

STARS: From Spatiotemporal Dynamics to Social Representations in Human-Robot Interaction

Nathan Tsoi*, Michael J. Munje*, Tejas Oberoi, Rishab Maheshwari,
Penggen Zheng, Tanush Chauhan, Peter Stone, Joydeep Biswas
Department of Computer Science, The University of Texas at Austin

* Equal Contribution

Abstract: Fielding socially competent robots requires joint reasoning over spatial information and the social information it can convey, such as group motion and personal space. While current models excel at either spatial reasoning (e.g., modeling physical dynamics) or social reasoning (e.g., parsing text-based interactions), few natively unify both. Moreover, data scarcity in Human-Robot Interaction (HRI) makes training large models from scratch challenging. To bridge this gap, we introduce **STARS: SpatioTemporal Autoencoded Representations for Social-interaction**, a method for learning *event-level relational representations* that models short windows of multi-agent interactions as spatiotemporal graphs. A self-supervised message-passing graph neural network autoencoder then compresses these physical dynamics into a compact latent space for each node. By explicitly modeling agents and their interactions as graph structure, STARS injects a strong relational inductive bias to naturally capture the underlying social situation. To assess the utility of these learned representations, we evaluate STARS’ representations on the SEAN Together dataset, which provides VR-based navigation data annotated with subjective human perceptions, and the SocialNav-SUB benchmark for social scene understanding. When trained on STARS representations, linear probes achieve highly competitive performance on perception prediction and pedestrian action classification tasks. Specifically, STARS exhibits strong data efficiency, particularly on pedestrian action classification, where STARS-He consistently outperforms unstructured baselines across labeled-data fractions, including in low-data settings. Furthermore, qualitative analysis of the learned latent space reveals that STARS representations can capture high-level social semantics, demonstrating cohesive clustering of distinct human action intents without optimizing the encoder with a supervised action classification objective. An overview of this paper along with the code can be found at <https://larg.github.io/stars>.

Keywords: Human-Robot Interaction, Social Navigation, Representation Learning, Graph Neural Networks

1 Introduction

Deploying socially competent robots in human spaces requires autonomous robots to reason jointly using spatial knowledge to plan collision-free paths and social knowledge to perceive context, predict behaviors, and adhere to social norms. A robot navigating through a crowded hallway, for example, must plan collision-free motion in the physical environment while also anticipating pedestrian behavior and reacting in a socially compliant manner [1, 2]. Joint reasoning over spatial and social information is central to Human-Robot Interaction (HRI), yet existing approaches tend to address only one or the other. Foundation spatial models parse LiDAR and visual observations into geometric representations, but lack the capacity to understand social dynamics [3, 4]. At the same

time, some autoregressive models conditioned on text (LLMs) or text and vision (VLMs) exhibit strong social perception [3], but struggle to process continuous spatial dynamics [5, 6] and precise temporal information [7]. Bridging this gap by training large models from scratch is challenging due to the scarcity of grounded HRI data. Collecting such data requires time and cost intensive human-subject studies, and yields datasets that are small, fragmented across sensor modalities, and highly prone to overfitting when applied to large, unstructured models.

Although most HRI datasets are small and fragmented, often collected in different environments, with different robots, sensors, tasks, and annotations, the premise of this paper is that their recurring social interaction structure [8] can be used to learn reusable representations for downstream social navigation tasks that require reasoning about agents, interactions, and shared context. This interaction structure is naturally relational, as agents move through shared spaces and influence one another over time. We hypothesize that graph-structured representations are well-suited to learning from small, fragmented HRI datasets because they can encode agents as nodes and interactions as edges, injecting strong *relational inductive bias* [9]. This inductive bias enables message-passing graph neural networks [10] to learn interaction patterns efficiently in limited-data regimes.

We introduce **STARS: SpatioTemporal Autoencoded Representations for Social-interaction**, a method for self-supervised learning of *event-level relational representations*. In contrast to the autoregressive prediction of world models, STARS learns a representation over a short time horizon to capture relevant contextual information. We define an *interaction scenario* as a fixed-duration sequence of events where agents share a physical space. We encode these interaction scenarios into graphs, where directed edges encode sequences of pairwise spatial features over time.

Though our interaction scenario graphs contain only spatiotemporal information, our target downstream tasks are inherently social. By optimizing a self-supervised reconstruction objective over the fixed-duration interaction scenario, STARS compresses physical dynamics into a compact latent space that captures the implicit *social situation* [11]. We demonstrate our method using two architectural variants that show the learned representations allow a simple linear probe to extract social meaning from physical motion, unlocking data-efficient generalization without requiring task-specific representation learning.

Our contributions are threefold:

1. We introduce STARS, a self-supervised representation learning method for social navigation, which compresses datasets with different feature dimensions and graph schemas into a latent representation containing features useful for diverse downstream tasks with limited task-specific labeled data.
2. We demonstrate that self-supervised pretraining of relational representations improves data efficiency. By comparing STARS to unstructured baselines in low-data regimes, we show that pretrained graph representations support robust downstream generalization with a fraction of the labeled data and transfer across the two evaluated datasets.
3. We establish that STARS’ self-supervised representations naturally encode social semantics. By analyzing the latent space topology, we illustrate the model’s inherent ability to distinguish complex social behaviors, such as action intents, without a supervised action classification objective.

2 Related Work

Spatial and social reasoning in social navigation. Evaluating multi-agent interactions has historically relied on pedestrian trajectory datasets [12, 13, 14] and egocentric robot observations [15, 16]. To model these complex shared environments, Graph Neural Networks (GNNs) have emerged as the standard architecture, as they naturally mirror the topological structure of human crowds [17, 18]. Recent heterogeneous and attention-based spatiotemporal GNNs excel at weighting relational dependencies between diverse entities to extract continuous kinematic interactions [19, 20]. Current learning efforts tend to focus on trajectory forecasting, which aims to minimize spatial displacement error, and therefore learn to reflect future spatial configurations, rather than reusable social semantics. Recent works have laid the groundwork to push beyond spatial forecasting. Zhang

et al. [21] provide a dataset and baseline for predicting human perceptions of robot performance and Munje et al. [6] established a benchmark for VLM understanding of social interactions in navigation. Still, existing trajectory-focused models struggle to generalize to *social* tasks because they do not explicitly model the normative behaviors and interpersonal associations that define social interactions [22, 11].

Foundation models and the HRI data bottleneck. Foundation and world models demonstrate impressive predictive capabilities for embodied decision-making [23, 24, 25, 26], including strong zero-shot transfer in Vision-Language-Action (VLA) controllers [27, 28, 29]. However, these models rely on massive, unified datasets. In contrast, grounded HRI data is scarce, geographically constrained, and highly fragmented across varying sensor modalities, survey instruments, and tasks [30, 31, 32, 33, 34, 35, 36, 37]. Applying foundation-scale learning to HRI is therefore hindered by data scarcity, leading unstructured models to overfit or fail to generalize across diverse social scenarios [4].

Relational inductive bias. To learn efficiently in low-data regimes, recent approaches have leveraged structured representations. Relational inductive bias [9] constrains neural architectures to represent entities as nodes and interactions as edges, encouraging networks to deduce underlying physical dynamics without relying on hand-engineered features [38]. While this bias has driven advancements in physical object manipulation, its application to extracting social context from interaction remains underexplored [39, 40]. At the same time, generative self-supervised graph representation learning has shown that semantically meaningful embeddings can be extracted without human annotation by reconstructing masked features or topological structures [41, 42, 43].

3 Method

In this section, we detail the STARS framework. We first formalize the problem of extracting representations from heterogeneous social navigation datasets, outline our graph construction process, and describe the unsupervised graph autoencoder used to learn the latent social semantics.

3.1 Problem Formulation

Let $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_K\}$ denote a collection of social navigation datasets (Fig. 2A), where index k identifies a dataset in the collection. Each dataset $\mathcal{D}_k$ contains a set of interaction scenarios $\{x_i^{(k)}\}$. An interaction scenario x is a fixed-duration observation of T_k timesteps containing multiple agents sharing a physical space. Rather than operating directly on arbitrary raw sensor inputs, STARS converts each scenario into a common graph representation with agent node attributes and temporal pairwise edge features. Let $\mathcal{A}$ denote the finite set of node types, $d_v^{(k)}$ the dimension of the node attributes supplied by dataset k , and d_e the dimension of the pairwise features. We represent each scenario with N nodes as a tuple $G = (\tau, \mathbf{X}^V, \mathbf{m}, \mathbf{X}^E)$ in the graph space $\mathcal{G}_k$, defined as:

$$\mathcal{G}_k = \bigsqcup_{N \geq 2} \underbrace{\mathcal{A}^N}_{\text{types } \tau} \times \underbrace{\mathbb{R}^{N \times d_v^{(k)}}}_{\text{node attributes } \mathbf{X}^V} \times \underbrace{\{0, 1\}^{N \times T_k}}_{\text{valid mask } \mathbf{m}} \times \underbrace{\mathbb{R}^{N \times N \times T_k \times d_e}}_{\text{edge features } \mathbf{X}^E} \quad (1)$$

The disjoint union over N allows the number of agents to vary across scenarios. Each node $v_i \in V$ corresponds to an agent and has type $\tau_i \in \mathcal{A}$. The node set is partitioned as $V = V_R \cup V_H$, where V_R contains robot nodes and V_H contains human nodes, and the type of each directed edge follows from its endpoints, $\rho_{ij} = (\tau_i, \tau_j)$. The valid mask entry $m_{i,t}$ records whether node i is observed at timestep t , and edge validity is given by $M_{ij,t}^E = m_{i,t} m_{j,t} \mathbb{I}[i \neq j]$, so the graph is complete over every ordered pair of nodes observed at the same timestep. Each row $\mathbf{x}_i$ of $\mathbf{X}^V$ contains the node attributes available for agent i , such as encoded occupancy grids or trajectory goals, and is zero-initialized otherwise. Each entry $\mathbf{e}_{ij} \in \mathbb{R}^{T_k \times d_e}$ of $\mathbf{X}^E$ is a temporal edge feature sequence containing the relative $SE(2)$ pose of node j in the body frame of node i and its temporal deltas, with $d_e = 8$ (Appendix B.1.2). Datasets may differ in sensor modality, sampling frequency,

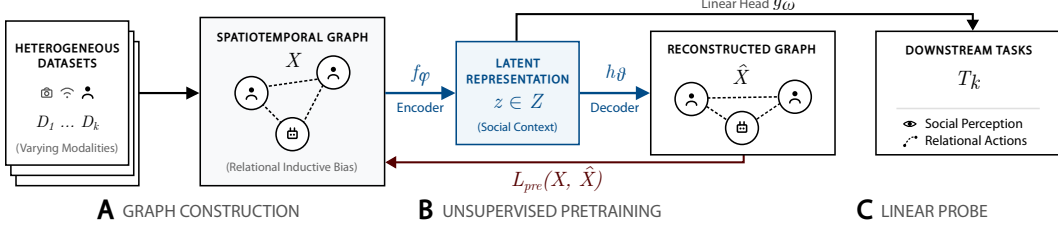


Figure 1: **STARS** three stages: **(A)** construct spatiotemporal graphs from social navigation datasets, **(B)** utilize an unsupervised message-passing graph autoencoder to compress physical dynamics into latent representations, and **(C)** adapt frozen latents to downstream tasks using lightweight task-specific prediction heads. See Sec. 3 for details.

agent types, and available annotations and in Eq. 1 that variation is confined to $d_v^{(k)}$ and T_k . To handle these diverse inputs, all raw sensor modalities are projected into a common hidden dimension prior to message passing. The relational message-passing core and latent heads are shared across datasets, while dataset-dependent input projections, temporal edge encoders, and edge decoders accommodate differences in feature dimensions and temporal windows (Appendix B.2.1).

The representation-learning problem maps an interaction scenario x , through its graph $G \in \mathcal{G}_k$, to compact per-node representations $\mathbf{Z}$ that support downstream social navigation tasks. For a downstream task $\mathcal{T}_m$, a subset of scenarios is paired with labels $\{(x_i, y_i^{(m)})\}$, where $y_i^{(m)} \in \mathcal{Y}_m$ represents the task-specific annotation, which in our evaluations include subjective human perceptions of robot behavior and discrete pedestrian action categories. A representation is considered better if it yields stronger held-out performance on $\mathcal{T}_m$ under the same labeled-data budget and downstream adaptation protocol. Thus, the goal is to obtain compact and reusable representations that support strong downstream performance when labeled examples for each task are scarce.

3.2 STARS Overview

Motivated by a need to learn from diverse social navigation datasets containing spatiotemporal interactions while preserving the relational structure needed for downstream social tasks, STARS is designed to disentangle spatiotemporal dynamics via autoencoding to per-node latents $\mathbf{Z} = f_\phi(G)$ that capture reusable social context [11].

As shown in Figure 1, STARS is a three-stage framework for learning representations from spatiotemporal data. First, STARS extracts features from social navigation datasets (examples shown in Figure 2) to construct unified spatiotemporal graphs. While STARS can apply generally to any spatiotemporal relational data, in this work we focus on the application area of social navigation. Explicitly modeling human and robot agents as nodes and their physical interactions as edges imposes a strong relational inductive bias. Second, a pretraining step compresses physical dynamics into a compact latent space via a self-supervised message-passing graph autoencoder. Self-supervised reconstruction reduces overfitting to scarce labels, enabling higher-capacity encoders compared to a supervised learning baseline (Appendix E.4.1). To evaluate the utility of these representations for social tasks, we subsequently apply a lightweight, task-specific linear prediction head to the frozen latents. We instantiate STARS using both homogeneous and heterogeneous graph formulations to study how explicit entity and relation types affect STARS’ learned representations.

3.3 Spatiotemporal Relational Graph Construction

Our construction makes two design choices. First, our graphs are *edge-feature dense*, capturing temporal relationships primarily through edge rather than node attributes. Second, rather than using distance-based thresholds, we construct *complete graphs* to capture interactions between all observed agents. An example graph construction is shown in Appendix Fig. 3, and full input projection specifications are given in Appendix B.1.

The homogeneous graph represents nodes as generic “agents” and all edges as “interactions.” Robot and human nodes retain distinct input attributes, and entity type enters only through a two-class node embedding; the graph does not include edge-type embeddings. Every dataset is resampled to a common temporal window, so one input projection and one edge encoder serve both datasets, and minibatches may mix sources.

The heterogeneous graph explicitly distinguishes nodes by entity type. Edges are also typed based on the connected node types. Robot nodes may include robot-specific attributes, such as encoded occupancy grids or trajectory goals when available, while human nodes are associated with a learnable semantic embedding representing the entity type, concatenated with a mean-pooled summary of their incident edge features. The edge set captures typed relations such as robot-human, human-robot, and human-human interactions, with relation types provided separately through edge-type embeddings.

3.4 Unsupervised Graph Autoencoder

The core of STARS is a graph encoder implemented as a Message-Passing Graph Neural Network (MPGNN) that maps the spatiotemporal graph into node-level latent distributions.

Message Passing: The graph encoder performs message passing over the complete interaction graph. In STARS’ *heterogeneous* graph formulation, message passing uses a HEAT-style update [44]. Let $\mathbf{h}_j^{(\ell)}$ and $\mathbf{h}_i^{(\ell)}$ denote the source and destination node embeddings at layer ℓ , and $\mathbf{t}_{ij}$ denote a learned embedding for the specific edge type (e.g., human-robot vs. human-human). Let $\tilde{\mathbf{e}}_{ij}^{(0)} = \text{Enc}_{\text{edge}}(\mathbf{e}_{ij})$ denote the initial encoded temporal edge representation. We construct a joint interaction context:

$$\mathbf{u}_{ij}^{(\ell)} = [\mathbf{h}_j^{(\ell)} \parallel \mathbf{h}_i^{(\ell)} \parallel \tilde{\mathbf{e}}_{ij}^{(\ell)} \parallel \mathbf{t}_{ij}] \quad (2)$$

The representation $\mathbf{u}_{ij}^{(\ell)}$ includes the full context of the interaction. During message passing, the network must determine both the content of the information being transmitted and the relative importance of that information. Therefore, we use $\mathbf{u}_{ij}^{(\ell)}$ to independently parameterize both the raw attention score $s_{ij}^{(\ell)}$ and the message content $\mathbf{m}_{ij}^{(\ell)}$ via two learnable functions:

$$s_{ij}^{(\ell)} = f_{\text{attn}}(\mathbf{u}_{ij}^{(\ell)}), \quad \mathbf{m}_{ij}^{(\ell)} = f_{\text{msg}}(\mathbf{u}_{ij}^{(\ell)}) \quad (3)$$

Attention weights are normalized via a destination-wise softmax $\alpha_{ij}^{(\ell)}$, and the aggregated message is used to update the node states via a residual LayerNorm step. The heterogeneous formulation also updates $\tilde{\mathbf{e}}_{ij}^{(\ell)}$ between message-passing layers using the corresponding edge message; full update equations are provided in Appendix B.2.3. Typed-edge attention enables the model to learn which distant interactions matter without needing separate message-passing modules per relation. The *homogeneous* graph formulation omits edge-type embeddings $\mathbf{t}_{ij}$ from $\mathbf{u}_{ij}^{(\ell)}$, using only the source node embedding, destination node embedding, and encoded temporal edge features $\tilde{\mathbf{e}}_{ij}$ in the update.

Latent Encoding: After L message-passing layers, shared linear projection heads parameterize a diagonal Gaussian latent distribution for each node:

$$q_{\phi}(\mathbf{z}_i \mid G) = \mathcal{N}(\boldsymbol{\mu}_i, \text{diag}(\boldsymbol{\sigma}_i^2)), \quad \boldsymbol{\mu}_i = f_{\mu}(\mathbf{h}_i^{(L)}), \quad \log \boldsymbol{\sigma}_i^2 = f_{\sigma}(\mathbf{h}_i^{(L)}) \quad (4)$$

Latent samples are drawn via the reparameterization trick, $\mathbf{z}_i = \boldsymbol{\mu}_i + \boldsymbol{\sigma}_i \odot \boldsymbol{\epsilon}$. Let $\mathbf{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_N]^T \in \mathbb{R}^{N \times d_z}$ denote the stacked node latents, where d_z is the latent dimension, so that $q_{\phi}(\mathbf{Z} \mid G) = \prod_i q_{\phi}(\mathbf{z}_i \mid G)$. This per-node latent formulation preserves the asymmetry of pairwise interactions, as edge reconstruction conditions on both endpoint latent nodes rather than a single pooled graph representation.

Decoding & Objective: Unlike autoregressive models, we employ an unsupervised autoencoding objective over the entire fixed-duration interaction scenario. Specifically, for temporal edge reconstruction, the decoder employs an MLP that takes the concatenated source and destination latents together with the edge-type embedding, $[\mathbf{z}_i \parallel \mathbf{z}_j \parallel \mathbf{t}_{ij}]$, as input.

For node-level reconstruction, we restrict the objective to nodes containing active input attributes. Reconstructing static, zero-initialized features, such as placeholders for entities without observed attributes, would introduce a strong bias toward trivial constant targets, diluting the gradient signal. Let $V_{\text{feat}} \subseteq V$ be the subset of nodes containing active attributes, determined by the dataset and node type rather than by testing whether $\mathbf{x}_i$ is zero (Appendix B.1). In practice, V_{feat} includes only nodes with observed, non-placeholder attributes, which in our datasets is the SEAN-T robot node alone. For nodes without active input attributes, we omit node reconstruction. Table 3 in Appendix B.1 summarizes the node features and reconstruction targets for each architecture and dataset. Crucially, the absence of human node reconstruction does not exclude human nodes from the learned representations or lead to a training mismatch, because the temporal edge decoder must reconstruct $\hat{\mathbf{e}}_{ij}$ from the concatenated endpoint latents alone. The model is forced to encode rich spatiotemporal social semantics and joint context into the human node latents to successfully reconstruct these edges.

The model is trained by minimizing the following VAE objective:

$$\mathcal{L}_{\text{pre}} = \lambda_v \sum_{v_i \in V_{\text{feat}}} \text{MSE}(\mathbf{x}_i, \hat{\mathbf{x}}_i) + \sum_{\rho} \lambda_{\rho} \sum_{\rho_{ij}=\rho} \text{MSE}(\mathbf{e}_{ij}, \hat{\mathbf{e}}_{ij}) + \beta \sum_{v_i \in V} D_{\text{KL}}(q_{\phi}(\mathbf{z}_i | G) \parallel \mathcal{N}(0, I)) \quad (5)$$

where λ_v and λ_{ρ} denote weighting factors for node and edge reconstruction, the latter indexed by relation type, allowing the network to learn across imbalanced datasets, for example, where human-human interactions are much more common than human-robot interactions.

3.5 Task-Agnostic Linear Probing

To evaluate the quality of our learned latent representations and adapt them to downstream tasks with limited labeled data, we apply linear probes [45].

Because STARS learns compressed node-level latent representations $\mathbf{z}_i$, it provides flexibility in selecting and aggregating these node latents to represent different structural components relevant to a given downstream task. After freezing the pretrained encoder weights ϕ , STARS selects and aggregates the necessary latents based on a given downstream task, and because N varies across scenarios this aggregation returns a fixed-dimensional vector that is invariant to permutations of nodes of the same type:

$$\mathbf{z}_{\text{probe}}^{\text{SEAN}} = [\mathbf{z}_r \parallel \bar{\mathbf{z}}_h], \quad \bar{\mathbf{z}}_h = \frac{1}{|V_H|} \sum_{v_i \in V_H} \mathbf{z}_i, \quad \mathbf{z}_{\text{probe}}^{\text{SNS}}(i) = [\mathbf{z}_r \parallel \mathbf{z}_i], \quad v_i \in V_H \quad (6)$$

the first for the graph-level SEAN-T perception tasks and the second for the node-level SNS action task, both of dimension $2d_z$. A linear probe with weights W is then optimized using supervised pairs (G, y) , where y is the task label, by minimizing $\mathcal{L}_{\text{task}} = \text{LossFn}(y, W\mathbf{z}_{\text{probe}} + b)$. This lightweight adaptation step keeps the representation fixed, so downstream performance reflects the information encoded in $\mathbf{z}_{\text{probe}}$ rather than task-specific representation learning.

4 Experiments

We evaluate the effectiveness of STARS to learn social semantics from physical dynamics. Specifically, we demonstrate that STARS achieves data-efficient downstream performance, and through analysis of the latent space topology, we show that the model can distinguish social behaviors, such as action intents, without optimizing the encoder with a supervised classification objective during pretraining. Our research questions are:

- RQ1: Downstream Performance.** How well does STARS work on downstream social navigation tasks compared to unstructured baselines when adapting pretrained representations?
- RQ2: Data Efficiency.** For a given downstream task, what is the impact of labeled data availability, from 1% of the labeled training set (60 samples) to 100%, when using pretrained STARS representations?
- RQ3: Latent Space Semantics.** Do STARS’ self-supervised representations naturally encode social semantics (e.g., action intents) without a supervised action classification objective?

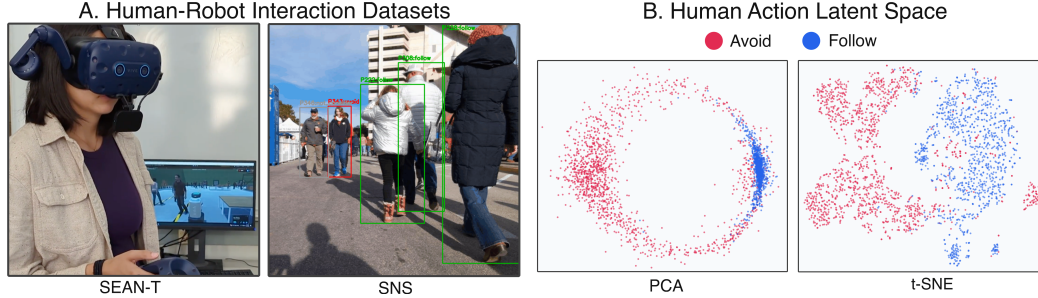


Figure 2: **(A) Dataset Overview.** We utilize two human-robot interaction datasets: **SEAN-T** [21] for evaluating subjective human ratings of robot navigation and **SNS** [6] for downstream relational action classification (e.g., *avoid*, *follow*). **(B) Latent Space Analysis (RQ3).** 2D Projections of the SNS Human Node Action Latent Space for the *avoid* and *follow* actions. We visualize the 128-dimensional representations of nodes extracted from the STARS-trained homogeneous GNN-VAE encoder using linear **PCA** and non-linear **t-SNE**. Each point represents a node embedding, color-coded by action label. See Sec. 4.2 for details.

Datasets and Tasks We evaluate STARS across two distinct datasets to assess unsupervised pre-training and downstream task adaptation. Our pretraining pool consists of multi-agent spatiotemporal data from **SEAN Together (SEAN-T)** [21], which provides 8-second immersive virtual-reality observations of robot navigation, and **SocialNav-SUB (SNS)** [6], a structured social reasoning benchmark containing pedestrian-robot interactions. Examples of the interaction scenarios are shown in Figure 2 and further details are provided in the Appendix A. To evaluate whether our autoencoder successfully captures reusable social semantics, we freeze the pretrained representations and train lightweight linear probes on two distinct downstream tasks: 1) **Perception Prediction (SEAN-T)**: Predicting binary subjective human ratings of a robot’s behavior across three distinct metrics: perceived competence, clarity of intention, and surprise [21]. This task evaluates the model’s ability to extract subjective social impressions directly from physical motion. 2) **Action Classification (SNS)**: Predicting discrete, categorical relational actions taken by pedestrians in dense crowd scenes with up to 30 people (e.g., *avoiding*, *following*, *yielding*) [6]. This task tests the model’s capacity to explicitly classify interaction intents in complex multi-agent environments.

Training and Evaluation All reported results are averaged across 10 random seeds and use early stopping [46] to limit overfitting. Model-specific hyperparameter selection, capacity, and complete training protocols are detailed in Appendix C.

4.1 RQ1 & RQ2: Downstream Performance and Data Efficiency

Experiment: To evaluate downstream performance (RQ1) and data efficiency (RQ2), we pretrain STARS on the combined training splits of the datasets using the self-supervised reconstruction objective. After freezing the encoder, we extract the relevant elements of the latent graph representation (details in the Appendix C) and train the linear probe using progressively larger fractions of the labels from the training split. We compare our approach against several unstructured baselines: a fully connected, feed-forward Multi-Layer Perceptron (MLP), an Autoencoder to ablate the graph structure while retaining data compression, and a Random Forest to provide a non-deep-learning baseline. Note that the Random Forest is omitted for SNS due to the dataset’s variable-sized, high-dimensional features. The Autoencoder receives the full temporal edge representation and follows the same pooled pretraining and frozen adaptation protocol across labeled-data fractions. We report F_1 -Scores for *competence*, *surprise*, and *intention* for SEAN-T and macro F_1 -Score over *avoid*, *follow*, and *not consider* for SNS, with per-class results reported in the Appendix E.1.

Results: Table 1 shows that the heterogeneous (STARS-He) and homogeneous (STARS-Ho) variants are competitive with unstructured baselines on SEAN-T and achieve stronger performance on

Table 1: Downstream Performance (RQ1) and Data Efficiency (RQ2) on SEAN-T and SNS. Results showcase F_1 -Scores ($\mu \pm \sigma$) across progressively scaled fractions of labeled data. We compare unstructured baseline methods against our proposed STARS variants (**STARS-He**: Heterogeneous, **STARS-Ho**: Homogeneous). To rigorously evaluate both representation quality and data efficiency, STARS utilizes a frozen pretrained encoder paired with a lightweight linear probe.

	Method	1%	5%	10%	25%	50%	100%
SEAN-T Competence	MLP	0.450 $\pm$ 0.22	0.546 $\pm$ 0.19	0.664 $\pm$ 0.15	0.761 $\pm$ 0.03	0.778 $\pm$ 0.02	0.797 $\pm$ 0.02
	Autoencoder	0.663 $\pm$ 0.08	0.678 $\pm$ 0.05	0.706 $\pm$ 0.05	0.734 $\pm$ 0.02	0.747 $\pm$ 0.02	0.742 $\pm$ 0.02
	Random Forest	0.557 $\pm$ 0.20	0.641 $\pm$ 0.12	0.680 $\pm$ 0.10	0.704 $\pm$ 0.09	0.715 $\pm$ 0.09	0.718 $\pm$ 0.08
	STARS-He (Ours)	0.635 $\pm$ 0.10	0.673 $\pm$ 0.07	0.726 $\pm$ 0.04	0.762 $\pm$ 0.03	0.781 $\pm$ 0.03	0.794 $\pm$ 0.02
	STARS-Ho (Ours)	0.612 $\pm$ 0.14	0.702 $\pm$ 0.08	0.738 $\pm$ 0.05	0.765 $\pm$ 0.04	0.787 $\pm$ 0.03	0.799 $\pm$ 0.02
SEAN-T Surprise	MLP	0.217 $\pm$ 0.22	0.250 $\pm$ 0.14	0.423 $\pm$ 0.27	0.450 $\pm$ 0.21	0.630 $\pm$ 0.23	0.720 $\pm$ 0.09
	Autoencoder	0.548 $\pm$ 0.10	0.597 $\pm$ 0.12	0.660 $\pm$ 0.07	0.674 $\pm$ 0.09	0.712 $\pm$ 0.10	0.739 $\pm$ 0.06
	Random Forest	0.291 $\pm$ 0.24	0.341 $\pm$ 0.19	0.373 $\pm$ 0.20	0.444 $\pm$ 0.16	0.470 $\pm$ 0.15	0.496 $\pm$ 0.15
	STARS-He (Ours)	0.455 $\pm$ 0.17	0.481 $\pm$ 0.17	0.526 $\pm$ 0.15	0.603 $\pm$ 0.12	0.655 $\pm$ 0.08	0.709 $\pm$ 0.04
	STARS-Ho (Ours)	0.440 $\pm$ 0.23	0.541 $\pm$ 0.21	0.548 $\pm$ 0.20	0.610 $\pm$ 0.16	0.713 $\pm$ 0.06	0.741 $\pm$ 0.03
SEAN-T Intention	MLP	0.479 $\pm$ 0.20	0.533 $\pm$ 0.13	0.606 $\pm$ 0.06	0.659 $\pm$ 0.03	0.693 $\pm$ 0.03	0.703 $\pm$ 0.01
	Autoencoder	0.569 $\pm$ 0.04	0.622 $\pm$ 0.03	0.659 $\pm$ 0.03	0.665 $\pm$ 0.05	0.661 $\pm$ 0.03	0.664 $\pm$ 0.02
	Random Forest	0.592 $\pm$ 0.18	0.665 $\pm$ 0.10	0.685 $\pm$ 0.09	0.692 $\pm$ 0.09	0.700 $\pm$ 0.08	0.698 $\pm$ 0.09
	STARS-He (Ours)	0.576 $\pm$ 0.08	0.619 $\pm$ 0.05	0.671 $\pm$ 0.04	0.694 $\pm$ 0.03	0.705 $\pm$ 0.03	0.721 $\pm$ 0.03
	STARS-Ho (Ours)	0.603 $\pm$ 0.10	0.671 $\pm$ 0.07	0.701 $\pm$ 0.06	0.715 $\pm$ 0.04	0.723 $\pm$ 0.03	0.734 $\pm$ 0.02
SNS Macro F_1	MLP	0.294 $\pm$ 0.00	0.297 $\pm$ 0.01	0.300 $\pm$ 0.01	0.306 $\pm$ 0.02	0.295 $\pm$ 0.01	0.293 $\pm$ 0.00
	Autoencoder	0.308 $\pm$ 0.02	0.309 $\pm$ 0.02	0.307 $\pm$ 0.02	0.310 $\pm$ 0.02	0.309 $\pm$ 0.02	0.332 $\pm$ 0.01
	STARS-He (Ours)	0.362 $\pm$ 0.06	0.447 $\pm$ 0.05	0.470 $\pm$ 0.04	0.493 $\pm$ 0.03	0.501 $\pm$ 0.03	0.503 $\pm$ 0.02
	STARS-Ho (Ours)	0.344 $\pm$ 0.05	0.431 $\pm$ 0.05	0.453 $\pm$ 0.04	0.468 $\pm$ 0.03	0.471 $\pm$ 0.02	0.475 $\pm$ 0.02

most evaluated settings. While both exhibit competitive results, STARS-Ho yields the highest overall scores in most SEAN-T categories. STARS-He achieves the highest SNS macro F_1 across all labeled-data fractions, with per-class results reported in Appendix E.1. On SNS, unstructured baselines fail to extract relational signal, plateauing near a macro F_1 -Score of 0.3, regardless of how much labeled data is provided. In contrast, STARS-He rapidly acquires task-relevant information, achieving 0.362 with just 1% of the data and achieves 0.503 at 100%, indicating that the heterogeneous formulation is particularly effective for this relational action classification task. We additionally compare against the same heterogeneous graph architecture trained end-to-end on SNS labels. STARS-He achieves a macro F_1 -Score of 0.503 at 100% labeled data, compared with 0.445 for the supervised model; full results are reported in Appendix E.4.1. To assess cross-dataset transfer, we pretrain on one dataset and probe on the other. At 100% labeled data, STARS-He pretrained only on SEAN-T achieves 0.497 macro F_1 on SNS compared with 0.503 under joint pretraining, while SNS-only pretraining achieves 0.801/0.710/0.729 on the SEAN-T competence/surprise/intention tasks compared with 0.794/0.709/0.721 under joint pretraining. STARS-Ho also transfers across datasets, although joint pretraining provides larger gains; full results across labeled-data fractions are reported in Appendix E.3. Finally, the benefit of STARS’s relational structure is most pronounced on SNS, where STARS-He consistently outperforms the temporal Autoencoder across all labeled-data fractions. On SEAN-T, the Autoencoder remains competitive in several settings, suggesting that explicit relational structure provides a greater benefit for pedestrian action classification than for the perception-prediction tasks.

4.2 RQ3: Latent Space Semantics

Experiment: To qualitatively evaluate whether the learned representations capture high-level social semantics without optimizing the encoder with a supervised classification objective (RQ3), we analyze the topology of the latent spaces. Using the pretrained latent representation from the homogeneous graph encoder reported in Table 1, we extract frozen representations from the 128-dimensional node-level embeddings for SNS. To focus on semantic navigation intents, we conduct our experiments on the dominant action labels in the SocialNav-SUB (SNS) dataset, *avoid* and *follow*. We apply both Principal Component Analysis (PCA) for linear projection and t-Distributed

Stochastic Neighbor Embedding (t-SNE) for non-linear dimensionality reduction to map the high-dimensional representations to a 2D space, color-coded by downstream class labels.

Results: Latent space projections shown in Figure 2 reveal structural self-organization corresponding to downstream human navigation semantics. The linear PCA projection (left) shows smooth variation across the latent space, while the non-linear t-SNE projection (right) reveals visually distinct groupings associated with action classes. Together, these projections provide qualitative evidence that the learned representations contain action relevant structure, despite the encoder not being optimized with a supervised action classification objective.

5 Limitations and Future Work

While STARS generalizes well, its autoencoder and message passing inherently smooth high-frequency pairwise geometries. Therefore, these latents excel at node-level tasks like intent prediction, but are less effective on edge-centric tasks (e.g., group detection) that require exact relational geometry. Additionally, standardizing diverse datasets into unified graphs can discard dataset-specific nuances. Future work can explore more flexible graph construction pipelines and the use of intermediate, pre-message-passing temporal edge embeddings to bypass the autoencoder bottleneck, preserving the precise geometric information necessary for fine-grained relational tasks. Our evaluation assumes fixed-duration interaction windows and processed agent tracks, and is limited to curated social navigation datasets. A promising direction for future work is to evaluate STARS under online segmentation, noisier perception, and more diverse real-world settings.

6 Conclusion

In this work, we introduced STARS for self-supervised representation learning in social navigation by bridging continuous physical dynamics and high-level social reasoning. By modeling scenarios as spatiotemporal graphs and using a self-supervised graph autoencoder, STARS injects a strong relational inductive bias. Evaluations show that our learned representations are highly reusable and data-efficient, and can encode complex social semantics without directly optimizing the encoder for downstream tasks. Ultimately, our results provide evidence that structural priors can help offset the scarcity of grounded HRI data, offering a promising direction toward reusable and data-efficient representations for social navigation.

Acknowledgments

A portion of work has taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by NSF (FAIN-2019844, NRT-2125858, OIA-2535195), ONR (N00014-24-1-2550), ARO (W911NF-17-2-0181, W911NF-23-2-0004, W911NF-25-1-0065), Lockheed Martin, Lyda Hill, and Good Systems. Peter Stone serves as the Chief Scientist of Sony CTC and receives financial compensation for that role. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research. Another portion of this work has taken place at the Autonomous Mobile Robotics Laboratory (AMRL) at the Artificial Intelligence Laboratory, The University of Texas at Austin. AMRL research is supported in part by the National Science Foundation (OIA-2535195, CAREER-2046955, OIA-2219236, DGE-2125858, CCF-2319471). The authors also acknowledge the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing high-performance computing resources that have contributed to the research results reported within this paper. Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

References

- [1] C. Mavrogiannis, F. Baldini, A. Wang, D. Zhao, P. Trautman, A. Steinfeld, and J. Oh. Core challenges of social robot navigation: A survey. *ACM Transactions on Human-Robot Interaction*, 12(3):1–39, 2023.
- [2] A. Francis, C. Pérez-d’Arpino, C. Li, F. Xia, A. Alahi, R. Alami, A. Bera, A. Biswas, J. Biswas, R. Chandra, et al. Principles and guidelines for evaluating social robot navigation algorithms. *ACM Transactions on Human-Robot Interaction*, 14(2):1–65, 2025.
- [3] X. Zhou, J. Liu, A. Yerukola, H. Kim, and M. Sap. Social world models. *arXiv preprint arXiv:2509.00559*, 2025.
- [4] M. Lisondra, B. Benhabib, and G. Nejat. Embodied ai with foundation models for mobile service robots: A systematic review. *Robotics*, 15(3):55, 2026.
- [5] S. K. Ramakrishnan, E. Wijmans, P. Kraehenbuehl, and V. Koltun. Does spatial cognition emerge in frontier models? *arXiv preprint arXiv:2410.06468*, 2024.
- [6] M. J. Munje, C. Tang, S. Liu, Z. Hu, Y. Zhu, J. Cui, G. Warnell, J. Biswas, and P. Stone. Socialnav-sub: Benchmarking vlms for scene understanding in social robot navigation. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2025. Project page: <https://larg.github.io/socialnav-sub>.
- [7] Q. Gao, X. Pi, K. Liu, J. Chen, R. Yang, X. Huang, X. Fang, L. Sun, G. Kishore, B. Ai, et al. Do vision-language models have internal world models? towards an atomic evaluation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 26170–26195, 2025.
- [8] M. M. de Graaf, S. Ben Allouch, and J. A. Van Dijk. What makes robots social?: A user’s perspective on characteristics for social human-robot interaction. In *International Conference on Social Robotics*, pages 184–193. Springer, 2015.
- [9] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.
- [10] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Message passing neural networks. In *Machine learning meets quantum physics*, pages 199–214. Springer, 2020.
- [11] N. Tsoi, A. Xiang, P. Yu, S. S. Sohn, G. Schwartz, S. Ramesh, M. Hussein, A. W. Gupta, M. Kapadia, and M. Vázquez. Sean 2.0: Formalizing and generating social situations for robot navigation. *IEEE Robotics and Automation Letters*, 7(4):11047–11054, 2022.
- [12] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool. You’ll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th international conference on computer vision*, pages 261–268. IEEE, 2009.
- [13] A. Lerner, Y. Chrysanthou, and D. Lischinski. Crowds by example. In *Computer graphics forum*. Wiley Online Library, 2007.
- [14] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In *European conference on computer vision*, pages 549–565. Springer, 2016.
- [15] R. Martín-Martín, H. Rezatofighi, A. Shenoi, M. Patel, J. Gwak, N. Dass, A. Federman, P. Goebel, and S. Savarese. Jrdb: A dataset and benchmark for visual perception for navigation in human environments. *arXiv preprint arXiv:1910.11792*, 2019.

- [16] H. Karnan, A. Nair, X. Xiao, G. Warnell, S. Pirk, A. Toshev, J. Hart, J. Biswas, and P. Stone. Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation. *IEEE Robotics and Automation Letters*, 7(4):11807–11814, 2022.
- [17] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14424–14432, 2020.
- [18] V. Kosaraju, A. Sadeghian, R. Martín-Martín, I. Reid, H. Rezatofighi, and S. Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. *Advances in neural information processing systems*, 32, 2019.
- [19] X. Li, Y. Liang, Z. Yang, and J. Li. Pedestrian trajectory prediction based on dual social graph attention network. *Applied Sciences*, 15(8):4285, 2025.
- [20] R. Li, T. Qiao, S. Katsigiannis, Z. Zhu, and H. P. Shum. Unified spatial-temporal edge-enhanced graph networks for pedestrian trajectory prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, 35(7):7047–7060, 2025.
- [21] Q. Zhang, N. Tsoi, M. Nagib, B. Choi, J. Tan, H.-T. L. Chiang, and M. Vázquez. Predicting human impressions of robot performance during navigation tasks. *arXiv preprint arXiv:2310.11590*, 2023.
- [22] S. Thompson, K. Candon, and M. Vázquez. The social context of human-robot interactions. *Annual Review of Control, Robotics, and Autonomous Systems*, 9, 2025.
- [23] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.
- [24] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.
- [25] J. Bruce, M. D. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes, M. Lai, A. Mavalankar, R. Steigerwald, C. Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.
- [26] J. Parker-Holder, P. Ball, J. Bruce, V. Dasagi, K. Holsheimer, C. Kaplanis, A. Moufarek, G. Scully, J. Shar, J. Shi, S. Spencer, J. Yung, M. Dennis, S. Kenjeyev, S. Long, V. Mnih, H. Chan, M. Gazeau, B. Li, F. Pardo, L. Wang, L. Zhang, F. Besse, T. Harley, A. Mitenkova, J. Wang, J. Clune, D. Hassabis, R. Hadsell, A. Bolton, S. Singh, and T. Rocktäschel. Genie 2: A large-scale foundation world model. URL: <https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model>, 2024. URL <https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model/>.
- [27] A. O’Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.
- [28] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.
- [29] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

- [30] J. A. Duncan, F. Alambeigi, and M. W. Pryor. A survey of multimodal perception methods for human–robot interaction in social environments. *ACM Transactions on Human-Robot Interaction*, 13(4):1–50, 2024.
- [31] J. Ondras, A. Anwar, T. Wu, F. Bu, M. Jung, J. J. Ortiz, and T. Bhattacharjee. Human-robot commensality: Bite timing prediction for robot-assisted feeding in groups. In *6th Annual Conference on Robot Learning*, 2022.
- [32] F. Bu and W. Ju. Ssup-hri: Social signaling in urban public human-robot interaction dataset. In *International Conference on Social Robotics*, pages 479–487. Springer, 2024.
- [33] J. S. Heinisch, J. Kirchhoff, P. Busch, J. Wendt, O. von Stryk, and K. David. Physiological data for affective computing in hri with anthropomorphic service robots: the affect-hri data set. *Scientific Data*, 11(1):333, 2024.
- [34] S. Heo, Y. Cho, J. Park, S. Cho, Z. Tsoy, H. Lim, and Y. Cha. Diverse humanoid robot pose estimation from images using only sparse datasets. *Applied Sciences*, 14(19):9042, 2024.
- [35] S. Thompson, A. Gupta, A. W. Gupta, A. Chen, and M. Vázquez. Conversational group detection with graph neural networks. In *Proceedings of the 2021 International Conference on Multimodal Interaction*, pages 248–252, 2021.
- [36] F. Yang, Y. Gao, R. Ma, S. Zojaji, G. Castellano, and C. Peters. A dataset of human and robot approach behaviors into small free-standing conversational groups. *PloS one*, 16(2):e0247364, 2021.
- [37] S. Shrestha, Y. Zha, S. Banagiri, G. Gao, Y. Aloimonos, and C. Fermuller. Natsgd: A dataset with speech, gestures, and demonstrations for robot learning in natural human-robot interaction. *arXiv preprint arXiv:2403.02274*, 2024.
- [38] F. Ferreira, L. Shao, T. Asfour, and J. Bohg. Learning visual dynamics models of rigid objects using relational inductive biases. *arXiv preprint arXiv:1909.03749*, 2019.
- [39] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. Battaglia. Learning to simulate complex physics with graph networks. In *International conference on machine learning*, pages 8459–8468. PMLR, 2020.
- [40] M. Malik and L. Isik. Relational visual representations underlie human social interaction recognition. *Nature Communications*, 14(1):7317, 2023.
- [41] T. N. Kipf and M. Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.
- [42] J. Liu, C. Zhang, Z. He, W. Zhang, and N. Li. Network-to-network: Self-supervised network representation learning via position prediction. *IEEE Transactions on Knowledge and Data Engineering*, 37(3):1354–1365, 2025.
- [43] P. Falqueto et al. *Human-Aware Robotics: Predict, Assist, and Plan for Seamless Interaction*. PhD thesis, University of Trento, 2025.
- [44] X. Mo, Y. Xing, and C. Lv. Heterogeneous edge-enhanced graph attention network for multi-agent trajectory prediction. In *arXiv preprint arXiv:2106.07161*, 2021. URL <https://arxiv.org/pdf/2106.07161>.
- [45] G. Alain and Y. Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.
- [46] L. Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the trade*, pages 55–69. Springer, 2002.

- [47] Y. Shi, Z. Huang, S. Feng, H. Zhong, W. Wang, and Y. Sun. Masked label prediction: Unified message passing model for semi-supervised classification, 2021. URL <https://arxiv.org/abs/2009.03509>.
- [48] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.
- [49] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

A Dataset and Graph Statistics

A.1 Datasets

We evaluate our method on two datasets providing physical robot-human interaction data: **SEAN Together** and **SocialNav-SUB**. Both datasets are converted into the common graph format described in Section 3.

SEAN Together (SEAN-T). SEAN Together contains 8-second (5 Hz) immersive virtual-reality observations of robot navigation behavior paired with subjective human ratings [21]. Participants rated interactions along three dimensions: perceived competence, clarity of intention, and surprise. We use this to evaluate the extraction of subjective social impressions from physical motion. We organize the SEAN-T dataset by participant and then split it into train, validation, and test subsets corresponding to 80%, 16% and 4% of participants.

SocialNav-SUB (SNS). SocialNav-SUB expands the SCAND dataset [16] into a structured social reasoning benchmark for navigation scenes [6]. The dataset contains robot-centric interaction scenarios with relational action labels, such as *avoiding*, *following*, and *yielding to*. We use this dataset to evaluate action classification from physical interaction dynamics. Training uses automatically generated action labels from the SocialNav-SUB heuristic labeling procedure, with the training portion split into 85% training and 15% validation, while evaluation uses a separate held-out test set annotated by human labelers.

Table 2: Dataset and graph construction details. The temporal window is fixed within each dataset. The maximum number of edges follows from the maximum number of nodes under the fully connected graph construction.

Dataset	# Scenarios	Sample Freq.	Window Size T	Window Duration	Max Nodes N	Max Edges
SEAN-T	2969	5 Hz	40	8s	16	240
SocialNav-SUB	3052	8 Hz	20	2.5s	31	930

B Implementation Details

B.1 Graph Construction Details

B.1.1 Shared Graph Representation

All variants represent each fixed-duration interaction scenario as the spatiotemporal graph G of Eq. 1, where nodes correspond to observed agents and directed edges correspond to ordered pairwise interactions. Each edge stores a temporal sequence of relative spatial and motion features, enumerated in Appendix B.1.2. The homogeneous and heterogeneous variants differ only in how entity and relation type information is represented, and in which components are instantiated per dataset (Appendix B.2.1).

B.1.2 Edge Feature Channels

Both variants use the same $d_e = 8$ edge channels, differing only in sequence length. The first four are the relative $SE(2)$ pose of node j in the body frame of node i , and the remaining four are their temporal deltas, $\delta_t u = u_t - u_{t-1}$:

$$\mathbf{e}_{ij,t} = [\Delta x_{ij,t}, \Delta y_{ij,t}, \cos \Delta \theta_{ij,t}, \sin \Delta \theta_{ij,t}, \delta_t \Delta x_{ij}, \delta_t \Delta y_{ij}, \delta_t \cos \Delta \theta_{ij}, \delta_t \sin \Delta \theta_{ij}] \quad (7)$$

Each delta is set to zero at the first observed timestep of a contiguous track segment, so a pedestrian who leaves and re-enters the window contributes no spurious jump across the gap. Because headings enter only through $(\cos, \sin)$ pairs, the encoding has no wrap-around discontinuity. The deltas are per-timestep differences rather than rates, so their scale depends on the sampling frequency; features are standard-scaled before training, which removes the resulting offset between datasets but not the difference in what one timestep represents physically.

B.1.3 Homogeneous Graph

Unified Representation: The homogeneous model processes both SEAN Together and SocialNav-SUB (SNS) as single-type graphs, while robot and human identity is included only as a 2-class input node embedding. All data is unified to a 40-timestep window, with SNS interpolated from 20 to 40 steps, and projected into a shared 592-dimensional node feature space.

Feature Sets:

- **Node Features:** SEAN robot nodes contain 512D map features and 80D goal trajectories. For SNS robot nodes and all human nodes across both datasets, these 592 dimensions are zero-initialized. A bias-free linear projection is used so that zeroed features contribute only through the learnable type embeddings.
- **Edge Features:** 8-dimensional relative $SE(2)$ poses and temporal deltas (Appendix B.1.2).

Processing: A *Temporal Edge Transformer* encodes the 40-timestep edge attributes into fixed-size embeddings. Message passing is performed using a homogeneous *TransformerConv* layer that operates on the entire graph simultaneously. An edge-level padding mask is derived from the *valid mask* to ensure the model ignores invalid timesteps during temporal pooling.

B.1.4 Heterogeneous Graph

Multi-Type Topology: The heterogeneous model utilizes a *HeteroData* structure with two node types (*robot*, *bystander*) and three directed edge types: (*robot*, *to*, *bystander*), (*bystander*, *to*, *robot*), and (*bystander*, *to*, *bystander*). Unlike the homogeneous case, SEAN (40 steps) and SNS (20 steps) retain their native temporal resolutions, and batches are restricted to a single source to prevent temporal mismatch. For SEAN-only heterogeneous experiments, we additionally retain the SEAN-specific *follower* node, adding directed edges between the follower and both robot and bystander nodes. This preserves the full interaction structure available in SEAN while keeping the combined heterogeneous model restricted to node and edge types shared across datasets.

Table 3: Node features by architecture and dataset. All configurations use 8D edge features comprising relative position, relative heading (sine and cosine), and their temporal differences. Both architectures use reconstruction pretraining followed by linear probing with a frozen encoder.

Model	Dataset	Robot features	Human features
STARS-He	SEAN-T	592D map and goal features	8D learned embedding + 8D pooled edge features
STARS-He	SNS	8D learned embedding + 8D pooled edge features	8D learned embedding + 8D pooled edge features
STARS-Ho	SEAN-T	592D map and goal features + type embedding	592D zero vector + type embedding
STARS-Ho	SNS	592D zero vector + type embedding	592D zero vector + type embedding

Feature Sets:

- **SEAN Nodes:** The robot uses 592D features (map + goal). Bystanders use a learnable 8D embedding concatenated with an 8-dimensional positional feature derived from the mean-pooled relative pose of the *robot-to-bystander* edge.
- **SNS Nodes:** Both robot and bystander nodes use a learnable 8D embedding concatenated with an 8-dimensional positional feature derived from the mean-pooled relative pose of the given node’s incident *robot-to-bystander* edges.
- **Edge Features:** 8-dimensional SE(2) relative pose and dynamics, matching the dimensions of the homogeneous case but varying in sequence length.

B.2 Graph Encoder Details

B.2.1 Per-Dataset and Shared Components

Because $d_v^{(k)}$ and T_k differ across datasets, components whose input or output dimensionality depends on these quantities are instantiated per dataset, while the relational message-passing core and latent heads are shared. Table 4 gives the split, which differs between the two variants and is the main structural distinction between them: the homogeneous variant unifies the datasets at conversion time and shares every component, while the heterogeneous variant keeps them separate through the input and output adapters and shares the relational core. In the heterogeneous variant this means four node feature projections, one per dataset and node type, three learned type embeddings, two temporal edge encoders, and two edge decoders.

Table 4: Components shared across datasets in each variant.

Component	Homogeneous	Heterogeneous
Node feature projection	shared	per dataset and node type
Temporal edge encoder	shared ($T=40$)	per dataset (40, 20)
Message-passing layers	shared	shared
Edge-type embedding	not used	shared
Latent heads f_μ, f_σ	shared	shared
Edge decoder	shared ($T=40$)	per dataset
Node decoder	SEAN-T robot node only	

B.2.2 Feature Preprocessing

Node attributes are first projected into a shared hidden dimension d_{hidden} via type-specific Multi-layer Perceptrons (MLPs):

$$\mathbf{h}_i^{(0)} = \text{MLP}_{\tau(i)}(\mathbf{x}_i) \quad (8)$$

where $\tau(i)$ denotes the node type. To encode temporal edge sequences across datasets, temporal edge sequences $\mathbf{e}_{ij}$ are passed through a Transformer edge encoder with learned positional embeddings. This compresses the full temporal sequence into a fixed-dimensional relation embedding $\tilde{\mathbf{e}}_{ij} \in \mathbb{R}^{d_{hidden}}$:

$$\tilde{\mathbf{e}}_{ij} = \text{Enc}_{\text{edge}}(\mathbf{e}_{ij}) \quad (9)$$

B.2.3 Message Passing

Heterogeneous Message Passing

The graph encoder performs message passing over the complete interaction graph. In our heterogeneous variant, we utilize a HEAT-style update [44]. Let $\mathbf{h}_j^{(k)}$ and $\mathbf{h}_i^{(k)}$ denote the source and destination node embeddings at layer k , and $\mathbf{t}_{ij}$ denote a learned embedding for the specific edge type. We construct an edge-conditioned interaction representation:

$$\mathbf{u}_{ij}^{(k)} = [\mathbf{h}_j^{(k)} \parallel \mathbf{h}_i^{(k)} \parallel \tilde{\mathbf{e}}_{ij}^{(k)} \parallel \mathbf{t}_{ij}] \quad (10)$$

This representation encapsulates the full context of the interaction. During message passing, the network must determine both the content of the information being transmitted and the relative importance of that information. Therefore, we use $\mathbf{u}_{ij}^{(k)}$ to independently parameterize both the raw attention score $s_{ij}^{(k)}$ and the message content $\mathbf{m}_{ij}^{(k)}$ via two learnable functions:

$$s_{ij}^{(k)} = f_{\text{attn}}(\mathbf{u}_{ij}^{(k)}), \quad \mathbf{m}_{ij}^{(k)} = f_{\text{msg}}(\mathbf{u}_{ij}^{(k)}) \quad (11)$$

Formally, the destination-wise softmax normalizes the raw attention scores across all incoming edges from the neighborhood $\mathcal{N}(i)$:

$$\alpha_{ij}^{(k)} = \frac{\exp(s_{ij}^{(k)})}{\sum_{m \in \mathcal{N}(i)} \exp(s_{im}^{(k)})} \quad (12)$$

These normalized weights are used to compute the aggregated neighborhood message $\bar{\mathbf{m}}_i^{(k)}$:

$$\bar{\mathbf{m}}_i^{(k)} = \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(k)} \mathbf{m}_{ij}^{(k)} \quad (13)$$

The node state is updated via a residual connection and Layer Normalization (LN) to maintain numerical stability during training:

$$\mathbf{h}_i^{(k+1)} = \text{LN}(\mathbf{h}_i^{(k)} + f^{\text{upd}}([\mathbf{h}_i^{(k)} \parallel \bar{\mathbf{m}}_i^{(k)}])) \quad (14)$$

To enhance our edges' influence in the GNN, and for future edge-level downstream task integration, we additionally update edge embeddings at each layer following the Group GNN convention [22]. The message content $\mathbf{m}_{ij}^{(k)}$, which was already computed for node aggregation previously, is reused as a residual update to the edge embedding:

$$\tilde{\mathbf{e}}_{ij}^{(k+1)} = \text{LN}_{\text{edge}}(\tilde{\mathbf{e}}_{ij}^{(k)} + \mathbf{m}_{ij}^{(k)}) \quad (15)$$

We perform a similar computation as the node update equation, except we omit a separate update function for $\mathbf{m}_{ij}^{(k)}$ since it has already been transformed by an MLP, making an additional transformation redundant.

Homogeneous Message Passing

For the homogeneous graph encoder, message passing does not use the HEAT-style update above. It instead applies a shared edge-conditioned attention convolution [47] in which the encoded temporal edge features $\tilde{\mathbf{e}}_{ij}$ enter the key and value terms, with no edge-type embedding.

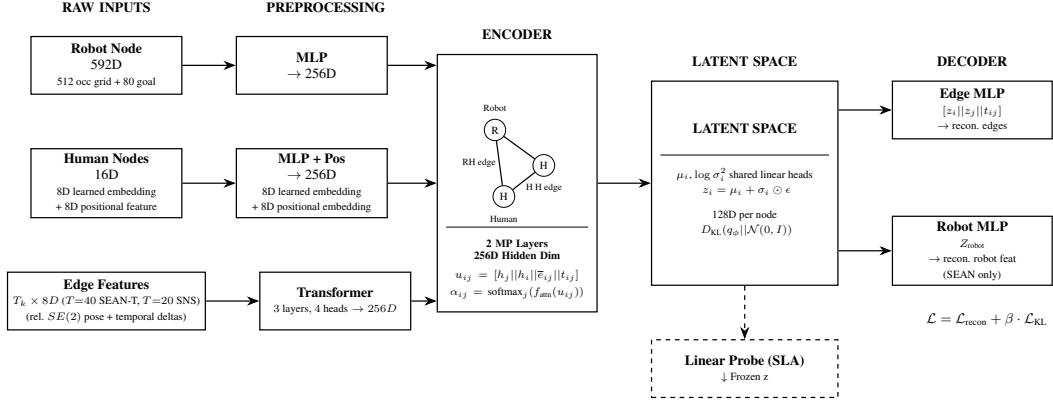


Figure 3: **STARS Heterogeneous architecture overview.** *Raw inputs* (robot occupancy + goal, learned human embeddings, 8D temporal edge features) are projected to a 256D shared hidden space via type-specific *preprocessing* (MLPs for nodes, a Transformer for edges). The *encoder* performs HEAT-style message passing (MP) over the fully connected heterogeneous graph (2 layers), with attention and messages parameterized over the concatenated edge-conditioned representation $\mathbf{u}_{ij}$. The resulting 256D node states are projected through shared linear heads to per-node Gaussian posteriors and sampled to 128D latents $\mathbf{z}_i$ via the reparameterization trick. The *decoder* reconstructs edge features (and robot features on SEAN) from latent pairs, trained jointly with a KL divergence term against $\mathcal{N}(0, I)$. Pretrained encoders are frozen for downstream linear probing (SLA).

C Training and Evaluation Protocol

All experiments were implemented in PyTorch [48] using mixed precision.

We applied standard feature scaling and fast early stopping [46] with patience 20. All reported results are averaged across 10 random seeds. Models were optimized with AdamW using decoupled weight decay [49]; model-specific learning rates and selected hyperparameters are reported in Appendix C.3.

C.1 Pretraining Protocol

Models were first pretrained using the ELBO self-supervised VAE objective described in Sec. 3. Graphs were processed independently within each minibatch, allowing each interaction scenario to contain a variable number of human nodes and edges. Optimization details and selected hyperparameters are provided in Appendix C.3.

C.2 Downstream Adaptation Protocol

After pretraining, encoder weights were frozen and reused for downstream prediction tasks. For supervised linear adaptation (SLA), latent graph representations were extracted from the pretrained encoder and passed into lightweight linear prediction heads. Linear probes were trained independently for each downstream task while keeping the graph encoder fixed.

The probe input follows Eq. 6: for the graph-level SEAN-T tasks it concatenates the robot latent with the mean of the human latents, giving one $2d_z = 256$ dimensional vector per scenario, and for the node-level SNS task it concatenates the robot latent with that pedestrian’s own latent, giving one 256-dimensional vector per labeled pedestrian. The three independent binary SEAN-T heads therefore hold $3 \times (256 + 1) = 771$ parameters and the single five-way SNS head $5 \times (256 + 1) = 1285$, matching Table 5. The SNS classifier retains all five action categories, but we report macro F_1 over *avoid*, *follow*, and *not consider*. We exclude *overtake* and *yield* from the aggregate metric because their limited labeled support does not permit stable per-class evaluation.

C.3 Hyperparameter Search

We tuned hyperparameters only for the unified SEAN+SNS models. Hyperparameter search was performed with Optuna using a TPE sampler and median pruning. Each trial first trained a unified GNN-VAE, then froze the encoder and trained an SNS linear probe. The Optuna objective was SNS probe validation loss using cross-entropy.

For the homogeneous graph model, the search selected a batch size of 32, dropout of 0.1, weight decay of 10^{-4} , adaptive maximum mixing coefficient of 0.75, $\beta_{\max} = 0.1$, hidden dimension 256, latent dimension 128, three Transformer layers, and learning rate 10^{-3} .

For the heterogeneous graph model, we ran three independent Optuna searches with $\beta_{\max}$ fixed at 0.1, 0.5, and 0.9, optimizing the remaining hyperparameters within each search. The $\beta_{\max} = 0.9$ search yielded the best downstream validation performance. The selected heterogeneous configuration used hidden dimension 256, latent dimension 128 per node, batch size 32, two HEAT-style message-passing layers, and a temporal edge Transformer with depth 3 and 4 attention heads. Training used a VAE learning rate of 1.156×10^{-4} , KL warmup of 25 epochs, dropout 0.2, adaptive sampling warmup of 20 epochs, adaptive ramp-up of 100 epochs, and maximum mixing coefficient 0.75.

All final unified models used AdamW optimization, VAE weight decay 10^{-4} , probe weight decay 10^{-4} , gradient clipping at 20.0, VAE early stopping patience 20, and probe early stopping patience 20. The unified search used SEAN graphs from `combined_v1.h5`, SNS graphs from `social_nav_v3.h5`, and the fixed scene split `split_indices_final.pt`. SNS node features were dropped, so SNS encodings used learned node-type embeddings and edge geometry.

For the SEAN-only baseline, we did not run a separate Optuna search. Instead, the SEAN-only model reused the selected unified hyperparameters wherever applicable, including hidden dimension, latent dimension, learning rate, KL weight, KL warmup, dropout, batch size, weight decay, and gradient clipping. Unified-only adaptive SNS sampling parameters were omitted for SEAN-only training.

C.4 Sampling and Class Imbalance

We used stratified and adaptive sampling during unified GNN training for both heterogeneous and homogeneous models to address the large SNS class imbalance, where approximately 82% of training examples correspond to the *not consider* class. Although the VAE pretraining objective itself is unsupervised, the adaptive sampler uses downstream SNS label statistics to improve the class balance of batches used during unified training. Empirically, this sampling strategy improved downstream F_1 scores for both SEAN and SNS tasks, with particularly noticeable gains for the homogeneous graph model.

D Compute and Reproducibility Details

To support reproducibility and clarify the computational requirements of STARS, we report dataset sizes, graph construction details, model capacity, training runtimes, inference runtimes, and hardware details for all experiments.

D.1 Hardware and Software

Hardware. All experiments were run primarily on the Lonestar6 and Stampede3 systems at the Texas Advanced Computing Center (TACC). Unless otherwise stated, models were trained on a single NVIDIA A100 PCIe GPU with 40 GB of GPU memory. Experiments were executed in an Ubuntu 22.04 containerized environment using Python 3.10, PyTorch 2.0.1, and CUDA-enabled GPU execution.

D.2 Model Capacity

Table 5 reports the number of trainable parameters for STARS. For STARS, we report the capacity of the pretrained encoder-decoder separately from the lightweight downstream prediction head.

Table 5: Model capacity comparison. For STARS, parameter counts are separated into the pretrained representation model and the downstream task-specific prediction head.

Model / Component	Stage	Trainable Params.	Frozen Params.	Notes
STARS-He Encoder/Decoder	Pretraining	6,420,810	N/A	Heterogeneous graph autoencoder
STARS-Ho Encoder/Decoder	Pretraining	2,936,848	N/A	Homogeneous graph autoencoder
Linear Prediction Head	Downstream adaptation	1,285 (SNS), 771 (SEAN)	6,420,810	Used with frozen STARS encoder

D.3 Runtime and Memory

Training and inference runtime. Table 6 reports the computational cost of each experimental pipeline. Pretraining time refers to the wall-clock time required to train the graph autoencoder. Downstream training time refers to fitting the task-specific prediction head on top of the frozen encoder. Inference runtime is reported per interaction scenario and includes representation extraction and prediction when applicable. Peak GPU memory is measured during training unless otherwise stated.

Table 6: Runtime and memory usage for each experimental pipeline. For STARS-based pipelines, pretraining is performed once and the frozen encoder is reused for downstream tasks. Downstream training time refers to fitting the task-specific prediction head. Inference runtime is reported per interaction scenario.

Model / Pipeline	Dataset / Task	Pretraining Time	Downstream Training Time	Inference Time / Scenario	Peak GPU Memory
STARS-He + Linear Head	SEAN-T + SNS	~2.5 hrs / run	<1 sec	~12 ms	~13 GB
STARS-Ho + Linear Head	SEAN-T + SNS	~3 hrs / run	~12 sec	~30 ms	~17 GB

Table 7: Extended STARS results on SEAN-T and SNS across labeled-data fractions.

Task	Method	1%	5%	10%	25%	50%	100%
Competence	STARS-He (Ours)	0.635 ± 0.10	0.673 ± 0.07	0.726 ± 0.04	0.762 ± 0.03	0.781 ± 0.03	0.794 ± 0.02
	STARS-Ho (Ours)	0.612 ± 0.14	0.702 ± 0.08	0.738 ± 0.05	0.765 ± 0.04	0.787 ± 0.03	0.799 ± 0.02
Surprise	STARS-He (Ours)	0.455 ± 0.17	0.481 ± 0.17	0.526 ± 0.15	0.603 ± 0.12	0.655 ± 0.08	0.709 ± 0.04
	STARS-Ho (Ours)	0.440 ± 0.23	0.541 ± 0.21	0.548 ± 0.20	0.610 ± 0.16	0.713 ± 0.06	0.741 ± 0.03
Intention	STARS-He (Ours)	0.576 ± 0.08	0.619 ± 0.05	0.671 ± 0.04	0.694 ± 0.03	0.705 ± 0.03	0.721 ± 0.03
	STARS-Ho (Ours)	0.603 ± 0.10	0.671 ± 0.07	0.701 ± 0.06	0.715 ± 0.04	0.723 ± 0.03	0.734 ± 0.02
Avoid	STARS-He (Ours)	0.098 ± 0.09	0.173 ± 0.11	0.190 ± 0.10	0.219 ± 0.10	0.230 ± 0.09	0.242 ± 0.08
	STARS-Ho (Ours)	0.126 ± 0.09	0.234 ± 0.06	0.252 ± 0.06	0.259 ± 0.06	0.257 ± 0.05	0.267 ± 0.05
Follow	STARS-He (Ours)	0.155 ± 0.15	0.327 ± 0.12	0.374 ± 0.08	0.404 ± 0.04	0.412 ± 0.03	0.405 ± 0.03
	STARS-Ho (Ours)	0.132 ± 0.13	0.289 ± 0.12	0.343 ± 0.07	0.374 ± 0.04	0.385 ± 0.04	0.396 ± 0.02
Not Consider	STARS-He (Ours)	0.834 ± 0.06	0.840 ± 0.04	0.846 ± 0.03	0.855 ± 0.02	0.861 ± 0.02	0.862 ± 0.02
	STARS-Ho (Ours)	0.773 ± 0.11	0.768 ± 0.05	0.765 ± 0.04	0.771 ± 0.03	0.770 ± 0.04	0.763 ± 0.04
Macro F_1	STARS-He (Ours)	0.362 ± 0.06	0.447 ± 0.05	0.470 ± 0.04	0.493 ± 0.03	0.501 ± 0.03	0.503 ± 0.02
	STARS-Ho (Ours)	0.344 ± 0.05	0.431 ± 0.05	0.453 ± 0.04	0.468 ± 0.03	0.471 ± 0.02	0.475 ± 0.02

Table 8: SEAN-T downstream performance after SEAN-T-only pretraining. Results report F_1 -Score ($\mu \pm \sigma$) across labeled-data fractions.

Task	Model	1%	5%	10%	25%	50%	100%
Competence	STARS-He	0.568 ± 0.10	0.636 ± 0.07	0.686 ± 0.06	0.733 ± 0.03	0.773 ± 0.03	0.792 ± 0.02
Surprise	STARS-He	0.455 ± 0.14	0.453 ± 0.16	0.480 ± 0.22	0.55 ± 0.15	0.653 ± 0.09	0.716 ± 0.05
Intention	STARS-He	0.528 ± 0.09	0.601 ± 0.05	0.644 ± 0.05	0.683 ± 0.03	0.705 ± 0.03	0.721 ± 0.02

E Additional Results and Ablations

E.1 Extended Main Results

Table 7 provides the full downstream results for STARS across all supervised data fractions. We report SEAN-T performance for the three subjective social-impression tasks and SNS performance using per-class F_1 scores together with macro- F_1 . Across both datasets, performance generally improves as the amount of labeled data available for supervised linear adaptation increases. The heterogeneous and homogeneous variants show similar trends on SEAN-T, while the heterogeneous model achieves higher SNS macro- F_1 at larger label fractions.

E.2 Pretraining Data Ablations

SEAN-only Pretraining. We compare the jointly pretrained SEAN-T+SNS model against a SEAN-T only model to assess the effect of adding cross-dataset interaction data on SEAN-T downstream performance. The SEAN-T only model remains competitive with joint pretraining, particularly at larger labeled-data fractions. At 100% labeled data, it achieves 0.792/0.716/0.721 on competence, surprise, and intention, compared with 0.794/0.709/0.721 under joint pretraining. These results indicate that adding SNS data does not substantially alter SEAN-T downstream performance.

E.3 Cross-Dataset Transfer

To evaluate whether STARS learns representations that transfer across datasets, we pretrain on one dataset and evaluate the frozen encoder on the other using the same linear-probing protocol as the main experiments. We compare single-dataset pretraining against joint SEAN-T+SNS pretraining.

At 100% labeled data, STARS-He achieves performance comparable to joint pretraining. SEAN-T-only pretraining achieves 0.497 macro F_1 on SNS compared with 0.503 under joint pretraining, while SNS-only pretraining achieves 0.801/0.710/0.729 on SEAN-T competence, surprise, and intention, compared with 0.794/0.709/0.721 under joint pretraining. STARS-Ho also transfers across datasets, although joint pretraining provides larger gains.

Table 9: Cross-dataset transfer to SNS. Models are pretrained either on SEAN-T only or jointly on SEAN-T and SNS and evaluated on SNS using progressively larger fractions of labeled training data. Results report macro F_1 -Score ($\mu \pm \sigma$).

Model	Pretraining	1%	5%	10%	25%	50%	100%
STARS-He	SEAN-T only	0.355 $\pm$ 0.06	0.442 $\pm$ 0.05	0.463 $\pm$ 0.04	0.489 $\pm$ 0.03	0.493 $\pm$ 0.03	0.497 $\pm$ 0.03
STARS-He	SEAN-T+SNS	0.362 $\pm$ 0.06	0.447 $\pm$ 0.05	0.470 $\pm$ 0.04	0.493 $\pm$ 0.03	0.501 $\pm$ 0.03	0.503 $\pm$ 0.02
STARS-Ho	SEAN-T only	0.274 $\pm$ 0.05	0.332 $\pm$ 0.04	0.351 $\pm$ 0.03	0.365 $\pm$ 0.03	0.407 $\pm$ 0.02	0.419 $\pm$ 0.01
STARS-Ho	SEAN-T+SNS	0.344 $\pm$ 0.05	0.431 $\pm$ 0.05	0.453 $\pm$ 0.04	0.468 $\pm$ 0.03	0.471 $\pm$ 0.02	0.475 $\pm$ 0.02

Table 10: Cross-dataset transfer to SEAN-T. Models are pretrained either on SNS only or jointly on SEAN-T and SNS and evaluated on the three SEAN-T perception tasks using progressively larger fractions of labeled training data. Results report F_1 -Score ($\mu \pm \sigma$).

Model	Pretraining	Task	1%	5%	10%	25%	50%	100%
STARS-He	SNS only	Competence	0.615 $\pm$ 0.10	0.674 $\pm$ 0.08	0.731 $\pm$ 0.05	0.772 $\pm$ 0.03	0.786 $\pm$ 0.03	0.801 $\pm$ 0.01
STARS-He	SNS only	Surprise	0.409 $\pm$ 0.20	0.446 $\pm$ 0.18	0.493 $\pm$ 0.17	0.595 $\pm$ 0.11	0.652 $\pm$ 0.08	0.710 $\pm$ 0.05
STARS-He	SNS only	Intention	0.571 $\pm$ 0.09	0.619 $\pm$ 0.08	0.668 $\pm$ 0.04	0.700 $\pm$ 0.03	0.717 $\pm$ 0.03	0.729 $\pm$ 0.03
STARS-He	SEAN-T+SNS	Competence	0.635 $\pm$ 0.10	0.673 $\pm$ 0.07	0.726 $\pm$ 0.04	0.762 $\pm$ 0.03	0.781 $\pm$ 0.03	0.794 $\pm$ 0.02
STARS-He	SEAN-T+SNS	Surprise	0.455 $\pm$ 0.17	0.481 $\pm$ 0.17	0.526 $\pm$ 0.15	0.603 $\pm$ 0.12	0.655 $\pm$ 0.08	0.709 $\pm$ 0.04
STARS-He	SEAN-T+SNS	Intention	0.576 $\pm$ 0.08	0.619 $\pm$ 0.05	0.671 $\pm$ 0.04	0.694 $\pm$ 0.03	0.705 $\pm$ 0.03	0.721 $\pm$ 0.03
STARS-Ho	SNS only	Competence	0.440 $\pm$ 0.27	0.507 $\pm$ 0.28	0.600 $\pm$ 0.24	0.735 $\pm$ 0.02	0.744 $\pm$ 0.03	0.766 $\pm$ 0.01
STARS-Ho	SNS only	Surprise	0.206 $\pm$ 0.28	0.285 $\pm$ 0.25	0.309 $\pm$ 0.26	0.481 $\pm$ 0.23	0.570 $\pm$ 0.14	0.655 $\pm$ 0.04
STARS-Ho	SNS only	Intention	0.493 $\pm$ 0.18	0.515 $\pm$ 0.19	0.573 $\pm$ 0.11	0.647 $\pm$ 0.02	0.655 $\pm$ 0.02	0.676 $\pm$ 0.02
STARS-Ho	SEAN-T+SNS	Competence	0.612 $\pm$ 0.14	0.702 $\pm$ 0.08	0.738 $\pm$ 0.04	0.765 $\pm$ 0.04	0.787 $\pm$ 0.03	0.799 $\pm$ 0.02
STARS-Ho	SEAN-T+SNS	Surprise	0.440 $\pm$ 0.23	0.541 $\pm$ 0.21	0.548 $\pm$ 0.20	0.610 $\pm$ 0.16	0.713 $\pm$ 0.06	0.741 $\pm$ 0.03
STARS-Ho	SEAN-T+SNS	Intention	0.603 $\pm$ 0.10	0.671 $\pm$ 0.07	0.701 $\pm$ 0.06	0.715 $\pm$ 0.04	0.723 $\pm$ 0.03	0.734 $\pm$ 0.02

E.4 Pretraining Objective Ablations

Random Encoder. The random encoder baseline uses the same heterogeneous GNN-VAE architecture as the full model but removes self-supervised pretraining. This isolates whether downstream performance comes from the learned pretrained representation or simply from the architecture and linear probe. The random encoder performs noticeably worse than the full model on most of the F_1 -scores, indicating that the self-supervised graph reconstruction objective learns task-relevant latent structure.

E.4.1 Self-Supervised vs. Supervised Heterogeneous Graph

To separate the value of self-supervised pretraining from the heterogeneous graph architecture itself, we compare the frozen STARS encoder with an otherwise similar heterogeneous model trained end-to-end using supervised SNS labels. As shown in Table 11, supervised training improves some minority-class scores such as *avoid*, but the self-supervised representation achieves stronger macro- F_1 across all label fractions. This suggests that the reconstruction-based pretraining objective provides a more stable representation for low-data downstream adaptation under the SNS class imbalance.

E.5 Representation Component Ablations

Robot-Latent Only. The robot-latent only ablation uses only the learned robot latent representation for downstream prediction, omitting human-node latents from the downstream readout. This tests how much task-relevant information is summarized directly in the robot representation. The full representation performs substantially better on SNS minority action classes and on most SEAN-T perception tasks, although the robot-latent only representation remains competitive on SEAN-T intention.

Bystander-Latent Only. The bystander-latent only ablation uses only the mean pooled bystander latent representation, omitting the robot latent from the downstream readout. This tests how much predictive information can be recovered from surrounding pedestrian context alone. The bystander

Table 11: SNS Action Classification: Self-Supervised vs. Supervised GNN-VAE. We compare STARS’ frozen self-supervised encoder with a linear probe against an identical architecture trained end-to-end with supervised classification. Both models use the heterogeneous graph formulation. Results report F1-Score ($\mu \pm \sigma$) across label fractions.

Task	Method	1%	5%	10%	25%	50%	100%
Avoid	Hetero Self-Sup	0.098 $\pm$ 0.09	0.173 $\pm$ 0.11	0.190 $\pm$ 0.10	0.219 $\pm$ 0.10	0.230 $\pm$ 0.09	0.242 $\pm$ 0.08
	Hetero Sup	0.178 $\pm$ 0.07	0.234 $\pm$ 0.05	0.258 $\pm$ 0.04	0.292 $\pm$ 0.03	0.297 $\pm$ 0.02	0.284 $\pm$ 0.02
Follow	Hetero Self-Sup	0.155 $\pm$ 0.15	0.327 $\pm$ 0.12	0.374 $\pm$ 0.08	0.404 $\pm$ 0.04	0.412 $\pm$ 0.03	0.405 $\pm$ 0.03
	Hetero Sup	0.112 $\pm$ 0.11	0.287 $\pm$ 0.07	0.305 $\pm$ 0.06	0.353 $\pm$ 0.04	0.373 $\pm$ 0.04	0.384 $\pm$ 0.04
Not Consider	Hetero Self-Sup	0.834 $\pm$ 0.06	0.840 $\pm$ 0.04	0.846 $\pm$ 0.03	0.855 $\pm$ 0.02	0.861 $\pm$ 0.02	0.862 $\pm$ 0.02
	Hetero Sup	0.218 $\pm$ 0.25	0.351 $\pm$ 0.22	0.429 $\pm$ 0.23	0.604 $\pm$ 0.13	0.662 $\pm$ 0.08	0.667 $\pm$ 0.07
Macro F_1	Hetero Self-Sup	0.362 $\pm$ 0.06	0.447 $\pm$ 0.05	0.470 $\pm$ 0.04	0.493 $\pm$ 0.03	0.501 $\pm$ 0.03	0.503 $\pm$ 0.02
	Hetero Sup	0.169 $\pm$ 0.09	0.291 $\pm$ 0.10	0.331 $\pm$ 0.10	0.417 $\pm$ 0.07	0.444 $\pm$ 0.04	0.445 $\pm$ 0.04

Table 12: Random encoder ablation results.

		Architecture	1%	5%	10%	25%	50%	100%
Random encoder:	Competence	Heterogeneous	0.530 $\pm$ 0.22	0.596 $\pm$ 0.16	0.635 $\pm$ 0.12	0.677 $\pm$ 0.09	0.691 $\pm$ 0.08	0.707 $\pm$ 0.07
	Surprise	Heterogeneous	0.251 $\pm$ 0.20	0.296 $\pm$ 0.21	0.372 $\pm$ 0.17	0.424 $\pm$ 0.13	0.467 $\pm$ 0.12	0.480 $\pm$ 0.14
	Intention	Heterogeneous	0.606 $\pm$ 0.20	0.648 $\pm$ 0.14	0.677 $\pm$ 0.10	0.700 $\pm$ 0.08	0.704 $\pm$ 0.08	0.713 $\pm$ 0.09
	F_1 Avoid	Heterogeneous	0.070 $\pm$ 0.11	0.251 $\pm$ 0.08	0.301 $\pm$ 0.03	0.304 $\pm$ 0.02	0.305 $\pm$ 0.02	0.302 $\pm$ 0.02
	F_1 Follow	Heterogeneous	0.070 $\pm$ 0.12	0.065 $\pm$ 0.11	0.101 $\pm$ 0.12	0.101 $\pm$ 0.11	0.082 $\pm$ 0.10	0.095 $\pm$ 0.10
	F_1 Not Consider	Heterogeneous	0.865 $\pm$ 0.03	0.849 $\pm$ 0.02	0.833 $\pm$ 0.02	0.839 $\pm$ 0.01	0.839 $\pm$ 0.01	0.839 $\pm$ 0.00
	PA	Heterogeneous	0.601 $\pm$ 0.03	0.590 $\pm$ 0.02	0.577 $\pm$ 0.02	0.583 $\pm$ 0.01	0.584 $\pm$ 0.01	0.583 $\pm$ 0.00

representation retains task relevant signal, but the full representation performs better overall, indicating that combining robot and pedestrian representations is more informative for downstream tasks.

Raw Robot Node Only. The raw robot-node baseline bypasses graph pretraining entirely and trains directly on the 592-dimensional SEAN robot feature vector. This tests whether the SEAN labels can be predicted from robot-centric observations alone without relational message passing. The raw robot-node baseline contains substantial predictive signal, particularly for competence and surprise. At 100% labeled data, it achieves 0.746/0.700/0.688 on competence, surprise, and intention, compared with 0.794/0.709/0.721 for the full heterogeneous STARS representation. Compared with the robot-latent-only ablation, the raw features perform better on competence and surprise but worse on intention, indicating that the different representations retain complementary task-relevant information.

E.6 Baseline and Lower-Bound Analyses

Weighted Random Baseline. The weighted random baseline samples predictions according to the empirical training-label distribution, providing a label-aware lower bound under class imbalance. Its strong performance on majority labels such as *not consider* highlights how misleading accuracy or single-class F1 can be on imbalanced SNS labels. However, overall, our full model substantially outperforms this baseline in nearly every F1 metric, showing that it learns beyond label-frequency priors.

E.7 SNS F_1 -Score Analysis

We evaluate the SocialNav-SUB vision-language-model (VLM) baselines on the same held-out SNS test split used for our downstream action classification task. Rather than independently rerunning the prompted VLM baselines, we obtained their per-example prediction outputs from the SocialNav-SUB authors and computed metrics on the corresponding examples in our evaluation split. For each scenario, the model predicts a discrete relational action label for the target pedestrian. We compute per-class F_1 -scores and Probability of Agreement (PA) following the SocialNav-SUB evaluation protocol [6]. Because these models are evaluated zero-shot through prompting and are not fine-

Table 13: Robot-latent only (128D) ablation results.

		Architecture	1%	5%	10%	25%	50%	100%
128D robot-z only	Competence	Heterogeneous	0.592 $\pm$ 0.13	0.627 $\pm$ 0.10	0.663 $\pm$ 0.08	0.695 $\pm$ 0.07	0.707 $\pm$ 0.07	0.711 $\pm$ 0.08
	Surprise	Heterogeneous	0.373 $\pm$ 0.12	0.400 $\pm$ 0.12	0.410 $\pm$ 0.11	0.432 $\pm$ 0.12	0.454 $\pm$ 0.12	0.479 $\pm$ 0.14
	Intention	Heterogeneous	0.623 $\pm$ 0.12	0.662 $\pm$ 0.08	0.696 $\pm$ 0.08	0.716 $\pm$ 0.07	0.730 $\pm$ 0.07	0.730 $\pm$ 0.07
	F_1 Avoid	Heterogeneous	0.017 $\pm$ 0.05	0.004 $\pm$ 0.02	0.001 $\pm$ 0.01	0.000 $\pm$ 0.00	0.000 $\pm$ 0.00	0.000 $\pm$ 0.00
	F_1 Follow	Heterogeneous	0.004 $\pm$ 0.02	0.008 $\pm$ 0.03	0.007 $\pm$ 0.02	0.003 $\pm$ 0.01	0.000 $\pm$ 0.00	0.000 $\pm$ 0.00
	F_1 Not Consider	Heterogeneous	0.875 $\pm$ 0.02	0.881 $\pm$ 0.01	0.883 $\pm$ 0.00	0.883 $\pm$ 0.00	0.883 $\pm$ 0.00	0.883 $\pm$ 0.00
	PA	Heterogeneous	0.610 $\pm$ 0.02	0.616 $\pm$ 0.01	0.618 $\pm$ 0.00	0.618 $\pm$ 0.00	0.618 $\pm$ 0.00	0.618 $\pm$ 0.00

Table 14: 128D byst only ablation results.

		Architecture	1%	5%	10%	25%	50%	100%
128D byst only	Competence	Heterogeneous	0.551 $\pm$ 0.17	0.533 $\pm$ 0.15	0.556 $\pm$ 0.15	0.621 $\pm$ 0.10	0.642 $\pm$ 0.09	0.650 $\pm$ 0.07
	Surprise	Heterogeneous	0.323 $\pm$ 0.15	0.337 $\pm$ 0.13	0.302 $\pm$ 0.13	0.232 $\pm$ 0.14	0.161 $\pm$ 0.15	0.149 $\pm$ 0.14
	Intention	Heterogeneous	0.574 $\pm$ 0.17	0.598 $\pm$ 0.15	0.634 $\pm$ 0.13	0.693 $\pm$ 0.09	0.713 $\pm$ 0.09	0.717 $\pm$ 0.09
	F_1 Avoid	Heterogeneous	0.134 $\pm$ 0.10	0.154 $\pm$ 0.08	0.167 $\pm$ 0.08	0.149 $\pm$ 0.09	0.148 $\pm$ 0.08	0.153 $\pm$ 0.09
	F_1 Follow	Heterogeneous	0.223 $\pm$ 0.12	0.317 $\pm$ 0.06	0.341 $\pm$ 0.04	0.344 $\pm$ 0.03	0.344 $\pm$ 0.03	0.343 $\pm$ 0.03
	F_1 Not Consider	Heterogeneous	0.661 $\pm$ 0.08	0.756 $\pm$ 0.06	0.786 $\pm$ 0.04	0.807 $\pm$ 0.03	0.807 $\pm$ 0.03	0.811 $\pm$ 0.02
	PA	Heterogeneous	0.419 $\pm$ 0.06	0.498 $\pm$ 0.05	0.524 $\pm$ 0.03	0.539 $\pm$ 0.03	0.539 $\pm$ 0.02	0.542 $\pm$ 0.02

tuned for this task, we report results only on the full test set rather than across labeled-data fractions. Table 17 summarizes the resulting per-class F_1 and PA scores.

Table 15: 592D robot node only (SEAN-ablation compared with combined model).

	Architecture	1%	5%	10%	25%	50%	100%
Competence	Heterogeneous	0.541 $\pm$ 0.11	0.641 $\pm$ 0.07	0.673 $\pm$ 0.07	0.714 $\pm$ 0.05	0.724 $\pm$ 0.04	0.746 $\pm$ 0.03
Surprise	Heterogeneous	0.528 $\pm$ 0.10	0.591 $\pm$ 0.09	0.610 $\pm$ 0.08	0.652 $\pm$ 0.05	0.682 $\pm$ 0.04	0.700 $\pm$ 0.04
Intention	Heterogeneous	0.498 $\pm$ 0.09	0.580 $\pm$ 0.06	0.590 $\pm$ 0.05	0.632 $\pm$ 0.05	0.649 $\pm$ 0.05	0.688 $\pm$ 0.05

Table 16: WRB ablation results.

	Architecture	1%	5%	10%	25%	50%	100%
WRB:	Competence	Heterogeneous	0.524 $\pm$ 0.14	0.528 $\pm$ 0.12	0.517 $\pm$ 0.10	0.517 $\pm$ 0.10	0.530 $\pm$ 0.11
	Surprise	Heterogeneous	0.345 $\pm$ 0.13	0.366 $\pm$ 0.12	0.384 $\pm$ 0.10	0.360 $\pm$ 0.08	0.368 $\pm$ 0.08
	Intention	Heterogeneous	0.563 $\pm$ 0.11	0.574 $\pm$ 0.09	0.575 $\pm$ 0.09	0.578 $\pm$ 0.08	0.575 $\pm$ 0.08
	F_1 Avoid	Heterogeneous	0.090 $\pm$ 0.07	0.104 $\pm$ 0.05	0.104 $\pm$ 0.04	0.108 $\pm$ 0.04	0.105 $\pm$ 0.05
	F_1 Follow	Heterogeneous	0.046 $\pm$ 0.06	0.062 $\pm$ 0.05	0.069 $\pm$ 0.05	0.083 $\pm$ 0.04	0.070 $\pm$ 0.05
	F_1 Not Consider	Heterogeneous	0.812 $\pm$ 0.05	0.806 $\pm$ 0.03	0.800 $\pm$ 0.02	0.811 $\pm$ 0.02	0.813 $\pm$ 0.02
	PA	Heterogeneous	0.542 $\pm$ 0.05	0.532 $\pm$ 0.03	0.527 $\pm$ 0.02	0.536 $\pm$ 0.02	0.539 $\pm$ 0.01

Table 17: SocialNav-SUB VLM baseline results on the held-out SNS action classification task [6]. Models are evaluated on the full downstream test set. Metrics report per-class F_1 -scores alongside Probability of Agreement (PA).

Model	F_1 Avoid	F_1 Follow	F_1 Overtake	F_1 Yield	F_1 NC	PA
Gemini	0.223	0.000	0.000	0.222	0.745	0.493
4o	0.241	0.176	0.000	0.000	0.521	0.344
o4-mini	0.222	0.363	0.000	0.111	0.852	0.582