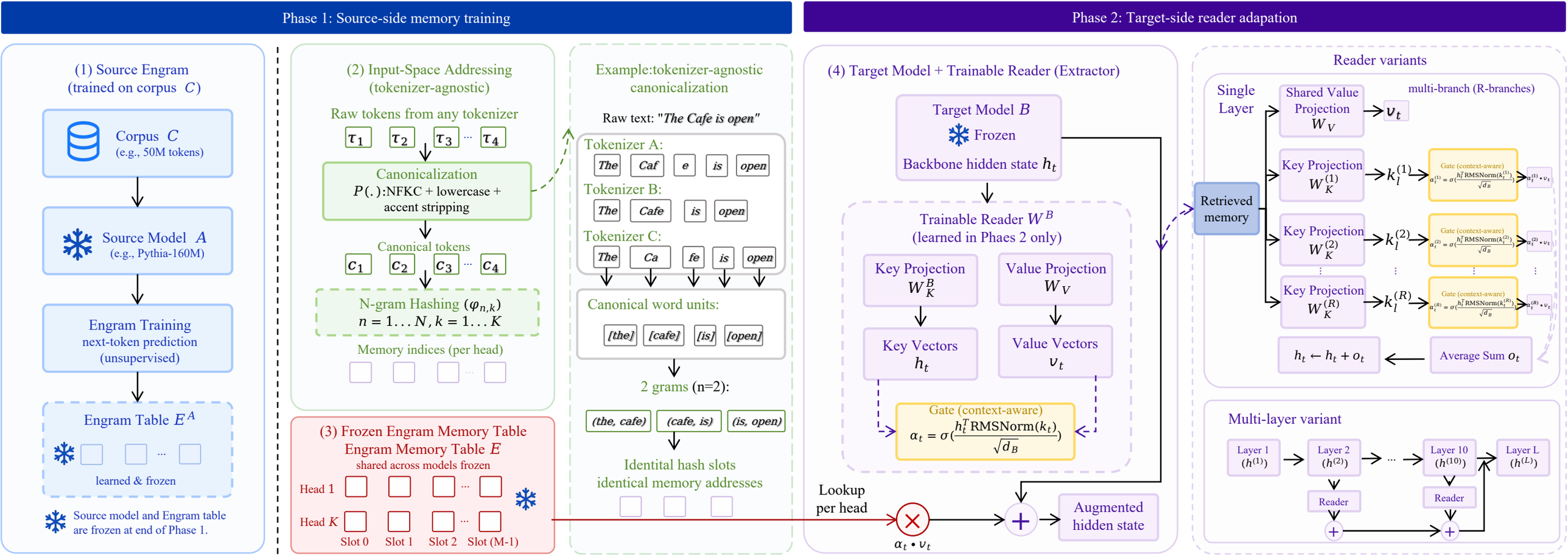
Phase 1: Source-side memory training
Phase 2: Target-side reader adapation
(1) Source Engram
(trained on corpus C)
Corpus C
(e.g., 50M tokens)
Source Model A
(e.g., Pythia-160M)
Engram Training
next-token prediction
(unsupervised)
Engram Table E^A
learned & frozen
Source model and Engram table are frozen at end of Phase 1.
(2) Input-Space Addressing
(tokenizer-agnostic)
Raw tokens from any tokenizer
Canonicalization
P(.):NFKC + lowercase + accent stripping
Canonical tokens
N-gram Hashing (φ_{n,k})
n = 1...N, k = 1...K
Memory indices (per head)
(3) Frozen Engram Memory Table
Engram Memory Table E
shared across models frozen
Head 1
Head K
Slot 0
Slot 1
Slot 2
Slot (M-1)
Example:tokenizer-agnostic canonicalization
Raw text: "The Cafe is open"
Tokenizer A:
Tokenizer B:
Tokenizer C:
Canonical word units:
2 grams (n=2):
(the, cafe)
(cafe, is)
(is, open)
Identital hash slots
identical memory addresses
(4) Target Model + Trainable Reader (Extractor)
Target Model B
Frozen
Backbone hidden state h_t
Trainable Reader W^B
(learned in Phaes 2 only)
Key Projection W_K^B
Value Projection W_V
Key Vectors h_t
Value Vectors v_t
Gate (context-aware)
α_t = σ(h_t^T RMSNorm(k_t) / √d_B)
Lookup per head
α_t • v_t
Augmented hidden state
Reader variants
Single Layer
multi-branch (R-branches)
Shared Value Projection W_V
Key Projection W_K^(1)
Key Projection W_K^(2)
Key Projection W_K^(R)
Gate (context-aware)
Retrieved memory
h_t ← h_t + o_t
Average Sum o_t
Multi-layer variant
Layer 1 (h^(1))
Layer 2 (h^(2))
Layer 10 (h^(10))
Layer L (h^(L))
Reader
Reader