SOPHIA Custom — Training Methodology Specification

Thomas Garren[1]

May 9, 2026

Abstract.

Abstract

This document specifies the training methodology used in SOPHIA Custom, the per-customer fine-tuning offering for retrieval-grounded masked-diffusion language model deployment. The methodology applies anchor-token masking — a training-objective modification empirically validated to produce a 1.67× pooled out-of-distribution generalization advantage over standard random-position masking on identical data — as the default fine-tuning regime, with V/O-only LoRA configuration and short training budgets (100-300 steps on 50-150 examples) chosen for the regime in which the technique is most effective. Each customer fine-tune produces a domain-specialized model adapter in 30-100 seconds on consumer-grade GPU hardware, with measurable held-out generalization advantages over vanilla LoRA fine-tuning.

---

1. Overview

SOPHIA Custom is a per-customer fine-tuning service for retrieval-grounded language model deployment. A customer supplies a knowledge corpus (technical documentation, knowledge base, structured records, etc.); SOPHIA Custom transforms that corpus into LTMi-XT format, trains a customer-specific LoRA adapter using the **anchor-token-masking V/O-only** technique on a base masked-diffusion language model, and deploys the adapter as a hosted inference endpoint. The resulting model produces grounded responses on the customer's domain with measurable generalization advantages over standard fine-tuning approaches.

This document specifies the training methodology used in the service. It is intended for technical buyers, integration partners, and researchers who want to understand or replicate the approach.

2. Pipeline

The end-to-end SOPHIA Custom pipeline:

```
Customer corpus (raw)
        │
        ▼
[1] Crystallization to LTMi-XT format
        │
        ▼
[2] (Q, A) pair generation from breadcrumb hierarchy
        │
        ▼
[3] V/O-only LoRA training with anchor-token masking
        │
        ▼
[4] Validation against held-out subset
        │
        ▼
[5] Deployment as customer-specific inference endpoint
```

Steps 1-2 transform the customer's content into a structured retrieval-grounded training substrate. Step 3 is the core training step. Steps 4-5 are operational.

3. Stage 1 — Crystallization

3.1 Input

Customer-supplied source material in any of: PDF, plain text, Markdown, HTML, JSON records, structured database export. Maximum size per ingest: 100 MB raw text equivalent.

3.2 Process

The crystallizer (an LLM-based pipeline using the LTMi-XT reference implementation) splits prose into self-contained atomic statements (loci). For each locus:

- **Single fact per locus** — compound claims are split
- **Pronouns resolved** — referents made explicit
- **Self-contained reading** — each locus stands alone without surrounding context
- **Source provenance preserved** — byte-offset back to original document
- **Confidence score** — crystallizer's confidence in faithfulness to source

3.3 Output

An LTMi-XT bundle (`.ltmi` file) containing typically 50-2000 atomic loci depending on input size.

3.4 Topology assignment

Each locus receives a four-level breadcrumb path: `[topic, subtopic, concept, claim]`. The breadcrumb is generated by the topologizer step using the LLM's understanding of the document's structural hierarchy. Lattice coordinates are then derived deterministically by hashing breadcrumb prefix levels (see LTMi-XT v0.1 spec).

4. Stage 2 — (Q, A) Pair Generation

For each locus in the crystallized bundle, three (Q, A) training pairs are generated by templated breadcrumb decomposition:

```
Q: "Tell me about {concept} in the context of {topic}."
A: <locus statement>

Q: "What does the source say about {concept}?"
A: <locus statement>

Q: "Regarding {topic}: {concept}?"
A: <locus statement>
```

This yields a training set of `3 × N_loci` (Q, A) pairs. For typical customer corpora this produces 150-6000 training pairs.

5. Stage 3 — V/O-only LoRA Training with Anchor-Token Masking

5.1 Training objective

Anchor-token masking restricts the diffusion-LM masking schedule to apply only to answer-statement (A) positions. Prefix question (Q) positions remain visible at every training step. Loss is computed only over masked answer positions.

Formally:

$$m_i = \begin{cases} 0 & \text{if position } i \in Q \text{ (anchor — never masked)} \\ \text{Bernoulli}(\gamma(t)) & \text{if position } i \in A \text{ (target — always subject to schedule)} \\ 0 & \text{if position } i \in \text{Pad} \end{cases}$$

Mask ratio $\gamma(t)$ sampled uniformly from $[0.5, 0.9]$ per training step.

5.2 V/O-only LoRA configuration

LoRA adapters wrap **only** the value (`v_proj`) and output (`o_proj`) projections. Query (`q_proj`) and key (`k_proj`) projections remain frozen at base weights. This configuration was empirically validated to outperform full LoRA (q/k/v/o wrapped) at half the trainable parameters [garren2026anchortokenmasking].

Parameter	Value
Wrapped projections	`v_proj` , `o_proj` only
LoRA rank	16
LoRA alpha	32
Trainable parameters	~3M (vs ~7M for full LoRA)
Initialization	A: Kaiming uniform; B: zero
Base model	frozen

5.3 Training hyperparameters

Hyperparameter	Value
Optimizer	AdamW
Learning rate	$1\times10^{-4} \rightarrow 1\times10^{-5}$ cosine decay
Betas	(0.9, 0.95)
Weight decay	0.01
Batch size	2 (microbatch)
Gradient accumulation	1-2 (effective batch 2-4)
Sequence length	192
Training steps	100-300 (default: 200)
Gradient norm clip	1.0
Random seed	configurable (42 default)
Mask ratio range	[0.5, 0.9] uniform

5.4 Training budget recommendations

The sample-efficiency curve from the validation work [garren2026anchortokenmasking] establishes that the anchor-token-masking advantage peaks at low training budgets and attenuates at high budgets due to overfitting on the focused training objective:

Customer corpus size	Recommended training steps	Expected wall-clock (3090 Ti)
< 50 unique loci	100 steps	~30 sec
50-150 loci	200-300 steps	~60-90 sec
150-500 loci	300-500 steps	~90 sec - 3 min
> 500 loci	500-1000 steps + early-stopping monitoring	3-10 min

For all corpus sizes: short training budgets are preferred. Held-out validation should monitor for advantage attenuation; if held-out generalization plateaus or degrades, training should be stopped.

5.5 Compute requirements

Configuration	VRAM	Wall-clock per fine-tune
RTX 3090 Ti (24 GB)	~3-5 GB used	30-100 seconds
RTX 4090 (24 GB)	~3-5 GB used	25-80 seconds
RTX A6000 (48 GB)	comfortable	25-80 seconds
H100 80GB	comfortable	15-50 seconds

V/O-only configuration leaves substantial memory headroom on consumer GPUs, enabling parallel training sessions on the same hardware (typically 2-3 concurrent fine-tunes per 24 GB GPU).

6. Stage 4 — Validation

Before deployment, each customer LoRA undergoes automated validation:

6.1 Held-out test set

A subset of the customer's loci (default: 10%, minimum 5 loci) is automatically held out from training and used as a held-out test set. The customer's questions about those held-out loci validate that the trained

model has acquired the topical inductive bias rather than just memorizing the training set.

6.2 Metrics

Metric	Threshold for deployment	Notes
Anchor preservation	100%	Guaranteed by forced-anchor decoding
Held-out corpus_overlap	> 0.20	Model produces topical content; below this indicates training failure
Held-out keyword hit	> 50%	Lenient metric checking expected keywords appear in output

If validation thresholds are not met, the fine-tune is automatically retried with adjusted hyperparameters (different mask ratio, longer training, etc.). Customer is notified if no configuration meets the thresholds.

6.3 Comparison baseline

For transparency, validation also runs a vanilla full-LoRA fine-tune on the same data and reports the comparative held-out metrics. Customers see the quantitative advantage of the anchor-token-masking V/O-only approach on their specific data.

7. Stage 5 — Deployment

7.1 Inference

The customer's trained LoRA is loaded onto a hosted inference endpoint. Inference uses **forced-anchor decoding** by default: when the customer's question matches a high-confidence retrieval target, the retrieved locus tokens are locked at fixed positions in the response slot, providing deterministic preservation of high-stakes identifiers (part numbers, citations, drug names, dates, etc.).

7.2 Endpoint formats

Endpoint	Purpose
`POST /api/v1/customer/{id}/query`	Standard query → grounded response
`POST /api/v1/customer/{id}/query/stream`	Streaming response
`POST /api/v1/customer/{id}/retrieve`	Retrieval-only (returns top loci, no generation)
`GET /api/v1/customer/{id}/loci/{locus_id}`	Direct locus inspection

7.3 Customer integration

Customer integration is a standard REST API call with API key authentication. Average integration time: 1-2 hours for an experienced developer.

8. Privacy and data handling

- **Customer corpus is processed in isolation**: no cross-customer data sharing
- **Crystallization may use third-party LLMs** (OpenAI, Anthropic, etc.) for the topologizer step; customer data flows through these APIs subject to their respective terms
- **Trained LoRAs are customer-specific**: deletion on customer request
- **Source provenance is preserved**: customer retains audit trail back to source byte offsets
- **Optional on-prem deployment**: full pipeline can run in customer infrastructure for sensitive data

9. Pricing and economics

(Subject to change; see sophiaxt.com/custom for current pricing.)

Tier	Setup	Per-call inference	Use case
Hobby	$99 one-time	$0.005 / 1K calls	Solo developers, side projects
Pro	$299 one-time	$0.005 / 1K calls + dedicated endpoint	Small teams, prod use
Scale	$999 one-time	Custom	High-volume programmatic use
Enterprise	Contact	Contact	On-prem, SLAs, custom integrations

The training compute cost per customer fine-tune (~$0.02-0.05 of GPU time) leaves substantial pricing margin even at the Hobby tier.

10. Empirical basis

The methodology specified in this document is based on empirical validation reported in [@garren2026anchortokenmasking]:

Finding	Result
Pooled OOD generalization advantage (anchor vs random)	1.67× (95% CI [1.51, 1.85])
OOD advantage statistically larger than ID advantage	CIs do not overlap
Direction of asymmetry is load-bearing	Reverse-asymmetric < random (P=0.006)
V/O-only outperforms full LoRA	0.416 vs 0.337 held-out overlap
Optimal training budget	100-300 steps on 50-150 examples

Customers' specific results will vary depending on corpus size, domain semantic distance from base model training, and other factors. SOPHIA Custom's automated validation (Stage 4) reports per-customer metrics for transparency.

11. Limitations and scope

- **The technique is best characterized as a low-shot fine-tuning advantage**, not a fundamental scaling advantage. Long training runs erode the anchor advantage.
- **Optimal regime is short fine-tunes on small-to-medium corpora**. Customers with very large corpora (>10K loci) may see attenuated benefits relative to standard approaches.
- **Tested only on masked-diffusion base models**; autoregressive base models would require a different (prefix-LM analog) training approach.
- **Tested at 1.3B base model scale**; behavior at frontier scale (>70B) is uncharacterized.

12. Versioning

This document specifies SOPHIA Custom training methodology **v1.0**.

Methodology updates will be reflected in subsequent versions. Customer deployments specify which methodology version they were trained with for reproducibility and audit purposes.

References

[garren2026anchortokenmasking] Garren, T. (2026). Anchor-Token Masking Produces an Out-of-Distribution Generalization Advantage in Retrieval-Grounded Masked-Diffusion Language Models. *Technical report, SOPHIA XT LLC.*

[garren2026ltmixt] Garren, T. (2026). LTMi-XT v0.1: A Retrieval Format with Hash-Derived Topological Indexing. *Technical specification, SOPHIA XT LLC.*

SOPHIA Custom Training Methodology Specification v1.0 · 2026-05-09 · SOPHIA XT LLC · CC BY 4.0 (this document) / Apache 2.0 (reference implementation).