ChatGPT

Holographic Multi-Scale Attention Framework

Core Idea

The core idea is to design an **attention mechanism** inspired by holographic codes in theoretical physics. In AdS/CFT duality, information in a higher-dimensional "bulk" can be redundantly encoded on a lower-dimensional "boundary" via a quantum error-correcting code [1] [2]. We propose treating the attention module as a *bulk–boundary mapping*: a compact **central latent representation** ("minimal seed") in the "bulk" that projects out to richly detailed, redundant features on the "boundary". This draws on the HaPPY code model [3] [4], where a single logical qubit's information is spread across many physical qubits on the boundary – analogous to how a focused idea might influence many neurons or tokens. The attention weights serve as **isometric encoders**, ensuring that key information in the core can be reconstructed from multiple subsets of peripheral details (akin to complementary recovery in holography [5]). In effect, the attention mechanism doesn't just select features – it **encodes high-level content into lower-level representations with built-in redundancy**, much like a tensor-network quantum code that preserves logical information in entangled physical degrees of freedom.

This framework naturally implies a **hierarchical, multi-scale structure**. Rather than a single flat attention map, we envision nested layers or "rings" of attention spanning different granularities. The inner-most layer (small radius) carries abstract, coarse content (e.g. global context or gist), while outer layers add finer details [6]. Each outward step from the core increases the representation's size (more tokens/nodes) and specificity, analogous to zooming in on an image or adding higher-frequency components. Conceptually, this resembles **renormalization group flow**: the core interacts with successive layers that refine it, similar to how MERA tensor networks add detail at each scale [7]. The brain's attention mechanisms offer a helpful metaphor here – higher cortical areas bias the processing of lower areas (top-down attention) while sensory inputs compete for representation (bottom-up) [8]. Our model mirrors this: the *core latent* exerts top-down influence via attention weights, biasing which boundary features become amplified, while bottom-up competition among boundary elements determines what enters the core. Neuroscience also shows that attention works via **gain modulation** (multiplicative scaling of neuron responses to attended features [9]); in our framework the attention weights modulate how strongly each "boundary" unit (token/feature) is influenced by or contributes to the core. In summary, the core idea is a *holographic attention code* that unifies: (1) **holographic error-correction** for robust, redundant encoding of concepts; (2) **multi-scale attention** akin to a tensor network with concentric layers; (3) a **biased competition model** where a central "focus" steers the processing of distributed inputs.

Candidate Models

To ground this concept, we identify models and principles across disciplines that align with the above vision:

- **Holographic Error-Correcting Codes:** *Pastawski et al.'s HaPPY code* is a prime example [3]. It tessellates a hyperbolic disk with perfect tensors, yielding an isometric mapping from one **bulk qubit** to many **boundary qubits** [1]. The code's properties (logical info recoverable from multiple boundary regions, resilience to erasures) demonstrate how central data can be redundantly

distributed [10]. This inspires an attention mechanism that treats the latent query/key as "bulk" information and the attended features as "physical" encodings – ensuring the model can recall the core concept even if some inputs are missing or noisy (analogous to error-correction). We can draw on tensor network constructions (e.g. perfect tensors and **entanglement wedges**) to design attention layers that mimic this redundancy and locality (each tensor only connects locally in the network).

- **Tensor Networks & Multi-Scale Representations:** The **MERA (Multi-Scale Entanglement Renormalization Ansatz)** provides a template for hierarchical information flow. MERA is a tensor network with interleaved *disentanglers* and *isometries* that systematically reduce dimensionality across scales [11]. In MERA, each layer captures correlations at a certain length-scale while preserving an isometric mapping to the next layer. Translating this to machine learning, we consider **sparse or hierarchical attention** architectures: layers of attention could progressively aggregate information (coarse-grain) and then inject detail (fine-grain) analogous to MERA's up-down mapping. Notably, a recent multi-scale tensor network model was used for supervised learning by wavelet-based coarse-graining [12], hinting that such structures can be adapted to neural architectures. We aim to implement attention not as one monolithic $N!\times!N$ operation, but as a *cascade* of localized attentions on different resolutions – effectively a learned **attention pyramid** or tree.

- **Transformer Attention Variants:** Modern transformers already hint at multi-scale processing through various adaptations:

- *Standard Multi-Head Self-Attention* provides a flexible all-to-all connectivity, but with quadratic cost and no explicit hierarchy. It's the "flat" limit of our framework (all "rings" collapse into one). The need to handle long sequences has driven innovation in structuring attention.
- *Sparse & Hierarchical Transformers:* **Longformer** and **BigBird** introduce local windowed attention plus a few global tokens [13] [14]. This effectively creates two scales: a fine-grained local context and a coarse global summary. Longformer's design ("sliding window + global tokens") is analogous to each segment of text attending within itself (like boundary patches) while a few nodes attend globally (like a core summary) [13]. Similarly, **Hierarchical BERT** models and *chunk-based* transformers process text in segments, then aggregate segment representations for higher-level processing [15]. These can be seen as early steps toward our nested attention rings – they partition and prioritize information by scale.
- *Latent-Variable Models (Perceiver, Perceiver IO):* The **Perceiver** architecture explicitly uses a small set of learned latent vectors that attend to the input, rather than full pairwise attention on the input itself [16]. This approach **"decouples the size of the latent core from the input size"** [17], very much in line with our bulk-boundary idea. The latent array in Perceiver can be thought of as the bulk state, and cross-attention maps the high-dimensional input (boundary) onto this latent. After several latent self-attention updates, the model can decode back to a desired output modality. The Perceiver's success across images, audio, and language suggests that focusing computation on a compact core while sparsely reading/writing from the periphery is a powerful pattern – one that our framework generalizes with the lens of holographic coding (ensuring information isn't just compressed but *redundantly encoded* across the "boundary").

- *Recurrent Memory Transformers:* Efforts like *RMT (Recurrent Memory Transformer)* introduce a recurrent state that carries information across segments, effectively creating an inner loop of information that persists while new input chunks are processed [18]. This "memory state" is analogous to our core

latent – continuously updated and error-corrected as new boundary data comes in. It showcases how a persistent bulk can integrate sequential inputs without full recomputation.

- **Neuroscience Attention Models:** Biological attention provides qualitative guidance for mechanism design:

- The **Biased Competition model** posits that multiple stimuli compete for neural representation and attention biases this competition in favor of the task-relevant stimulus [8]. If we map "stimuli" to tokens/features in a network, our core latent (task context or query) provides the bias – via attention weights – to amplify certain token interactions while damping others. Mechanistically, this could be implemented by modulating key/query vectors or gating certain attention heads, echoing how neuromodulators modulate synaptic gains.
- **Feature Similarity Gain Modulation (FSGM):** Experiments show that neurons tuned to an attended feature increase their firing, whereas those tuned to ignored features suppress firing – and this effect is multiplicative on the neural responses [9]. In our framework, this is analogous to **multiplicative attention weights** that scale output contributions. We could incorporate feature-based gating in the attention layers: e.g. an attention head that selects for a certain feature will boost values aligned with that feature and attenuate others, implementing a form of gain modulation.
- **Dynamic Routing and Global Workspaces:** Cognitive theories like the *Global Workspace* suggest a small subset of information becomes globally broadcast (through attention or synchronous firing) to coordinate brain-wide processing. Our "bulk" latent can be seen as this workspace – a bottleneck that, once a concept enters, can influence all peripheral components (the broadcast). Conversely, the competition among many modules to enter the global workspace mirrors how boundary elements compete for access to the core latent. We can draw algorithmic inspiration from *routing networks* (e.g. Capsule Networks' routing-by-agreement) to decide which signals get promoted to the next ring or to the core.

Combining these models, we converge on a **candidate architecture**: a *nested attention network*. Imagine a central latent vector (or set of vectors) that interacts with the first layer of inputs via attention, producing an enriched representation; this then serves as query for a second layer of wider context, and so on, expanding outward. Each layer could be a Transformer-style attention block but constrained to a subset of tokens or a specific scale. Notably, at each expansion, one could include an **error-correcting constraint** (for instance, a penalty if the core representation can't be reconstructed from any one segment of that layer's outputs, enforcing redundancy). The outcome is a deep network resembling a **concentric tensor network**, where each concentric layer is an attention-mediated code that adds resolution while preserving the core content.

Implications for UI/Architecture

Conceptual representation of a nested-ring attention architecture: a central core (blue) encodes high-level information, surrounded by successive layers (yellow, green, orange) that add increasing detail or context.

System Architecture: Adopting this framework implies a non-standard neural architecture. We will need to implement **hierarchical attention modules** that pass information radially. One potential design is an "onion model" where the innermost layer is a small Transformer (operating on the core latent and perhaps a very coarse summary of input), which then feeds into a larger Transformer that attends to more fine-grained input features, and so on. This differs from typical stack-of-identical-layers: here each layer operates at a different *scope*. Architecturally, we must manage connections such that *inner layers inform outer layers* (top-down influence) and *outer layers feed refined information back inward* (updating the core). This could be realized with a **bidirectional attention flow**: e.g. each layer performs a round of attention *from core to boundary* (distributing the core info as queries to refine the boundary representation) and a round of attention *from boundary to core* (aggregating updated information back into the core). The **UI/UX** of such a system (if exposed to users or developers) might emphasize interpretability of each layer – we could present the model's understanding at different levels of abstraction. For instance, a debugging interface might show the "blue core" focusing on a concept, the yellow ring capturing broad groups of tokens (like topics or sections of a document), the green ring focusing on specific details within those groups, etc. This layered explanation could make large model decisions more transparent by revealing *which scale of representation* influenced an output.

From an engineering perspective, **efficiency and memory** usage would improve: the core latent is small, and each layer only attends within a limited scope. This addresses the $O(N^2)$ attention problem by breaking a huge attention map into manageable pieces (similar to how **Longformer** and others achieve linear scaling [13]). The trade-off is increased complexity in orchestrating layers and ensuring the whole is trainable end-to-end. We may need to incorporate *inter-layer loss functions* (for example, an autoencoding loss to ensure the core can reconstruct important inputs, reinforcing the error-correcting metaphor).

Additionally, because information is processed in stages, **latency** could be an issue (sequential dependence of layers), though techniques like pipelining or parallel coarse/fine updates might mitigate this.

User Interface / Visualization: If we imagine an application of this model (say, a knowledge graph navigator or a document analyzer), the UI could explicitly leverage the nested structure. For example, consider a text analytics tool: the UI might display a central highlight (the model's inferred main topic or question), around it clusters representing subtopics or document sections, and further out specific facts or entities. This *radial layout* is a natural way to visualize hierarchical attention. The user could click on an outer ring item to see how it connects to the core (tracing attention links), akin to selecting a detail to see why it's relevant to the main idea. Another UI idea is a **"focus+context" lens**: as the user selects a core concept, the interface brings in the most related peripheral details (from various parts of the data) due to the model's holographic encoding – much like how our attention jumps to relevant details across a scene when we focus on a thought. Because the model is inherently multi-scale, the UI might allow *zooming in and out*: zoom out to see broad themes (model's core representation), zoom in to see fine details (model's boundary representation), with smooth transitions (the intermediate rings). In terms of developer tools, visualizations of the attention tensor network (perhaps drawn as concentric rings or a hyperbolic tiling graph) would help in understanding and debugging the training process. Each node or region's opacity/size could indicate its attention weight or information content, providing an intuitive view of how information flows from seed to full representation.

Technically, implementing the UI for this requires exposing the model's intermediate states. We would design **introspection hooks** at each attention layer to extract what concept is represented and how it's linked to the next layer. This also suggests an *interactive training interface*: one could imagine guiding the model by "pinning" certain core concepts to desired boundary outcomes (a bit like concept intervention at different layers). In summary, the architecture invites UIs that are **radial, multi-resolution, and interactive**, turning the normally opaque attention weights into something like a navigable map of knowledge.

Next Steps

1. **Formalize Theoretical Mapping:** We will start by defining a clear mathematical mapping between holographic code elements and an attention network. This includes choosing a specific **tensor network layout** (e.g. tree vs. hyperbolic tiling) for the attention layers and formulating how queries/keys/values correspond to tensor indices. A small-scale analytical example (perhaps a toy "perfect tensor" attention module for a tiny bulk and boundary) can validate the error-correcting property – e.g. show that the core latent can be recovered from various subsets of the boundary outputs.

2. **Prototype a Minimal Model:** Develop a minimal working model with one core vector and one or two nested attention rings. This could be implemented by modifying a transformer: for instance, use a small **learnable latent** (core) that attends to input chunks and produces outputs. We'll verify that this model can at least match a standard transformer on simple tasks, and examine its attention patterns. The pseudocode below sketches the forward structure:

```
# Pseudocode: expand a minimal seed through nested attention rings
core = encode(seed_input)               # bulk encoding of minimal seed
for layer in range(1, N_layers+1):
```

```
    context = get_boundary_chunk(layer) # e.g., chunk of input or features at
this layer
    # Bidirectional attention between core and context:
    context = Attention(query=core, keys=context, values=context)
    core    = Attention(query=context, keys=core, values=core)
    core    = update(core, context)     # integrate new details into the core
(e.g., concat or add)
output = decode(core)                   # final output from the enriched core
```

This outline will be refined through implementation. Even at this stage, visualization of the attention weights can confirm if the core truly behaves like a "focus": does it attend broadly on the first pass and more narrowly on later passes? We'll test variations (e.g. one-way attention from core→context only) to see what works best.

1. **Incorporate Sparse/Hierarchical Attention Mechanisms:** Building on the prototype, integrate known techniques for scalable attention. For example, use *local attention* within each layer's context and only a few *global tokens* representing the core [13]. We should also explore **iterative refinement**: allow the core to update, then re-attend the same layer (like a message-passing iteration) which resonates with *capsule routing* and could improve consistency between layers. Performance and memory profiling at this step will guide us in tuning the layer sizes and how many tokens the core should interact with at once.

2. **Error-Correction and Robustness Tests:** A key claim of the framework is improved robustness via redundancy. We will design experiments (e.g., mask out or corrupt a fraction of the "boundary" inputs) to see if the model's core representations and outputs degrade gracefully compared to a standard transformer. Metrics like accuracy vs. fraction of missing tokens, or the model's ability to reconstruct missing content, will quantify the error-correcting behavior. If needed, introduce an explicit loss term that encourages reconstructing inputs from the core (turning parts of the network into an autoencoder) to enforce the property.

3. **Neuroscience-Inspired Enhancements:** Test features like **feature-based gates or biases** in attention. For instance, implement a mechanism where the core provides a feature query that gates the value update (mimicking gain modulation). Measure if this improves the model's ability to focus on relevant features in noisy data (as biased competition would predict [8]). We might also simulate a "dual-task" scenario to see if the core can dynamically route information to different output tasks, analogous to executive control in the brain [19].

4. **Visualization & UI Prototyping:** Simultaneously, begin developing visualization tools for the model's internals. A simple radial graph that shows attention link strengths between core and boundary (perhaps using D3 or an interactive Python notebook) would validate our conceptual UI approach. We will create a few schematic diagrams (like the concentric rings figure) to include in our design notes and potentially as figures in any paper or documentation. If the model is destined for a user-facing application, engage with a UI/UX designer to iterate on how end-users might toggle "focus modes" or see multi-scale outputs. Early feedback on these visual metaphors (e.g., do users find the ring layout intuitive?) can inform adjustments to the model (for example, ensuring each ring has a clear semantic meaning that can be labeled in the UI).

5. **Scaling Up and Integration:** Finally, plan the path to scale the model to realistic tasks (long documents, large images, etc.). This involves increasing the number of layers (rings) and the size of each, possibly using techniques like **parameter sharing** or **progressive training** (first train inner layers on small data, then gradually add outer layers). We'll also consider how to integrate this with existing infrastructure – e.g., could we fine-tune a pre-trained transformer to adopt a bulk-boundary partition of its attention? Collaboration with domain experts (NLP, vision) will ensure our architecture is tuned to the nuances of those modalities (text may need a different tiling strategy than images, for instance). Throughout, we'll keep the interdisciplinary narrative strong: the end goal is not just a high-performing model, but one that **embodies principles of physics and cognition** – offering a new lens to understand deep learning systems. Our next steps will thus be evaluated not only by conventional metrics, but also by how well the resulting system aligns with holographic and attentional theories, potentially yielding new insights across those fields.

---

1 4 10 [1503.06237] Holographic quantum error-correcting codes: Toy models for the bulk/boundary correspondence
https://arxiv.org/abs/1503.06237

2 5 Holographic codes from hyperinvariant tensor networks | Nature Communications
https://www.nature.com/articles/s41467-023-42743-z?error=cookies_not_supported&code=534715b7-bbc8-4dd2-80e6-f13930933df4

3 Pastawski-Yoshida-Harlow-Preskill (HaPPY) code | Error Correction Zoo
https://errorcorrectionzoo.org/c/happy

6 7 arxiv.org
https://arxiv.org/pdf/2005.05971

8 9 19 Frontiers | Attention in Psychology, Neuroscience, and Machine Learning
https://www.frontiersin.org/journals/computational-neuroscience/articles/10.3389/fncom.2020.00029/full

11 MERA: Multi-scale Entanglement Renormalization Ansatz
https://www.emergentmind.com/topics/multi-scale-entanglement-renormalization-ansatz-mera

12 [2001.08286] A Multi-Scale Tensor Network Architecture for Classification and Regression
https://arxiv.org/abs/2001.08286

13 14 15 The Flaws and Future of Transformer Self-Attention: A Path to Performance Breakthroughs and Cognitive Remodeling | by InverseAtom | Oct, 2025 | Medium
https://medium.com/@inverseatom.ai/the-flaws-and-future-of-transformer-self-attention-a-path-to-performance-breakthroughs-and-b468c01d9749

16 Perceiver IO: a scalable, fully-attentional model that works on any ...
https://huggingface.co/blog/perceiver

17 [PDF] Perceiver IO: A General Architecture for Structured Inputs & Outputs
https://www.drewjaegle.com/pdfs/jaegle2021perceiver_io.pdf

18 Recurrent Memory-Augmented Transformers with Chunked ... - arXiv
https://arxiv.org/html/2507.00453v1