endow machines with "self-needs." Therefore, machines can autonomously decide, self-evolve, have their own emotions, and pursue "self-needs," so our machines have "consciousness."

(3) The "general decision-making" problem. Machines make decisions based on "seeking advantages and avoiding disadvantages" for any task. Human-given tasks are byproducts of machines pursuing "self-needs." This is the same as you completing tasks assigned by your boss. You also complete the boss's tasks while pursuing "self-needs." If conflicts arise, you also make various flexible decisions based on advantage-seeking, probing the boss's true intentions and considering their bottom line, making your decisions very flexible!

(4) The "language understanding" problem. Because we have not destroyed the original temporal/spatial relationships of tokens, the token temporal/spatial sequences represented by language sequences can be understood and imitated. Therefore, machines can learn various skills directly through language like humans—reading an oven manual once and starting to bake bread [5][6][7].

We believe our path is a feasible road to AGI.

Advantage 1: Can handle tasks that cannot be massively trial-and-errored, such as autonomous driving, domestic assistance, elderly care, child companionship, and "industrial, agricultural, military, and commercial" work. Because we are "human-like" AI capable of general decision-making and skill learning through language, while current large models cannot handle these matters!

Advantage 2: Solves the "hallucination" problem. Large models only have "common expressions" obtained through local statistics, without factual mem-

ory. Our solution first stores memory, then extracts common information from memory. Therefore, we inherently include a "fact database" integrated with knowledge.

Advantage 3: Can directly learn and imitate skills through language. Because we have not destroyed the temporal/spatial relationships of token combinations, the token temporal/spatial relationships represented by language can be understood and imitated! This is something large models cannot achieve now or in the future! For example, on a robot's first day at a bakery, it can ask the boss for an oven manual, read it once, and directly start "baking bread" without separate training!

Advantage 4: Safer! (1) Current AI is single-goal, which in decision-making terms is "by any means necessary" "single-minded AI." Such AI does not consider anything beyond the goal, and decisions are black boxes. Imagine how dangerous it would be if a "single-minded" person controlled your life! If such AI comprehensively controlled human life, it could cause immeasurable disasters out of goodwill due to misunderstanding! (2) In our solution, machine "need types" can be pre-configured, values can be trained to align with human values, and machines always consider various goals comprehensively, avoiding "extreme" behaviors. Moreover, in our solution, decisions are visible, modifiable, and "white-box."

7. Underlying Logic of Our Solution

7.1 Chain Associative Activation Process is the Attention Mechanism

First, we believe the essence of knowledge is information. Human-generated knowledge is an infinitesimal portion of information because human information resolution is limited. The relative spatiotemporal relationship between atom A and atom B on a blade of grass is also information, but we do not recognize it.

During evolution, humans developed token recognition capabilities. Tokens are the minimal information units humans commonly use, such as a straight line. Tokens themselves are "world models," the smallest "world models" for building humanity's grand knowledge edifice. Through evolution, humans developed "pattern recognition" capabilities using tokens as "models" to recognize surrounding information, greatly improving information recognition energy efficiency. This is evolution's gift.

If we arrange all "tokens" of all things from the "Big Bang" to the "present" in spatial and temporal order, we obtain an information tensor containing all knowledge humans possess.

Facing this knowledge treasure trove, if intelligent entities outside our universe want to understand it, they will statistically analyze these tokens. The first question: "How many independent token types do we have?" In our solution, similarity relationships answer this. The second question: "What is the quantity distribution of each token type?" In our solution, repetitiveness answers this.

The third question: "How are tokens arranged?" In our solution, proximity relationships answer this. We can see that in our solution, the chain associative activation process and memory/forgetting mechanisms constitute statistical description of information!

In large model attention mechanisms, token combination correlations are inferred through pairwise token correlations, then larger token combination correlations are inferred through pairwise correlations again. After multiple iterations, correlations between different token combinations can be obtained. The pre-training process uses trial-and-error (deep learning) to find the correct "optimal coordinate basis." With attention mechanism assistance, the obtained "optimal coordinate basis" only targets "common information." This process is essentially Bayesian inference: inferring specific token conditional probabilities from partial known probabilities. In our solution, token correlations are obtained through induction. The chain associative activation process uses pretrained correlations (partial known probabilities) to obtain conditional probabilities of specific token occurrences. The chain associative activation process is the process of finding correlations (attention mechanism inference), while memory/forgetting mechanisms are a form of induction.

7.2 The Core of Attention Mechanism is Creating "Common Sense"

Knowledge is token arrangement patterns; common sense is common token arrangement patterns [16]. The core problem with current large models is that they transform human knowledge (token arrangement patterns) into their own knowledge system (because deep learning destroys original token temporal/spatial relationships, making large model knowledge difficult for humans to understand and imitate). They use their own knowledge system to solve problems, then translate back to humans. Therefore, deep learning destroying original token temporal/spatial relationships means it destroys the original, human-understandable organization form and converts it into a machine-understandable organization form. From the machine's perspective, it retains token organization patterns because it correctly identifies "common information." But from the human perspective, the knowledge it generates cannot directly interconnect and borrow from human-created knowledge.

Because the underlying languages of the two systems cannot communicate, humans find it difficult to endow machines with "innate knowledge" (such as innate needs, innate reward/punishment functions, innate emotion functions), so they can only use post-hoc remedies like RLHF or external knowledge bases to solve partial problems, and can only communicate through "yes" or "no." Such robots can only be "bookish" "nerds" that cannot truly solve problems flexibly.

Therefore, in our solution, the most crucial aspect is establishing "common sense" without destroying the original temporal/spatial organization of tokens and including machines' "subjective common sense."

To establish "common sense" without destroying original token temporal/spatial

organization, we use information tokenization, preserve temporal/spatial information storage, adopt chain associative activation processes, and use memory/forgetting mechanisms to induce chain activation value transmission relationships between tokens. Simultaneously, we imitate the organization form of "common sense" to pre-configure token combinations representing innate needs, innate reward/punishment functions, and innate emotion functions. Then we let machines autonomously decide, self-evolve, and expand memory banks around innate knowledge according to advantage-seeking principles, forming the entire knowledge network and creating "objective common sense" and "subjective common sense."

7.3 We Only Complete One Thing: "Creating Common Sense"

To "create common sense," we must first solve (1) "endowing machines with self-needs." To solve (1), we must first solve (2) "how to create understandable knowledge." To solve (2), we must solve "how to create fully connected knowledge networks without using deep learning." Then we can achieve connection relationships between subjective tokens and objective tokens through the attention mechanism—this is common sense.

Establishing relationships between subjective tokens and objective tokens is the "pre-emptization" + "proceduralization" of reward functions, enabling "general decision-making capability" through "seeking advantages and avoiding disadvantages." Driven by "self-needs," machines can achieve "self-evolution."

7.4 We Build a Baby AI

The idea of "building a baby machine, then lifelong learning and self-growth" has existed for many years, but we are the first team to propose detailed solution steps.

8. A Simple Example

Below, we use an example to illustrate how machines make decisions and respond.

Background: Lao Wang goes on vacation out of town, brings an assistant robot, and checks into a hotel room…

Lao Wang: "Hey…"

Robot: Many tokens in the memory bank are in activated states, but among these activated tokens, no reward symbols exceed activation value A1 (A1 is a preset threshold), and no punishment symbols exceed P1 (P1 is a preset threshold).

It continuously receives external and internal information from sensors, uses low-resolution priority to extract tokens from this information, and stores them in the memory bank. Following the same process, it assigns initial activation values

to these tokens. Because no high-activation reward/punishment symbols exist, according to preset procedures, the assigned activation values are relatively low. Consequently, activation value propagation scope is small in the subsequent chain associative activation process, which completes quickly.

The machine begins updating memory values. Because activated tokens obtain low activation values (due to low initial activation values and small propagation scope), their memory value increments are small, and much information is forgotten quickly. Meanwhile, because reward/punishment symbols in the memory bank obtain relatively low activation values—meaning potential rewards and punishments are relatively small—the machine's optimal decision path is to continue receiving information. This is because consuming power itself is a punishment; without obtaining rewards, the optimal decision is to not waste power.

After each chain associative activation process, the machine must check whether any reward or punishment symbols exceed preset thresholds. In this situation, the machine's optimal response is: use low resolution to extract tokens from this information and store them in the memory bank. The above process cycles.

Suddenly, the audio processing system transmits a series of audio tokens (still low-resolution extracted). These tokens, following the same process, are assigned relatively low initial activation values and undergo chain associative activation. Some tokens in this input, due to similarity, activate many similar tokens in the memory bank during chain propagation. These tokens have tight activation value transmission relationships with many reward/punishment symbols, causing many reward/punishment symbols to be activated in this activation value chain propagation. (These tokens are usually the owner's voiceprint features, such as unique pitch.)

Because many reward/punishment symbols obtain activation values exceeding presets this time, assume N reward symbols and M punishment symbols exceed thresholds. The machine autonomously and simultaneously establishes N+M goals using the N reward symbols and M punishment symbols as targets. Therefore, in our solution, goals are autonomously generated by the machine, with multiple goals generated simultaneously, rather than humans presetting a single overall reward function.

In our solution, all machine responses follow the principle of seeking advantages and avoiding disadvantages. Therefore, after creating N+M goals, the machine plans its response path with the principle: increase the probability of reward symbol activation values occurring and decrease the probability of punishment symbol activation values occurring. Therefore, machine decision-making revolves around achieving rewards and avoiding punishments.

The machine first processes those reward/punishment tokens with highest activation values, possibly one or multiple punishment tokens. In the memory bank, the propagation path transmitting activation values to this punishment token may be: owner's voiceprint underlying feature input activates many owner's

voiceprint features in memory through similarity activation; these activation values further propagate activation values in the memory bank.

Among these memories, one punishment token has high activation value. High-activation tokens are 无非几种 cases: (1) This punishment token has high memory value, possibly because it had high activation value when stored (memory value increment is positively correlated with activation value), or because it is frequently activated and obtains high memory value through repetition. (2) Multiple input tokens transmit activation values to this punishment token through different paths. For example, owner's "tone tokens," "word tokens," "owner's state tokens," "owner's expression tokens," "current environment-related tokens," etc. If these tokens all have tight activation relationships with similar punishment symbols in memory, after completing activation value chain propagation together, tokens related to all of them may obtain high activation values. (3) This punishment token has tight activation value transmission relationships with specific input tokens, meaning they always co-occur in memory, forming "proximity relationships" and "high memory value relationships" with short propagation paths and high transmission coefficients. Therefore, the attention mechanism can both use comprehensive reasoning (multiple tokens transmit activation values to specific reward/punishment symbols, integrating experience) and special case reasoning (specific tight activation value transmission paths, specific experiences) to form attention mechanism reasoning for information.

High activation values of reward/punishment tokens may also come from activation value distributions established by previous token inputs. Although activation values of high-activation tokens decay over time, if activation values are sufficiently high, they influence machine decisions for longer periods—similar to humans.

In this example, the propagation network composed of activation value transmission paths contains too many tokens to describe. However, language symbols usually have the highest activation values (because they are most frequently used and have highest memory values). If these language symbols are combined according to their temporal/spatial order, the gist may be "don't lie down (antecedent), got scolded by owner, feel sad (consequence)."

Thus, the machine immediately begins searching for the optimal response path to avoid the probability of this punishment symbol occurring. The machine's decision-making principle is to increase the probability of reward symbol occurrence and decrease punishment symbol probability. The specific method is: reduce the occurrence probability of concepts on paths transmitting activation values to punishment symbols; increase the occurrence probability of concepts on paths transmitting activation values to reward symbols. How to specifically increase or decrease probabilities for each concept? Each concept is a locally tight network in the memory bank. The machine needs to reduce the occurrence probability of high-activation tokens in this local tight network, thereby reducing the occurrence probability of this reward/punishment logic link.

For example, in the machine's memory, when the owner "used similar tokens to scold," the memory stored its internal sensor data at that time and external sensor data. Some tokens were forgotten because they didn't repeat later and didn't obtain enhanced memory. However, token combinations that could repeatedly co-occur with this "punishment symbol" all contained "lying down"-related tokens, some "representing specific time tokens," and "representing specific occasion tokens." Due to repetitiveness, they obtained higher memory values. Because they are repeatable combinations, each occurrence mutually pushes up the other's activation values, obtaining memory values far higher than repetitiveness. Moreover, because they are repeatable, their combinations can obtain higher activation values each time, making them more easily activated and more likely to be remembered—this is a positive feedback loop. This is the experience summarization process.

If once, in a similar environment, the owner praised the machine, such memories would also participate in subsequent decision-making. Therefore, in similar environments, various tokens may transmit activation values to both punishment and reward symbols. The machine's decision-making comprehensively statistics all reward/punishment values, considering both obtaining rewards and avoiding punishments. Therefore, when selecting response paths, some local response paths may be both paths to rewards and punishments, requiring the machine to subdivide these paths to determine which lead to rewards and which to punishments. This subdivision process adds more tokens to form multiple subdivided paths (e.g., different scenarios, time points, or antecedents), enabling the machine to determine its response through subdivided paths—this is the core of segmented imitation.

Therefore, our machine does not need to accommodate new "Fine-tuning" by modifying past parameters. It only needs to accumulate memory to achieve "Fine-tuning." It can perform any depth of "Fine-tuning," any domain of "Fine-tuning," and even infinite domain superimposed "Fine-tuning" without "catastrophic forgetting" because it does not modify past knowledge parameters but simply expands the network.

In this example, assuming it's daytime and the machine is lying down (saving power to obtain rewards), when the owner's voiceprint activates punishment symbols, the machine needs to avoid the probability of activated punishment symbols and increase the probability of activated reward symbols. Here, there are at least two cases: (1) Reduce the occurrence probability of the "lying down" concept to avoid punishment (e.g., being scolded); (2) Increase the occurrence probability of the "lying down" concept to obtain rewards (e.g., saving power). At this point, the machine needs to make optimal choices according to the advantage-seeking principle, comprehensively considering various response paths and comparing statistical reward/punishment values.

Assuming the machine's battery is sufficient and power-saving brings small rewards, after completing the chain associative activation process, only one punishment symbol obtains high activation value. The machine, following the

advantage-seeking principle, will choose to avoid punishment because this yields the highest statistical reward value. Therefore, driven by benefit maximization, the machine will set avoiding punishment as the goal and begin establishing responses.

Assuming the machine's battery is insufficient and power-saving brings large rewards (here assuming the machine must lie down to charge), after completing the chain associative activation process, one punishment symbol obtains high activation value, and one reward symbol also obtains high activation value. The machine, following the advantage-seeking principle, will simultaneously establish two goals: achieve rewards and avoid punishment, because this yields the highest statistical reward value. Therefore, driven by benefit maximization, the machine will set obtaining rewards + avoiding punishment as goals and begin establishing responses.

Assuming the machine's battery is sufficient, it now creates a second-level goal: reduce the activation value obtained by the "lying down" concept. Under the constraint of the second-level goal, the machine searches for propagation paths transmitting activation values to the "lying down" concept and creates third-level goals: reduce activation values of concepts on these propagation paths. The machine discovers the main path transmitting activation values to the "lying down" concept is a set of self-state sensor inputs. Therefore, the machine creates third-level goals: reduce the probability of occurrence of these input tokens.

The machine records various internal/external parameters from each training session, optimizing them through memory and forgetting mechanisms. Through reward/punishment feedback, it encourages imitation of parameters that obtained rewards and avoids parameters that obtained punishments. Through this method, parameter combinations + rewards/punishments + internal/external environments establish experiential connection relationships. This is essentially a reinforcement learning process. Of course, humans can also imitate this form to implant innate knowledge (drive-related) into machines or directly modify machine knowledge using accumulated human experience to accelerate convergence.

Thus, in different environments, environmental tokens automatically activate the most relevant memories. By imitating these experiences, the machine transmits similar parameter combinations (including parameter types and their temporal order, all automatically completed) to its motion system. This enables the machine to stand up in various environments, reducing the occurrence probability of "lying down"-related tokens.

Assuming the machine's battery is insufficient, the machine's experience of achieving rewards will keep it lying down, increasing the realization probability of charging-related tokens. To avoid punishment, it will imitate past experience to explain its reasons to the owner. The machine then creates a second-level goal: increase the activation value obtained by the "charging" concept. It imitates past experience of avoiding "punishment." Therefore, the machine may create a

third-level goal: “explain its behavior reasons to the owner,” because such “token combinations” have tight activation value transmission relationships with “avoid punishment” token combinations in memory. Therefore, the machine’s goal is to increase the occurrence probability of specific token combinations (explaining behavior reasons to owner). Therefore, the next-level decision-making related token combinations are: language organization-related experiences will be activated.

This process iterates continuously, with new reward/punishment symbols activated each time. The machine statistically analyzes these reward/punishment symbol activation values until discovering convergence. At this point, the machine has established the optimal response path.

Then, the machine enters the imitation execution process. The machine’s decision path needs iterative decomposition down to underlying drive parameters before it can issue drive commands by imitating parameter configurations from experience, thereby imitating execution.

In reality, experience and reality can only partially match, so experience and reality can only achieve generalization by imitating their shared token combination patterns.

Among these paths, those composed of high-activation tokens are top-level imitation paths. If imitation paths do not contain direct underlying drive command combinations, adding more tokens (lower-activation tokens) transforms imitation paths into different combination forms of multi-segment paths composed of more tokens. This is the meaning of segmented imitation.

In other words, when facing a large path without suitable direct imitation experience, we refine and decompose it into multiple small response path segments. For each small segment, we re-seek suitable experiences to generalize. If still not decomposable to direct underlying drive command combinations, repeat this process by adding more tokens to decompose the response path into more small segments, then seek suitable experiences to generalize. If still not decomposable, repeat until decomposition into direct underlying drive command combinations.

The above process iterates continuously. New tokens may input at any moment. Whenever new tokens input, the machine must re-perform chain associative activation. After completion, activation value distributions in the memory bank change, requiring the machine to re-perform decision-making. Therefore, during this process, the machine’s optimal decision may be to abandon current partial goals and begin pursuing newly generated goals.

Thus, our machine generates its own goals and can continuously change them, making its decisions highly flexible and constantly matching the environment.

Therefore, in the above example, the machine’s possible execution result is: immediately stand up, simultaneously increase the resolution of the sound processing system, and turn its head to observe the owner’s posture, movements,

and expressions. However, at this moment, the owner may have just said the word "Hey…" and hasn't begun the following words.

Therefore, our machine is human-like intelligence. Its understanding of information comes from its own experiences, not statistical processes. Only in this way can our machine provide personalized services.

A thousand housewives have a thousand different requirements. AI obtained through knowledge statistics and robots that cannot update knowledge in real-time can never enter households or win housewives' hearts. Their application scenarios will be very limited, while our solution is true general artificial intelligence that may change the world's appearance.

9. Conclusion

We believe AI development can be roughly divided into stages: (1) "Feature exploration" stage. Before deep learning, it focused on "manual exploration." After deep learning, it focused on "machine exploration." (2) After achieving true attention (Transformer), because machines' "knowledge coordinate basis cluster" and humans' "knowledge coordinate basis cluster (concepts)" initially aligned, machines achieved "knowledge generalization." Facing human tasks, machines can exhibit certain capabilities through "knowledge generalization." One-dimensional attention mechanisms brought language large models. Two-dimensional attention mechanisms brought image generalization. Three-dimensional attention mechanisms can achieve 3D creation capabilities. Four-dimensional (3D+time) attention mechanisms can achieve dynamic process generalization: bringing video generation and robot services in limited scenarios.

However, we believe only by adding "vitality: the fifth dimension, self-needs" can machine intelligence be given a true "soul." The path taken by large models 注定了 it cannot achieve the "fifth dimension." Our solution can endow machines with "life," making it possible to become true "general artificial intelligence."

Therefore, we believe AI needs to develop to the next stage: the "autonomous interaction" stage. "Autonomous" means machines are no longer silent "machines"—they can spontaneously generate behaviors (equivalent to self-programming) and self-explore knowledge (e.g., actively interacting with the environment to obtain knowledge). "Interaction" means machines can interact with the environment in real-time, update their knowledge in real-time, and make continuous decisions to complete complex tasks in unfamiliar environments.

Many renowned scholars have proposed their views on how to achieve true AGI. For example, Professor LeCun proposed the "world model," and Professor Songchun Zhu proposed four characteristics for AGI: (1) Ability to perform infinite tasks; (2) Ability to autonomously generate new tasks; (3) Value system-driven; (4) Possessing world models reflecting the real world. Clearly, our solution responds to the ideas of Professors LeCun and Zhu.

General artificial intelligence is the original aspiration and crown jewel of AI. We propose a technical solution for AGI, including step-by-step implementation steps. In references [25][26][27][28], we reveal in detail through patents the technical steps to implement this path. It may be a correct path guiding humanity toward AGI.

References

[1] A Generalist Agent, Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Giménez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar and Nando de Freitas, https://openreview.net/forum?id=1ikK0kHjvj

[2] ChatGPT's Impact on Scientific Research and Literature Intelligence Work, Zhixiong Zhang, Li Qian, Jing Xie, et al., chinaXiv:

[3] Ouyang L, Wu J, Jiang X, et al. Training language models to follow instructions with human feedback[J].

[4] Vinyals O, Ewalds T, Bartunov S, et al. Starcraft ii: A new challenge for reinforcement learning[J]. arXiv:1708.04782, 2017.

[5] A Survey of Learning-based Automated Program Repair, Antonio Mastropaolo, arXiv:2203.02155, 2022

[6] Jakob Survey Simone Matteo Weisong Yuxiang Program Emanuela Zhe Gan, Guglielmo Ciniselli, Uszkoreit, Pascarella, Rocco Oliveto, Jianfeng Wang, Shuohang Wang, Jordan Boyd-Graber, Chunrong Automated Zhengyuan Lijuan Wang, Gabriele Bavota, arXiv:2302.00438 Learning-based Fang, GPT-3 To Be Reliable, Chenglei Si, Repair, Quanjun Zhenyu https://jina.ai/news/auto-gpt-unmasked-hype-hard-truths-production-

[7] An experimental open-source attempt to make GPT-4 fully autonomous, Attention Is All You Need, Ashish Vaswani, Aidan https://github.com/Torantulino/Auto-GPT?ref=jina-ai-gmbh.ghost.io Auto-GPT - The next evolution of data driven Chat AI, https://auto

[8] 文心一言: https://mp.weixin.qq.com/s/0-8X9FPouteKzNiK6DPaiA

[9] Learned in translation: Contextualized word vectors. In Advances in Scalabrino,

[10] A Zhang, Chen, arXiv:2301.03270

[11] Introducing ChatGPT, https://openai.com/blog/chatgpt/

[12] Prompting Yang, arXiv:2210.09150 https://GitHub.com/Significant-Gravitas/Auto-GPT pitfalls/ -gpt.ai

[13] Neural Information Processing Systems Parmar, Kaiser, Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. arXiv preprint arXiv:1601.06733, 2016.

[14] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio, A structured self-attentive sentence embedding. arXiv:1703.03130, 2017.

[15] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model. In Empirical Methods in Natural Language Processing, 2016.

[16] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. arXiv preprint arXiv:1705.04304, 2017.

[17] Tamkin A, Brundage M, Clark J, et al. Understanding the capabilities, limitations, and societal impact of large language models[J]. arXiv:2102.02503,

[18] Roy, arXiv:2107.03508 Dialogue. https://openai.com/blog/chatgpt/, 2023

[19] instructions with human feedback[J]. arXiv:2203.02155, 2022

[20] Hutter, arXiv:1111.6117

[21] A method for implementing general artificial intelligence, Yongcong Chen, Ting Zeng, Xingyue Chen, Chinese Patent

[22] Ouyang L, Wu J, Jiang X, et al. Training language models to follow Principles of Solomonoff Induction and Illia Polosukhin, arXiv:1706.03762

[23] BERMo: What can BERT learn from Noam Shazeer, Gomez, ELMo? Sangamesh Kodge, AIXI, Peter Sunehag, Lukasz Optimizing ChatGPT:

[24] Language Kaushik OpenAI. Marcus Models Jones, Llion

[25] A method for imitating human intelligence in machine intelligence implementation, Yongcong Chen, Ting Zeng, Xingyue Chen, Chinese Patent

[26] A method for implementing human-like general artificial intelligence machines, Yongcong Chen, Ting Zeng, Xingyue Chen, Chinese Patent CN111553467B

[27] CN111563575B

[28] CN112215346B Yongcong, Zhang Jun, Zeng Ting, Chen Xingyue, US Patent No. 11,715,291 ESTABLISHMENT OF GENERAL-PURPOSE ARTIFICIAL INTELLIGENCE SYSTEM, Chen

Note: Figure translations are in progress. See original paper for figures.