Module A: Input Processing

Part of the Eidos Unified Framework for Persistent, Dynamic, and Adaptive Multimodal Intelligence

—

Abstract

This module describes the *Input Processing* component of the Eidos framework. Its purpose is to acquire raw data from external sources (e.g., text, images, sensor signals) and preprocess it into a standardized format suitable for downstream processing. We detail the raw input, the preprocessing function, and the transformation into a canonical form. This module forms the essential first step in the Eidos pipeline, ensuring that all subsequent modules receive clean and consistent input.

Contents

1 Introduction and Motivation

In any advanced intelligent system, the quality and consistency of input data are critical. The **Input Processing** module of the Eidos framework is responsible for acquiring raw data from diverse sources and transforming it into a standardized representation. This ensures that downstream components—such as tokenization, embedding, and deep models—operate on consistent, noise-reduced data. The primary objectives of this module are:

(a) To standardize heterogeneous raw inputs (e.g., text, images, sensor data) into a canonical format.

(b) To clean, normalize, and pre-process data so that subsequent modules can reliably extract features.

(c) To provide a well-defined interface for the conversion of raw input X_{raw} to processed input X_{proc}.

2 Preliminaries and Notation

- **Raw Input:** Let $X_{\text{raw}} \in \Sigma^*$ denote the raw input data, where Σ is the base alphabet or signal set (e.g., Unicode for text, pixel values for images).
- **Preprocessed Input:** We denote the standardized, cleaned input by $X_{\text{proc}} \in \mathcal{X}_{\text{proc}}$, where $\mathcal{X}_{\text{proc}}$ is the canonical data domain used by subsequent modules.
- **Preprocessing Function:** The transformation from raw to preprocessed data is performed by a function
$$\mathcal{P}_{\text{in}} : \Sigma^* \to \mathcal{X}_{\text{proc}}.$$

3 Formal Definitions and Mathematical Formulation

Definition 1 (Preprocessing Function)

We define the preprocessing function $\mathcal{P}_{\text{in}}$ to be a mapping that cleans, normalizes, and standardizes raw input data:
$$\mathcal{P}_{\text{in}}(X_{\text{raw}}) = X_{\text{proc}},$$
where the operation includes:

(i) **Normalization:** Converting input data to a standard format (e.g., Unicode normalization for text).

(ii) **Noise Removal:** Eliminating extraneous symbols, correcting errors, and filtering irrelevant content.

(iii) **Formatting:** Structuring the data into a consistent format (e.g., tokenizable strings for text, standardized dimensions for images).

Properties

The function $\mathcal{P}_{\text{in}}$ is designed to be:

- **Deterministic:** Given the same X_{raw}, it always produces the same X_{proc}.
- **Robust:** Capable of handling various input anomalies and noise.
- **Modular:** Its output is in a format that is compatible with subsequent modules, ensuring loose coupling.

4 Algorithmic Description

Below is the pseudocode for the Input Processing module:

Algorithm 1 Input Processing

Input: Raw input $X_{\text{raw}} \in \Sigma^*$
Output: Preprocessed input $X_{\text{proc}} \in \mathcal{X}_{\text{proc}}$
Begin:
  Apply normalization: $X_{\text{norm}} \leftarrow \text{Normalize}(X_{\text{raw}})$
  Remove noise: $X_{\text{clean}} \leftarrow \text{NoiseRemoval}(X_{\text{norm}})$
  Format data: $X_{\text{proc}} \leftarrow \text{FormatData}(X_{\text{clean}})$
Return: X_{proc}

5 Theoretical Analysis and Guarantees

Theorem 1 (Determinism of $\mathcal{P}_{\text{in}}$)

Statement: For any fixed raw input $X_{\text{raw}} \in \Sigma^*$, the preprocessing function $\mathcal{P}_{\text{in}}$ produces a unique, well-defined output $X_{\text{proc}} \in \mathcal{X}_{\text{proc}}$.

Proof Sketch: Normalization, noise removal, and formatting are standard operations that are defined algorithmically. Each step is deterministic (given fixed parameters and rules). Therefore, the composition $\mathcal{P}_{\text{in}} = \text{FormatData} \circ \text{NoiseRemoval} \circ \text{Normalize}$ is deterministic. □

6 Integration with the Overall Eidos Framework

The Input Processing module is the first stage of the Eidos pipeline. Its outputs serve as the input for the Multidimensional Vocabulary and Tokenization system (Module D). By standardizing the raw data into a canonical form X_{proc}, this module ensures that all downstream components (including embedding, knowledge graph construction, and deep model processing) receive consistent and high-quality input.

7 Implementation Considerations

- The implementation of $\mathcal{P}_{\text{in}}$ should consider language-specific normalization rules, especially for multilingual data.
- For text, standard libraries (such as ICU for Unicode normalization) may be used.

- For other modalities (e.g., images or sensor data), analogous normalization and noise reduction techniques must be developed.
- The module should be highly optimized for streaming data, ensuring minimal latency for real-time applications.

8 Conclusion

The Input Processing module is a critical first component of the Eidos framework. It transforms raw input X_{raw} into a standardized, high-quality representation X_{proc} that is used throughout the system. Its deterministic, robust, and modular design ensures that all subsequent modules operate on clean data, thereby enhancing overall system performance and reliability.

Module Summary:
Completed: Module A – Input Processing.
Remaining Modules: Module B: Universal Communication & Data Handling Interface and Coordination,
Module C: Universal Streaming/Handling/Loading/Indexing Module,
Module D: Multidimensional Vocabulary and Tokenization System,
Module E: Contextual NLU/NLP Embedding and Multidimensional Tokenization,
Module F: Deep Knowledge Graphs System (Base and Personal),
Module G: Infinite RoPE Context Scaling and Dynamic Vocabulary Updating,
Module H: Core Model Architectures (RWKV and Transformer Modules, Mixture-of-Experts Style),
Module I: Titans Memory Architecture (Multi-Layer Memory Module),
Module J: Recursive Adaptive Dynamic Idempotent Feedback and State-Based Runtime Learning and Inference,
Module K: Universal Training System,
Module L: Final Decoding and Multimodal Output.