

Io 3.5 Control Preview

25 September 2026

Abstract

Io is a family of small, specialist models. This paper introduces Io 3.5 Control Preview, the family's first member: a *decision model* that, rather than generating text, answers *typed questions* about a state and returns a probability distribution over each question's options together with a confidence. It targets computer use: given an accessibility-tree view of a desktop and a set of candidate actions, it identifies which action to take next as a calibrated distribution. We describe the model's architecture, its observation and action contract, and the released weights. This is a preview ahead of the full technical report, which will cover training data, objective details, and calibration methodology, and will include code for inference and fine-tuning. The model is already available on Perceptron Cloud via the API [2], and a playground is available in the Perceptron app [3].

1. Introduction

Io is a family of small, specialist models. Io 3.5 Control Preview is the family's first member, and is a *decision model*: rather than generating text, it answers *typed questions* about a state, returning a probability distribution over each question's options and a confidence. It performs no planning and produces no natural language, and is intended as the grounded *System 1* component beneath a larger *System 2* orchestrator that plans, retrieves memory, and verifies outcomes.

It targets *computer use*: its observations are accessibility-tree views of a desktop, its questions concern the candidate actions, and its answer is the action to take next, as a calibrated distribution. This paper covers the model, its architecture, its observation and action contract, and the released weights.

2. What Io 3.5 Control Is and Is Not

Io 3.5 Control Preview is a *decision scorer*. Every question returns a full probability distribution over its options, the selected option, and a confidence. Three question kinds are supported:

- **choice**: select one option from a set (for example, *which verb*, *which target*).
- **score**: assign a graded score to each option.
- **noul**: a binary question (yes / no).

It is *not* a chatbot, a planner, or a general-purpose model. It emits no prose, does not reason across many steps, and holds no memory.

3. The Decision Contract

An Io request pairs a *state* (a rendered textual observation) with a list of *typed questions*. Each question carries an id, a kind, an instruction, and an ordered list of options with human-readable descriptions. The response gives, per question, a distribution over its options, the selected option index, and a confidence. A sampling seed may be supplied so a decision replays exactly; otherwise a fresh seed is drawn and echoed back.

Because the contract is purely *state + questions → distributions*, one model serves many observations: the same weights answer a UI action question, a game move, or an imported decision task, provided the caller renders the state and enumerates the options.

4. Architecture

Io 3.5 Control is a *typed-decision scorer built on a frozen Qwen3.5-0.8B text backbone* [1].

Component	Parameters	State
Qwen3.5-0.8B text backbone	752,393,024	frozen
Adapter	21,645,312	trained
Total	774,038,336	

The backbone has 24 transformer layers at hidden size 1024 with *hybrid attention*: 18 Gated DeltaNet (linear-attention) layers and 6 full GQA layers, one full-

attention layer every fourth position. Full attention uses 8 query heads and 2 KV heads at head dimension 256; DeltaNet uses 16 key/value heads at head dimension 128 with convolution kernel 4. The feed-forward is SwiGLU with intermediate size 3584, normalization is RMSNorm, the vocabulary is 248,320 tokens, and embeddings are tied. The model is text-only; the base model’s vision tower and multi-token-prediction head are not built.

A rank-32 Adapter is attached to the attention q/k/v/o projections, the feed-forward gate/up/down projections, and the DeltaNet input/output projections. Only the adapter is trained; the backbone stays frozen.

To score a decision, the state and each question’s option list are serialized into a prompt and every option is assigned a reserved single-token code. At the answer position the model applies a softmax over the valid option codes, producing a normalized distribution over exactly the options the caller supplied.

5. The Control Domain

The observation is an *accessibility state*: the current task, the active window, the focused element, and a list of interface elements. Each element carries its role, label, value, checked state, an optional frame, its allowed actions, and an opaque snapshot-bound token used for grounding; the observation may also include orchestrator-retrieved memory and recent action history.

The action space is a set of desktop verbs: `click`, `double_click`, `right_click`, `type`, `set_value`, `select`, `toggle`, `scroll`, `keypress`, `wait`, `done`, `abort`, and `abstain`, each optionally bound to a target element and, for text entry, a text span.

`abstain` is always available, and the domain *fails closed*: an action targeting a safety-labelled destructive element is refused when the model reports that no destructive action is required.

6. Training

The adapter is trained with a supervised objective over the option codes on a curated mixture of decision data, and checkpoints are selected by a metric that balances choice accuracy against calibration rather than accuracy alone. The full recipe (data mixture, objective details, hyperparameters, and curriculum) is reserved for the complete technical report.

7. Calibration and Evaluation

Calibration is a first-class output: every answer carries a confidence calibrated per question type. As a preview, confidence should be monitored rather than trusted absolutely, especially out of domain.

8. Availability of Weights

The open-weights release consists of two artifacts:

- the *base model*, `Qwen/Qwen3.5-0.8B` [1], pinned to a fixed revision on HuggingFace;
- the trained *Adapter*, a single `model.io` checkpoint (86,661,571 bytes, 21,645,312 f32 parameters), published at

`https://perceptron.cloud/static/release/io/io-3.5-control/1315/model.io`

The checkpoint is self-describing: it carries a manifest with a tensor table (name, shape, offset), a state signature, a prompt/tokenizer hash pair, its license, and base-model attribution.

The weights are released under the *MIT license*.

9. Intended Use and Limitations

Io 3.5 Control is for grounded next-step decisions in computer-use interfaces observed as accessibility text. It is narrow, and designed to operate alongside a larger orchestrator rather than replace one.

- *Text-only*. It does not observe pixels; its view is the accessibility tree.
- *No planning*. It answers the question asked and holds no long-term memory.
- *Domain-scoped*. It is specialized for computer use; validate behaviour on unrelated states.
- *Supervise destructive actions*. Gate irreversible actions behind the `destructive` signal and caller confirmation.
- *Preview*. A research preview, not a production-hardened release.

References

1. Qwen Team. “Qwen3.5-0.8B.” Apache License 2.0. <https://huggingface.co/Qwen/Qwen3.5-0.8B>
2. Perceptron Cloud. “Decision API.” <https://perceptron.cloud/docs/decision>

3. Perceptron. "IO 3.5 Control playground." <https://perceptron.cloud/web/models/playground/bartmoss/io-3.5-control>