# SOLVER-JEPA: A JOINT-EMBEDDING PREDICTIVE ARCHITECTURE FOR VISUAL PROBLEM SOLVING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Visual problem solving requires transforming visually presented problems into structured solutions under global constraints and objectives. While methods learn to generate solution images or reasoning trajectories, visual problem solving still lacks a representation-learning objective designed to organize the information required for solution construction. Such learning is challenging because the visually observed structure and the underlying decision structure are not equivalent: small local differences can induce discontinuous changes in global feasibility or optimality, while a final solution image reveals only one successful endpoint and leaves the consequences and interactions of alternative decisions unspecified. We introduce **Solver-JEPA**, a solver-supervised joint-embedding predictive architecture that turns each problem into a family of decision-conditioned prediction tasks. During training, partial commitments specify tentative local decisions, and task solvers provide targets describing their consequences for problem structure, global feasibility, attainable solution quality, and compatible complete solutions. Predicting these targets in representation space encourages the visual encoder to preserve solution-relevant local-to-global dependencies. A separately trained decoder reuses the frozen representation to construct complete solution images without solver access at inference. Across four visual problem solving tasks, Solver-JEPA improves strict solution accuracy by up to $11.6$ percentage points and provides $15$–$150\times$ faster inference than generation-based baselines. Controlled ablations show that joint-embedding alignment and composite-commitment supervision improve the readability of complete solutions from the learned representations.

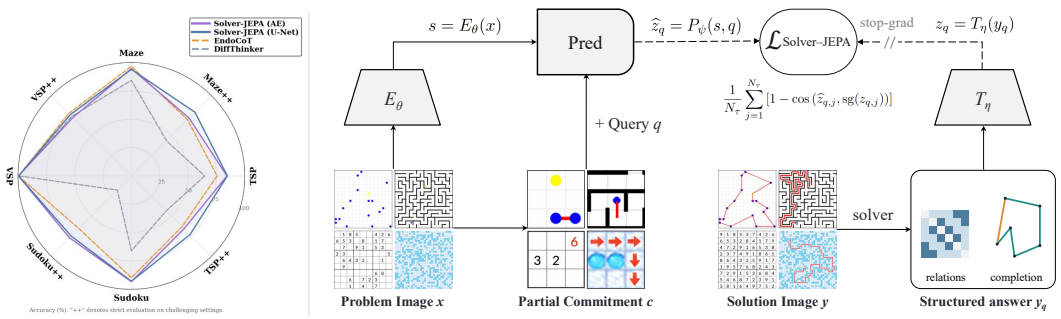
## 1 INTRODUCTION

Visual problem solving (VPS) aims to construct structured solutions from visually presented problems under task-specific constraints and objectives. Representative tasks span path planning (*e.g.*, maze navigation), combinatorial optimization (*e.g.*, the traveling salesman problem), and constraint satisfaction (*e.g.*, Sudoku). Beyond recognizing the elements in an image, solving these problems requires recovering their relational structure, evaluating interdependent decisions, and coordinating them into a globally valid or optimal solution. VPS therefore provides a controlled setting for studying how visual perception supports structured decision making, with related demands arising in multimodal reasoning (Team, 2025), visual navigation (Shah et al., 2023), robotic manipulation (Huang et al., 2023), and constraint-aware layout generation (Nauata et al., 2021).

Unlike visual recognition (Caron et al., 2021) and vision-language understanding (Radford et al., 2021), where representation learning provides a standard foundation for downstream models, VPS has largely been approached from the output side. Existing methods generate textual reasoning from visual inputs (**?**), construct plans as sequences of visual states (Xu et al., 2025), or produce complete solutions through visual or latent reasoning (He et al., 2025; Dai et al., 2026), but do not explicitly optimize the input representation for solution construction. Consequently, decision-relevant structure may remain entangled in the generation process rather than being exposed in a reusable latent state, coupling problem understanding with solution decoding. This coupling motivates a representation-first question for VPS: *what should a visual problem representation encode, and how can such a representation be learned?*



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.




**(a) Overall Performance** **(b) Training Process of Solver-JEPA**

Figure 1: **Overview. (a)** Overall performance across four VPS tasks, where "++" denotes strict evaluation on challenging settings. **(b)** Solver-JEPA encodes a problem image into a shared spatial representation and predicts solver-derived consequences conditioned on a partial commitment and query. Structured answers provide target representations for joint-embedding alignment.

Developing such a representation is challenging for three reasons. *First*, the observed visual structure and the underlying decision structure are asymmetric. Images expose appearance and spatial arrangement, whereas solving depends on implicit relations, constraints, feasible sets, objectives, and complete solution structure. A representation learned only within the visual domain may therefore capture how a problem looks without exposing the information needed to solve it. *Second*, VPS requires a different sensitivity profile from conventional visual representation learning. A small visual change (*e.g.*, modifying one wall in Maze or shifting one city in TSP) can decisively alter the underlying constraints, feasible solution set, or global optimum. A useful representation must therefore remain robust to irrelevant appearance variation while being highly sensitive to constraint-relevant differences. *Third*, VPS solutions are compositional structures formed by interdependent local components. Components that are individually admissible may become incompatible when combined, and correctness may emerge only at the level of the complete configuration. A useful representation must therefore expose structural compatibility, global feasibility, attainable solution quality, and compatible complete solutions.

To this end, we introduce **Solver-JEPA**, a solver-supervised joint-embedding predictive architecture with query-based prediction. It moves decision structure from terminal solution supervision into representation learning by predicting how local and joint constraints reshape the solution space. To bridge visual and decision structures, we adopt the JEPA framework (LeCun, 2022), using the problem image as context and solver-derived answers as semantic targets for solution-relevant structure and outcomes. This encourages the encoder to expose decision-side information rather than preserve visual appearance alone. Unlike canonical image (Assran et al., 2023) and video (Assran et al., 2025) JEPA formulations that primarily specify targets by spatial or temporal location, Solver-JEPA defines predictions through problem-solving semantics. A partial commitment induces a constrained problem, a query specifies which consequence to predict, and the corresponding solver answer provides the target. This turns the potentially discontinuous local-to-global effects of constraint-relevant choices into explicit supervision. Because VPS solutions comprise interdependent local components, composite commitments impose multiple constraints jointly, explicitly supervising interactions that isolated changes do not reveal. After representation learning, a lightweight decoder constructs complete solutions from the frozen visual encoder.

Our main contributions are threefold:

- We introduce **Solver-JEPA**, a representation-first framework for visual problem solving that bridges visual and decision structures through solver-supervised joint-embedding prediction, enabling complete-solution construction from a frozen visual encoder.
- We develop a query-based prediction formulation that predicts solver-derived consequences under partial commitments, with composite commitments exposing interactions among multiple local constraints.
- Across four verifiable VPS tasks, Solver-JEPA improves strict solution accuracy by up to 11.6 percentage points and achieves 15–150$\times$ faster inference than generation-based baselines. Controlled ablations further support the effectiveness of the key design choices.

# 2 Related Works

## 2.1 Visual Problem Solving

Visual problem solving requires models to recover problem structure from visual inputs and construct solutions under task-specific rules and objectives. Existing benchmarks cover combinatorial puzzles (Ghosal et al., 2025), spatial planning and maze navigation (Wu et al., 2025b; Xu et al., 2025; Yang et al., 2025), visual Sudoku (Li et al., 2023; Wewer et al., 2025), TSP (He et al., 2025; Dai et al., 2026), and jigsaw puzzles (Gu et al., 2025; He et al., 2025). These tasks provide controlled settings for studying how visual understanding supports structured reasoning.

Existing approaches address VPS through symbolic inference, visual generation, or latent reasoning. Neuro-symbolic methods learn symbol grounding and logical constraints (Li et al., 2023), while MGSD distills planning capabilities from a teacher with privileged symbolic states into a visual student (Luo et al., 2026). Visual Planning autoregressively predicts image-state trajectories (Xu et al., 2025), Spatial Reasoning Models denoise unobserved variables using variable-specific schedules (Wewer et al., 2025), and DiffThinker generates solution images conditioned on problem images and textual instructions (He et al., 2025). ThinkMorph interleaves textual and visual reasoning (Gu et al., 2025), whereas EndoCoT iteratively refines latent thought states with terminal-state supervision derived from ground-truth reasoning (Dai et al., 2026). Our work focuses on learning a shared visual problem representation through solver-derived predictive supervision.

## 2.2 Visual Self-Supervised Learning

Visual self-supervised learning constructs training signals from images without manual semantic labels. Contrastive methods, such as SimCLR (Chen et al., 2020) and MoCo (He et al., 2020), learn representations by aligning different augmented views of the same image while separating unrelated samples. Non-contrastive methods, including BYOL (Grill et al., 2020) and DINO (Caron et al., 2021), further learn consistent representations across views through self-distillation.

Another line of work learns representations through visual prediction. MAE (He et al., 2022) reconstructs the pixels of masked image regions, while BEiT (Bao et al., 2022) predicts discrete visual tokens derived from an image tokenizer. These methods show that meaningful visual representations can emerge from predicting missing or transformed visual content. This predictive perspective is closely related to our formulation, although Solver-JEPA uses solver-derived targets to organize representations around the structure of the underlying problem rather than visual content alone.

## 2.3 Joint-Embedding Predictive Architectures

Joint-embedding predictive architectures learn representations by predicting targets in feature space rather than reconstructing observations (LeCun, 2022). I-JEPA (Assran et al., 2023) predicts representations of masked image regions from visible context, conditioned on the target positions. V-JEPA (Bardes et al., 2024) extends this feature-prediction principle to spatio-temporal regions in video, showing that representation prediction can capture both appearance and motion without reconstructing pixels. V-JEPA 2 (Assran et al., 2025) further scales action-free video pretraining and subsequently trains an action-conditioned predictor over frozen visual representations using a small amount of robot interaction data. The resulting V-JEPA 2-AC model supports goal-conditioned planning by evaluating candidate action sequences in the learned representation space.

The JEPA framework has also been extended beyond image and video masking through different input modalities and conditioning signals. A-JEPA (Fei et al., 2023) applies masked latent prediction to audio, while Point-JEPA (Saito et al., 2025) adapts the same principle to 3D point-cloud representations. CrossJEPA (Perera et al., 2025) introduces cross-modal prediction between 3D inputs and 2D visual targets, and VL-JEPA (Chen et al., 2025) conditions prediction on textual queries to estimate answer representations. These studies extend the JEPA beyond the original masked prediction setting by exploring different contexts, conditioning information, and prediction targets.

Another line of work introduces actions, policies, or task information into latent prediction for control and planning. TD-JEPA (Bagatella et al., 2025) conditions latent prediction on policy information for zero-shot reinforcement learning. LeWorldModel (Maes et al., 2026) jointly trains a visual

encoder and an action-conditioned next-representation predictor end-to-end from pixels, and performs planning with learned latent dynamics. INTACT (Sun et al., 2026) further explores decision-oriented predictive representations by learning a state-conditioned intent-to-action interface. Together with V-JEPA 2-AC, these works establish joint-embedding prediction as a useful framework for learning action-conditioned dynamics and task-relevant latent representations for control.

# 3 PRELIMINARY

Joint-embedding predictive architectures (JEPAs) learn representations by predicting target features in an embedding space rather than reconstructing the target observation itself (LeCun, 2022). A context encoder maps the available observation into a representation, and a predictor uses this representation together with conditioning information to estimate the corresponding target embedding.

Formally, let $x_{\mathrm{ctx}}$ denote a context input, $\kappa$ the conditioning information that specifies the prediction, and $x_{\mathrm{tgt},\kappa}$ the associated target input. Given a context encoder $E_\theta$, a predictor $P_\psi$, and a target encoder $T_\eta$, the predicted and target representations are

$$s = E_\theta(x_{\mathrm{ctx}}), \qquad \widehat{z}_\kappa = P_\psi(s, \kappa), \qquad z_\kappa = T_\eta(x_{\mathrm{tgt},\kappa}). \tag{1}$$

A common asymmetric prediction objective is

$$\mathcal{L}_{\mathrm{pred}} = \mathbb{E}_{(x_{\mathrm{ctx}},\kappa,x_{\mathrm{tgt},\kappa})\sim\mathcal{D}} \left[\ell_{\mathrm{pred}}(\widehat{z}_\kappa, \mathrm{sg}(z_\kappa))\right], \tag{2}$$

where sg stops gradients and $\ell_{\mathrm{pred}}$ measures discrepancy in the embedding space. The construction and optimization of the target representation are formulation dependent. For example, I-JEPA updates its target encoder using an EMA of the context encoder (Assran et al., 2023).

# 4 METHOD

Solver-JEPA learns solution-relevant visual representations from solver-derived query–answer pairs defined over commitment-constrained problems. We first formulate the prediction tasks and introduce their key components (Sec. 4.1), then instantiate them in a joint-embedding predictive architecture and present the learning objectives (Secs. 4.2). Finally, we describe the solution decoding procedure (Sec. 4.3). Additional implementation details are provided in Sec. 5.1 and the Appendix

## 4.1 SOLVER-DERIVED PREDICTION COMPONENTS

Visual problem solving requires more than identifying individual elements in an image. It also requires understanding the relations among them and how they combine to form valid complete solutions. Accordingly, a model should be able to (1) understand problem structure, (2) evaluate the consequences of decisions, and (3) represent and verify complete solutions.

To this end, we introduce a "query–consequence" embedding-prediction objective in Solver-JEPA to learn visual representations that capture solution-relevant information about both the original problem and the consequences of partial commitments. Specifically, each prediction task comprises **a commitment** $c$ defining the conditions, **a query** $q$ specifying what to predict under $c$, and **a solver-derived answer** $y_q$ providing the target.

**Partial commitments.** A partial commitment $c$ is a set of local constraints on a complete solution, such as requiring or excluding an edge in TSP. For an instance $\xi$ depicted by an image $x$, with valid solution space $\Omega(\xi)$, it induces

$$\Omega(\xi, c) = \{w \in \Omega(\xi) : w \models c\}, \tag{3}$$

where $w \models c$ means that $w$ satisfies all constraints in $c$. The empty commitment $c = \varnothing$ recovers the original problem and provides supervision for problem structure and complete solutions. Nonempty commitments extend this supervision to the consequences of tentative decisions: first-order commitments impose one constraint, whereas composite commitments impose multiple constraints jointly.

**Algorithm 1** Solver-JEPA representation training

**Input:** Dataset $\mathcal{D} = \{(x_i, \mathcal{B}_i)\}$ with cached query–answer banks, batch size $B$, query-selection schedules, and loss weights $\lambda_{\text{sem}}, \lambda_{\text{reg}}$

Initialize modules with parameters $\Theta = \{\theta, \psi, \eta, \omega\}$
**while** not converged **do**
  ▷ Sample images and select queries
  $\{(x_b, \mathcal{B}_b)\}_{b=1}^{B} \sim \mathcal{D}$
  $\mathcal{Q}_b \leftarrow \text{Select}(\mathcal{B}_b)$ for each $b$
  ▷ Share image encodings across queries
  $s_b \leftarrow E_\theta(x_b)$ for $b = 1, \dots, B$
  $\widehat{z}_{b,q} \leftarrow P_\psi(s_b, q)$ for all $b$ and $(q, y_q) \in \mathcal{Q}_b$
  $z_{b,q} \leftarrow T_\eta(y_q)$ for selected structured answers
  ▷ Compute losses and update
  $\mathcal{L}_{\text{JEPA}} \leftarrow \mathbb{E}_{\text{str}}\left[\frac{1}{N}\sum_{j=1}^{N}(1 - \cos(\widehat{z}_{q,j}, \text{sg}(z_{q,j})))\right]$ (Eq. 7)
  $\mathcal{L}_{\text{sem}} \leftarrow \mathbb{E}_{\text{all}}\big[\ell_q^{\text{pred}}\big(H_\omega^{\text{pred}}(\widehat{z}_q, q), y_q\big)\big] + \mathbb{E}_{\text{str}}\big[\ell_q^{\text{tgt}}\big(H_\omega^{\text{tgt}}(z_q, q), y_q\big)\big]$ (Eq. 8)
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{JEPA}} + \lambda_{\text{sem}}\mathcal{L}_{\text{sem}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}}$
  Update $\Theta$ with AdamW using $\nabla_\Theta \mathcal{L}$
**end while**
**return** $E_\theta$

**Prediction queries.** Partial commitments define the problem conditions, while queries specify what to predict under those conditions. We represent a query as

$$q = (\tau, c, u), \tag{4}$$

where $\tau$ denotes the prediction type and $u$ contains any additional query-specific inputs, such as selected problem elements or a candidate solution. We use `RELATION` queries for task-specific structural relations, `FEASIBILITY` queries for the existence of a completion under $c$, and `VALUE` queries for changes in the optimal objective relative to the original problem. `WITNESS` queries specify complete valid or optimal solutions under $c$, while `MEMBERSHIP` queries assess candidate validity and optimality under the given commitment. Details are provided in Appendix A.

**Solver-derived targets.** For each query $q$, we construct the corresponding answer $y_q$ using task-specific solvers, structural computations, or candidate checks. For an optimization task with a minimization objective $J_\xi$, a feasible commitment defines

$$v_c = \min_{w \in \Omega(\xi, c)} J_\xi(w), \qquad \Delta v_c = v_c - v_\varnothing. \tag{5}$$

The `VALUE` query concerns the cost increase $\Delta v_c$, while an optimal `WITNESS` attains $v_c$. Such a witness can be suboptimal for the original problem when $v_c > v_\varnothing$. Binary and scalar targets encode feasibility, candidate acceptance, and cost changes, whereas structured targets encode relations among problem elements or complete solutions. Undefined answer components, including a value or witness for an infeasible commitment, are masked in the corresponding supervision.

### 4.2 Joint-Embedding Prediction

Solver-JEPA is trained on the triplets $\langle x, q, y_q \rangle$ constructed in Sec. 4.1. Multiple queries associated with the same problem image share a single visual encoding.

**Joint-embedding prediction formulation.** An image encoder $E_\theta$ produces a shared representation, and a query-conditioned predictor $P_\psi$ produces features for each selected query. For structured answers, a target encoder $T_\eta$ additionally constructs the corresponding target features:

$$s = E_\theta(x), \qquad \widehat{z}_q = P_\psi(s, q), \qquad z_q = T_\eta(y_q), \tag{6}$$

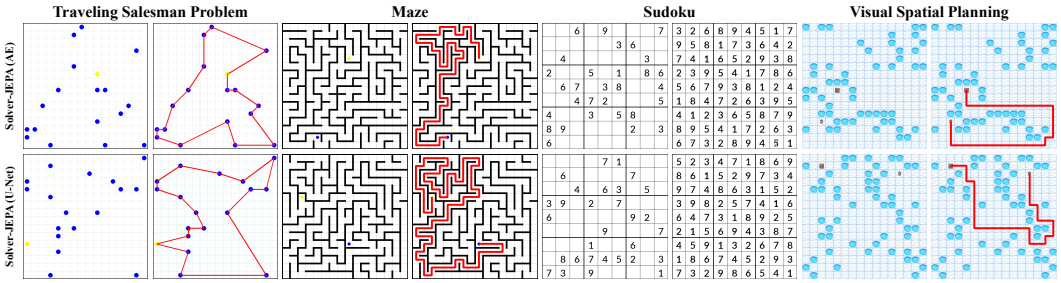



Figure 2: **Qualitative examples of visual problem solving with Solver-JEPA.** Each pair shows an input problem and a complete solution generated from the frozen Solver-JEPA representation.

where $s \in \mathbb{R}^{N \times d}$ retains a spatial grid of visual tokens and $z_q$ is defined only for structured-answer queries. Within a structured branch of type $\tau$, the predicted and target features share a token count $N_\tau$ and embedding dimension $d_\tau$. Their alignment objective is

$$\mathcal{L}_{\text{JEPA}} = \mathbb{E}_{\text{str}} \left[ \frac{1}{N_\tau} \sum_{j=1}^{N_\tau} (1 - \cos(\widehat{z}_{q,j}, \text{sg}(z_{q,j}))) \right], \tag{7}$$

where $\mathbb{E}_{\text{str}}$ denotes task-weighted averaging over structured queries.

**Efficient multi-query training.** Each image is associated with many queries, making it inefficient to encode the image separately for every query. For each image $x_b$, we cache query–answer pairs in $\mathcal{B}_b = \{(q, y_q)\}$ and select a subset $\mathcal{Q}_b \subseteq \mathcal{B}_b$ at each update. All selected queries reuse a single image encoding $s_b = E_\theta(x_b)$. Where possible, related targets are grouped into spatial or pairwise tensors for parallel prediction. Query grouping and sampling details are provided in Appendix D.

**Semantic supervision and regularization.** We add lightweight query-specific readouts $H_\omega^{\text{pred}}$ and $H_\omega^{\text{tgt}}$ to the predicted and target features, respectively:

$$\mathcal{L}_{\text{sem}} = \mathbb{E}_{\text{all}} \left[ \ell_q^{\text{pred}} \left( H_\omega^{\text{pred}}(\widehat{z}_q, q), y_q \right) \right] + \mathbb{E}_{\text{str}} \left[ \ell_q^{\text{tgt}} \left( H_\omega^{\text{tgt}}(z_q, q), y_q \right) \right]. \tag{8}$$

Here, $\mathbb{E}_{\text{all}}$ averages over selected queries of all types, and the query-specific losses incorporate masks for undefined answer components. This objective anchors the predicted and target features to the corresponding answer semantics. The target-side term trains $T_\eta$, while stop-gradient in Eq. 7 blocks updates to $T_\eta$ from the alignment loss. We further apply variance regularization $\mathcal{L}_{\text{reg}}$ to discourage representation collapse. The overall objective is

$$\mathcal{L} = \mathcal{L}_{\text{JEPA}} + \lambda_{\text{sem}} \mathcal{L}_{\text{sem}} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}. \tag{9}$$

Alg. 1 summarizes representation training. Details are discussed in Appendix C.

### 4.3 Solution Decoding

After representation training, we freeze $E_\theta$ and independently train two types of solution-image decoders on $s = \text{sg}(E_\theta(x))$: a deterministic RAE-style decoder Zheng et al. (2025) and a conditional image-diffusion decoder Song et al. (2021). Both reuse the same frozen representations. The query predictors, target encoders, and semantic readouts are discarded for solution decoding, and neither decoding pipeline accesses a task solver at inference.

**Deterministic decoding.** Let $x_{\text{sol}}$ denote a reference solution image paired with $x$. A decoder $D_\phi$ predicts the solution image in one forward pass:

$$\begin{aligned} \widehat{x}_{\text{sol}} &= D_\phi(s), \\ \mathcal{L}_{\text{AE}} &= \mathbb{E}[\ell_{\text{img}}(D_\phi(s), x_{\text{sol}})] + \lambda_{\text{AE}} \mathcal{L}_{\text{aux}}^{\text{AE}}, \end{aligned} \tag{10}$$

where $\ell_{\text{img}}$ is the image reconstruction loss and $\mathcal{L}_{\text{aux}}^{\text{AE}}$ comprises task-specific auxiliary loss.

**Diffusion decoding.** We alternatively condition a denoising network $v_\varphi$ on $s$. Using the normalized solution image, we construct

$$x_t = \alpha_t x_{\text{sol}} + \sigma_t \epsilon, \qquad v_t = \alpha_t \epsilon - \sigma_t x_{\text{sol}}, \tag{11}$$

where $\epsilon \sim \mathcal{N}(0, I)$, $t$ is sampled uniformly over the diffusion steps, and $\alpha_t, \sigma_t$ are fixed noise-schedule coefficients satisfying $\alpha_t^2 + \sigma_t^2 = 1$. The core $v$-prediction objective is

$$\mathcal{L}_v = \mathbb{E}_{x, x_{\text{sol}}, t, \epsilon}\left[\|v_\varphi(x_t, t, s) - v_t\|_2^2\right]. \tag{12}$$

At inference, the diffusion decoder generates a solution image from Gaussian noise using DDIM, with $s$ fixed throughout denoising. Architecture configurations, optimization settings, sampling steps, and auxiliary objectives are provided in Appendix F.

# 5 Experiments

## 5.1 Experimental Setup

We train a task-specific Solver-JEPA model from scratch by first learning the visual encoder with solver-derived supervision, then freezing it and independently training either a deterministic RAE-style decoder (AE) or an image-space diffusion U-Net to generate solution images. All stages are trained for 100 epochs on 4 NVIDIA A800 GPUs using AdamW with a learning rate of $1 \times 10^{-4}$. The representation-learning and AE stages each take approximately 15 hours, while U-Net training takes approximately 30 hours. Further details are provided in Appendices B and E.

## 5.2 Quantitative Comparison

**Task setup.** Following prior work (He et al., 2025; Dai et al., 2026), we evaluate Solver-JEPA on four verifiable visual problem solving tasks. **FrozenLake** evaluates visual spatial planning (VSP and VSP-Super with larger map sizes), while **Maze** evaluates longer-horizon path planning. **TSP** and **Sudoku** evaluate combinatorial optimization and constraint satisfaction, respectively.

**Compared methods.** We compare Solver-JEPA with strong publicly available multimodal foundation models and generation-based approaches on VPS tasks. For Qwen3-VL-32B (Team, 2025), we report both the vanilla model and variants post-trained on task-specific data using SFT or GRPO. We also include the vanilla Qwen-Image-Edit-2511 model (Wu et al., 2025a), together with generation-based methods DiffThinker, DiffThinker++ (He et al., 2025), and EndoCoT (Dai et al., 2026). Solver-JEPA is evaluated separately with AE and U-Net decoders.

**Evaluation setup.** Tab. 1 reports accuracy under the official test sets, task-specific parsers, and verifiers (He et al., 2025). A prediction is considered correct if the extracted candidate satisfies the task constraints and, where applicable, the optimization objective. However, such verification may overlook visual errors in the rendered image, such as stray line segments in Maze or redundant edges in TSP. We therefore additionally evaluate 1,000 instances in each challenging setting using a strict image-level verifier (Tab. 2) with the official generation scripts. This protocol directly verifies the complete rendered solution and rejects artifacts that make the solution structure invalid.

**Results.** As shown in Tab. 1, Solver-JEPA achieves the highest mean accuracy over the 18 evaluation settings, reaching $93.6\%$ with the AE decoder and $93.7\%$ with the U-Net decoder, compared with $92.1\%$ for EndoCoT and $83.8\%$ for DiffThinker++. On the most challenging TSP and Sudoku settings, the U-Net decoder reaches $80\%$ on 18-city TSP and the AE decoder reaches $74\%$ on 30-clue Sudoku, exceeding the strongest baseline by 7 and 10 percentage points, respectively. Solver-JEPA remains competitive on navigation tasks, although EndoCoT performs better in some settings, including Maze-32 and VSP-Super-32.

Under strict image-level verification (Tab. 2), Solver-JEPA achieves $79.4\%$ on Maze-32 and $73.2\%$ on TSP-18 with the U-Net decoder, and $77.2\%$ on Sudoku-30 with the AE decoder. These results exceed the strongest baseline by $9.3$, $9.9$, and $11.6$ percentage points, respectively. The gains therefore extend to evaluations that require image-level verification of complete solutions, rather than correctness of an extracted candidate alone.

Table 1: **Quantitative comparison across VPS tasks.** Accuracy (%) is evaluated using official parsers and verifiers. Avg. denotes the unweighted mean over the 18 evaluation settings.

| Models | Maze | | | TSP | | | Sudoku | | | | VSP | | | | | | VSP-Super | | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8 | 16 | 32 | 12 | 15 | 18 | 45 | 40 | 35 | 30 | 3 | 4 | 5 | 6 | 7 | 8 | 16 | 32 | |
| ▼ *Zero-shot* | | | | | | | | | | | | | | | | | | | |
| Qwen3-VL-32B† | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 75 | 51 | 47 | 25 | 23 | 26 | 0 | 0 | 13.7 |
| Qwen-Image-Edit-2511† | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 50 | 55 | 44 | 16 | 23 | 23 | 0 | 0 | 11.7 |
| ▼ *Task-Specific Training* | | | | | | | | | | | | | | | | | | | |
| Qwen3-VL-32B† (SFT) | 91 | 57 | 3 | 69 | 59 | 52 | 32 | 22 | 2 | 0 | 96 | 99 | 98 | **100** | 99 | 90 | 85 | 21 | 59.7 |
| Qwen3-VL-32B† (GRPO) | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 99 | 90 | 95 | 69 | 73 | 58 | 1 | 0 | 27.2 |
| DiffThinker | **100** | 97 | 56 | 74 | 62 | 58 | 98 | 95 | 57 | 14 | 99 | **100** | 98 | 99 | **100** | **100** | 96 | 83 | 82.6 |
| DiffThinker++ | **100** | **100** | 65 | 76 | 72 | 59 | 97 | 94 | 55 | 13 | **100** | **100** | **100** | 98 | **100** | **100** | **99** | 80 | 83.8 |
| EndoCoT | **100** | **100** | **90** | 77 | 77 | 73 | **100** | **100** | 95 | 64 | **100** | **100** | **100** | 99 | **100** | 99 | **99** | **85** | 92.1 |
| **Ours w/. AE Decoder** | **100** | 98 | 86 | 88 | **89** | 76 | **100** | **100** | **99** | **74** | **100** | **100** | **100** | 99 | **100** | 99 | 97 | 79 | 93.6 |
| **Ours w/. U-Net Decoder** | **100** | 98 | 84 | **89** | 85 | **80** | **100** | **100** | **99** | 73 | **100** | **100** | **100** | 98 | **100** | 99 | 98 | 83 | **93.7** |

*Note.* † Results are taken from the evaluations reported in (He et al., 2025).

Table 2: **Strict evaluation on challenging settings.** Each task contains 1,000 test instances and predictions are evaluated with a strict image-level verifier. Any invalid or ambiguous visual artifact as an error.

| Models | Maze-32 | TSP-18 | Sudoku-30 | VSP-Super-32 |
|---|---|---|---|---|
| DiffThinker | 43.9 | 43.6 | 17.6 | 74.3 |
| DiffThinker++ | 49.4 | 46.2 | 16.7 | 73.2 |
| EndoCoT | 70.1 | 63.3 | 65.6 | **78.1** |
| **Ours (AE)** | 72.1 | 65.8 | **77.2** | 72.9 |
| **Ours (U-Net)** | **79.4** | **73.2** | 75.1 | 76.4 |

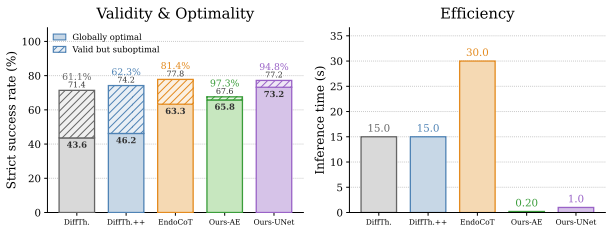



Figure 3: **Validity, optimality, and efficiency on TSP-18.** Left: total bar height denotes the strict-valid-tour rate, solid segments denote globally optimal tours.

### 5.3 Optimality and Efficiency

Beyond end-task correctness, we focus on TSP because its explicit optimization objective distinguishes valid tours from globally optimal ones. We use tour optimality to assess how well Solver-JEPA models global solution structure, and inference efficiency to examine the practical value of its learned representations for lightweight solution decoding.

**Optimality of TSP.** We compute the conditional optimality rate over the 1,000 samples in Tab. 2, measuring the fraction of strictly valid Hamiltonian cycles that also attain the optimal tour length. As shown in Fig. 3, Solver-JEPA achieves $97.3\%$ with the AE decoder and $94.8\%$ with the U-Net decoder, compared with $81.4\%$ for EndoCoT and $61.1\%$/$62.3\%$ for DiffThinker/DiffThinker++. Thus, once Solver-JEPA constructs a valid tour, it is highly likely to be globally optimal. Together with the overall success rates, these results support the effectiveness of its learned representations in enabling globally informed solution construction beyond feasibility alone.

**Efficiency.** Fig. 3 also demonstrates a substantial efficiency advantage. Solver-JEPA requires only $0.2$ seconds per instance with the AE decoder and $1.0$ second with the U-Net decoder, compared with $15.0$ seconds for DiffThinker and DiffThinker++ and $30.0$ seconds for EndoCoT. All runtimes are measured on an NVIDIA A6000 GPU and averaged over 1,000 instances. These results correspond to speedups of $75$–$150\times$ for AE and $15$–$30\times$ for U-Net. Together with the strong strict-evaluation results on TSP, these speedups highlight the practical value of learning solution-relevant representations that support fast, accurate solution construction with lightweight decoders.

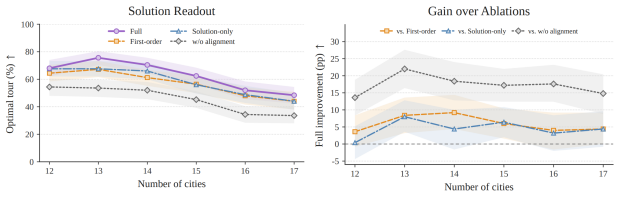
### 5.4 Ablation Studies

**Evaluation setup.** We conduct ablations on TSP because its explicit optimality objective and strong local-to-global dependencies make it well suited for evaluating solution-relevant represen-

Table 3: **Representation training ablations on TSP 12-17.**

| Training variant | Opt. tour ↑ (%) | Value MAE ↓ | Member BA (%) ↑ |
|---|---|---|---|
| Image-only | 7.40 | 0.0252 | 94.86 |
| Direct solution | 31.13 | 0.0208 | 96.83 |
| Empty $c$ only | 58.33 | 0.0207 | 97.17 |
| w/. First-order $c$ | 56.87 | 0.0201 | 97.53 |
| w/o embed. align. | 45.53 | 0.0204 | 97.37 |
| **Full design** | **62.80** | **0.0200** | **97.61** |

*Note.* Probes are trained from scratch on frozen representations using identical configurations and a fixed probe-training budget. w/. First-order retains empty-commitment but excludes composite commitments.




(a) Optimal-tour accuracy (b) Paired gains of Full

Figure 4: **Frozen solution readout across TSP problem sizes.** (a) Optimal-tour accuracy on 250 held-out instances per size. (b) Paired accuracy differences between Full and each ablated variant on the same instances, in percentage points. Both panels use the same frozen-probe protocol.

tations. Under matched budgets, we freeze each encoder and train identically configured solution and query probes from scratch, without reusing training-time predictors. **Optimal tour** is the fraction of instances where a shallow edge readout yields a valid, globally optimal tour without solver repair, measuring solution-structure readability rather than RGB generation. **Value MAE** measures normalized error in predicting commitment-induced cost increases. **Member BA** measures balanced accuracy in verifying whether a candidate tour is valid and optimal under a commitment. Fig. 4 further disaggregates the solution-probe results in Tab. 3 across 12–17 cities.

**Embedding-prediction strategy.** We first evaluate two straightforward baselines: image-only JEPA, which predicts solution-image embeddings from an EMA target encoder, and direct solution prediction, which predicts solver-generated structures without a target encoder or embedding alignment. They achieve optimal-tour rates of $7.40\%$ and $31.13\%$, respectively, compared with $62.80\%$ for Full. These results favor the full prediction design over these two alternatives, while the following ablations examine the contributions of commitment supervision and embedding alignment.

Solution-only JEPA retains embedding prediction and semantic supervision for the original problem under the empty commitment, achieving $58.33\%$ but remaining $4.47$ percentage points below Full. Restricting nonempty supervision to first-order commitments, each involving a single edge constraint, yields $56.87\%$, below Solution-only and $5.93$ percentage points below Full. This pattern does not support a monotonic benefit from adding nonempty commitments. Full's improvement over first-order is instead consistent with the value of explicitly supervising the joint consequences of multiple constraints. Finally, to isolate embedding alignment, we retain Full's targets and semantic supervision but remove the alignment loss. This reduces the optimal-tour rate from $62.80\%$ to $45.53\%$, supporting the contribution of embedding alignment beyond semantic supervision alone.

**Readability across problem sizes.** Among the four variants in Fig. 4(a), Full achieves the highest optimal-tour point estimate at every evaluated size. The paired comparisons in Fig. 4(b) show gains of $13.6$–$22.0$ percentage points over w/o alignment and $3.6$–$9.2$ points over first-order. These results extend the aggregate findings in Tab. 3: the improvements in complete-solution readout are observed across multiple problem sizes rather than being confined to one size.

## 6 Conclusion

We introduced **Solver-JEPA**, a joint-embedding predictive architecture for learning solution-relevant visual representations from solver-derived supervision. Queries over original and commitment-constrained problems organize supervision for problem structure, decision consequences, and complete-solution representation and verification. Joint-embedding prediction shapes a shared representation that independently trained decoders reuse to construct complete solution images without solver access at inference. Experiments demonstrate strong solution accuracy and inference efficiency, while ablations highlight the importance of embedding alignment and the additional value of composite-commitment supervision. These findings support a representation-centered approach to visual problem solving, in which solver-derived knowledge is learned during training and reused for subsequent solution construction

## LLM USE STATEMENT

We used large language models (LLMs) to assist with language editing and to provide limited feedback on experimental design. Specifically, LLMs were used to improve the clarity and presentation of the manuscript and to suggest possible ways of organizing or evaluating experiments. All methodological choices, experimental configurations, implementations, analyses, results, and scientific conclusions were independently reviewed and validated by the authors. The authors made all final research decisions and take full responsibility for the content and contributions of this work.

## REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael G. Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *CVPR*, pp. 15619–15629, 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew J. Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *CoRR*, abs/2506.09985, 2025.

Marco Bagatella, Matteo Pirotta, Ahmed Touati, Alessandro Lazaric, and Andrea Tirinzoni. TD-JEPA: latent-predictive representations for zero-shot reinforcement learning. *CoRR*, abs/2510.00739, 2025.

Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT pre-training of image transformers. In *ICLR*, 2022.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Trans. Mach. Learn. Res.*, 2024.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, pp. 9630–9640, 2021.

Delong Chen, Mustafa Shukor, Théo Moutakanni, Willy Chung, Jade Yu, Tejaswi Kasarla, Allen Bolourchi, Yann LeCun, and Pascale Fung. VL-JEPA: joint embedding predictive architecture for vision-language. *CoRR*, abs/2512.10942, 2025.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pp. 1597–1607, 2020.

Xuanlang Dai, Yujie Zhou, Long Xing, Jiazi Bu, Xilin Wei, Yuhong Liu, Beichen Zhang, Kai Chen, and Yuhang Zang. EndoCoT: Scaling endogenous chain-of-thought reasoning in diffusion models. *CoRR*, abs/2603.12252, 2026.

Zhengcong Fei, Mingyuan Fan, and Junshi Huang. A-JEPA: joint-embedding predictive architecture can listen. *CoRR*, abs/2311.15830, 2023.

Deepanway Ghosal, Vernon Toh, Yew Ken Chia, and Soujanya Poria. AlgoPuzzleVQA: Diagnosing multimodal reasoning challenges of language models with algorithmic multimodal puzzles. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pp. 9615–9632, 2025.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent - A new approach to self-supervised learning. In *NeurIPS*, 2020.

Jiawei Gu, Yunzhuo Hao, Huichen Will Wang, Linjie Li, Michael Qizhe Shieh, Yejin Choi, Ranjay Krishna, and Yu Cheng. Thinkmorph: Emergent properties in multimodal interleaved chain-of-thought reasoning. *CoRR*, abs/2510.27492, 2025.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pp. 9726–9735, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pp. 15979–15988, 2022.

Zefeng He, Xiaoye Qu, Yafu Li, Tong Zhu, Siyuan Huang, and Yu Cheng. DiffThinker: Towards generative multimodal reasoning with diffusion models. *CoRR*, abs/2512.24165, 2025.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. In *Conference on Robot Learning, CoRL 2023, 6-9 November 2023, Atlanta, GA, USA*, pp. 540–562, 2023.

Yann LeCun. A path towards autonomous machine intelligence, version 0.9.2, 2022-06-27. *OpenReview*, 62(1):1–62, 2022.

Zenan Li, Yunpeng Huang, Zhaoyu Li, Yuan Yao, Jingwei Xu, Taolue Chen, Xiaoxing Ma, and Jian Lu. Neuro-symbolic learning yielding logical constraints. In *NeurIPS*, 2023.

Haocheng Luo, Jiahui Liu, Ruicheng Zhang, Zhizhou Zhong, Jiaqi Huang, Zunnan Xu, Quan Shi, Jun Zhou, and Xiu Li. Learning visual spatial planning from symbolic state via modality-gap-aware self-distillation. *CoRR*, abs/2606.06076, 2026.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorldModel: Stable end-to-end joint-embedding predictive architecture from pixels. *CoRR*, abs/2603.19312, 2026.

Nelson Nauata, Sepidehsadat Hosseini, Kai-Hung Chang, Hang Chu, Chin-Yi Cheng, and Yasutaka Furukawa. House-gan++: Generative adversarial layout refinement network towards intelligent computational agent for professional architects. In *CVPR*, pp. 13632–13641, 2021.

Avishka Perera, Kumal Hewagamage, Saeedha Nazar, Kavishka Abeywardana, Hasitha Gallella, Ranga Rodrigo, and Mohamed Afham. CrossJEPA: Cross-modal joint-embedding predictive architecture for efficient 3d representation learning from 2d images. *CoRR*, abs/2511.18424, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, pp. 8748–8763, 2021.

Ayumu Saito, Prachi Kudeshia, and Jiju Poovvancheri. Point-JEPA: A joint embedding predictive architecture for self-supervised learning on point cloud. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2025, Tucson, AZ, USA, February 26 - March 6, 2025*, pp. 7348–7357, 2025.

Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. Vint: A foundation model for visual navigation. In *Conference on Robot Learning, CoRL 2023, 6-9 November 2023, Atlanta, GA, USA*, pp. 711–733, 2023.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

Junhan Sun, Hao Zhao, and Guofeng Zhang. INTACT: isomorphic intent-to-action learning for search-free world models. *CoRR*, abs/2607.26056, 2026.

Qwen Team. Qwen3-vl technical report. *CoRR*, abs/2511.21631, 2025.

Christopher Wewer, Bartlomiej Pogodzinski, Bernt Schiele, and Jan Eric Lenssen. Spatial reasoning with denoising models. In *ICML*, 2025.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Shengming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report. *CoRR*, abs/2508.02324, 2025a.

Qiucheng Wu, Handong Zhao, Michael Saxon, Trung Bui, William Yang Wang, Yang Zhang, and Shiyu Chang. VSP: diagnosing the dual challenges of perception and reasoning in spatial planning tasks for MLLMS. In *ICCV*, pp. 2270–2280, 2025b.

Yi Xu, Chengzu Li, Han Zhou, Xingchen Wan, Caiqi Zhang, Anna Korhonen, and Ivan Vulic. Visual planning: Let's think only with images. *CoRR*, abs/2505.11409, 2025.

Cheng Yang, Haiyuan Wan, Yiran Peng, Xin Chen, Zhaoyang Yu, Jiayi Zhang, Junchi Yu, Xinlei Yu, Xiawu Zheng, Dongzhan Zhou, and Chenglin Wu. Reasoning via video: The first evaluation of video models' reasoning abilities through maze-solving tasks. *CoRR*, abs/2511.15065, 2025.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. *CoRR*, abs/2510.11690, 2025.

## A Notation, query semantics, and information boundaries

An instance $\xi$ is presented as a problem image $x$. Its set of complete feasible solutions is $\Omega(\xi)$, and a commitment $c$ restricts this set to $\Omega(\xi, c) = \{w \in \Omega(\xi) : w \models c\}$. The empty commitment recovers the original problem. First-order commitments contain one constraint; composite commitments contain multiple constraints imposed simultaneously. A query $q = (\tau, c, u)$ specifies a prediction family $\tau$, the commitment, and any additional public query input $u$. A candidate solution is supplied only to a membership query, not to a value or witness query.

For an optimization task, let $J_\xi(w)$ be the cost of a feasible solution, $v_c = \min_{w \in \Omega(\xi,c)} J_\xi(w)$ for feasible $c$, and $v_\varnothing$ the original optimum. Table 4 distinguishes the query meanings from their task-specific numerical encodings. A membership query tests acceptance under the queried commitment; a constrained-optimal solution need not be optimal for the original problem.

Table 4: **General query semantics.** Task-specific encodings and sampling schedules follow the corresponding task configurations.

| **Query** | **Answer under commitment** $c$ |
|---|---|
| Relation | Structural information about selected elements or a dense representation of the task structure. |
| Feasibility | $f_c = \mathbb{I}[\Omega(\xi, c) \neq \varnothing]$. |
| Value | Change in attainable optimal cost, with the task-specific normalization stated below. Undefined when $f_c = 0$. |
| Witness | A complete solution in $\Omega(\xi, c)$, attaining $v_c$ for an optimization task. Undefined when $f_c = 0$. |
| Membership | Whether an explicitly supplied candidate satisfies the problem and commitment constraints and, where applicable, is optimal under $c$. |

**Training-only information.** Solver answers, optimal costs, target maps, witness-mode labels, and answer-validity masks are used for target construction and loss computation. They are not image-encoder inputs. The encoder computes $s = E_\theta(x)$ independently of $q$. Query predictors may use the public commitment and task-specific element addresses; independent solution-image decoders use the frozen representation without a task solver.

**Feasibility is not optimality.** For any feasible commitment and candidate $\widehat{w} \in \Omega(\xi, c)$,

$$J_\xi(\widehat{w}) - v_\varnothing = (v_c - v_\varnothing) + (J_\xi(\widehat{w}) - v_c). \tag{13}$$

The first term measures the cost imposed by the commitment; the second measures suboptimality within the constrained problem. This elementary decomposition explains why value and membership/witness queries provide different supervision. It is an interpretation of the target definitions, not a claim that the model explicitly evaluates this decomposition at inference.

# B Network Architecture and Target Representations

**Shared image encoder** maps a $512 \times 512$ RGB image in $[0, 1]$ to $s \in \mathbb{R}^{B\times 1024\times 384}$. The trainable overlapping convolutional stem has channels $3 \to 64 \to 128 \to 384$, kernels $7, 3, 3$, and strides $4, 2, 2$. Group normalization follows each convolution, with GELU after the first two normalized convolutions. The resulting $32 \times 32$ feature map is flattened into spatial tokens. Learned positional embeddings are added before eight pre-normalized Transformer blocks, each with six attention heads and an MLP expansion ratio of four. A final LayerNorm produces $s$.

Table 5: Verified representation configuration.

| **Component** | |
|---|---|
| Input resolution | $512 \times 512$ |
| Spatial token grid / width | $32 \times 32$ / 384 |
| Encoder depth / heads | 8 / 6 |
| Predictor width / heads | 256 / 4 |
| Structural / value-related / solution predictor depths | 2 / 2 / 1 |
| Original-problem target map channels | Task-specific |
| Conditional scalar or query width | 128 |
| Conditional attention heads | 4 |
| External visual pretraining | None |

# C Training Objectives, Masks, and Gradient Paths

## C.1 Structured prediction and semantic supervision

For structured branch $\tau$, $\widehat{z}_q = P_{\psi,\tau}(s, q)$ and $z_q = T_{\eta,\tau}(y_q)$ contain $N_\tau$ tokens. Alignment is

$$\mathcal{L}_{\text{align},q} = \frac{1}{N_\tau} \sum_{j=1}^{N_\tau} \left( 1 - \frac{\widehat{z}_{q,j}^\top \operatorname{sg}(z_{q,j})}{\|\widehat{z}_{q,j}\|_2 \, \|\operatorname{sg}(z_{q,j})\|_2} \right), \tag{14}$$

with the implementation's numerical stabilization. Scalar and binary answers use their prediction readouts rather than an artificial scalar target embedding. Dense grouping can combine several answer components in one branch; a shared latent map does not imply that scalar answers receive a separate target-side alignment loss.

The predicted-side semantic loss trains the predictor and encoder. The target-side semantic loss trains the target encoder. Accordingly, the main-text predicted/target readout notation denotes their roles and does not require untied parameters in every branch.

**Masked reductions.** For losses $\ell_i$ and validity mask $m_i$, the implemented reduction is

$$\operatorname{mean}_m(\ell) = \frac{\sum_i m_i \ell_i}{\max(1, \sum_i m_i)}. \tag{15}$$

Broadcast masks are expanded to the supervised tensor shape where required. Padded cities, self-edges, padded queries, unavailable membership pairs, and undefined answer components are excluded from their respective losses. An infeasible value stored as zero is a placeholder and is not a zero-regret training example.

### C.2 VARIANCE REGULARIZATION AND PARAMETER UPDATES

Let $Z \in \mathbb{R}^{M \times d}$ be features flattened across images and spatial positions. The variance regularizer is

$$\mathcal{V}(Z) = \frac{1}{d} \sum_{j=1}^{d} \max\left(0, 1 - \sqrt{\mathrm{Var}_i(Z_{ij}) + 10^{-4}}\right), \tag{16}$$

where the population variance is used. The variance regularizer is applied to the representation branches specified by each task configuration, with task-specific weights. Target encoders are ordinary trainable modules: their outputs are detached only on the target side of the alignment loss, while target-side semantic losses retain gradients. No EMA update is used for the target encoders.

Both implementations also train a diagnostic head on $\mathrm{sg}(s)$. Its input is detached, so its loss updates only the diagnostic head and does not shape the image encoder. The primary complete-solution probe in the ablation protocol is trained afresh after freezing the encoder rather than reusing this online diagnostic head. Direct raw-$s$ answer losses and pretrained-feature anchor losses are disabled in the documented TSP strict representation configuration.

## D DATA CONSTRUCTION, CACHING, AND MULTI-QUERY TRAINING

### D.1 DATASET CONSTRUCTION AND SPLIT

We follow the provided DiffThinker-style data-generation configurations for the four tasks. Only the generated training pools are used in this construction. Each task contains 30,000 generated problem–solution pairs, which are partitioned with a validation-to-training ratio of 1:9, yielding 3,000 validation instances and 27,000 training instances per task.

Table 6: Training-data generation profiles. Each generated pool is split into validation and training subsets at a 1:9 ratio.

| **Task** | |
|---|---|
| TSP | City counts 12–17, with 5,000 instances generated for each city count. |
| Maze | Grid sizes 8, 16, and 32, each contributing 10,000 instances across the configured minimum-path-length buckets. |
| Sudoku | Clue counts 30, 35, 40, and 45, with 7,500 instances generated for each clue count. |
| FrozenLake | Sizes 3–6 jointly contribute 10,000 instances, while sizes 16 and 32 contribute 10,000 instances each. |

*Note.* Each task contains 30,000 generated instances, yielding 27,000 training and 3,000 validation instances under the 1:9 validation-to-training split.

For Maze, the $8 \times 8$ and $16 \times 16$ pools use five minimum-path-length buckets with 2,000 instances per bucket: $\{1, 6, 11, 16, 21\}$ and $\{1, 9, 17, 25, 33\}$, respectively. The $32 \times 32$ pool uses 2,000 instances for each of minimum lengths 1, 14, and 27, and 4,000 instances for minimum length 90. For FrozenLake, sizes 3, 4, 5, and 6 contain 500, 1,000, 2,500, and 6,000 instances, respectively. The $16 \times 16$ pool uses minimum lengths $\{1, 9, 17, 25, 33\}$ with 2,000 instances per setting, and the $32 \times 32$ pool uses $\{1, 14, 27, 40, 90\}$ with 2,000 instances per setting. The generation parameter is $p = 0.8$ except for the $32 \times 32$, minimum-length-90 setting, which uses $p = 0.6$.

### D.2 CACHED SUPERVISION AND SHARED MULTI-QUERY ENCODING

For every problem image $x_i$, preprocessing constructs a cache $\mathcal{B}_i = \{(q, y_q)\}$ of solver-derived query–answer pairs. The cache includes supervision for the original problem through the empty commitment and additional supervision from sampled nonempty commitments. Answers and validity masks are computed offline and stored separately for the training and validation splits; target embeddings are not cached and are instead produced online by the trainable target encoders.

At each optimization step, the loader selects a subset $\mathcal{Q}_i \subseteq \mathcal{B}_i$ for each sampled image. The image encoder is evaluated once, $s_i = E_\theta(x_i)$, and all selected queries reuse the same spatial representation. Queries of compatible type are grouped into dense tensors where possible, while padding and

undefined answer components are excluded by masks. This organization reduces repeated image encoding without changing the semantics of the individual prediction tasks. No task solver is invoked during the optimization step.

### D.3 Consistent spatial processing

When spatial augmentation is applied, the same transformation is used for the problem image, solution image, structured target maps, and any spatial commitment or candidate representation. This preserves correspondence between the visual input and all solver-derived supervision. Task-specific rendering and coordinate conventions follow the corresponding data generator.

## E Optimization and Training Configuration

Each task uses the same two-stage training protocol. First, a task-specific Solver-JEPA representation model is trained from scratch for 100 epochs with AdamW, using an initial learning rate of $1 \times 10^{-4}$ on four NVIDIA A800 GPUs. The solver-derived query objectives train the image encoder, query-conditioned predictors, target encoders, and semantic readouts jointly. The held-out validation partition is used for training-time monitoring and model selection.

After representation learning, the image encoder is frozen and a task-specific solution-image decoder is trained for 100 epochs on the resulting representations. We train the deterministic AE and diffusion U-Net independently, so neither decoder updates the representation encoder. Task-specific architectural dimensions, query sampling schedules, auxiliary losses, and loss weights are provided by the corresponding task configuration, while the overall optimization protocol is shared. In the reported setup, representation training and AE decoding each require approximately 15 hours, and diffusion U-Net training requires approximately 30 hours.

## F Complete Solution-Image Decoding

After representation training, the image encoder is frozen and the decoder receives only the spatial representation $s = \mathrm{sg}(E_\theta(x))$. Both decoder families are trained from paired problem and solution images. The query predictors, target encoders, semantic readouts, and task solvers used during representation learning are not used for solution generation at inference.

### F.1 Deterministic decoding

A task-specific deterministic decoder $D_\phi$ maps the frozen representation directly to a complete solution image,

$$\widehat{x}_{\mathrm{sol}} = D_\phi(s). \tag{17}$$

Its general training objective is

$$\mathcal{L}_{\mathrm{AE}} = \mathbb{E}[\ell_{\mathrm{img}}(D_\phi(s), x_{\mathrm{sol}})] + \lambda_{\mathrm{AE}}\mathcal{L}^{\mathrm{AE}}_{\mathrm{aux}}, \tag{18}$$

where $\ell_{\mathrm{img}}$ supervises the rendered solution and $\mathcal{L}^{\mathrm{AE}}_{\mathrm{aux}}$ denotes optional task-specific structural losses. These auxiliary terms are used only to train the decoder and do not update the frozen image encoder. Inference requires a single decoder forward pass.

### F.2 Diffusion decoding

The diffusion decoder conditions a denoising U-Net $v_\varphi$ on the same frozen representation. For a normalized solution image $x_{\mathrm{sol}}$, Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$, and a sampled timestep $t$, we construct

$$x_t = \alpha_t x_{\mathrm{sol}} + \sigma_t \epsilon, \qquad v_t = \alpha_t \epsilon - \sigma_t x_{\mathrm{sol}}, \qquad \alpha_t^2 + \sigma_t^2 = 1. \tag{19}$$

The shared core objective is

$$\mathcal{L}_v = \mathbb{E}_{x, x_{\mathrm{sol}}, t, \epsilon}\left[\left\|v_\varphi(x_t, t, s) - v_t\right\|_2^2\right], \tag{20}$$

optionally augmented by task-specific image-structure losses. At inference, the decoder starts from Gaussian noise and uses DDIM with the configured sampling schedule while keeping $s$ fixed throughout denoising. No solver call, candidate repair, or training-time query module is used.