Precision-Weighted Predictive Coding with Adaptive Re-entrant Inference in Neural Networks

Open research specification 2 of 7

Krrish Choudhary

Verace Pvt. Ltd.

This document was first drafted in the form of a patent specification, which is why it speaks of an invention and of claims. No patent application was filed and none is intended. Where the text mentions a related or co-pending patent application, it means a companion document in this set, and none of them was filed. The work is published as open research.

Published under the PolyForm Noncommercial License 1.0.0

Copyright (c) 2026 Verace Pvt. Ltd. (https://verace.in)

Contents

1 FIELD **3**

2 BACKGROUND **3**

2.1 Prior Art in Neural Network Uncertainty 3

2.2 Prior Art in Predictive Coding 4

2.3 Need for the Present Invention 5

3 SUMMARY **5**

3.1 Content-Dependent Precision Estimation 5

3.2 Depth-Scaled Precision Initialisation 6

3.3 Multi-Scale Precision Bands 6

3.4 Re-entrant Predictive Coding Inference 7

3.5 Conflict-Driven Adaptive Depth 7

3.6 Analytical Precision Learning 7

4 DETAILED DESCRIPTION **8**

4.1 System Overview 8

4.2 Precision Estimation Module 9

4.2.1 Architecture 9

4.2.2 Initialisation of the Weight Matrix 9

4.2.3 Depth-Scaled Bias Initialisation 10

4.2.4 Sigma-Squared Adaptive Dendritic Modulation 10

4.3 Multi-Scale Precision Bands 10

4.4 Prediction Generation Module 11

4.5 Precision-Weighted Prediction Error Loss 12

4.6 Re-entrant Predictive Coding Inference Loop 12

4.6.1 Initial Bottom-Up Pass 12

4.6.2 Prediction Error Computation 13

4.6.3 Re-entrant Iterations 13

4.6.4 Gradient-Free Inference 14

4.7 Conflict-Driven Adaptive Depth 14

4.7.1 Conflict Computation 14

4.7.2 Adaptive Step Allocation 14

4.7.3 Health Monitoring 15

4.8 Analytical Precision Learning 15

4.8.1 Derivation of the Analytical Gradient 15

4.8.2 Computational Form 16

4.8.3 Advantages Over Automatic Differentiation 16
4.9 Prediction Weight Update 17
4.10 Integration with Hierarchical Neural Network 17
4.11 Embodiment Parameters 18

5 DESCRIPTION OF THE FIGURES **19**

6 CLAIMS **39**

TITLE

Precision-Weighted Predictive Coding with Adaptive Re-entrant Inference in Neural Networks

ABSTRACT

The present invention discloses a system and method for precision-weighted predictive coding in hierarchical neural networks, wherein each processing unit in a multi-level hierarchy generates both a prediction of its input and a content-dependent precision estimate (σ^2) quantifying the certainty of said prediction. The precision estimate is computed by a learned linear transformation ($\mathbf{W}_\sigma$) applied to the normalised hidden state, passed through a positivity-preserving softplus activation, yielding a per-token, per-level scalar that varies with input content. The system implements depth-scaled initialisation of precision parameters following cortical maturation gradients, wherein shallow processing units begin with lower initial uncertainty than deep processing units. A multi-scale precision band architecture partitions the processing hierarchy into token-level, phrase-level, sentence-level, and discourse-level bands, each feeding distinct downstream subsystems. The system further comprises a re-entrant predictive coding inference loop that iteratively refines representations through top-down prediction and bottom-up error correction under a gradient-free regime, with conflict-driven adaptive depth control that allocates additional inference iterations when inter-unit disagreement exceeds a threshold. All precision parameters are learned via analytical derivative computation without automatic differentiation, using closed-form expressions for the gradient of a precision-weighted prediction error loss through the softplus nonlinearity.

1. FIELD

The present invention relates generally to the field of artificial neural networks and machine learning systems. More particularly, the invention relates to systems and methods for uncertainty estimation and predictive coding in hierarchical neural network architectures, including methods for computing content-dependent precision weights, depth-scaled initialisation thereof, multi-scale precision band organisation, iterative re-entrant inference with adaptive computational depth, and analytical learning rules for precision parameters without automatic differentiation.

2. BACKGROUND

2.1. Prior Art in Neural Network Uncertainty

Contemporary deep neural networks, including transformer-based language models as described by Vaswani et al. (2017), produce point estimates without any measure of certainty or uncertainty associated with their predictions. When a transformer model produces a next-token probability

distribution, there is no mechanism by which the model internally represents *how confident* it is in that distribution, nor can different layers of the model communicate their respective levels of certainty to one another.

This absence of uncertainty estimation creates several fundamental limitations:

(a) **Uniform treatment of predictions.** All predictions from all layers are treated with equal weight regardless of whether the model is processing familiar or novel content. A layer processing a common grammatical pattern and a layer processing a rare technical term contribute identically to downstream computation.

(b) **Fixed computational depth.** Standard neural networks apply the same number of processing steps to every input regardless of difficulty. An unambiguous sentence and a deeply ambiguous sentence receive identical computational resources.

(c) **No per-token precision.** Existing approaches that attempt uncertainty estimation, such as Monte Carlo dropout (Gal & Ghahramani, 2016) or deep ensembles (Lakshminarayanan et al., 2017), compute uncertainty as a post-hoc aggregate over the entire model output rather than as an intrinsic per-token, per-layer signal available during forward computation.

(d) **No depth-dependent initialisation.** In all known neural network architectures, uncertainty parameters (when present) are initialised uniformly across all layers. There is no mechanism by which shallow layers, which process lower-level features, begin with different uncertainty characteristics than deep layers, which process higher-level abstractions.

2.2. Prior Art in Predictive Coding

Rao and Ballard (1999) introduced predictive coding as a computational theory of visual cortex processing, wherein each cortical area generates predictions of the activity in the area below and computes prediction errors. Friston (2005) generalised this into the free energy principle, proposing that the brain minimises a variational free energy bound.

However, existing computational implementations of predictive coding for artificial neural networks suffer from several deficiencies:

(a) **Constant precision.** Prior implementations of predictive coding in neural networks use a fixed, scalar precision term that does not vary with input content. The precision is either a global hyperparameter or a per-layer constant that is the same for every token in a sequence.

(b) **No multi-scale organisation.** Prior predictive coding implementations treat all levels of the hierarchy identically with respect to temporal scale. There is no mechanism by which different bands of the hierarchy specialise for different temporal receptive fields (token-level versus discourse-level).

(c) **Fixed inference depth.** Prior predictive coding implementations run a fixed number of inference iterations regardless of the difficulty or ambiguity of the input, wasting computation on easy inputs and under-allocating on difficult inputs.

(d) **Reliance on automatic differentiation.** Prior implementations of learned precision parameters require backpropagation through the computational graph, which conflicts with the local learning principle of predictive coding and incurs substantial memory overhead from storing activations for the backward pass.

2.3. Need for the Present Invention

There exists a need for a neural network architecture that:

(i) computes *content-dependent* precision estimates that vary with input;

(ii) initialises precision parameters with a depth-dependent gradient reflecting the principle that shallow processing units should start less uncertain than deep ones;

(iii) organises precision into multi-scale bands aligned with different temporal receptive fields;

(iv) employs iterative re-entrant inference whose depth adapts to input difficulty;

(v) learns all precision parameters via analytical derivatives without automatic differentiation.

The present invention addresses all of the foregoing limitations.

3. SUMMARY

The present invention provides a system and method for precision-weighted predictive coding with adaptive re-entrant inference in hierarchical neural networks. The invention comprises six interrelated innovations that collectively enable content-dependent uncertainty estimation, biologically-informed initialisation, multi-scale precision organisation, adaptive inference depth, and autograd-free precision learning.

3.1. Content-Dependent Precision Estimation

A first aspect of the invention provides a precision estimation module within each processing unit of a hierarchical neural network. Unlike prior art systems that employ a fixed or constant precision value, the present invention computes a content-dependent precision estimate σ^2 for each token at each hierarchical level by applying a learned linear transformation $\mathbf{W}_\sigma$ to the normalised hidden state:

$$z = \mathbf{W}_\sigma \cdot \text{RMSNorm}(h) + b_\sigma \tag{1}$$

$$\sigma^2 = \text{softplus}(z) + \epsilon \tag{2}$$

where h is the hidden state of the processing unit, RMSNorm is root-mean-square layer normalisation, b_σ is a bias term, $\text{softplus}(x) = \ln(1 + e^x)$ is a positivity-preserving activation function, and ϵ is a small constant (e.g., 10^{-6}) preventing numerical degeneracy. The weight matrix $\mathbf{W}_\sigma$ is initialised with small non-zero random values (standard deviation ≈ 0.01), ensuring that σ^2 is content-dependent from the first forward pass. A zero-initialised $\mathbf{W}_\sigma$ would render σ^2 equal to $\text{softplus}(b_\sigma)$ for all inputs — a constant precision, analogous to the neurological condition of anosognosia wherein a patient cannot distinguish between states of certainty and uncertainty.

3.2. Depth-Scaled Precision Initialisation

A second aspect of the invention provides a depth-dependent initialisation scheme for the precision bias term b_σ. For a hierarchy of N processing units indexed $i = 0, 1, \ldots, N-1$, the initial precision value for unit i is:

$$\sigma^2_{\text{init},i} = \sigma^2_{\text{init}} \times \left(1 + \alpha_{\text{depth}} \times \frac{i}{\max(N-1, 1)}\right) \tag{3}$$

where σ^2_{init} is a base initial precision (e.g., 1.0) and α_{depth} is a depth scaling factor (e.g., 1.0). The bias b_σ is then set to $\text{softplus}^{-1}(\sigma^2_{\text{init},i}) = \ln(e^{\sigma^2_{\text{init},i}} - 1)$. This produces a monotonically increasing initial uncertainty from shallow to deep units, reflecting the empirically observed cortical maturation gradient (Huttenlocher, 1990): primary sensory areas mature earlier and operate with lower initial uncertainty, while associative areas mature later and begin with greater uncertainty.

3.3. Multi-Scale Precision Bands

A third aspect of the invention organises the N processing units into four precision bands, each specialising for a different temporal receptive field:

- **Token band** (units 0 to $q - 1$): σ^2_{token}, precision over individual tokens.
- **Phrase band** (units q to $2q - 1$): σ^2_{phrase}, precision over local phrases.
- **Sentence band** (units $2q$ to $3q - 1$): $\sigma^2_{\text{sentence}}$, precision over sentence-level context.
- **Discourse band** (units $3q$ to $N - 1$): $\sigma^2_{\text{discourse}}$, precision over discourse-level coherence.

where $q = \max(1, \lfloor N/4 \rfloor)$. Each band's aggregate precision feeds a distinct downstream subsystem: the token band informs local prediction, the sentence band modulates episodic memory encoding, and the discourse band guides global workspace broadcasting. This multi-scale organisation mirrors the hierarchy of temporal receptive fields observed in the human cortex (Hasson et al., 2008).

3.4. Re-entrant Predictive Coding Inference

A fourth aspect of the invention provides an iterative re-entrant inference loop that refines representations through top-down prediction and bottom-up error correction. After an initial bottom-up pass through the hierarchy, the system performs K additional inference iterations (where K ranges from a configurable minimum to a configurable maximum), each comprising:

(i) top-down prediction generation from each unit to the unit below;

(ii) bottom-up error computation: $\epsilon_l = \text{target}_l - \text{prediction}_l$;

(iii) re-computation of lateral and global workspace signals;

(iv) convergence testing: $|\Delta E|/(E_{\text{prev}} + \epsilon) < \tau_{\text{conv}}$.

All inference iterations execute under a gradient-free regime (no computational graph is constructed), consistent with the biological principle that inference (perception) and learning (synaptic modification) are separate processes.

3.5. Conflict-Driven Adaptive Depth

A fifth aspect of the invention provides an adaptive depth mechanism that allocates additional inference iterations based on inter-unit conflict. A metacognitive monitoring module computes the variance across the output representations of all processing units:

$$C = \text{Var}_{i \in \{0,\ldots,N-1\}} \left[\bar{h}_i\right] \tag{4}$$

where $\bar{h}_i$ is the mean output of unit i across the batch and sequence dimensions. When C exceeds a conflict threshold τ_C, additional inference steps are allocated:

$$K_{\text{extra}} = \min\left(K_{\text{max,extra}}, \left\lfloor \frac{C - \tau_C}{\tau_C} \times K_{\text{max,extra}} \right\rfloor\right) \tag{5}$$

This implements a form of metacognitive resource allocation: the system "thinks harder" when its processing units disagree, analogous to the function of the anterior cingulate cortex in conflict monitoring (Botvinick et al., 2001). Removal of this mechanism yields a system that applies uniform computation to all inputs regardless of difficulty, analogous to the neurological condition of anosognosia.

3.6. Analytical Precision Learning

A sixth aspect of the invention provides a method for updating the precision parameters $\mathbf{W}_\sigma$ and b_σ using analytically derived gradients, without invoking automatic differentiation. The

precision-weighted prediction error loss for a single level is:

$$\mathcal{L}_l = \frac{\|\epsilon_l\|^2}{2 \cdot \sigma_l^2} + \frac{1}{2} \ln(\sigma_l^2) \tag{6}$$

The derivative with respect to σ^2 is:

$$\frac{\partial \mathcal{L}}{\partial \sigma^2} = \frac{-\|\epsilon\|^2}{2 \cdot (\sigma^2)^2} + \frac{1}{2 \cdot \sigma^2} \tag{7}$$

The chain rule through the softplus activation yields the derivative with respect to the pre-activation z:

$$\frac{\partial \mathcal{L}}{\partial z} = \frac{\partial \mathcal{L}}{\partial \sigma^2} \times \text{sigmoid}(z) \tag{8}$$

where $\text{sigmoid}(z) = 1/(1 + e^{-z})$ is the derivative of softplus. The weight update is then:

$$\Delta \mathbf{W}_\sigma = \frac{1}{N} \sum_{n=1}^{N} \frac{\partial \mathcal{L}_n}{\partial z_n} \cdot \text{RMSNorm}(h_n)^\top \tag{9}$$

$$\Delta b_\sigma = \frac{1}{N} \sum_{n=1}^{N} \frac{\partial \mathcal{L}_n}{\partial z_n} \tag{10}$$

All computations are performed under a no-gradient regime, eliminating the memory overhead of storing activations for backpropagation and reducing the computational cost of the learning step.

4. DETAILED DESCRIPTION

The following detailed description, taken in conjunction with the accompanying drawings, describes preferred embodiments of the present invention. It is to be understood that the invention is not limited to these specific embodiments but extends to all alternatives, modifications, and equivalents as defined by the appended claims.

4.1. System Overview

Referring to Figure 1, the system of the present invention comprises a hierarchical neural network (100) having a plurality of processing units (110-1, 110-2, …, 110-N) arranged in a feedforward hierarchy. Each processing unit receives input from the unit below and produces output for the unit above. In a preferred embodiment, the processing units are cortical column modules as described in a co-pending patent application by the same applicant. However, the precision-weighted predictive coding system described herein is applicable to any hierarchical neural network architecture, including transformer layers, recurrent modules, or convolutional stages.

The system further comprises:

- a precision estimation module (120) within each processing unit;

- a prediction generation module (130) within each processing unit;
- a re-entrant inference controller (140) that orchestrates iterative refinement;
- a conflict detection module (150) that monitors inter-unit agreement;
- an analytical precision learning module (160) that updates precision parameters.

4.2. Precision Estimation Module

Referring to Figure 2, the precision estimation module (120) resides within each processing unit (110-i) and computes a per-token precision estimate $\sigma_i^2(t)$ for each position t in the input sequence.

4.2.1. Architecture

The precision estimation module comprises:

(a) A normalisation layer (121), specifically a root-mean-square normalisation (RMSNorm) operation, that normalises the hidden state $h_i \in \mathbb{R}^{B\times T\times D_{\text{col}}}$ to produce $\hat{h}_i = \text{RMSNorm}(h_i)$.

(b) A learned linear projection (122), denoted $\mathbf{W}_\sigma \in \mathbb{R}^{1\times D_{\text{col}}}$ with bias $b_\sigma \in \mathbb{R}^1$, that maps the normalised hidden state to a scalar pre-activation:

$$z_i(t) = \mathbf{W}_\sigma \cdot \hat{h}_i(t) + b_\sigma \tag{11}$$

(c) A positivity-preserving activation (123), specifically the softplus function $\text{softplus}(z) = \ln(1 + e^z)$, that ensures the output is strictly positive:

$$\sigma_i^2(t) = \text{softplus}(z_i(t)) + \epsilon \tag{12}$$

where $\epsilon = 10^{-6}$ prevents numerical degeneracy.

4.2.2. Initialisation of the Weight Matrix

The weight matrix $\mathbf{W}_\sigma$ is initialised by sampling from a zero-mean Gaussian distribution with standard deviation $\sigma_{\text{init}} = 0.01$:

$$\mathbf{W}_\sigma[j] \sim \mathcal{N}(0, \sigma_{\text{init}}^2), \quad j = 1, \ldots, D_{\text{col}} \tag{13}$$

This non-zero initialisation is critical. If $\mathbf{W}_\sigma$ were initialised to zero, the precision estimate would reduce to $\sigma^2 = \text{softplus}(b_\sigma) + \epsilon$ for all inputs, rendering precision content-independent. The small but non-zero initialisation ensures that different hidden states produce different precision values from the first forward pass, enabling the system to distinguish between inputs it finds familiar (low σ^2) and inputs it finds novel (high σ^2).

4.2.3. Depth-Scaled Bias Initialisation

The bias term b_σ is initialised differently for each processing unit in the hierarchy. For unit i in a hierarchy of N units:

$$r_i = \frac{i}{\max(N-1, 1)} \tag{14}$$

$$\sigma^2_{\text{init},i} = \sigma^2_{\text{base}} \times (1 + \alpha_{\text{depth}} \times r_i) \tag{15}$$

$$b_{\sigma,i} = \begin{cases} \ln(e^{\sigma^2_{\text{init},i}} - 1) & \text{if } \sigma^2_{\text{init},i} > 0.01 \\ 0.0 & \text{otherwise} \end{cases} \tag{16}$$

In a preferred embodiment, $\sigma^2_{\text{base}} = 1.0$ and $\alpha_{\text{depth}} = 1.0$, yielding:

- Unit 0 (shallowest): $\sigma^2_{\text{init},0} = 1.0$, $b_{\sigma,0} \approx 0.541$
- Unit $N-1$ (deepest): $\sigma^2_{\text{init},N-1} = 2.0$, $b_{\sigma,N-1} \approx 1.313$

This depth-dependent initialisation reflects the principle that shallow processing units, which operate on lower-level features (analogous to primary sensory cortex), should begin with lower uncertainty, while deep processing units, which operate on higher-level abstractions (analogous to prefrontal cortex), should begin with higher uncertainty. This mirrors the empirically observed cortical maturation timeline described by Huttenlocher (1990), wherein primary sensory areas achieve adult-level synaptic density before associative areas.

4.2.4. Sigma-Squared Adaptive Dendritic Modulation

In a preferred embodiment, the precision estimate from a preceding processing unit is used to modulate the dendritic computation of the current processing unit. A linear projection $\mathbf{W}_{\sigma\text{-adapt}} \in \mathbb{R}^{D_{\text{ff}} \times 1}$ maps the preceding unit's σ^2 into the feedforward dimension, where it additively modulates the dendritic gate logit:

$$g_{\text{dend}} = g_{\text{dend}} + \mathbf{W}_{\sigma\text{-adapt}} \cdot \sigma^2_{\text{prev}} \tag{17}$$

This enables a processing unit to adjust its computation based on how certain the preceding unit is — high preceding uncertainty loosens the dendritic gate, allowing more information through; low preceding uncertainty tightens it.

4.3. Multi-Scale Precision Bands

Referring to Figure 3, the N processing units of the hierarchy are organised into four precision bands, each corresponding to a different temporal receptive field:

$$q = \max\left(1, \left\lfloor \frac{N}{4} \right\rfloor\right) \tag{18}$$

Band	Units	Temporal Scale
Token	0 to $q-1$	Individual token features
Phrase	q to $2q-1$	Local phrase structure
Sentence	$2q$ to $3q-1$	Sentence-level context
Discourse	$3q$ to $N-1$	Discourse-level coherence

The aggregate precision for each band is the mean σ^2 across all units in that band:

$$\sigma^2_{\text{band}} = \frac{1}{|\text{band}|} \sum_{i \in \text{band}} \sigma^2_i \tag{19}$$

Each band's aggregate precision feeds a distinct downstream subsystem:

(a) The **token band** precision modulates local next-token prediction confidence, influencing how strongly the model's immediate prediction is weighted.

(b) The **phrase band** precision modulates lateral column communication, controlling the strength of horizontal information flow between processing units.

(c) The **sentence band** precision modulates episodic memory encoding thresholds, determining which representations are sufficiently surprising (high σ^2) or sufficiently certain (low σ^2) to be committed to episodic memory.

(d) The **discourse band** precision modulates global workspace broadcasting, controlling which processing units gain access to the shared global workspace based on their discourse-level certainty.

In a preferred embodiment with $N = 12$ processing units, $q = 3$, yielding bands of 3 units each.

4.4. Prediction Generation Module

Each processing unit (110-i) contains a prediction generation module (130) comprising a learned linear projection:

$$\hat{x}_i = \mathbf{W}_{\text{predict}} \cdot \hat{h}_i \tag{20}$$

where $\mathbf{W}_{\text{predict}} \in \mathbb{R}^{D_{\text{model}} \times D_{\text{col}}}$ maps the normalised hidden state to the inter-unit dimension. This prediction represents unit i's expectation of the input it will receive from unit $i-1$ (or from the embedding layer for unit 0).

$\mathbf{W}_{\text{predict}}$ is initialised to zero, meaning that at initialisation no unit generates any prediction. This ensures that prediction errors are maximal at the start of training, driving rapid initial learning.

The prediction error at each level is:

$$\epsilon_l = \text{target}_l - \hat{x}_l \tag{21}$$

where target_l is the actual output of unit $l-1$ (detached from the computational graph) for $l > 0$, or the embedding output for $l = 0$.

4.5. Precision-Weighted Prediction Error Loss

The precision-weighted prediction error loss for a single level l is:

$$\mathcal{L}_l = \frac{1}{2 \cdot \sigma_l^2} \|\epsilon_l\|^2 + \frac{1}{2} \ln(\sigma_l^2) \tag{22}$$

This loss has a natural Bayesian interpretation as the negative log-likelihood of a Gaussian distribution with mean equal to the prediction and variance equal to σ^2. The first term penalises large prediction errors, weighted inversely by precision. The second term penalises over-confidence (low σ^2 when errors are large), preventing the trivial solution of setting $\sigma^2 \to \infty$ to minimise the first term.

The total predictive coding loss across all N levels is:

$$\mathcal{L}_{\text{PC}} = \sum_{l=0}^{N-1} \mathcal{L}_l \tag{23}$$

At equilibrium, the optimal σ^2 for each level equals the mean squared prediction error at that level:

$$\sigma^{2*}_l = \mathbb{E}\left[\|\epsilon_l\|^2\right] \tag{24}$$

This means the precision naturally calibrates itself: levels with large errors develop high σ^2 (low precision, low influence), while levels with small errors develop low σ^2 (high precision, high influence).

4.6. Re-entrant Predictive Coding Inference Loop

Referring to Figure 4, the re-entrant inference controller (140) orchestrates an iterative refinement process after the initial bottom-up forward pass.

4.6.1. Initial Bottom-Up Pass

The initial bottom-up pass propagates input through all N processing units sequentially:

$$h_0 = f_0(x), \quad h_i = f_i(h_{i-1}), \quad i = 1, \ldots, N-1 \tag{25}$$

where f_i denotes the forward function of unit i. During this pass, each unit produces:

- an output h_i for the next unit;
- a prediction $\hat{x}_i$ of its input from the unit below;
- a precision estimate σ_i^2 of its prediction certainty.

4.6.2. Prediction Error Computation

After the bottom-up pass, prediction errors are computed for all levels:

$$\epsilon_l = \text{target}_l - \hat{x}_l, \quad l = 0, 1, \ldots, N-1 \tag{26}$$

4.6.3. Re-entrant Iterations

The system then performs up to $K_{\text{max}} - 1$ additional inference iterations (one pass having already been completed). Each iteration proceeds as follows:

Step 1: Top-down context assembly. For each unit i, a top-down context signal is assembled from:

- the global workspace modulation signal;
- the lateral inter-unit communication signal;
- the prediction from unit $i+1$ (for $i < N-1$).

Step 2: Forward pass with top-down context. Each unit re-processes its input with both the bottom-up input (routed through a thalamic relay module) and the assembled top-down context:

$$h_i^{(k)} = f_i\left(\text{thal}(h_{i-1}^{(k)}),\ \text{td_ctx}_i^{(k-1)}\right) \tag{27}$$

Step 3: Error recomputation. Prediction errors are recomputed with the updated representations.

Step 4: Lateral and workspace recomputation. Lateral inter-unit signals and global workspace broadcast signals are recomputed from the updated outputs and errors.

Step 5: Convergence test. The total squared error is compared to the previous iteration:

$$E^{(k)} = \sum_{l=0}^{N-1} \|\epsilon_l^{(k)}\|^2 \tag{28}$$

$$\delta^{(k)} = \frac{|E^{(k)} - E^{(k-1)}|}{E^{(k-1)} + \epsilon} \tag{29}$$

If $k \geq K_{\text{min}}$ and $\delta^{(k)} < \tau_{\text{conv}}$, the iteration terminates.

In a preferred embodiment, $K_{\text{min}} = 2$, $K_{\text{max}} = 5$, and $\tau_{\text{conv}} = 0.01$.

4.6.4. Gradient-Free Inference

A critical aspect of the re-entrant inference loop is that all iterations after the initial bottom-up pass execute under a gradient-free regime (e.g., `torch.no_grad()` in PyTorch). No computational graph is constructed during inference iterations. This has two important consequences:

(a) **Memory efficiency.** No intermediate activations are stored for backpropagation, reducing memory consumption by approximately the factor of inference iterations.

(b) **Biological plausibility.** Inference (representation settling) and learning (synaptic modification) are separated, consistent with the biological principle that neural computation and synaptic plasticity operate on different timescales.

4.7. Conflict-Driven Adaptive Depth

Referring to Figure 5, the conflict detection module (150) implements a metacognitive monitoring function that adjusts the number of inference iterations based on the degree of inter-unit disagreement.

4.7.1. Conflict Computation

The conflict score C is computed as the variance of the mean output representations across all processing units:

$$\bar{h}_i = \frac{1}{B \cdot T} \sum_{b=1}^{B} \sum_{t=1}^{T} h_i(b,t), \quad i = 0, \ldots, N-1 \tag{30}$$

$$C = \frac{1}{D} \sum_{d=1}^{D} \mathrm{Var}_i \left[\bar{h}_i[d] \right] \tag{31}$$

where B is the batch size, T is the sequence length, D is the representation dimension, and Var_i denotes variance across the N processing units.

4.7.2. Adaptive Step Allocation

The number of extra inference steps is computed as:

$$K_{\text{extra}} = \begin{cases} 0 & \text{if } C < \tau_C \text{ or } C \text{ is non-finite} \\ \min\left(K_{\text{max,extra}}, \left\lfloor \frac{C - \tau_C}{\max(\tau_C, 10^{-6})} \times K_{\text{max,extra}} \right\rfloor\right) & \text{otherwise} \end{cases} \tag{32}$$

The total maximum inference steps becomes $K_{\text{max}} + K_{\text{extra}}$.

In a preferred embodiment, $\tau_C = 0.5$ and $K_{\text{max,extra}} = 2$, meaning the system can allocate up to 2 additional inference iterations (for a total of 7) when processing highly ambiguous or conflicting inputs.

4.7.3. Health Monitoring

The conflict detection module further maintains exponential moving averages of per-unit σ^2 and convergence rates:

$$\sigma^{2(t)}_{\text{EMA},i} = (1-\alpha) \cdot \sigma^{2(t-1)}_{\text{EMA},i} + \alpha \cdot \overline{\sigma^{2(t)}_i} \tag{33}$$

where $\alpha = 0.01$ is a slow averaging coefficient. The module monitors for:

- **Precision divergence**: $\sigma^2_{\text{EMA},i} > \tau_{\text{max}}$ (e.g., 100.0), indicating the unit's uncertainty is growing without bound. Corrective action: scale down the unit's output weights.
- **Precision collapse**: $\sigma^2_{\text{EMA},i} < \tau_{\text{min}}$ (e.g., 0.001), indicating the unit has become pathologically overconfident. Corrective action: increase the bias b_σ.
- **Persistent non-convergence**: convergence rate exceeding 0.8, indicating the inference loop consistently hits the maximum step count.

4.8. Analytical Precision Learning

Referring to Figure 6, the analytical precision learning module (160) updates the precision parameters $\mathbf{W}_\sigma$ and b_σ using closed-form gradient expressions, without invoking automatic differentiation.

4.8.1. Derivation of the Analytical Gradient

Starting from the precision-weighted loss (Equation 22):

$$\mathcal{L} = \frac{\|\epsilon\|^2}{2 \cdot \sigma^2} + \frac{1}{2}\ln(\sigma^2) \tag{34}$$

Step 1: Derivative with respect to σ^2.

$$\frac{\partial \mathcal{L}}{\partial \sigma^2} = \frac{-\|\epsilon\|^2}{2 \cdot (\sigma^2)^2} + \frac{1}{2 \cdot \sigma^2} \tag{35}$$

Note that the error norm $\|\epsilon\|^2$ is computed as the mean squared error across the model dimension (i.e., $\frac{1}{D}\sum_{d=1}^{D} \epsilon_d^2$) to ensure the gradient magnitude is independent of the dimensionality D.

Step 2: Chain rule through softplus. Since $\sigma^2 = \text{softplus}(z) + \epsilon$ and $\frac{d}{dz}\text{softplus}(z) = \text{sigmoid}(z)$:

$$\frac{\partial \mathcal{L}}{\partial z} = \frac{\partial \mathcal{L}}{\partial \sigma^2} \times \text{sigmoid}(z) \tag{36}$$

The sigmoid value is clamped to $[0.01, 0.99]$ to prevent numerical instability at extreme pre-activations.

Step 3: Outer product for weight gradient. Since $z = \mathbf{W}_\sigma \cdot \hat{h} + b_\sigma$:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_\sigma} = \frac{1}{N} \sum_{n=1}^{N} \frac{\partial \mathcal{L}_n}{\partial z_n} \cdot \hat{h}_n^\top \tag{37}$$

$$\frac{\partial \mathcal{L}}{\partial b_\sigma} = \frac{1}{N} \sum_{n=1}^{N} \frac{\partial \mathcal{L}_n}{\partial z_n} \tag{38}$$

where $N = B \times T$ is the total number of tokens.

4.8.2. Computational Form

In matrix notation, let $\mathbf{Z} \in \mathbb{R}^{N \times 1}$ denote the flattened pre-activations and $\hat{\mathbf{H}} \in \mathbb{R}^{N \times D_{\text{col}}}$ denote the flattened normalised hidden states. The updates are:

$$\Delta \mathbf{W}_\sigma = \frac{1}{N} \mathbf{G}^\top \hat{\mathbf{H}} \tag{39}$$

$$\Delta b_\sigma = \frac{1}{N} \sum_{n=1}^{N} G_n \tag{40}$$

where $\mathbf{G} \in \mathbb{R}^{N \times 1}$ is the vector of per-token gradient scalars $\partial \mathcal{L}_n / \partial z_n$.

The gradient norm $\|\Delta \mathbf{W}_\sigma\|$ is clipped to a maximum value (e.g., 1.0) to prevent destabilising updates. The weight update is then:

$$\mathbf{W}_\sigma \leftarrow \mathbf{W}_\sigma - \eta \cdot \Delta \mathbf{W}_\sigma \tag{41}$$

$$b_\sigma \leftarrow b_\sigma - \eta \cdot \Delta b_\sigma \tag{42}$$

where η is the learning rate, optionally modulated by a neuromodulatory plasticity gate.

4.8.3. Advantages Over Automatic Differentiation

The analytical gradient computation offers the following advantages:

(a) **No activation storage.** Automatic differentiation requires storing all intermediate activations for the backward pass. The analytical method requires only the pre-activation z (cached from the forward pass), the hidden state $\hat{h}$ (already available), the prediction error ϵ, and the current σ^2 value.

(b) **Reduced computational cost.** The gradient computation requires two matrix multiplications (one for the weight gradient, one for the bias gradient) and element-wise operations

on vectors of length N, versus the full backward pass through the softplus, linear projection, and normalisation layers.

(c) **Compatibility with no-gradient inference.** Since the inference loop runs without gradient tracking, the precision parameters can be updated immediately after inference settles, without requiring a separate backward pass.

(d) **Numerical stability.** The analytical form allows explicit clamping of intermediate quantities (sigmoid clamping, precision floor) at precise points in the computation, providing finer control over numerical stability than automatic differentiation.

4.9. Prediction Weight Update

The prediction generation weights $\mathbf{W}_{\text{predict}}$ are updated via a similar analytical rule. The gradient of the precision-weighted loss with respect to $\mathbf{W}_{\text{predict}}$ is:

$$\Delta \mathbf{W}_{\text{predict}} = -\frac{1}{N} \left(\frac{\epsilon}{\sigma^2} \right)^{\top} \hat{\mathbf{H}} \tag{43}$$

where ϵ/σ^2 denotes element-wise division of the prediction error by the precision, broadcasting the scalar σ^2 across the model dimension. This is a precision-weighted outer product: predictions from high-precision (low σ^2) units receive larger gradient updates, accelerating learning where the unit is confident, and smaller updates where the unit is uncertain.

4.10. Integration with Hierarchical Neural Network

In a preferred embodiment, the precision-weighted predictive coding system is integrated into a hierarchical neural network as follows:

Phase 1: Embedding. Input tokens are converted to continuous representations via an embedding module.

Phase 2: Bottom-up pass. The embedded input propagates through all N processing units. Each unit produces an output, a prediction, and a precision estimate. The precision from unit i is passed to unit $i + 1$'s dendritic modulation.

Phase 3: Prediction error computation. Top-down predictions are compared against bottom-up targets at each level.

Phase 4: Conflict assessment. The metacognitive module computes inter-unit conflict and determines the number of additional inference steps.

Phase 5: Re-entrant inference. The system iterates, refining representations through top-down predictions and bottom-up error correction, until convergence or the maximum step count.

Phase 6: Output generation. The final representation from the top of the hierarchy is passed through a language model head to produce a probability distribution over the vocabulary.

Phase 7: Precision learning. After the forward pass, the analytical precision learning module updates $\mathbf{W}_\sigma$, b_σ, and $\mathbf{W}_{\text{predict}}$ for each processing unit using the cached errors, precisions, and hidden states.

The precision estimates from different bands feed into downstream subsystems:

- The sentence-band σ^2 modulates the episodic memory encoding threshold, controlling which experiences are stored.
- The discourse-band σ^2 modulates global workspace access, controlling which units broadcast to the shared workspace.
- The aggregate σ^2 across all units feeds into the neuromodulatory brainstem module, where it influences dopamine, norepinephrine, serotonin, and acetylcholine production levels.

4.11. Embodiment Parameters

Table 1 lists the configurable parameters of a preferred embodiment.

Table 1: Preferred embodiment parameters.

Parameter	Value	Description
σ^2_{base}	1.0	Base initial precision
α_{depth}	1.0	Depth scaling factor
σ_{init}	0.01	Weight init standard deviation
K_{min}	2	Minimum inference steps
K_{max}	5	Maximum inference steps (before adaptive)
τ_{conv}	0.01	Convergence threshold
τ_C	0.5	Conflict threshold
$K_{\text{max,extra}}$	2	Maximum additional adaptive steps
η_{PC}	3×10^{-4}	PC learning rate
τ_{max}	100.0	Precision divergence threshold
τ_{min}	0.001	Precision collapse threshold
ϵ	10^{-6}	Numerical stability constant

5. DESCRIPTION OF THE FIGURES

The accompanying drawings, which are incorporated herein and constitute a part of this specification, illustrate embodiments of the invention and, together with the description, serve to explain the principles of the invention.

Figure 1 is a system block diagram showing the overall architecture of the precision-weighted predictive coding system (Claim 1), including the hierarchical arrangement of processing units with precision estimation modules, prediction generation modules, the re-entrant inference controller, the conflict detection module, and the analytical precision learning module.

Figure 2 is a diagram showing the multi-scale precision band organisation, with the N processing units divided into four bands (token, phrase, sentence, discourse) and arrows showing how each band's aggregate precision feeds a distinct downstream subsystem.

Figure 3 is a computational graph of the analytical precision learning module, showing the step-by-step derivation from the precision-weighted loss through the derivative with respect to σ^2, the chain rule through softplus (yielding the sigmoid factor), and the outer product with the normalised hidden state to produce the weight gradient $\Delta\mathbf{W}_\sigma$ and bias gradient Δb_σ.

Figure P2-1

is a detailed system block diagram of the complete precision-weighted predictive coding system (Claim 1), showing (a) hierarchical processing units, (b) precision estimation pipeline, (c) prediction generation, (d) re-entrant inference controller, (e) conflict detection module, and (f) analytical precision learning module.

Figure P2-2

is a detailed block diagram of the depth-dependent bias initialisation (Claim 2), showing the initialisation formula and resulting bias values across the hierarchy.

Figure P2-3

is a diagram of the multi-scale precision band organisation (Claim 3), showing four bands (token, phrase, sentence, discourse) with downstream routing.

Figure P2-4

is a timeline diagram of the re-entrant inference iteration control (Claim 4), showing mandatory iterations, convergence testing, and extra conflict-allocated iterations.

Figure P2-5

is a diagram of the conflict score computation and adaptive step allocation (Claim 5), showing inter-unit variance computation and K_{extra} determination.

Figure P2-6

is a computational graph of the analytical precision learning (Claim 6), showing the four-step closed-form gradient derivation.

Figure P2-7

is a diagram of the dendritic modulation mechanism (Claim 7), showing how σ^2 from a preceding unit modulates the dendritic gate of the current unit.

Figure P2-8

is a diagram of the precision health monitoring system (Claim 8), showing EMA tracking, divergence detection, collapse detection, and persistent non-convergence monitoring.

Figure P2-9

is a complete method flowchart of the PWPC inference method (Claim 9), showing all steps from bottom-up pass through analytical gradient update.

Figure P2-10

is a bar chart of the depth-dependent bias initialisation (Claim 10), showing b_σ values across a hierarchy of $N = 12$ processing units.

Figure P2-11

is a diagram of precision band routing to downstream subsystems (Claim 11), showing sentence-band routing to episodic memory and discourse-band routing to global workspace.

Figure P2-12

is a state machine diagram of the re-entrant iteration control (Claim 12), showing the MANDATORY, TEST, EXTRA, and SETTLED states.

Figure P2-13

is a detailed four-step pipeline of the analytical gradient computation (Claim 13), showing mean squared error, derivative w.r.t. σ^2, sigmoid clamping, and outer product averaging.

Figure P2-14

is a diagram of the precision-weighted prediction update (Claim 14), contrasting high-precision and low-precision units to illustrate how σ^2 acts as a per-unit adaptive learning rate.

Figure P2-15

is a chain diagram of inter-unit precision propagation (Claim 15), showing how σ^2 propagates through the hierarchy modulating dendritic gates.

Figure P2-16

is a block diagram of the complete PWPC computer-readable medium instruction set (Claim 16), showing all stored instructions from instantiation through band routing.

Figure P2-17

is a diagram of the four precision bands with mean aggregation (Claim 17), showing example σ^2 values and aggregates for $N = 12$ units.

Figure P2-18

is a diagram of the EMA precision monitoring and corrective actions (Claim 18), showing divergence and collapse detection with time-series graphs.

Figure P2-19

is a side-by-side comparison diagram of the gradient-free inference regime (Claim 19), contrasting standard autodiff memory usage with the gradient-free approach.

Figure P2-20

is a diagram of the depth-dependent initialisation boundary values (Claim 20), showing the linear interpolation from σ^2_{base} at the shallowest unit to $\sigma^2_{\text{base}} \times (1 + \alpha_{\text{depth}})$ at the deepest unit.

Note: The drawings described above are schematic representations of the system architecture and data flow. Actual implementation may vary in specific arrangement and connectivity without departing from the scope of the invention.

DRAWING SHEETS

FIG. 5
Re-Entrant Predictive Coding Inference Loop

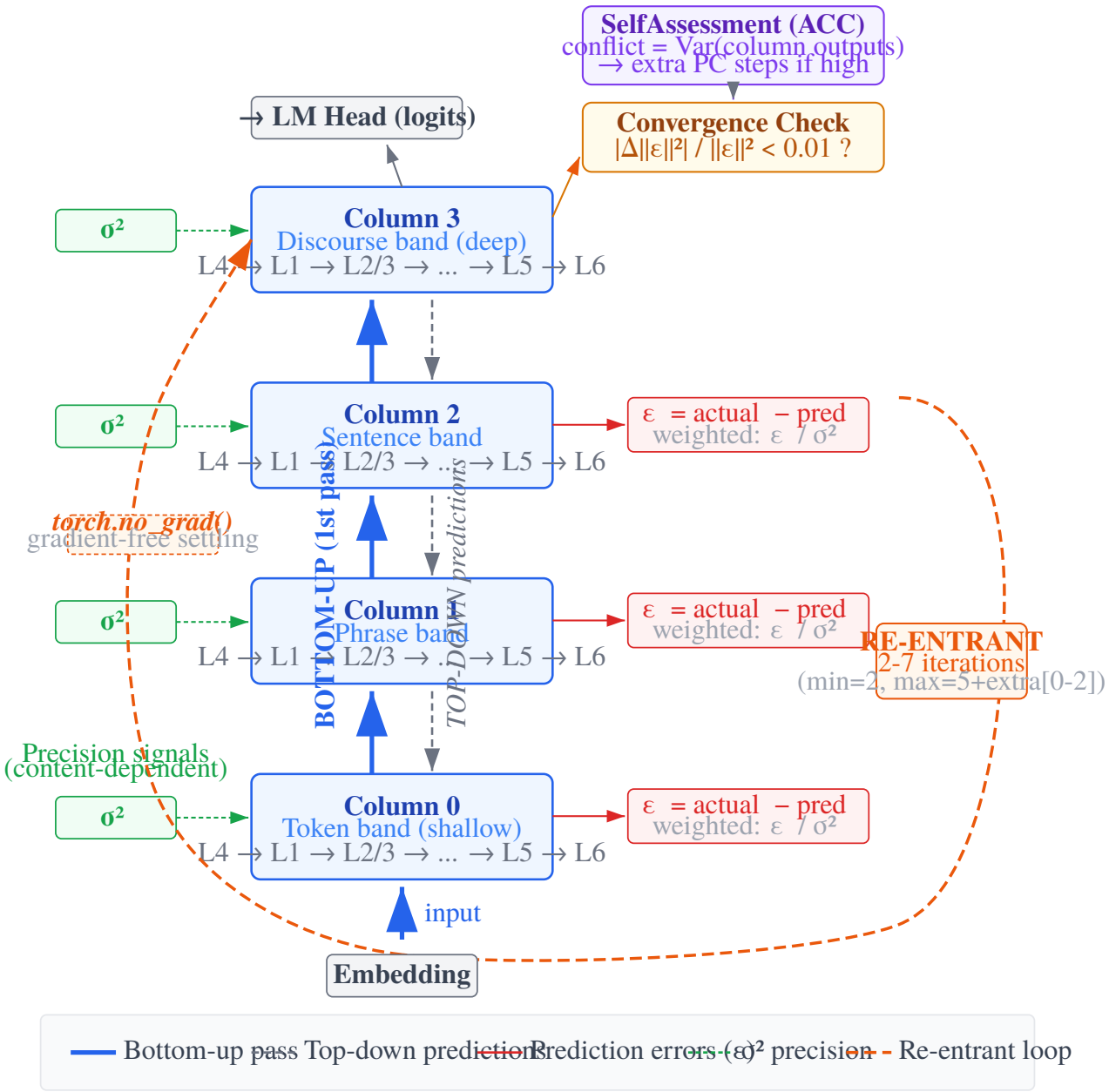


Figure 1: FIG. 1 — Precision-Weighted Predictive Coding Loop

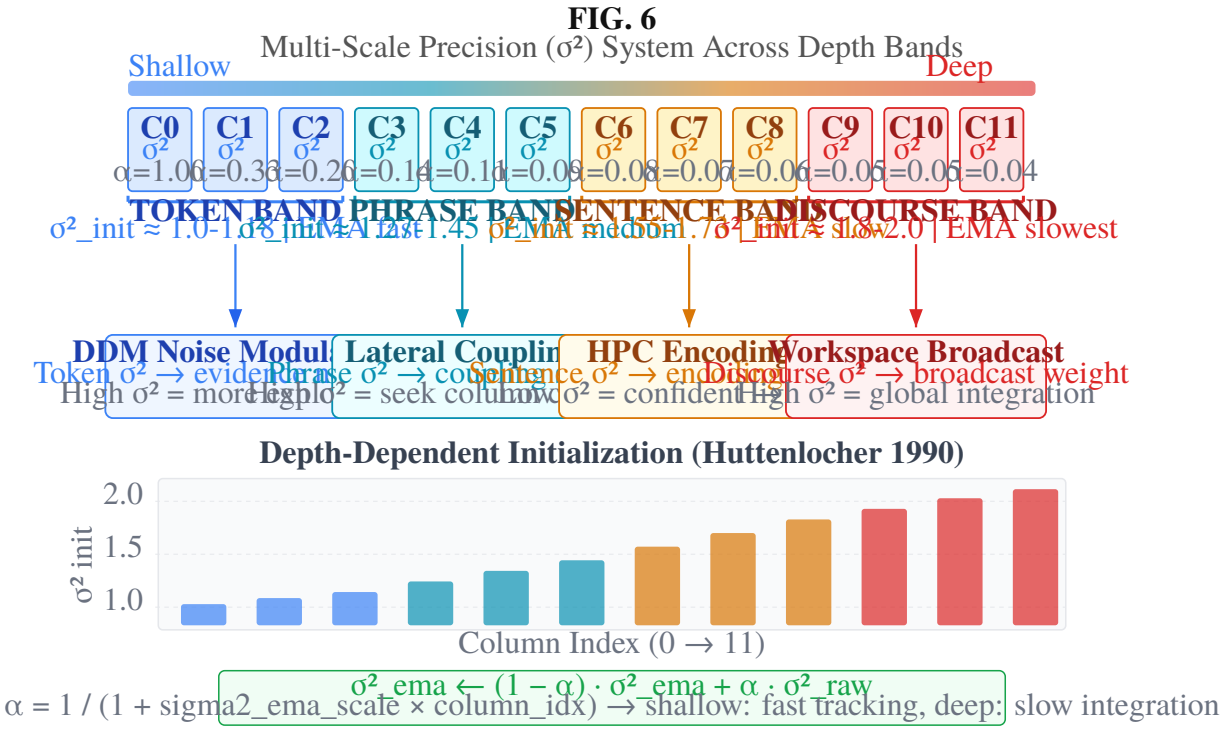


Figure 2: FIG. 2 — Multi-Scale σ^2 Band Organisation

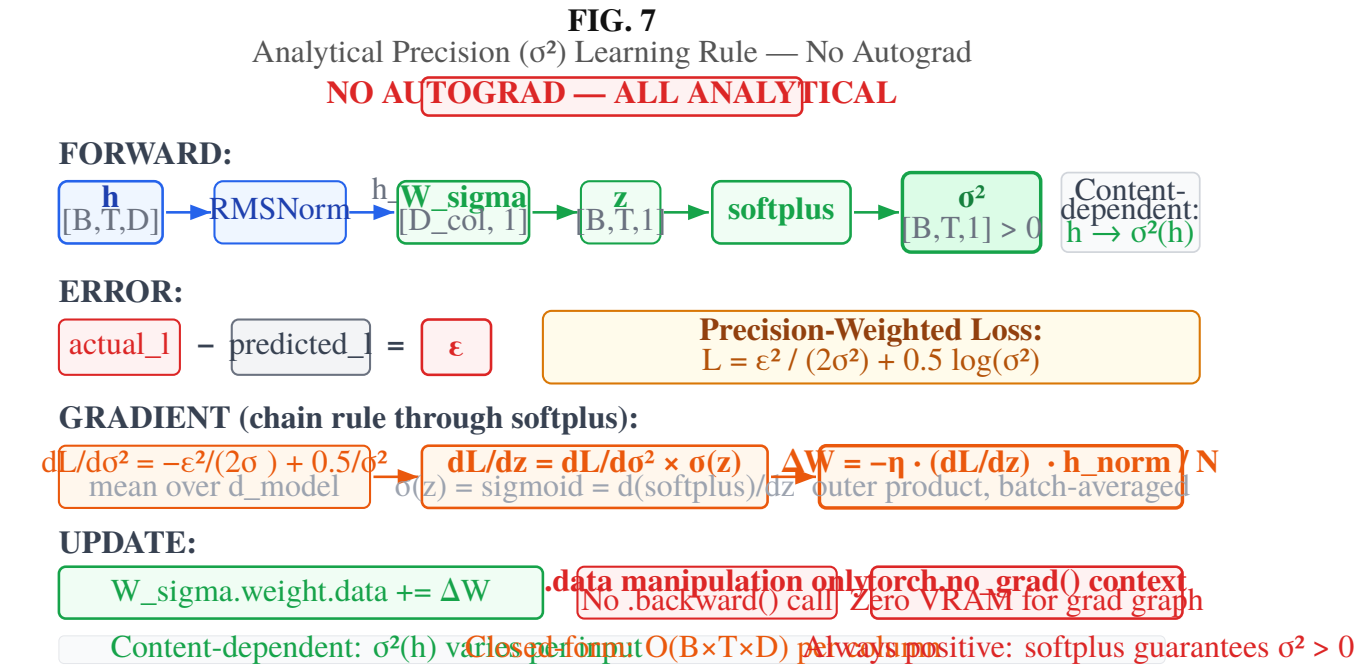
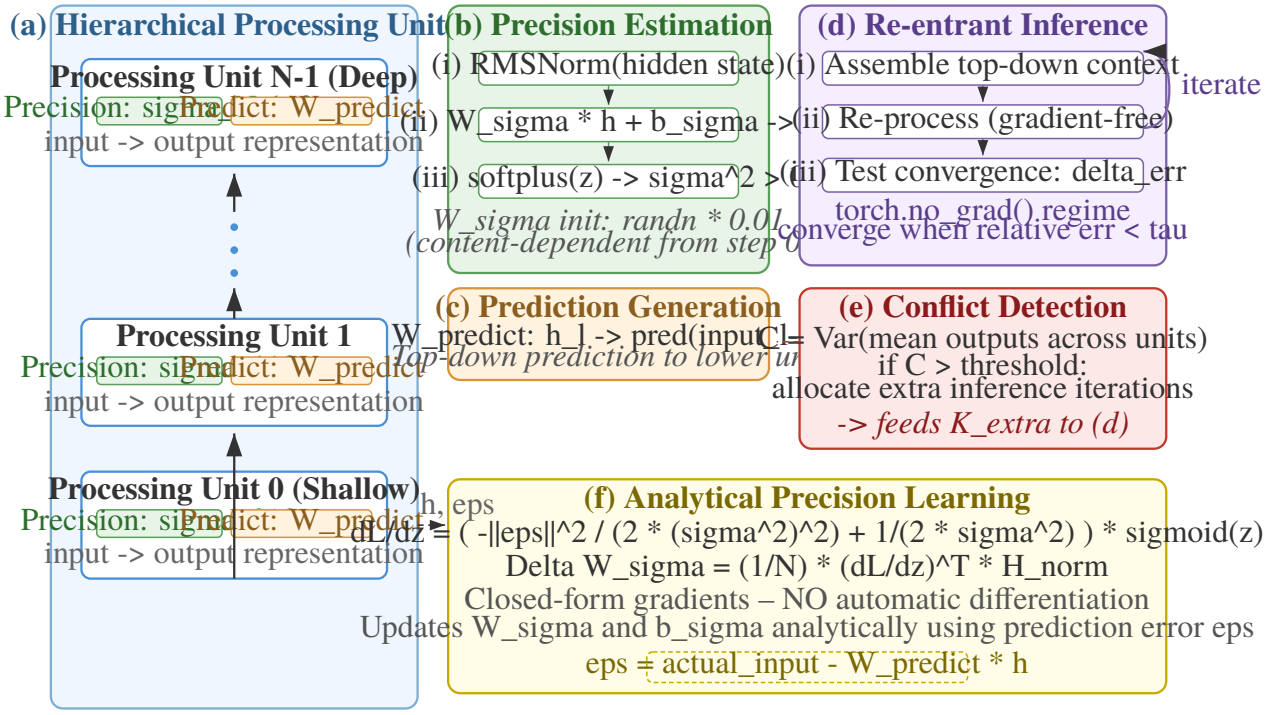


Figure 3: FIG. 3 — Analytical Precision (σ^2) Learning Rule

CLAIM ILLUSTRATION SHEETS

FIG. P2-1: Precision-Weighted Predictive Coding System (Claim 1)

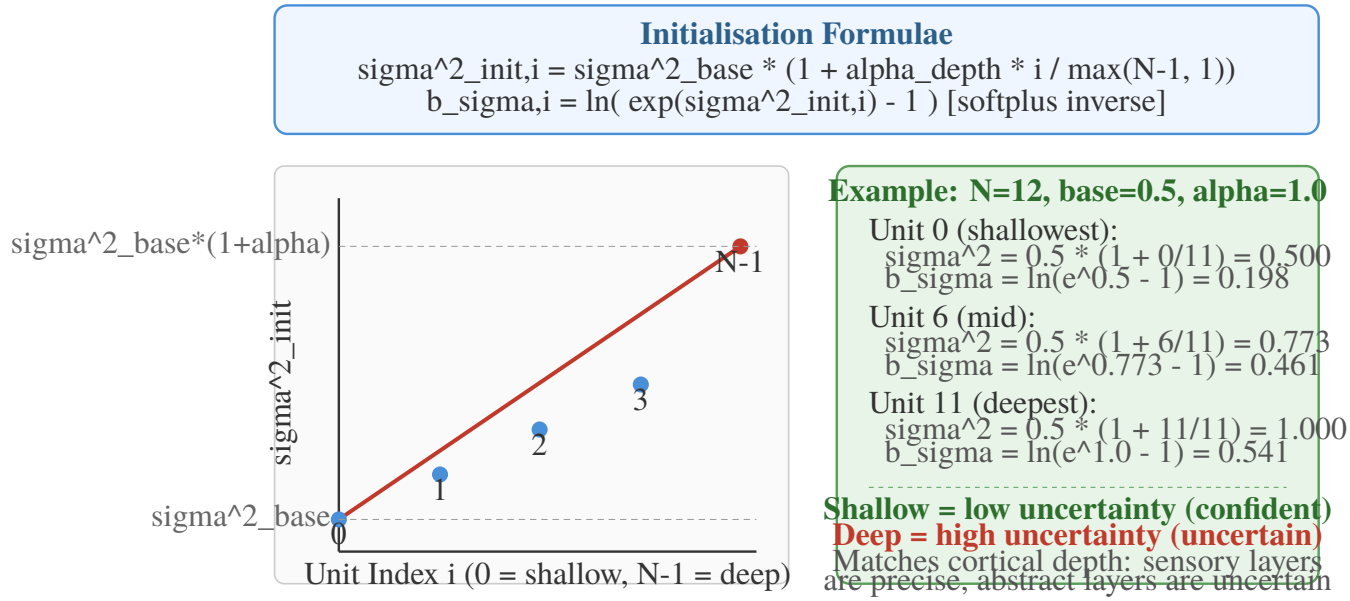
lete PWPC system with hierarchy, precision estimation, prediction, re-entrant inference, conflict detection, and analytical lear



Patent 2: Precision-Weighted Predictive Coding – Independent Claim 1 (System)
Source: column/l5_output_prediction.py, model/network.py, systems/self_assessment.py

Figure 4: FIG. P2-1 — Precision-Weighted Predictive Coding System (Claim 1). Complete system with (a) hierarchical processing units, (b) precision estimation (RMSNorm $\rightarrow$ $\mathbf{W}_\sigma$ $\rightarrow$ softplus), (c) prediction generation ($\mathbf{W}_{\text{predict}}$), (d) re-entrant inference controller (gradient-free), (e) conflict detection (inter-unit variance), and (f) analytical precision learning (closed-form gradients).

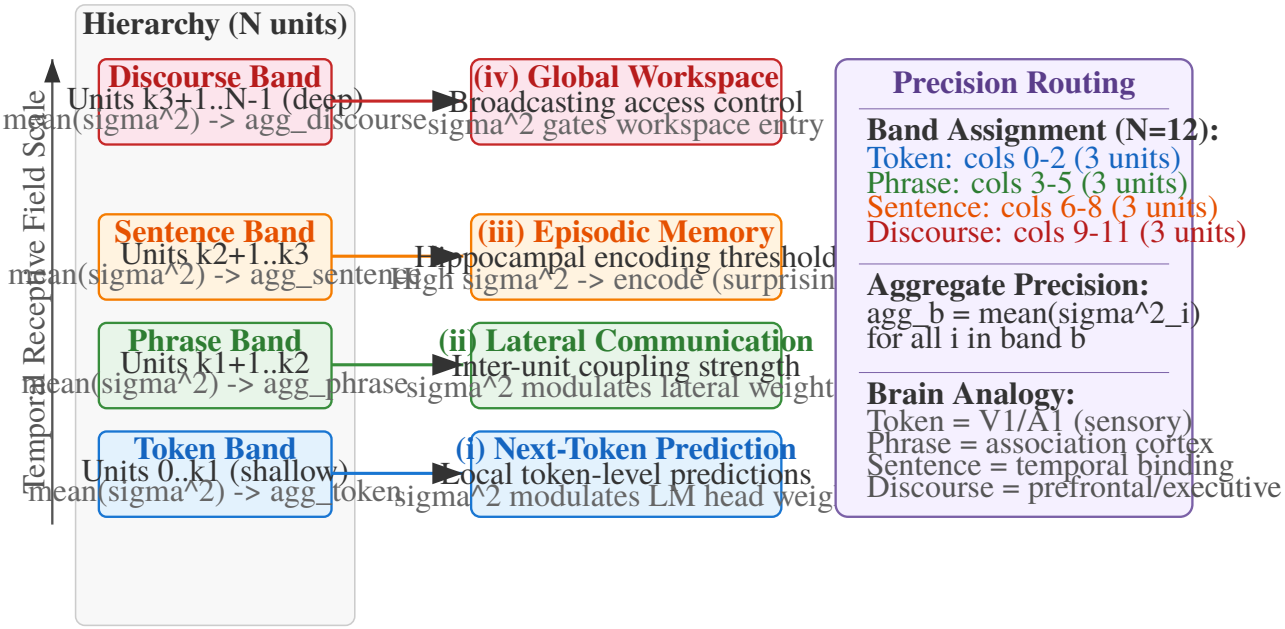
FIG. P2-2: Depth-Dependent Precision Initialisation (Claim 2)
Bias b_sigma initialised so shallow units start with lower uncertainty than deep units



Patent 2: PWPC – Claim 2 (Depth-Dependent Initialisation)
Source: column/l5_output_prediction.py (sigma2_init computation)

Figure 5: FIG. P2-2 — Depth-Dependent Bias Initialisation (Claim 2). Bias $b_{\sigma,i} = \ln(\exp(\sigma^2_{\text{base}} \times (1 + \alpha_{\text{depth}} \times i / \max(N-1, 1))) - 1)$; shallow units receive lower initial bias (lower uncertainty), deep units receive higher initial bias (higher uncertainty).

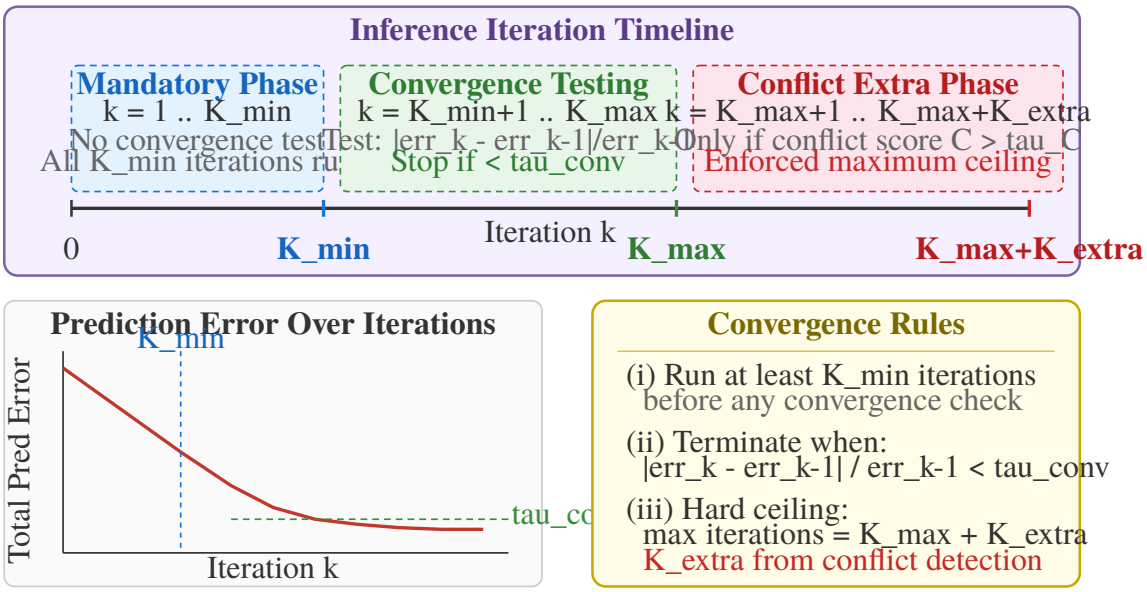
FIG. P2-3: Multiscale Precision Bands (Claim 3)
Processing units organised into 4 precision bands, each routing to a distinct subsystem



Patent 2: PWPC – Claim 3 (Multiscale Precision Bands)
Source: model/network.py (precision band routing), systems/workspace.py, systems/hippocampus.py

Figure 6: FIG. P2-3 — Multi-Scale Precision Bands (Claim 3). Four bands (token, phrase, sentence, discourse) with aggregate precision routing: token → NTP, phrase → lateral coupling, sentence → episodic memory, discourse → global workspace.

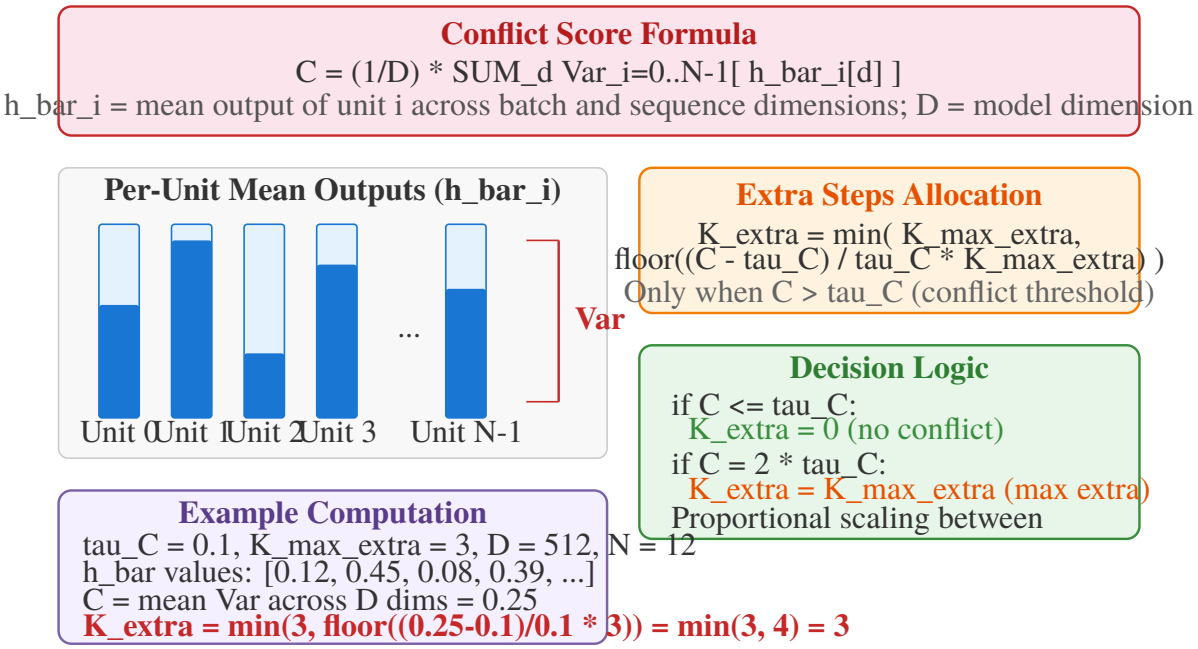
FIG. P2-4: Re-entrant Inference Convergence (Claim 4)
K_min mandatory iterations, convergence test, K_max + K_extra ceiling



Patent 2: PWPC – Claim 4 (Re-entrant Inference Convergence Control)

Figure 7: FIG. P2-4 — Re-entrant Inference Iteration Control (Claim 4). Timeline showing K_{min} mandatory iterations, convergence testing ($\delta < \tau_{\text{conv}}$) from K_{min} to K_{max}, and K_{extra} conflict-allocated additional iterations.

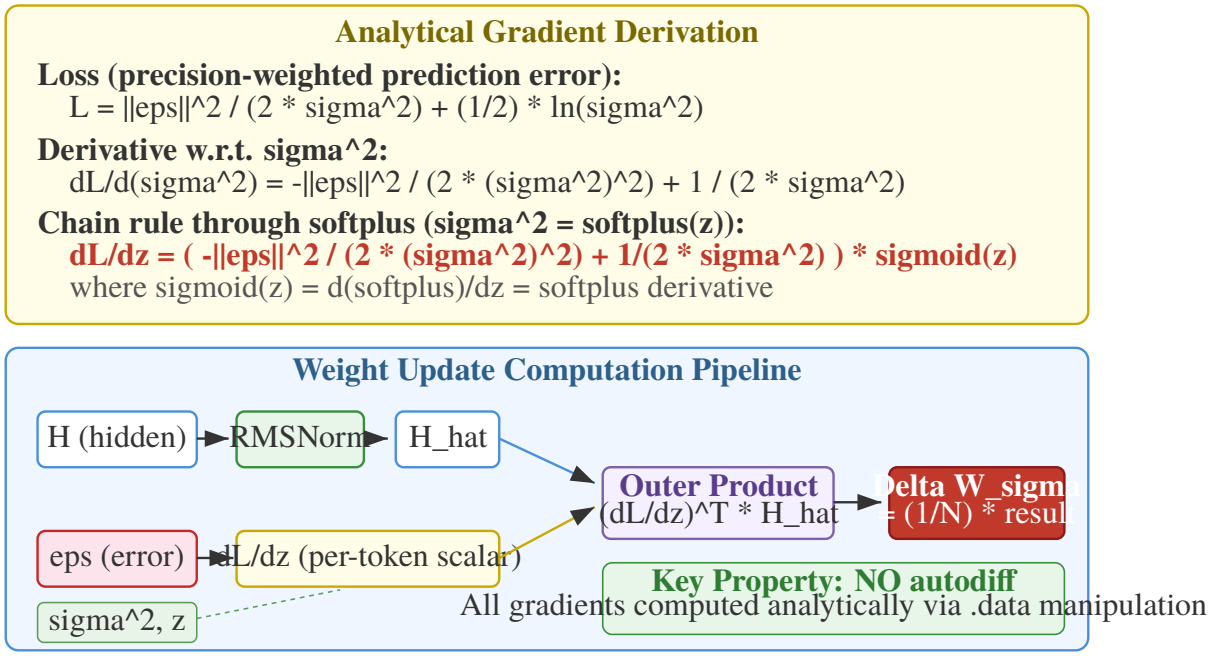
FIG. P2-5: Conflict Score Computation (Claim 5)
Inter-unit output variance as conflict measure, proportional extra step allocation



Patent 2: PWPC – Claim 5 (Conflict Score and Extra Iteration Allocation)

Figure 8: FIG. P2-5 — Conflict Score Computation (Claim 5). Inter-unit output variance $C = (1/D)\sum_d \text{Var}_i[\bar{h}_i[d]]$; extra iterations $K_{\text{extra}} = \min(K_{\text{max,extra}}, \lfloor (C - \tau_C)/\tau_C \times K_{\text{max,extra}} \rfloor)$.

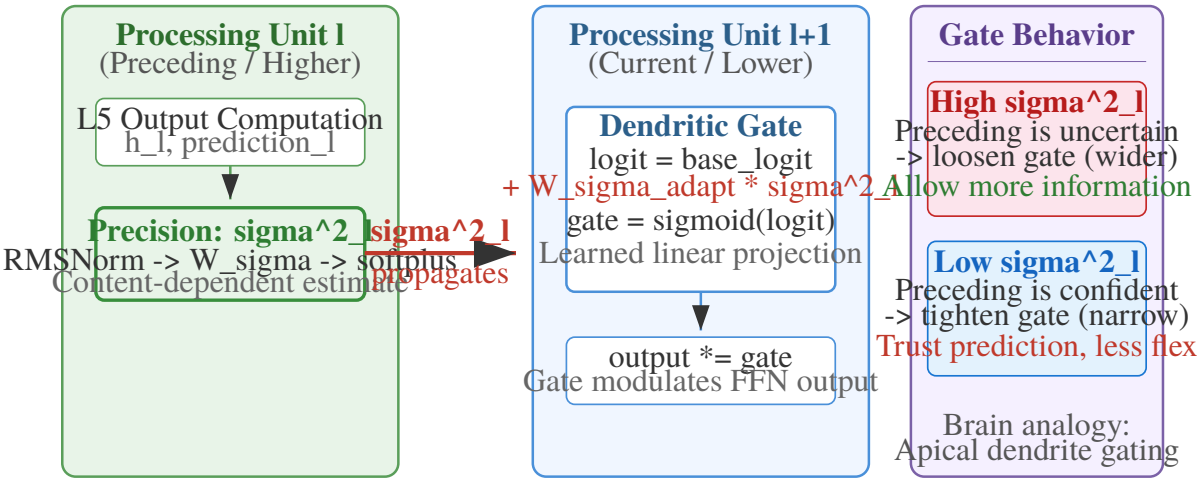
FIG. P2-6: Analytical Precision Learning (Claim 6)
Closed-form gradient computation for W_sigma and b_sigma, no autodiff



Patent 2: PWPC – Claim 6 (Analytical Precision Gradient)
Source: learning/predictive_coding.py, column/l5_output_prediction.py

Figure 9: FIG. P2-6 — Analytical Precision Learning (Claim 6). Closed-form gradient: $\partial\mathcal{L}/\partial z = (-\|\epsilon\|^2/(2(\sigma^2)^2) + 1/(2\sigma^2)) \times \text{sigmoid}(z)$; weight update $\Delta\mathbf{W}_\sigma = (1/N)(\partial\mathcal{L}/\partial z)^\top\hat{\mathbf{H}}$.

FIG. P2-7: Dendritic Modulation by Preceding Precision (Claim 7)
sigma^2 from unit l modulates dendritic gate of unit l+1



Patent 2: PWPC – Claim 7 (Dendritic Precision Modulation)
Source: column/l5_output_prediction.py (dendritic gate computation)

Figure 10: FIG. P2-7 — Dendritic Modulation by Preceding Precision (Claim 7). σ^2 from unit l modulates dendritic gate of unit $l+1$ via $\mathbf{W}_{\sigma\text{-adapt}}$: high preceding σ^2 loosens gate, low preceding σ^2 tightens gate.

FIG. P2-8: Precision Health Monitoring (Claim 8)
EMA tracking, divergence detection, collapse detection, persistent non-convergence

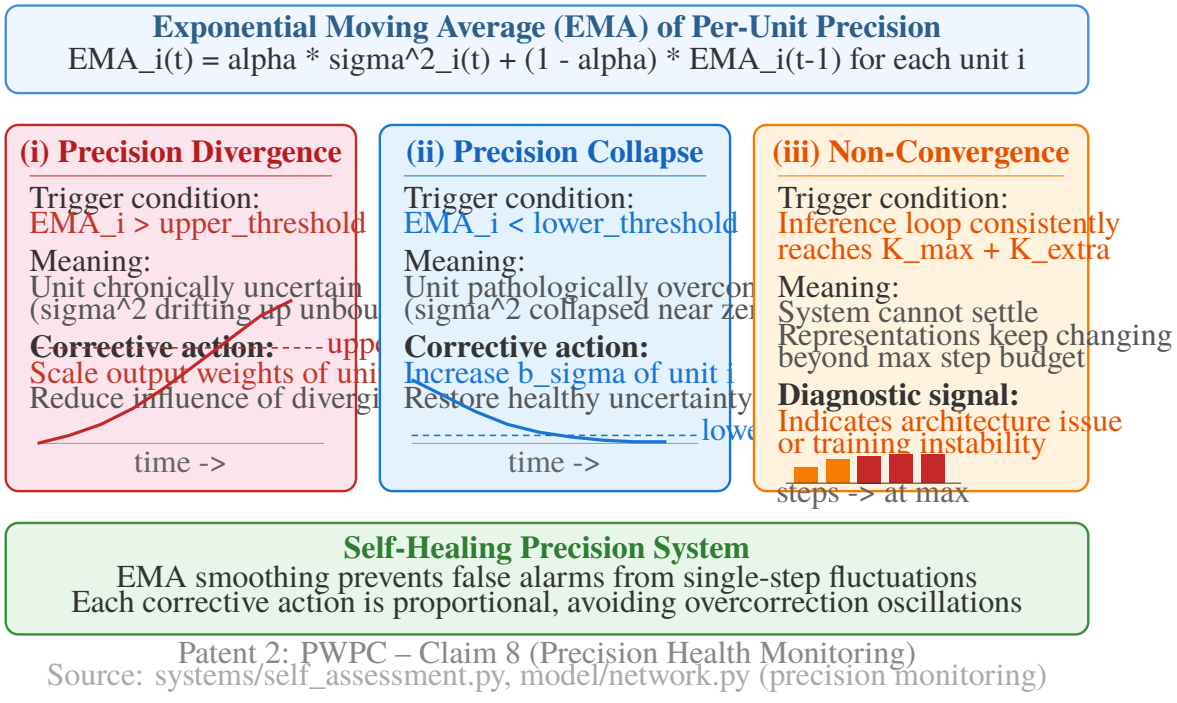
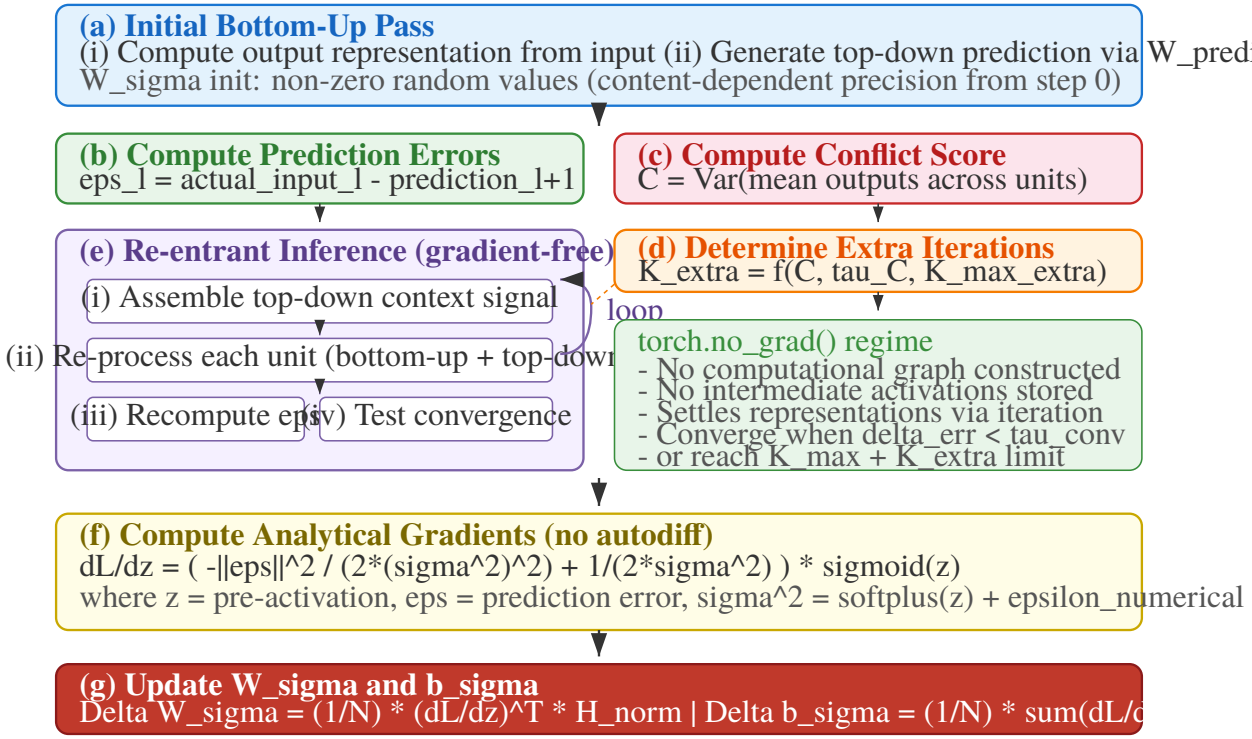


Figure 11: FIG. P2-8 — Precision Health Monitoring (Claim 8). EMA tracking with three monitoring conditions: (i) divergence (EMA $>$ upper threshold $\rightarrow$ scale output weights), (ii) collapse (EMA $<$ lower threshold $\rightarrow$ increase b_σ), (iii) persistent non-convergence.

FIG. P2-9: PWPC Inference Method (Claim 9)
Complete method flow: bottom-up pass, prediction errors, conflict, re-entrant inference, analytical learning



Patent 2: PWPC – Independent Claim 9 (Method)
Source: model/network.py (forward + PC inference loop), learning/predictive_coding.py

Figure 12: FIG. P2-9 — PWPC Inference Method (Claim 9, independent). Complete method: (a) bottom-up pass, (b) prediction errors, (c) conflict score, (d) extra iterations, (e) re-entrant inference (gradient-free), (f) analytical gradients, (g) parameter update.

FIG. P2-10: Depth-Dependent Bias Initialisation (Claim 10)
Method for setting b_sigma per unit index; shallow = low uncertainty, deep = high uncertainty

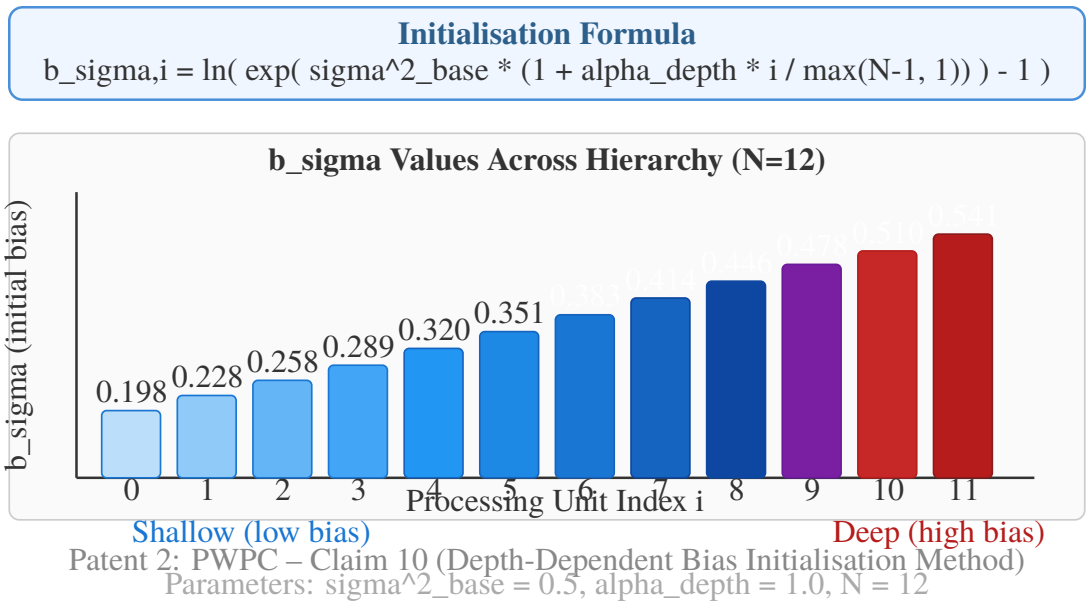


Figure 13: FIG. P2-10 — Depth-Dependent Bias Initialisation (Claim 10). Bar chart of b_σ values for $N = 12$ units with $\sigma^2_{\text{base}} = 0.5$, $\alpha_{\text{depth}} = 1.0$: values increase monotonically from 0.198 (shallowest) to 0.541 (deepest).

FIG. P2-11: Precision Band Routing to Subsystems (Claim 11)
Sentence-level precision modulates episodic memory; discourse-level modulates global workspace

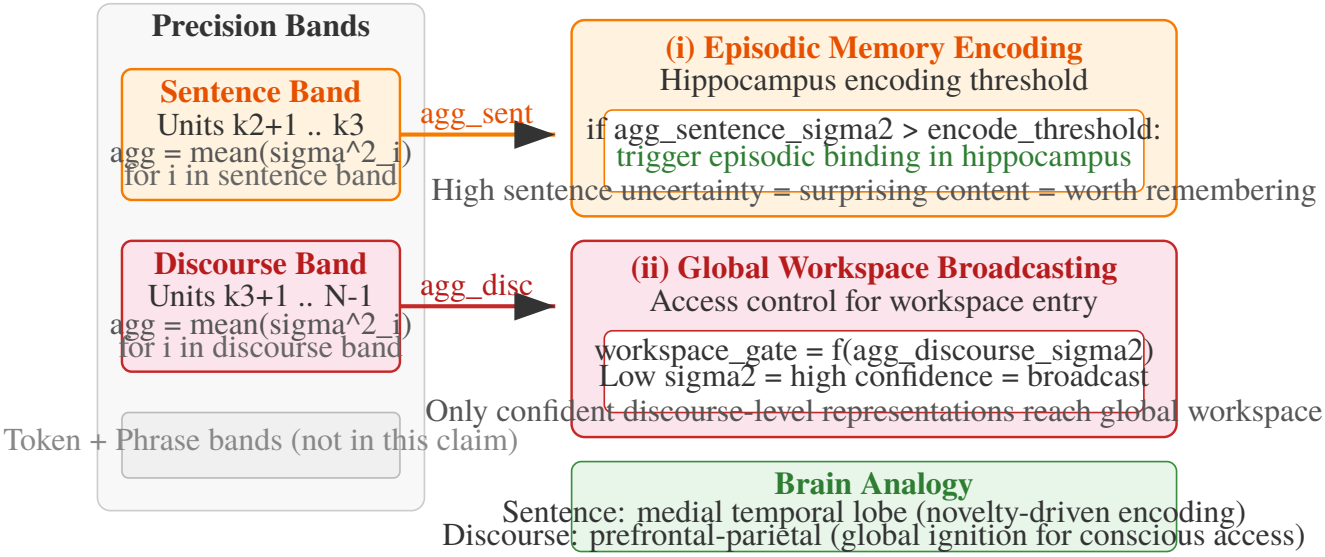
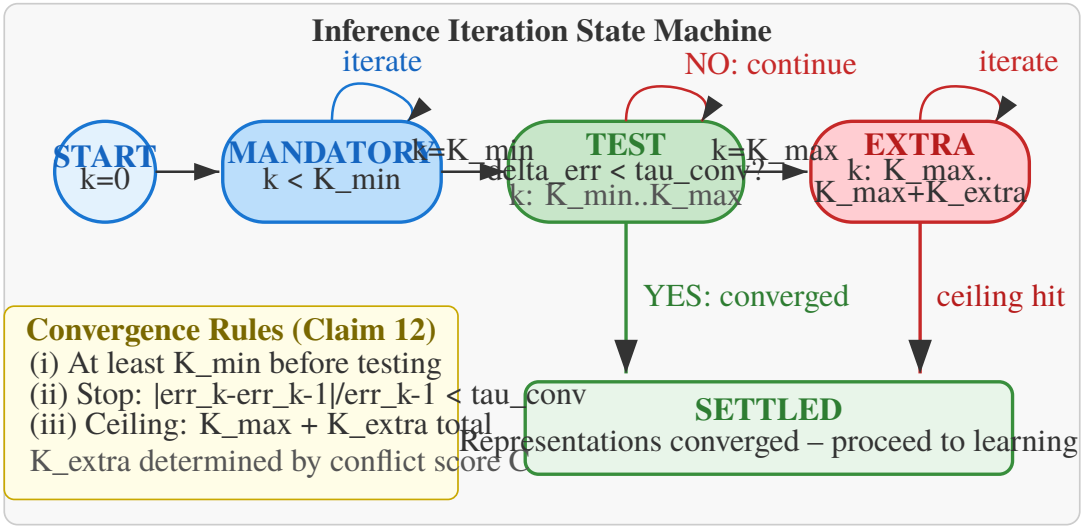


Figure 14: FIG. P2-11 — Precision Band Routing to Subsystems (Claim 11). Sentence-band aggregate σ^2 modulates episodic memory encoding threshold; discourse-band aggregate σ^2 modulates global workspace broadcasting access.

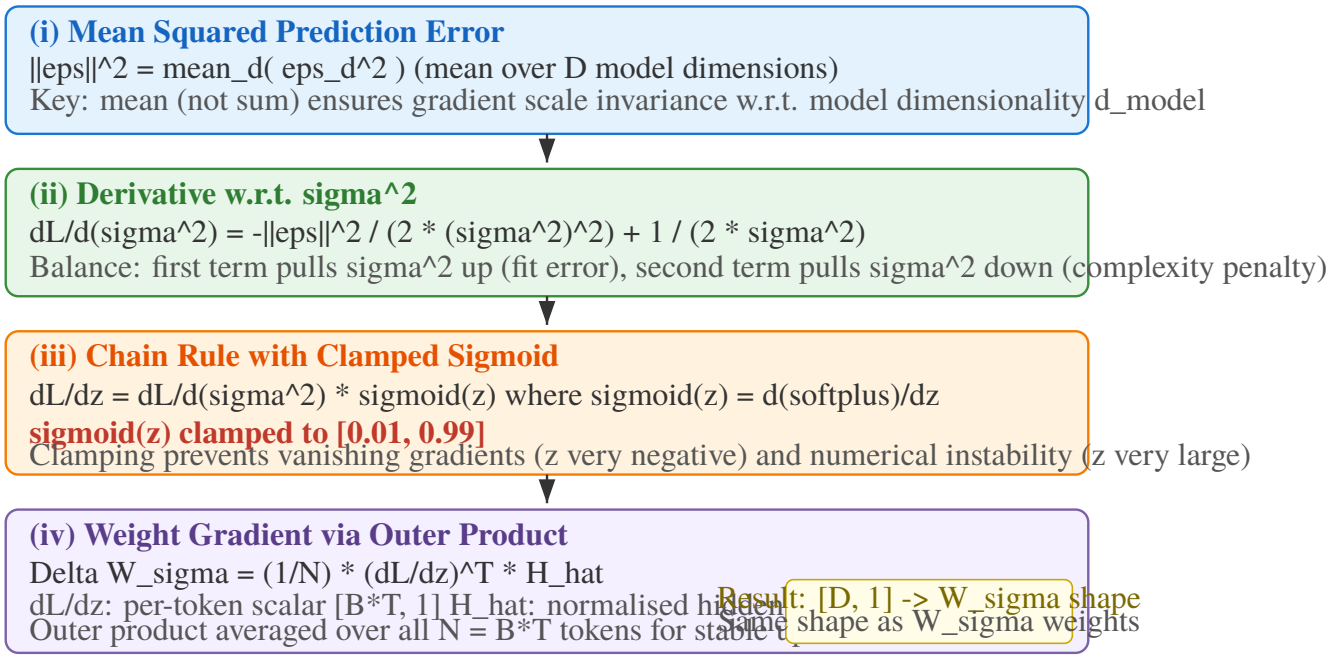
FIG. P2-12: Re-entrant Iteration Control (Claim 12)
K_min mandatory, tau_conv termination, K_max + K_extra ceiling (method claim)



Patent 2: PWPC – Claim 12 (Re-entrant Iteration Control Method)

Figure 15: FIG. P2-12 — Re-entrant Iteration Control State Machine (Claim 12). States: START $\rightarrow$ MANDATORY ($k < K_{\text{min}}$) $\rightarrow$ TEST ($\delta_{\text{err}} < \tau_{\text{conv}}$?) $\rightarrow$ EXTRA ($K_{\text{max}}..K_{\text{max}} + K_{\text{extra}}$) $\rightarrow$ SETTLED.

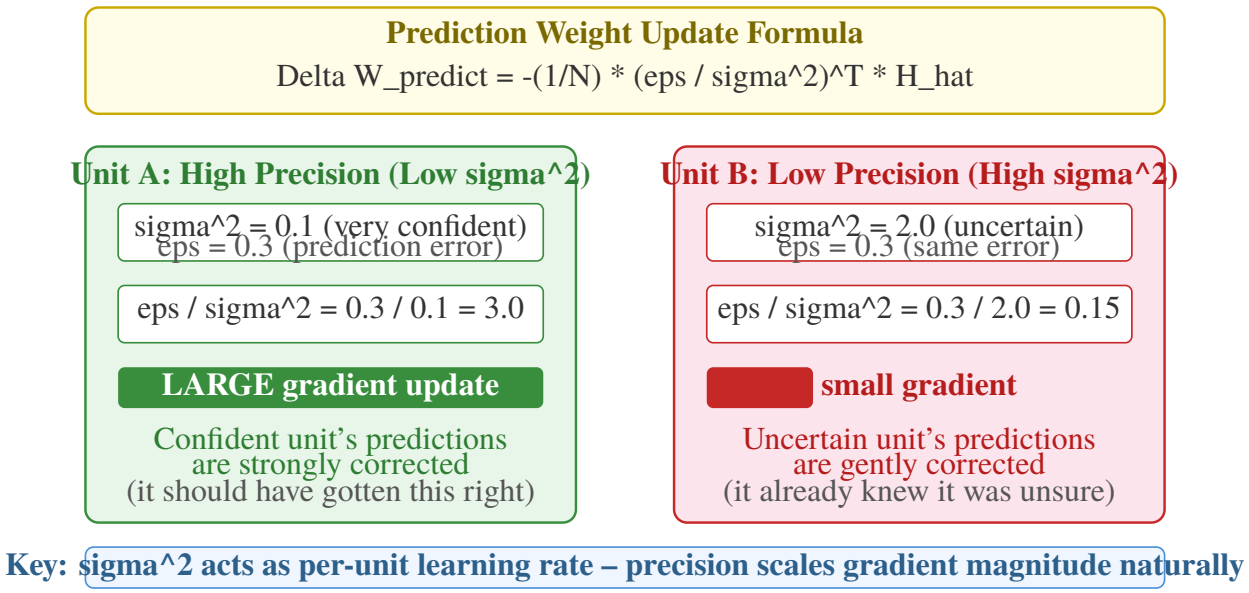
FIG. P2-13: Analytical Gradient Computation Details (Claim 13)
Four-step gradient pipeline: mean MSE, dL/d(sigma^2), sigmoid clamping, outer product averaging



Patent 2: PWPC – Claim 13 (Analytical Gradient Computation Details)
Source: learning/predictive_coding.py (analytical_sigma2_update)

Figure 16: FIG. P2-13 — Analytical Gradient Computation Details (Claim 13). Four-step pipeline: (i) mean MSE, (ii) $\partial\mathcal{L}/\partial\sigma^2$, (iii) sigmoid clamping $[0.01, 0.99]$, (iv) outer product weight gradient $\Delta\mathbf{W}_\sigma$.

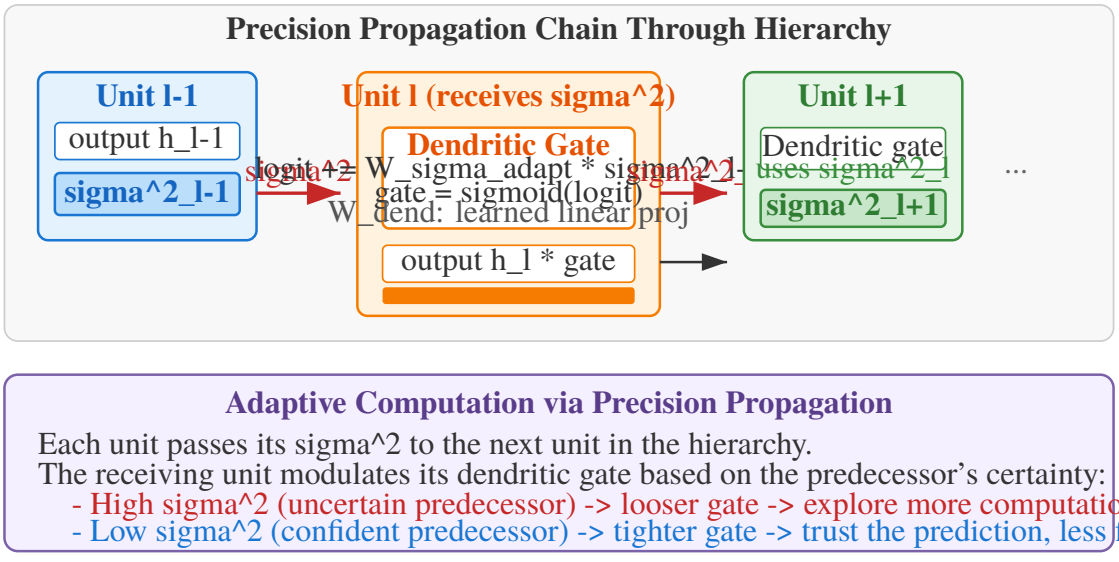
FIG. P2-14: Precision-Weighted Prediction Update (Claim 14)
W_predict updated proportionally to precision: high-precision units learn faster



Patent 2: PWPC – Claim 14 (Precision-Weighted Prediction Gradient)

Figure 17: FIG. P2-14 — Precision-Weighted Prediction Update (Claim 14). $\Delta \mathbf{W}_{\text{predict}} = -(1/N)(\epsilon/\sigma^2)^\top \hat{\mathbf{H}}$: high-precision units (low σ^2) receive large updates, low-precision units (high σ^2) receive small updates.

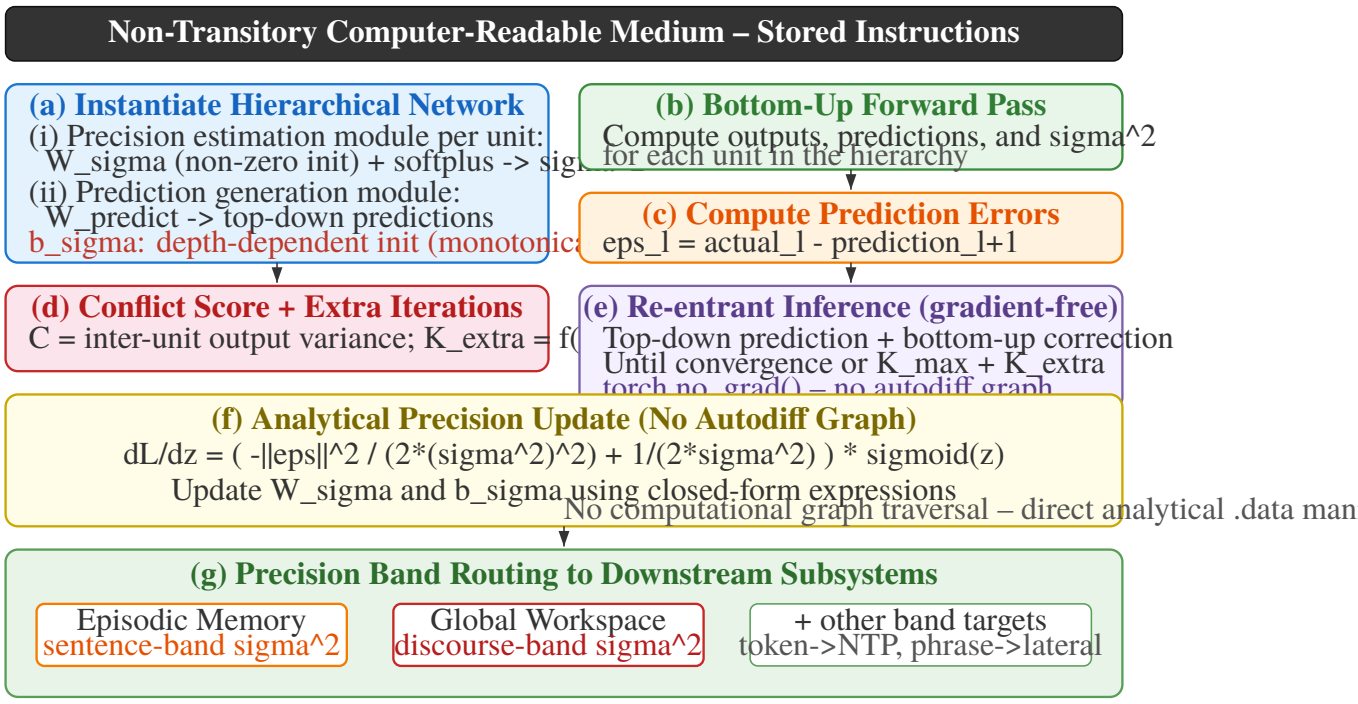
FIG. P2-15: Inter-Unit Precision Propagation (Claim 15)
sigma^2 propagated from each unit to the next; modulates dendritic gate via learned projection



Patent 2: PWPC – Claim 15 (Inter-Unit Precision Propagation and Dendritic Gating)

Figure 18: FIG. P2-15 — Inter-Unit Precision Propagation (Claim 15). Chain showing σ^2 passed from each unit to the next, modulating the receiving unit's dendritic gate via learned projection $\mathbf{W}_{\sigma\text{-adapt}}$.

FIG. P2-16: PWPC Computer-Readable Medium (Claim 16)
Complete instruction set: instantiate, forward pass, errors, conflict, inference, analytical update, band routing



Patent 2: PWPC – Independent Claim 16 (Non-Transitory Computer-Readable Medium)
Source: Complete PWPC pipeline as stored processor instructions

Figure 19: FIG. P2-16 — PWPC Computer-Readable Medium (Claim 16, independent). Stored instructions: (a) instantiate, (b) bottom-up pass, (c) prediction errors, (d) conflict + extra iterations, (e) re-entrant inference (gradient-free), (f) analytical update, (g) band routing.

FIG. P2-17: Four Precision Bands (Claim 17)
Token, phrase, sentence, discourse bands; aggregate = mean sigma^2 per band

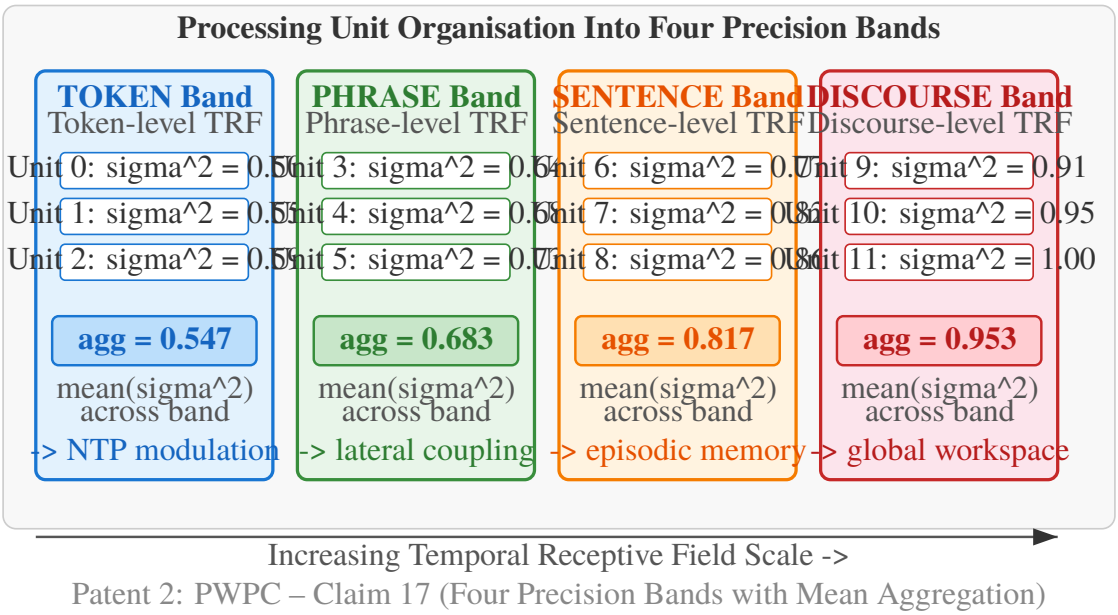


Figure 20: FIG. P2-17 — Four Precision Bands (Claim 17). $N = 12$ units in four bands (token, phrase, sentence, discourse) with mean aggregation per band. Example aggregates: token 0.547, phrase 0.683, sentence 0.817, discourse 0.953.

FIG. P2-18: EMA Precision Monitoring and Correction (Claim 18)
EMA tracking with automatic corrective actions for divergence and collapse

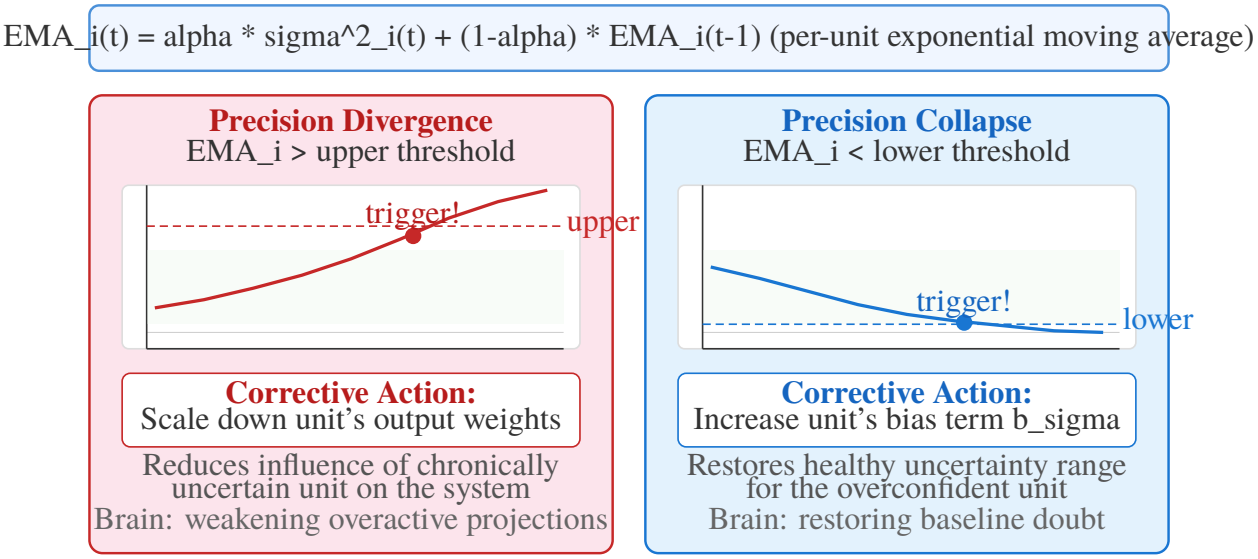


Figure 21: FIG. P2-18 — EMA Precision Monitoring and Correction (Claim 18). Divergence (EMA $>$ upper $\rightarrow$ scale output weights) and collapse (EMA $<$ lower $\rightarrow$ increase b_σ) with time-series graphs.

FIG. P2-19: Gradient-Free Inference Regime (Claim 19)
No computational graph, no intermediate activations stored during re-entrant iterations

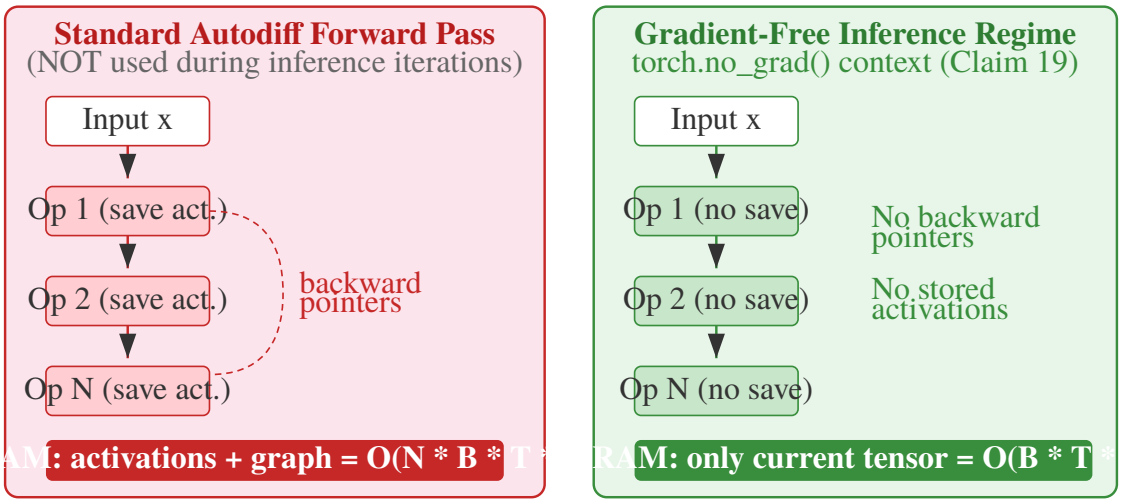


Figure 22: FIG. P2-19 — Gradient-Free Inference Regime (Claim 19). Comparison: standard autodiff ($O(N \times B \times T \times D)$ VRAM) vs. gradient-free ($O(B \times T \times D)$ VRAM), saving ∼3 GiB for the 106M model at batch 96.

FIG. P2-20: Depth-Dependent Init Boundary Values (Claim 20)
Shallowest unit: sigma^2_base; deepest unit: sigma^2_base * (1 + alpha_depth)

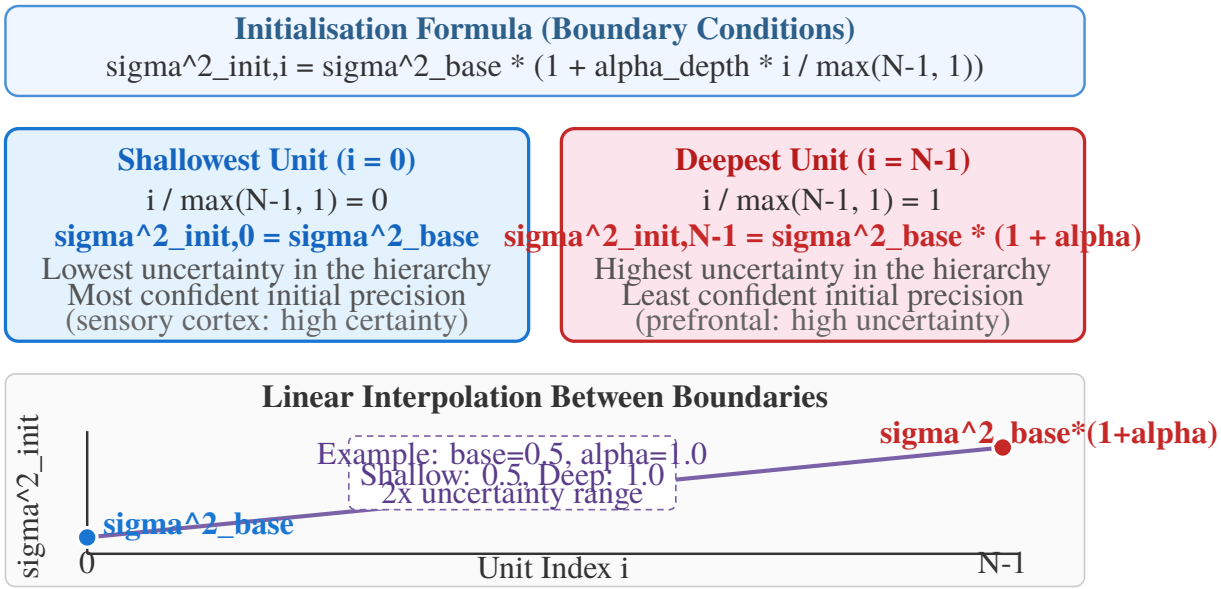


Figure 23: FIG. P2-20 — Depth-Dependent Init Boundary Values (Claim 20). Shallowest unit: $\sigma^2_{\text{init},0} = \sigma^2_{\text{base}}$; deepest unit: $\sigma^2_{\text{init},N-1} = \sigma^2_{\text{base}} \times (1 + \alpha_{\text{depth}})$. Linear interpolation with example: base$= 0.5$, $\alpha = 1.0$ gives range $[0.5, 1.0]$.

Note: In the actual patent filing, each figure described above shall be provided as a separate drawing sheet conforming to Rule 15 of the Patents Rules, 2003. The drawings are schematic representations of the system architecture, data flow, and computational methods. Actual implementation may vary in specific arrangement and connectivity without departing from the scope of the invention.

6. CLAIMS

We claim:

Claim 1. A precision-weighted predictive coding system for a hierarchical neural network, the system comprising:

(a) a plurality of processing units arranged in a feedforward hierarchy, each processing unit configured to receive an input representation and produce an output representation;

(b) a precision estimation module within each processing unit, comprising:

(i) a normalisation layer configured to normalise the hidden state of the processing unit;

(ii) a learned linear projection having a weight matrix $\mathbf{W}_\sigma$ and a bias term b_σ, configured to map the normalised hidden state to a scalar pre-activation value; and

(iii) a positivity-preserving activation function, specifically a softplus function, configured to transform the scalar pre-activation into a per-token precision estimate σ^2 that is strictly positive,

wherein the weight matrix $\mathbf{W}_\sigma$ is initialised with non-zero random values such that the precision estimate varies with input content from the first forward pass;

(c) a prediction generation module within each processing unit, comprising a learned linear projection $\mathbf{W}_{\text{predict}}$ configured to generate a top-down prediction of the input to the next lower processing unit in the hierarchy;

(d) a re-entrant inference controller configured to iteratively refine the representations of the processing units by:

(i) assembling a top-down context signal for each processing unit from predictions of higher units;

(ii) re-processing each processing unit with the assembled top-down context under a gradient-free computational regime; and

(iii) testing for convergence by computing a relative change in total prediction error across the hierarchy;

(e) a conflict detection module configured to compute a conflict score as the variance of mean output representations across the processing units, and to allocate additional inference iterations to the re-entrant inference controller when the conflict score exceeds a predetermined threshold; and

(f) an analytical precision learning module configured to update the weight matrix $\mathbf{W}_\sigma$ and bias b_σ of each precision estimation module using closed-form gradient expressions derived from a precision-weighted prediction error loss, without invoking automatic differentiation.

Claim 2. The system of Claim 1, wherein the bias term b_σ of each precision estimation module is initialised according to a depth-dependent initialisation scheme, such that for a processing unit at index i in a hierarchy of N units:

$$\sigma^2_{\text{init},i} = \sigma^2_{\text{base}} \times \left(1 + \alpha_{\text{depth}} \times \frac{i}{\max(N-1,1)}\right)$$

$$b_{\sigma,i} = \ln\left(e^{\sigma^2_{\text{init},i}} - 1\right)$$

wherein σ^2_{base} is a base initial precision value and α_{depth} is a depth scaling factor, such that shallow processing units are initialised with lower uncertainty than deep processing units.

Claim 3. The system of Claim 1, wherein the plurality of processing units are organised into a plurality of precision bands, each precision band corresponding to a different temporal receptive field scale, comprising:

(i) a token band comprising shallow processing units, whose aggregate precision modulates local next-token prediction;

(ii) a phrase band comprising intermediate-shallow processing units, whose aggregate precision modulates lateral inter-unit communication;

(iii) a sentence band comprising intermediate-deep processing units, whose aggregate precision modulates episodic memory encoding thresholds; and

(iv) a discourse band comprising deep processing units, whose aggregate precision modulates global workspace broadcasting access.

Claim 4. The system of Claim 1, wherein the re-entrant inference controller is configured to:

(i) perform a minimum of K_{min} inference iterations before testing for convergence;

(ii) terminate inference iterations when a relative change in total prediction error falls below a convergence threshold τ_{conv}; and

(iii) enforce a maximum of $K_{\text{max}} + K_{\text{extra}}$ inference iterations, where K_{extra} is determined by the conflict detection module.

Claim 5. The system of Claim 1, wherein the conflict detection module computes the conflict score as:

$$C = \frac{1}{D} \sum_{d=1}^{D} \text{Var}_{i \in \{0,\ldots,N-1\}} \left[\bar{h}_i[d] \right]$$

where $\bar{h}_i$ is the mean output of processing unit i across batch and sequence dimensions, and allocates extra inference steps according to:

$$K_{\text{extra}} = \min \left(K_{\text{max,extra}}, \left\lfloor \frac{C - \tau_C}{\tau_C} \times K_{\text{max,extra}} \right\rfloor \right)$$

where τ_C is the conflict threshold and $K_{\text{max,extra}}$ is the maximum number of additional steps.

Claim 6. The system of Claim 1, wherein the analytical precision learning module computes the gradient of the precision-weighted prediction error loss with respect to the pre-activation z as:

$$\frac{\partial \mathcal{L}}{\partial z} = \left(\frac{-\|\epsilon\|^2}{2 \cdot (\sigma^2)^2} + \frac{1}{2 \cdot \sigma^2} \right) \times \text{sigmoid}(z)$$

and updates the weight matrix as:

$$\Delta \mathbf{W}_\sigma = \frac{1}{N} \left(\frac{\partial \mathcal{L}}{\partial z} \right)^\top \hat{\mathbf{H}}$$

where $\hat{\mathbf{H}}$ is the matrix of normalised hidden states, ϵ is the prediction error, σ^2 is the current precision estimate, and $\text{sigmoid}(z)$ is the derivative of the softplus function.

Claim 7. The system of Claim 1, further comprising a dendritic modulation mechanism within each processing unit, wherein the precision estimate σ^2 from a preceding processing unit modulates the dendritic gate logit of the current processing unit via a learned linear projection, such that high preceding uncertainty loosens the dendritic gate and low preceding uncertainty tightens it.

Claim 8. The system of Claim 1, wherein the conflict detection module further maintains exponential moving averages of per-unit precision estimates and monitors for:

(i) precision divergence, wherein a unit's exponential moving average of σ^2 exceeds an upper threshold, triggering scaling of the unit's output weights;

(ii) precision collapse, wherein a unit's exponential moving average of σ^2 falls below a lower threshold, triggering an increase in the unit's bias term b_σ; and

(iii) persistent non-convergence, wherein the inference loop consistently reaches the maximum step count.

Claim 9. A method for performing precision-weighted predictive coding inference in a hierarchical neural network comprising a plurality of processing units, the method comprising:

(a) performing an initial bottom-up pass through the plurality of processing units, wherein each processing unit:

(i) computes an output representation from its input;

(ii) generates a top-down prediction of the input to the next lower processing unit via a learned linear projection $\mathbf{W}_{\text{predict}}$; and

(iii) computes a content-dependent precision estimate σ^2 by applying a learned linear projection $\mathbf{W}_\sigma$ to a normalised hidden state and passing the result through a softplus activation function, wherein the weight matrix $\mathbf{W}_\sigma$ is initialised with non-zero random values;

(b) computing prediction errors at each level of the hierarchy as the difference between actual inputs and generated predictions;

(c) computing a conflict score as the variance of mean output representations across the processing units;

(d) determining a number of additional inference iterations based on whether the conflict score exceeds a predetermined threshold;

(e) performing a plurality of re-entrant inference iterations under a gradient-free computational regime, each iteration comprising:

(i) assembling a top-down context signal for each processing unit;

(ii) re-processing each processing unit with both bottom-up input and the top-down context signal;

(iii) recomputing prediction errors; and

(iv) testing for convergence based on relative change in total prediction error;

(f) computing analytical gradients of a precision-weighted prediction error loss with respect to the parameters $\mathbf{W}_\sigma$ and b_σ without invoking automatic differentiation, using the closed-form expression:

$$\frac{\partial \mathcal{L}}{\partial z} = \left(\frac{-\|\epsilon\|^2}{2 \cdot (\sigma^2)^2} + \frac{1}{2 \cdot \sigma^2} \right) \times \text{sigmoid}(z)$$

where z is the pre-activation, ϵ is the prediction error, and $\sigma^2 = \text{softplus}(z) + \epsilon$; and

(g) updating $\mathbf{W}_\sigma$ and b_σ using the computed analytical gradients.

Claim 10. The method of Claim 9, further comprising initialising the bias term b_σ of each processing unit according to a depth-dependent scheme wherein shallow processing units receive

a lower initial bias value corresponding to lower initial uncertainty and deep processing units receive a higher initial bias value corresponding to higher initial uncertainty, according to:

$$b_{\sigma,i} = \ln\left(\exp\left(\sigma^2_{\text{base}} \times \left(1 + \alpha_{\text{depth}} \times \frac{i}{\max(N-1,1)}\right)\right) - 1\right)$$

where i is the index of the processing unit in the hierarchy and N is the total number of processing units.

Claim 11. The method of Claim 9, further comprising organising the plurality of processing units into a plurality of precision bands and routing the aggregate precision of each band to a distinct downstream subsystem, wherein:

(i) a sentence-level precision band modulates a threshold for episodic memory encoding; and

(ii) a discourse-level precision band modulates access to a global workspace broadcasting mechanism.

Claim 12. The method of Claim 9, wherein performing the plurality of re-entrant inference iterations comprises:

(i) performing at least K_{min} iterations before evaluating the convergence criterion;

(ii) terminating iterations when the relative change in total prediction error falls below a threshold τ_{conv}; and

(iii) limiting the total number of iterations to $K_{\text{max}} + K_{\text{extra}}$, where K_{extra} is determined by the conflict score.

Claim 13. The method of Claim 9, wherein computing the analytical gradients comprises:

(i) computing the mean squared prediction error $\|\epsilon\|^2$ as the mean over the model dimension to ensure gradient scale invariance with respect to model dimensionality;

(ii) computing the derivative of the loss with respect to σ^2 as $-\|\epsilon\|^2/(2 \cdot (\sigma^2)^2) + 1/(2 \cdot \sigma^2)$;

(iii) multiplying by $\text{sigmoid}(z)$, clamped to the interval $[0.01, 0.99]$, to obtain the derivative with respect to the pre-activation; and

(iv) computing the weight gradient as the outer product of the per-token gradient scalars with the normalised hidden states, averaged over all tokens.

Claim 14. The method of Claim 9, further comprising updating the prediction weights $\mathbf{W}_{\text{predict}}$ of each processing unit via an analytical precision-weighted gradient:

$$\Delta\mathbf{W}_{\text{predict}} = -\frac{1}{N}\left(\frac{\epsilon}{\sigma^2}\right)^{\top} \hat{\mathbf{H}}$$

wherein predictions from high-precision units receive proportionally larger gradient updates than predictions from low-precision units.

Claim 15. The method of Claim 9, further comprising passing the precision estimate σ^2 from each processing unit to the next processing unit in the hierarchy, wherein the precision estimate modulates a dendritic gate of the receiving processing unit via a learned linear projection, enabling computation to adapt based on the certainty of the preceding unit.

Claim 16. A non-transitory computer-readable medium storing instructions that, when executed by one or more processors, cause the one or more processors to:

(a) instantiate a hierarchical neural network comprising a plurality of processing units, each processing unit including:

(i) a precision estimation module comprising a learned linear projection $\mathbf{W}_\sigma$ with non-zero initialised weights and a softplus activation, configured to compute a content-dependent per-token precision estimate σ^2; and

(ii) a prediction generation module comprising a learned linear projection $\mathbf{W}_{\text{predict}}$ configured to generate top-down predictions;

wherein the bias term b_σ of each precision estimation module is initialised according to a depth-dependent scheme producing monotonically increasing initial uncertainty from shallow to deep processing units;

(b) perform a bottom-up forward pass through the processing units, computing output representations, predictions, and precision estimates;

(c) compute prediction errors at each hierarchical level;

(d) compute a conflict score based on inter-unit output variance and determine additional inference iterations;

(e) perform re-entrant inference iterations under a gradient-free regime, refining representations through top-down prediction and bottom-up error correction, until convergence or a maximum step count;

(f) update the precision parameters $\mathbf{W}_\sigma$ and b_σ using analytical gradient expressions computed as:

$$\frac{\partial \mathcal{L}}{\partial z} = \left(\frac{-\|\epsilon\|^2}{2 \cdot (\sigma^2)^2} + \frac{1}{2 \cdot \sigma^2} \right) \times \text{sigmoid}(z)$$

without constructing or traversing an automatic differentiation graph; and

(g) route aggregate precision values from different hierarchical bands to distinct downstream subsystems, including episodic memory encoding and global workspace broadcasting.

Claim 17. The non-transitory computer-readable medium of Claim 16, wherein the instructions further cause the one or more processors to organise the processing units into four precision bands corresponding to token-level, phrase-level, sentence-level, and discourse-level temporal receptive fields, wherein each band's aggregate precision is computed as the mean σ^2 across all units in that band.

Claim 18. The non-transitory computer-readable medium of Claim 16, wherein the instructions further cause the one or more processors to maintain exponential moving averages of per-unit precision estimates and to take corrective action when a unit's precision diverges above an upper threshold or collapses below a lower threshold.

Claim 19. The non-transitory computer-readable medium of Claim 16, wherein the gradient-free regime for re-entrant inference iterations comprises executing all computations within the iterative loop without constructing a computational graph for automatic differentiation, such that no intermediate activations are stored for a backward pass during the inference iterations.

Claim 20. The non-transitory computer-readable medium of Claim 16, wherein the depth-dependent initialisation scheme sets the initial precision value for processing unit i in a hierarchy of N units to:

$$\sigma^2_{\text{init},i} = \sigma^2_{\text{base}} \times \left(1 + \alpha_{\text{depth}} \times \frac{i}{\max(N-1, 1)}\right)$$

such that the shallowest processing unit has initial precision σ^2_{base} and the deepest processing unit has initial precision $\sigma^2_{\text{base}} \times (1 + \alpha_{\text{depth}})$.