

MANEUVER- AND DELAY-ROBUST DISTRIBUTED SATELLITE SSA VIA SPATIO-TEMPORAL GRAPH FORECASTING

Hamada Rizk[1,2], Hirozumi Yamaguchi[1,2], Sumio Morioka[3] & Takahiro Inagawa[3]

[1]The University of Osaka, Osaka, Japan
[2]RIKEN R-CCS, Kobe, Japan
[3]Interstellar Technologies Inc., Hokkaido, Japan

Abstract

The rapid growth of LEO constellations is turning Space Situational Awareness (SSA) into a real-time bottleneck: conjunction screening must be refreshed frequently, yet catalog updates arrive with latency and gaps, and maneuvers can abruptly invalidate propagator-only predictions, destabilizing closest-approach estimates and risk ranking. We present a distributed SSA framework designed to remain reliable under delayed and incomplete information by combining interaction-aware trajectory forecasting with uncertainty-calibrated triage. Inputs are time-aligned inertial state sequences derived from TLE/SGP4 propagation, and outputs are multi-step forecasts of state means together with calibrated uncertainty. We model SSA as a dynamic spatio-temporal interaction graph whose edges encode relative motion and risk-relevant neighborhoods informed by CPA/TCA. A spatio-temporal graph forecaster predicts short-horizon trajectories, and candidate pairs are ranked using an uncertainty-aware score to produce a stable top-K triage list. Distributed execution exchanges only compact boundary summaries (forecast states, uncertainty, and timestamps) rather than raw catalogs, and robustness is evaluated under message delay and loss. In a 90-day catalog-driven test with 4,500 RSOs, our method reduces 60-min position RMSE by 33% (2.84→1.91 km) and improves triage quality by 27% in Recall@50 (0.62→0.79), 27% in NDCG@50 (0.55→0.70), and 39% in PR-AUC (0.41→0.57). Under distributed delay/loss ($\tau = 3$, $p = 0.3$), Recall@50 remains 0.73 with 7.6 kB/update/node.

Keywords: space situational awareness; conjunction screening; spatio-temporal graphs; uncertainty calibration; distributed inference

1. Introduction

Low Earth orbit (LEO) is becoming increasingly congested [1]. Large satellite constellations are growing rapidly [2], and debris objects continue to accumulate. As a result, Space Situational Awareness (SSA) systems must screen a very large number of resident space objects (RSOs) and repeatedly identify which object pairs may come dangerously close. This task is becoming a real-time operational bottleneck: the number of possible pairs grows quickly with catalog size, while operators need timely and stable conjunction warnings.

Most large-scale public screening pipelines rely on Two-Line Elements (TLEs) and fast analytical propagation, especially SGP4 [3]. This approach is useful because it is computationally efficient and can be applied to large catalogs. However, catalog-based SSA also has important limitations. TLEs are not perfect measurements of the true orbit. They may be delayed, sparse, inconsistent, or affected by unmodeled maneuvers. When this happens, a propagated trajectory may drift away from the true motion. Even a moderate prediction error can change the estimated closest approach between two objects, which may then change the risk ranking used by operators.

This problem becomes harder in distributed SSA. In practice, space objects may be observed, processed, and managed by different sensors, ground stations, operators, or organizations. Data updates are asynchronous, and not every node has immediate access to the same information. Some

Table 1 – Position of the proposed framework relative to common SSA and learning-based screening approaches.

Method family	**Multi-step**	**Interaction**	**Uncertainty**	**Delay/loss**	**Top-K**
SGP4 [3]	Yes	No	Limited	No	Yes
dSGP4 [4]	Yes	No	Limited	No	Yes
LSTM/Transformer [5, 6]	Yes	No	Usually no	No	Indirect
AI screening [7]	Limited	Pairwise	Usually no	No	Yes
Proposed framework	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

operators may also be unable or unwilling to share raw catalogs because of bandwidth, policy, privacy, or operational constraints. Therefore, a realistic SSA system must work even when information is incomplete, delayed, or only partially shared. It is not enough to assume that all objects and all updates are available in one centralized database at every decision time.

The key difficulty addressed in this paper is therefore not only orbit prediction accuracy. Lower position error is useful, but the operational question is more specific: can the system keep the truly important object pairs near the top of the triage list when the input information is stale, missing, or delayed? In conjunction monitoring, operators usually inspect only a limited number of candidate pairs. A method that slightly improves average trajectory error but drops a dangerous pair from the top-K list is not reliable for decision support. For this reason, we focus on uncertainty-aware top-K conjunction triage, not only on trajectory forecasting.

Existing methods address different parts of this problem, but not the complete setting. SGP4 and related propagators provide fast multi-step trajectory estimates, but they mainly propagate each object independently and do not explicitly model risk-relevant interactions between objects. Differentiable SGP4 and learning-based correction methods improve propagation or residual modeling, but they still mainly focus on individual-object prediction. LSTM and Transformer-based orbit predictors can learn temporal patterns from historical states, but they are often applied independently to each object and do not directly model evolving local conjunction neighborhoods. Learning-based conjunction screening methods can classify or filter candidate pairs, but they usually focus on pairwise screening efficiency rather than jointly forecasting trajectories, estimating calibrated uncertainty, and handling distributed delay and packet loss.

Table 1 summarizes this gap. The proposed method is designed to combine the missing capabilities: multi-step forecasting, interaction-aware modeling, predictive uncertainty, distributed robustness, and top-K triage.

To address this gap, we propose a distributed SSA framework based on spatio-temporal graph forecasting and uncertainty-aware conjunction triage. The main idea is to treat SSA as an interaction-aware forecasting problem. Each RSO is represented as a graph node, and edges connect objects that are relevant for short-horizon conjunction screening. These edges are not arbitrary. They are constructed using predicted relative motion and closest-approach information, so the graph focuses computation on pairs that are likely to matter. A spatio-temporal graph forecaster then predicts future states and calibrated uncertainty. Finally, candidate pairs are ranked using both predicted closest approach and uncertainty, so pairs with small predicted separation or high uncertainty are kept visible in the top-K triage list.

The framework is also designed for distributed execution. Instead of requiring all nodes to share raw TLE histories or full catalogs, each node exchanges compact boundary summaries for objects that may interact across partitions. A boundary summary contains only the predicted future state, uncertainty, timestamp, and TLE age. This design reduces communication cost while still allowing cross-node conjunction reasoning. The method is evaluated under explicit message delay and packet loss to test whether the ranking remains reliable when distributed information is imperfect.

The main contributions of this paper are as follows:

- We formulate distributed SSA as a dynamic spatio-temporal graph forecasting problem, where RSOs are modeled as nodes and risk-aware interaction neighborhoods are constructed using CPA/TCA-informed edges.
- We propose a multi-step graph forecasting model that predicts both future trajectory means and calibrated predictive uncertainty, allowing the system to express when forecasts should be trusted less.
- We introduce an uncertainty-aware top-K conjunction triage method that ranks candidate pairs using both predicted closest approach and forecast uncertainty, improving robustness under stale or incomplete information.
- We design a distributed inference protocol in which compute nodes exchange compact boundary summaries instead of raw catalogs, enabling communication-efficient cross-partition risk reasoning.
- We conduct a catalog-driven evaluation using public TLE data, including explicit tests under message delay, packet loss, maneuver-like perturbation, and communication-cost constraints.

Overall, this paper argues that future SSA systems should not be evaluated only by average orbit-prediction error. For operational conjunction monitoring, the more important question is whether the system can maintain a stable and trustworthy ranking of risky object pairs under imperfect information. The proposed framework addresses this need by jointly modeling object interactions, predictive uncertainty, and distributed communication constraints.

2. Related Work

2.1 Catalog-Based Orbit Propagation and Conjunction Assessment

Catalog-driven Space Situational Awareness (SSA) commonly relies on Two-Line Element (TLE) sets and fast analytical propagation. The most widely used propagation model for public catalog products is SGP4, whose implementation and assumptions were clarified and standardized by Vallado et al. [3]. SGP4 is attractive because it is fast, scalable, and suitable for screening large catalogs. However, its prediction quality depends on the freshness and consistency of the TLEs. When catalog updates are delayed, sparse, or affected by unmodeled maneuvers, propagated states may deviate from the true object motion, which can directly affect closest-approach estimates and risk ranking.

Classical conjunction assessment methods usually evaluate the relative motion between two objects near the time of closest approach and may use miss distance, covariance information, and probability of collision. Akella and Alfriend [8] presented a widely used formulation for estimating collision probability between space objects, while Patera [9] proposed a general numerical method for computing satellite collision probability from state vectors and covariance matrices. These methods provide important foundations for high-fidelity conjunction assessment, but they typically assume that candidate pairs and propagated states have already been generated. Therefore, large-scale SSA still requires efficient short-horizon forecasting and triage mechanisms that can identify which pairs deserve detailed analysis.

The proposed work does not replace classical conjunction-assessment methods. Instead, it addresses the preceding decision-support problem: how to forecast short-horizon motion and maintain a stable top-K list of risk-relevant pairs when the available catalog information is stale, incomplete, or distributed across multiple compute nodes.

2.2 Learning-Based Orbit Prediction

Recent studies have explored machine learning as a way to compensate for the limitations of purely analytical or dynamical propagation. Acciarini et al. [4] introduced dSGP4, a differentiable implementation of SGP4 that enables gradient-based optimization, covariance-related applications, and integration with machine learning models. This line of work is important because it preserves the computational efficiency of SGP4 while allowing learning-based correction.

Sequence models have also been investigated for short-term orbit prediction. Zhang et al. [5] used Long Short-Term Memory (LSTM) networks to improve short-term LEO orbit prediction by learning future orbit-prediction errors from historical errors and auxiliary features. Jeong and Shin [6] proposed the Lightweight Transformer Encoder (LTE) for satellite orbit prediction, showing that Transformer-based temporal modeling can improve prediction accuracy while considering computational efficiency. These works demonstrate that temporal neural models can learn useful correction patterns from historical orbit data.

However, most learning-based orbit-prediction methods treat each object independently or focus mainly on improving the state prediction of individual satellites. This is suitable for reducing trajectory error, but conjunction screening is fundamentally a relative-motion problem. A small state error can change the estimated closest approach between two objects, and a good single-object forecast does not necessarily produce a stable risk ranking. In contrast, our framework explicitly models interactions among risk-relevant neighboring objects through a dynamic spatio-temporal graph and uses the resulting forecasts for uncertainty-aware conjunction triage.

2.3 Learning-Based Conjunction Screening

Another related direction is the use of machine learning to accelerate large-scale conjunction screening. Stevenson et al. [7] benchmarked deep learning approaches for all-vs-all conjunction screening and framed the problem as a machine-learning classification task over candidate object pairs. Their study showed that deep learning can help address the computational burden of all-vs-all screening by learning to filter or classify candidate pairs.

This direction is highly relevant to large-catalog SSA because the number of possible object pairs grows quadratically with catalog size. Nevertheless, pair-classification approaches mainly focus on screening efficiency: they decide whether a pair should be considered risky or not based on pair-level features. They do not necessarily produce multi-step trajectory forecasts, do not explicitly model evolving multi-object neighborhoods, and usually do not address distributed execution with delayed or missing cross-node information.

Our work is complementary to learning-based screening. Instead of treating conjunction screening only as a binary classification problem, we formulate it as an interaction-aware forecasting and ranking problem. The model predicts future states and calibrated uncertainty, computes closest-approach quantities from the predicted trajectories, and ranks candidate pairs using an uncertainty-aware score. This design is intended to improve not only filtering efficiency but also the stability of the top-K triage list under imperfect information.

2.4 Spatio-Temporal Graph Learning and Multi-Agent Forecasting

Graph neural networks provide a natural way to model systems in which entities interact through structured relationships. Graph Attention Networks (GATs) allow each node to assign different importance weights to its neighbors [10]. This idea is useful for SSA because not all nearby objects are equally important: an object moving away from the target may be less relevant than another object that is predicted to approach closely within the screening horizon.

Spatio-temporal graph models have been widely used in other forecasting domains. Yu et al. [11] proposed Spatio-Temporal Graph Convolutional Networks (STGCN) for traffic forecasting, showing that graph-based models can capture spatial dependencies together with temporal evolution. Mohamed et al. [12] proposed Social-STGCNN for pedestrian trajectory prediction, where interacting agents are represented as nodes in a spatio-temporal graph. These works show that graph-based temporal models are effective when future motion depends not only on an individual entity's past behavior, but also on interactions with surrounding entities.

SSA shares this interaction-aware structure, but it also has important differences from traffic or pedestrian forecasting. Orbital motion is governed by astrodynamics, relative velocities can be very high, and the main decision variable is often the closest approach between object pairs rather than the next position of a single agent. Moreover, catalog-driven SSA must handle stale TLEs, delayed updates, and partial observability across distributed data sources. Therefore, directly applying generic spatio-temporal graph models is insufficient. Our framework adapts the graph-learning idea to SSA by constructing edges using short-horizon risk relevance, encoding relative position, relative velocity,

CPA, and TCA as edge features, and coupling graph forecasting with uncertainty-aware conjunction triage.

2.5 Transformers and Temporal Forecasting

Transformers were introduced by Vaswani et al. [13] as an attention-based architecture for sequence modeling. Their self-attention mechanism allows a model to capture dependencies across a sequence without relying only on recurrent updates. This is useful for orbit forecasting because the model can attend to different parts of the historical state sequence, such as periods where the TLE residual appears to grow or where neighbor geometry changes rapidly.

In our framework, the Transformer is used as a causal temporal encoder. The causal constraint ensures that the model uses only past and present information when generating forecasts. This is important for deployment because future states are unavailable at decision time. The Transformer therefore complements the spatial graph encoder: the graph encoder captures which neighboring objects matter at each time, while the temporal encoder captures how these interaction patterns evolve over the historical window.

2.6 Position of This Work

The proposed framework combines and extends ideas from catalog-based propagation, learning-based orbit correction, conjunction screening, and spatio-temporal graph forecasting. Compared with SGP4-based screening, it learns residual corrections and uncertainty from historical catalog behavior. Compared with independent LSTM or Transformer orbit predictors, it explicitly models risk-relevant interactions among neighboring objects. Compared with pair-classification screening, it produces multi-step trajectory forecasts and a ranked top-K triage list. Compared with generic spatio-temporal graph models, it uses SSA-specific graph construction, edge features, CPA/TCA computation, and distributed boundary-summary exchange.

The main gap addressed by this work is therefore the lack of a unified framework for short-horizon, interaction-aware, uncertainty-calibrated, and distributed conjunction triage under delayed and incomplete catalog information.

3. Methodology

This section describes the proposed distributed SSA framework. The main goal is to predict the short-term motion of satellites and debris objects while also producing a reliable ranking of potentially risky object pairs. The key idea is simple: an object in orbit should not be predicted only from its own past trajectory, because conjunction risk depends mainly on its relative motion with nearby objects. Therefore, we represent the catalog as a dynamic interaction graph, forecast future states using a spatio-temporal graph model, estimate prediction uncertainty, and use this uncertainty to rank risky pairs conservatively. The same framework is designed to operate in a distributed setting, where different compute nodes may have delayed or incomplete information about objects managed by other nodes.

3.1 Problem Formulation

Let the catalog contain N resident space objects (RSOs), where each object is indexed by $i \in \{1,\dots,N\}$. At time t, the state of object i is represented as

$$\mathbf{x}_i(t) = \begin{bmatrix} \mathbf{r}_i(t) \\ \mathbf{v}_i(t) \end{bmatrix} \in \mathbb{R}^6, \tag{1}$$

where $\mathbf{r}_i(t) \in \mathbb{R}^3$ is the position vector and $\mathbf{v}_i(t) \in \mathbb{R}^3$ is the velocity vector. In other words, each object is described by six values: three for where it is and three for how fast and in which direction it is moving.

Given a past observation window of length T, sampled at a fixed time interval Δt, the historical sequence of object i is

$$\mathbf{X}_i(t) = \{\mathbf{x}_i(t-T+1),\dots,\mathbf{x}_i(t)\}. \tag{2}$$

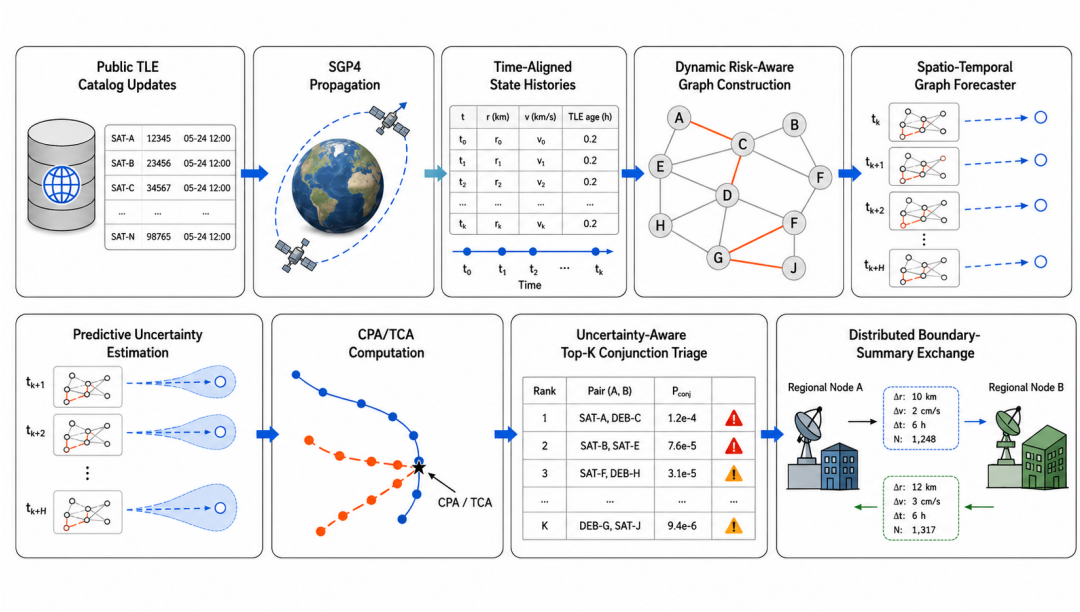


Figure 1 – Overview of the proposed distributed SSA framework. TLE-derived state histories are propagated using SGP4, converted into dynamic risk-aware interaction graphs, processed by a spatio-temporal graph forecaster, and ranked using uncertainty-aware CPA/TCA triage. Distributed nodes exchange compact boundary summaries rather than full catalogs.

The forecasting task is to predict the future trajectory of each object for the next H time steps:

$$\widehat{\mathbf{Y}}_i(t) = \{\widehat{\mathbf{x}}_i(t+1), \ldots, \widehat{\mathbf{x}}_i(t+H)\}. \tag{3}$$

However, for SSA, accurate trajectory forecasting alone is not enough. Operators are usually interested in which object pairs require attention. Therefore, the framework also estimates the uncertainty of each prediction:

$$\widehat{\Sigma}_i(t+k), \quad k = 1, \ldots, H, \tag{4}$$

where $\widehat{\Sigma}_i(t+k)$ describes how confident the model is about the predicted state at future step k. A small uncertainty means that the model is confident, while a large uncertainty means that the prediction should be treated carefully. This is important because stale TLEs, maneuvers, missing neighbor information, and communication delay can all make trajectory predictions unreliable.
The final output of the framework is a ranked top-K list of candidate object pairs:

$$\mathscr{R}_K(t) = [(i_1, j_1), \ldots, (i_K, j_K)], \tag{5}$$

where pairs at the top of the list are considered more important for conjunction monitoring. The objective is not only to minimize position error, but also to keep truly risky pairs near the top of the triage list, even when the available information is delayed or incomplete.

3.2 Catalog-Based State Construction

The system starts from publicly available catalog information represented by Two-Line Elements (TLEs). At each decision time t, the latest available TLE of each object is propagated using SGP4 to obtain position and velocity sequences. Because TLE updates do not arrive at exactly the same time for all objects, some objects may be represented by newer TLEs while others may be represented by older ones. This staleness is explicitly encoded as

$$a_i(t) = t - t_i^{\mathrm{TLE}}, \tag{6}$$

where t_i^{TLE} is the epoch of the latest available TLE for object i. The value $a_i(t)$ is called the TLE age. It tells the model how old the catalog information is. This is important because an old TLE is usually less reliable than a recent one.

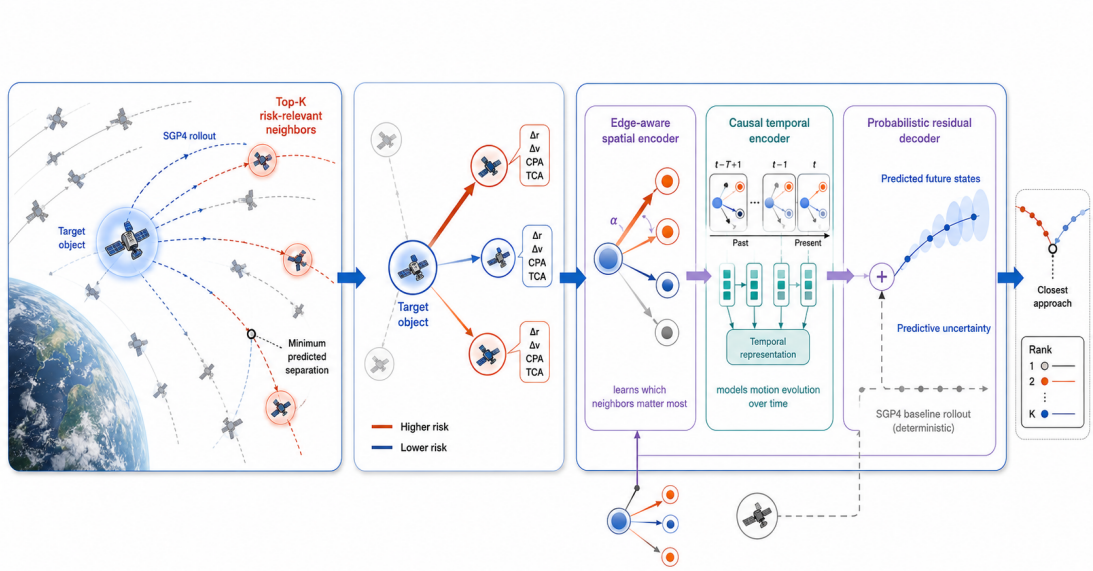


Figure 2 – Overview of the proposed risk-aware spatio-temporal forecasting framework for satellite SSA. Short-horizon SGP4 rollouts are used to identify risk-relevant neighbors, construct sparse dynamic interaction graphs with relative-motion and closest-approach features, and generate uncertainty-aware multi-step trajectory forecasts through an edge-aware spatial encoder, causal temporal encoder, and probabilistic residual decoder.

The node feature vector of object i is defined as

$$\mathbf{u}_i(t) = [\mathbf{x}_i(t), a_i(t), \eta_i(t)], \tag{7}$$

where $\eta_i(t)$ denotes optional orbital-context features derived from the TLE, such as inclination, eccentricity, mean motion, right ascension of the ascending node, argument of perigee, mean anomaly, and drag-related terms. These features provide physical context about the orbit. For example, two objects with similar altitude and inclination may have different interaction patterns from two objects in very different orbital regimes.

The intuition behind this representation is that SGP4 provides a fast physics-based baseline, while the learning model corrects its short-term errors. The model is not asked to learn orbital dynamics from zero. Instead, it learns when and how the SGP4-based prediction should be adjusted, especially when the catalog is stale or when nearby objects create risk-relevant interactions.

3.3 Dynamic Risk-Aware Graph Construction

At each decision time, the catalog is represented as a dynamic graph:

$$\mathcal{G}_t = (\mathcal{V}, \mathcal{E}_t), \tag{8}$$

where $\mathcal{V}$ is the set of objects and $\mathcal{E}_t$ is the set of edges. Each edge connects two objects that may be relevant for short-term conjunction screening.

A fully connected graph would contain all possible object pairs, but this is unnecessary and computationally expensive. Most objects in the catalog are far from each other and do not affect each other's short-term risk. Therefore, for each object, we select only a small set of risk-relevant neighbors.

First, SGP4 is used to produce a provisional short-horizon rollout for each object. Let $\widetilde{\mathbf{r}}_i(t+\tau)$ be the provisional position of object i at future time offset τ. For every pair (i, j), we compute the minimum provisional separation over a graph-screening horizon H_s:

$$d_{ij}^{\text{roll}}(t) = \min_{\tau \in [0, H_s]} \left\| \widetilde{\mathbf{r}}_i(t+\tau) - \widetilde{\mathbf{r}}_j(t+\tau) \right\|_2. \tag{9}$$

This value measures how close two objects are expected to become according to the fast SGP4 rollout. A small value means that the pair deserves attention.

The neighborhood of object i is then selected as

$$\mathcal{N}_i(t) = \mathrm{TopK}_{j\neq i}\left(-d_{ij}^{\mathrm{roll}}(t)\right), \tag{10}$$

where K_g is the number of graph neighbors. In our implementation, $K_g = 32$ unless otherwise stated. The negative sign is used because the smallest distances should be selected as the most important neighbors.

This graph construction has a clear operational meaning: each object listens mainly to the objects that are most likely to approach it within the screening horizon. This is different from ordinary trajectory prediction, where each object may be predicted independently. Here, the model is explicitly told which nearby objects may influence the risk ranking.

For each selected edge (i, j), we define the relative position and relative velocity:

$$\Delta\mathbf{r}_{ij}(t) = \mathbf{r}_j(t) - \mathbf{r}_i(t), \quad \Delta\mathbf{v}_{ij}(t) = \mathbf{v}_j(t) - \mathbf{v}_i(t). \tag{11}$$

These quantities describe the motion of object j from the viewpoint of object i. They are more useful for conjunction screening than absolute position alone, because collision risk depends mainly on relative distance and relative speed.

The edge feature vector is

$$\mathbf{e}_{ij}(t) = \left[\Delta\mathbf{r}_{ij}(t), \Delta\mathbf{v}_{ij}(t), \left\|\Delta\mathbf{r}_{ij}(t)\right\|_2, \left\|\Delta\mathbf{v}_{ij}(t)\right\|_2, \mathrm{CPA}_{ij}^{\mathrm{roll}}(t), \mathrm{TCA}_{ij}^{\mathrm{roll}}(t)\right]. \tag{12}$$

Here, $\mathrm{CPA}_{ij}^{\mathrm{roll}}(t)$ is the provisional closest approach distance computed from the SGP4 rollout, and $\mathrm{TCA}_{ij}^{\mathrm{roll}}(t)$ is the corresponding time of closest approach. These two values provide direct risk information to the graph model.

3.4 Closest Approach and Time of Closest Approach

The closest approach distance is one of the most important quantities in conjunction screening. For a predicted pair (i, j), we first compute the future time step at which their predicted distance is smallest:

$$\tau_{ij}^{*} = \arg\min_{\tau\in\{1,\ldots,H\}} \left\|\widehat{\mathbf{r}}_i(t+\tau) - \widehat{\mathbf{r}}_j(t+\tau)\right\|_2. \tag{13}$$

The symbol $\arg\min$ means "the value of τ that gives the minimum distance." Therefore, τ_{ij}^{*} is the predicted future step where the two objects are closest.

The predicted closest approach distance is then

$$\widehat{\mathrm{CPA}}_{ij}(t) = \left\|\widehat{\mathbf{r}}_i(t+\tau_{ij}^{*}) - \widehat{\mathbf{r}}_j(t+\tau_{ij}^{*})\right\|_2. \tag{14}$$

A smaller $\widehat{\mathrm{CPA}}_{ij}(t)$ means a more concerning pair. The predicted time of closest approach is

$$\widehat{\mathrm{TCA}}_{ij}(t) = t + \tau_{ij}^{*}\Delta t. \tag{15}$$

Thus, CPA tells us "how close," while TCA tells us "when."

For evaluation, a pair can be labeled as high-risk using a distance threshold:

$$y_{ij}(t) = \begin{cases} 1, & \mathrm{CPA}_{ij}^{\mathrm{ref}}(t) < d_{\mathrm{th}} \text{ and } \mathrm{TCA}_{ij}^{\mathrm{ref}}(t) \in [t, t+H\Delta t], \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

Here, d_{th} is the screening threshold, while $\mathrm{CPA}_{ij}^{\mathrm{ref}}$ and $\mathrm{TCA}_{ij}^{\mathrm{ref}}$ are computed using the reference trajectory. This definition means that a pair is considered high-risk if it becomes closer than the chosen threshold within the prediction horizon.

3.5 Spatio-Temporal Graph Forecasting Model

The proposed model has three main parts: a spatial graph encoder, a temporal encoder, and a probabilistic decoder. The spatial encoder learns which neighboring objects are important at each time. The temporal encoder learns how the motion evolves over time. The decoder predicts future states and uncertainty.

3.5.1 Edge-Aware Spatial Encoder

The spatial encoder updates the representation of each object by aggregating information from its risk-aware neighbors. For each edge (i, j), the model first builds a message from neighbor j to target object i:

$$\mathbf{m}_{ij}(t) = \phi_m\left([\mathbf{u}_j(t), \mathbf{e}_{ij}(t)]\right), \tag{17}$$

where $\phi_m(\cdot)$ is a small neural network and $[\cdot,\cdot]$ denotes feature concatenation. This message contains two types of information: the state of the neighbor and the relative-motion relationship between the neighbor and the target object.

Not all neighbors are equally important. A neighbor that is far away or moving away may be less important than a neighbor that is approaching quickly. Therefore, the model computes an attention weight:

$$\alpha_{ij}(t) = \frac{\exp\left(\phi_a\left([\mathbf{u}_i(t), \mathbf{m}_{ij}(t)]\right)\right)}{\sum_{\ell \in \mathcal{N}_i(t)} \exp\left(\phi_a\left([\mathbf{u}_i(t), \mathbf{m}_{i\ell}(t)]\right)\right)}. \tag{18}$$

The attention weight $\alpha_{ij}(t)$ tells the model how much object i should pay attention to neighbor j. The denominator normalizes the weights so that the importance of all neighbors is compared within the same neighborhood.

The final spatial embedding of object i is

$$\mathbf{h}_i(t) = \sigma\left(\mathbf{W}_0\mathbf{u}_i(t) + \sum_{j \in \mathcal{N}_i(t)} \alpha_{ij}(t)\mathbf{W}_1\mathbf{m}_{ij}(t)\right). \tag{19}$$

The first term, $\mathbf{W}_0\mathbf{u}_i(t)$, preserves the object's own state. The second term adds information from neighboring objects, weighted by their learned importance. The function $\sigma(\cdot)$ is a nonlinear activation function.

The intuition is similar to an SSA analyst looking at one object and asking: "Which nearby objects matter most for this object's short-term risk?" The attention mechanism allows the model to answer this question automatically from data.

3.5.2 Causal Temporal Encoder

After computing the spatial embedding at each time step, the model processes the sequence of embeddings:

$$\mathbf{H}_i(t) = \{\mathbf{h}_i(t-T+1), \ldots, \mathbf{h}_i(t)\}. \tag{20}$$

This sequence is passed to a causal Transformer:

$$\mathbf{z}_i(t) = \mathrm{Transformer}_{\mathrm{causal}}\left(\mathbf{H}_i(t)\right). \tag{21}$$

The word "causal" means that the model is only allowed to use past and present information. It cannot look into the future during training. This prevents information leakage and makes the training setting consistent with real deployment.

The role of the temporal encoder is to learn motion patterns over the historical window. For example, it can learn whether the SGP4 residual error is increasing, whether the TLE is becoming stale, or whether the local interaction pattern is changing. This is important because conjunction risk is not determined by a single snapshot, but by how relative motion evolves over time.

3.5.3 Probabilistic Residual Decoder

The decoder predicts future states as corrections to the SGP4 rollout:

$$\widehat{\mathbf{x}}_i(t+k) = \widetilde{\mathbf{x}}_i(t+k) + \Delta\widehat{\mathbf{x}}_i(t+k), \quad k = 1, \ldots, H. \tag{22}$$

Here, $\widetilde{\mathbf{x}}_i(t+k)$ is the SGP4-predicted state, and $\Delta\widehat{\mathbf{x}}_i(t+k)$ is the learned correction. This residual design is important. Instead of replacing physics-based propagation, the model improves it. SGP4 provides a physically meaningful first estimate, while the neural model learns systematic short-term errors caused by catalog noise, stale updates, and local interaction patterns.

The decoder also predicts a diagonal covariance matrix:

$$\widehat{\Sigma}_i(t+k) = \mathrm{diag}\left(\widehat{\sigma}_i^2(t+k)\right). \tag{23}$$

The diagonal covariance means that the model predicts one uncertainty value for each state dimension. To ensure that variance values are always positive, we use

$$\widehat{\sigma}_i^2(t+k) = \mathrm{softplus}\left(\mathbf{o}_i^{\sigma}(t+k)\right) + \varepsilon, \tag{24}$$

where $\mathrm{softplus}(\cdot)$ is a smooth function that outputs positive values, and ε is a small constant for numerical stability.
This uncertainty output is not only a mathematical detail. It is central to the proposed triage system. If the model is unsure about an object's future state, the risk ranking should reflect that uncertainty instead of treating the prediction as perfectly reliable.

3.6 Uncertainty-Aware Conjunction Triage

After predicting future trajectories, the framework ranks candidate pairs. For pair (i, j), the predicted relative position at future step k is

$$\widehat{\Delta\mathbf{r}}_{ij}(t+k) = \widehat{\mathbf{r}}_j(t+k) - \widehat{\mathbf{r}}_i(t+k). \tag{25}$$

The relative-position covariance is approximated as

$$\widehat{\Sigma}_{ij}^{r}(t+k) = \widehat{\Sigma}_i^{r}(t+k) + \widehat{\Sigma}_j^{r}(t+k), \tag{26}$$

where $\widehat{\Sigma}_i^{r}$ and $\widehat{\Sigma}_j^{r}$ are the position uncertainty blocks of the two objects. This equation means that if both objects are uncertain, then the uncertainty of their relative position becomes larger.
At the predicted closest-approach step τ_{ij}^{*}, we summarize the relative-position uncertainty as

$$\widehat{\sigma}_{ij}^{\mathrm{CPA}} = \sqrt{\mathrm{tr}\left(\widehat{\Sigma}_{ij}^{r}(t+\tau_{ij}^{*})\right)}. \tag{27}$$

The trace operator $\mathrm{tr}(\cdot)$ sums the diagonal elements of the covariance matrix. Therefore, this value gives a single scalar measure of how uncertain the predicted closest approach is.
The triage score is defined as

$$s_{ij}(t) = -\left(\widehat{\mathrm{CPA}}_{ij}(t) - \gamma\widehat{\sigma}_{ij}^{\mathrm{CPA}}(t)\right), \tag{28}$$

where $\gamma \geq 0$ controls how conservative the ranking is. If $\gamma = 0$, the score depends only on the predicted closest approach distance. If $\gamma > 0$, pairs with high uncertainty are ranked more cautiously.
This score can be understood as an uncertainty-aware lower confidence bound on the closest approach distance. A pair is considered more important if the predicted distance is small, or if the uncertainty is large enough that the true distance may be smaller than predicted. This is especially useful in distributed SSA, where missing or delayed information should not cause the system to ignore a potentially risky pair.
The final top-K triage list is

$$\mathscr{R}_K(t) = \mathrm{TopK}_{(i,j)\in\mathscr{P}(t)}\, s_{ij}(t), \tag{29}$$

where $\mathscr{P}(t)$ is the candidate-pair set. The highest-scoring pairs are returned to the operator or passed to a downstream high-fidelity screening module.

3.7 Distributed Inference

In practical SSA systems, data and computation may be distributed across multiple sensors, ground stations, operators, or organizations. Therefore, the catalog is divided across M compute nodes:

$$\mathcal{V} = \bigcup_{m=1}^{M} \mathcal{V}_m, \quad \mathcal{V}_m \cap \mathcal{V}_n = \emptyset \quad \text{for} \quad m \neq n. \tag{30}$$

Each node m is responsible for a subset of objects $\mathcal{V}_m$. The node can perform local SGP4 propagation, local graph construction, local forecasting, and local triage. However, conjunction risk does not always respect partition boundaries. An object handled by one node may have a risk-relevant neighbor handled by another node.

To handle this case without sharing full catalogs, nodes exchange compact boundary summaries. For boundary object i, the transmitted summary is

$$\mathbf{b}_i(t) = \left\{ i, t, \widehat{\mathbf{x}}_i(t+1:t+H), \widehat{\Sigma}_i(t+1:t+H), a_i(t) \right\}. \tag{31}$$

This summary contains the object ID, timestamp, predicted future states, prediction uncertainty, and TLE age. It does not contain raw TLE histories or the full local catalog.

The intuition is that neighboring nodes do not need to know everything about each other's catalogs. They only need enough information to evaluate short-term cross-partition risk. By sharing forecast states and uncertainty, the system supports distributed conjunction triage while keeping communication cost low.

The distributed inference procedure consists of eight steps: (1) receiving or updating TLEs for local objects; (2) propagating local objects with SGP4 and building synchronized state histories; (3) constructing risk-aware local neighborhoods; (4) identifying boundary objects near partition interfaces; (5) exchanging compact boundary summaries for these objects; (6) running the spatio-temporal graph forecaster using both local and received boundary information; (7) scoring candidate pairs using the uncertainty-aware triage score; and (8) merging local top-K lists into a regional or global top-K triage list.

The communication cost for node m can be approximated as

$$C_m \approx B_m H \left(D_x + D_\Sigma\right) q + C_{\text{meta}}, \tag{32}$$

where B_m is the number of transmitted boundary objects, H is the number of prediction steps, D_x is the state dimension, D_Σ is the number of uncertainty values, q is the number of bytes per scalar, and C_{meta} is the metadata overhead. This expression shows why the proposed protocol is communication-efficient: the cost grows with the number of boundary objects, not with the full catalog size.

3.8 Robustness to Delay, Packet Loss, and Missing Context

Distributed SSA is affected by communication delay and packet loss. A boundary message may arrive late, or it may not arrive at all. We model this using two parameters: a delay τ and a packet loss probability p. A delayed boundary message from object i is represented as

$$\mathbf{b}_i(t-\tau), \tag{33}$$

which means that the receiving node only has an old summary instead of the latest one.

If a delayed summary is available, the receiving node uses the timestamp to align it with the current decision time. If no summary is available, the node uses one of three fallback strategies: the most recently received summary, a local SGP4 rollout if a public TLE is available, or a masked-neighbor representation indicating that the neighbor context is missing. The missing-context mask is given to the model as an input feature, so the model can learn that the information is incomplete.

The intuition is straightforward: the model should behave differently when information is fresh and when it is stale. If a neighbor state is delayed or missing, the model should not make an overconfident prediction. Instead, it should increase uncertainty and rank potentially risky pairs more cautiously.

To train this behavior, we corrupt the training context using five types of impairments expected at inference time: (1) delayed boundary messages; (2) random packet dropout; (3) neighbor masking; (4) artificial TLE aging; and (5) maneuver-like perturbations.
This makes the model robust because it learns from imperfect conditions rather than only from clean data.

3.9 Maneuver-Robust Forecasting

Maneuvers can abruptly change an object's trajectory and make propagator-only predictions unreliable. In public catalog data, exact maneuver labels are often unavailable. Therefore, the proposed framework handles maneuver robustness in two ways: residual correction and uncertainty inflation.
First, the model predicts residual corrections to SGP4 instead of predicting the full state from scratch. This helps the model learn deviations from the nominal propagated trajectory. Second, the model is trained to increase uncertainty when the input sequence contains maneuver-like behavior.
During training, maneuver-like perturbations are injected by applying a small velocity change at a selected time t_m:

$$\mathbf{v}_i(t_m^+) = \mathbf{v}_i(t_m^-) + \Delta\mathbf{v}_i, \tag{34}$$

where $\mathbf{v}_i(t_m^-)$ and $\mathbf{v}_i(t_m^+)$ are the velocities before and after the perturbation, and $\Delta\mathbf{v}_i$ is a small sampled velocity impulse. This creates a controlled trajectory deviation that resembles a maneuver or an abrupt catalog inconsistency.
The goal is not to identify the exact maneuver command. Rather, the goal is to prevent the model from becoming confidently wrong. When a maneuver-like deviation appears, the model should either correct the forecast or increase the uncertainty so that the triage system treats the affected pairs carefully.

3.10 Training Objective

The model is trained using three complementary objectives:

$$\mathscr{L} = \mathscr{L}_{\text{pred}} + \lambda_{\text{cal}}\mathscr{L}_{\text{cal}} + \lambda_{\text{rob}}\mathscr{L}_{\text{rob}}. \tag{35}$$

The first term improves trajectory accuracy, the second term improves uncertainty calibration, and the third term improves robustness under corrupted or incomplete inputs. The parameters λ_{cal} and λ_{rob} control the relative importance of the calibration and robustness terms.
The prediction loss is

$$\mathscr{L}_{\text{pred}} = \sum_i \sum_{k=1}^{H} \text{Huber}\left(\mathbf{x}_i^{\text{ref}}(t+k) - \widehat{\mathbf{x}}_i(t+k)\right). \tag{36}$$

Here, $\mathbf{x}_i^{\text{ref}}(t+k)$ is the reference state, and the Huber loss penalizes prediction errors. Huber loss is used because it behaves like squared error for small mistakes but is less sensitive to large outliers than ordinary mean squared error. This is useful for TLE-based data, where occasional large residuals can occur.
The calibration loss is based on Gaussian negative log-likelihood:

$$\mathscr{L}_{\text{cal}} = \frac{1}{2}\sum_i \sum_{k=1}^{H} \left[\mathbf{e}_i(t+k)^\top \widehat{\Sigma}_i(t+k)^{-1}\mathbf{e}_i(t+k) + \log\left|\widehat{\Sigma}_i(t+k)\right|\right], \tag{37}$$

where

$$\mathbf{e}_i(t+k) = \mathbf{x}_i^{\text{ref}}(t+k) - \widehat{\mathbf{x}}_i(t+k). \tag{38}$$

This loss encourages the model to produce meaningful uncertainty. If the model predicts very small uncertainty but makes a large error, it is penalized. If it predicts very large uncertainty for every sample, it is also penalized through the log-determinant term. Therefore, the model is encouraged to be confident only when it is accurate.
The robustness loss compares predictions under clean input context and corrupted input context:

$$\mathscr{L}_{\text{rob}} = \sum_i \sum_{k=1}^{H} \left\|\widehat{\mathbf{x}}_i^{\text{clean}}(t+k) - \widehat{\mathbf{x}}_i^{\text{corr}}(t+k)\right\|_1 + \rho \max\left(0, \sigma_i^{\text{clean}}(t+k) - \sigma_i^{\text{corr}}(t+k)\right). \tag{39}$$

Table 2 – Summary of the proposed methodology.

Component	Setting
Input state	Position and velocity from TLE/SGP4 propagation
Catalog staleness	TLE age included as node feature
Sampling cadence	$\Delta t = 10$ minutes
History window	$T = 12$ hours
Forecast horizons	30, 60, and 180 minutes
Graph construction	Top-K_g risk-aware neighbors from SGP4 rollout
Default neighbors	$K_g = 32$
Edge features	Relative position, relative velocity, CPA, and TCA
Spatial encoder	Edge-aware graph attention
Temporal encoder	Causal Transformer
Decoder	Residual correction over SGP4 rollout
Uncertainty output	Diagonal predictive covariance
Triage score	Uncertainty-aware closest-approach ranking
Distributed exchange	Boundary forecasts, uncertainty, timestamp, and TLE age
Robustness factors	Delay, packet loss, missing neighbors, and maneuver-like changes

The first part encourages the prediction to remain stable when some context is corrupted. The second part encourages the model to increase uncertainty under corrupted conditions. In simple terms, the model should remain accurate when possible, but when the information quality becomes poor, it should honestly express higher uncertainty.

3.11 Reference Trajectory Construction

Public TLEs are useful but noisy. A single TLE propagated by SGP4 should not be treated as perfect ground truth. To obtain a more stable training and evaluation reference, we construct a multi-TLE smoothing reference from consecutive public TLEs.

For each object, we collect TLEs within a sliding window of 24–48 hours. Each TLE is propagated over its valid interval, and overlapping propagated arcs are combined to estimate a locally consistent trajectory. The reference trajectory is obtained by minimizing

$$\mathbf{X}_i^{\mathrm{ref}} = \arg\min_{\mathbf{X}} \sum_{q \in \mathscr{Q}_i} \sum_{\tau \in \Omega_q} \left\| \mathbf{x}_i(\tau) - \mathrm{SGP4}\left(\mathrm{TLE}_{i,q}, \tau\right) \right\|_2^2 + \mu \mathscr{S}(\mathbf{X}). \tag{40}$$

Here, $\mathscr{Q}_i$ is the set of TLEs for object i inside the smoothing window, Ω_q is the time interval covered by TLE q, and $\mathscr{S}(\mathbf{X})$ is a smoothness regularizer. The first term makes the reference trajectory agree with the available TLE arcs, while the smoothness term prevents unrealistic jumps.

This reference is used only to create training labels and evaluation targets. At inference time, the model does not use future TLEs. It only receives the latest available TLE-derived history, exactly as it would in an operational setting. Chronological train, validation, and test splits are used to avoid temporal leakage.

3.12 Implementation Summary

Table 2 summarizes the main methodological settings. These values define the default configuration and can be varied in ablation studies.

The overall design converts catalog-driven SSA into an interaction-aware and uncertainty-aware forecasting problem. The graph module captures which objects matter to each other, the temporal module captures how their motion evolves, the uncertainty module tells the triage system how much to trust each prediction, and the distributed protocol allows the method to work when information is fragmented across multiple nodes.

4. Experimental Setup and Results

This section evaluates the proposed framework from five perspectives: (1) trajectory forecasting accuracy; (2) top-K conjunction triage quality; (3) robustness to distributed message delay and packet

loss; (4) uncertainty calibration; and (5) communication and runtime cost. The evaluation is designed to test not only whether the method reduces trajectory error, but also whether it keeps risk-relevant object pairs near the top of the triage list when catalog information is stale, delayed, or incomplete.

4.1 Dataset and Preprocessing

The evaluation uses publicly available TLE data from Space-Track, with CelesTrak used to cross-check catalog metadata. We focus on LEO resident space objects because LEO is increasingly congested and requires frequent conjunction screening. The dataset was constructed offline from historical public TLE records. Starting from all available catalog entries within a 90-day study window, we filtered for LEO objects with sufficient TLE coverage and retained 4,500 RSOs for training, validation, and testing. At each decision time, the latest available TLE for each object was propagated using SGP4 and sampled at a fixed cadence of $\Delta t = 10$ minutes. Each propagated state contains position and velocity, and the TLE age was included as an input feature to represent catalog staleness.

For each object, the input history window is set to 12 hours. The model predicts future states at horizons of 30, 60, and 180 minutes. We use a chronological split of 70% for training, 10% for validation, and 20% for testing. This split prevents temporal leakage by ensuring that future catalog behavior is not used to train models evaluated on earlier test intervals.

Table 3 summarizes the dataset and evaluation configuration.

Table 3 – Dataset and evaluation configuration.

Item	Setting
Catalog sources	Space-Track and CelesTrak TLEs
Orbit regime	LEO RSOs
Number of RSOs	4,500
Evaluation period	90 days
Sampling cadence	10 minutes
Input history window	12 hours
Forecast horizons	30, 60, and 180 minutes
Train/validation/test split	70/10/20 chronological split
Reference construction	24–48 hour multi-TLE smoothing window
Graph neighbors	$K_g = 32$ per object
Distributed nodes	$M = 8$
Message delay	$\tau \in \{0, 1, 3\}$ update steps
Packet loss	$p \in \{0, 0.1, 0.3\}$

4.2 Reference Trajectory and Risk Labels

A single TLE propagated by SGP4 is not treated as exact ground truth because public TLEs contain modeling error and may be affected by delayed updates or maneuver-like changes. Therefore, training and evaluation use a multi-TLE smoothing reference. For each object, consecutive TLEs within a 24–48 hour sliding window are propagated and aligned over overlapping intervals. A locally consistent reference trajectory is then obtained by minimizing propagation-consistent residuals with a smoothness constraint.

This reference trajectory is used only to construct training labels and evaluation targets. At inference time, the model receives only the latest available TLE-derived history, exactly as it would in a catalog-driven operational setting. Future TLEs from the smoothing window are never used as model input.

For conjunction triage, a candidate pair (i, j) is labeled as high-risk if its reference closest approach distance falls below a screening threshold within the forecast horizon:

$$y_{ij}(t) = 1 \quad \text{if} \quad \mathrm{CPA}_{ij}^{\mathrm{ref}}(t) < d_{\mathrm{th}} \quad \text{and} \quad \mathrm{TCA}_{ij}^{\mathrm{ref}}(t) \in [t, t + H\Delta t]. \tag{41}$$

Otherwise, the pair is labeled as non-risky. Unless otherwise stated, we use $d_{\mathrm{th}} = 10$ km. This threshold defines close-approach candidates for evaluating screening and ranking quality. It is not intended to replace high-fidelity collision-probability analysis.

4.3 Baselines

We compare the proposed method with five baselines.

SGP4 risk ranking. The latest available TLE for each object is propagated using SGP4 [3]. Candidate pairs are ranked directly by predicted closest approach distance. This represents a standard fast catalog-driven screening approach.

dSGP4. This baseline uses differentiable SGP4-based correction [4] to improve short-horizon propagation while retaining the efficiency of analytical propagation.

Per-object LSTM. An LSTM model is trained to predict future state residuals from each object's historical state sequence [5]. This baseline captures temporal error patterns but does not use graph-neighbor information.

Per-object Transformer. A causal Transformer is trained as an independent sequence predictor for each object [6]. It uses the same state history and TLE-age feature as the proposed method but does not include interaction-aware graph aggregation.

AI conjunction screening. A pairwise learning-based screening baseline is trained using candidate-pair features, including relative position, relative velocity, provisional CPA, and provisional TCA [7]. It outputs a pairwise risk score but does not produce full multi-step object trajectories.

All learning baselines use the same train/validation/test split, input history length, sampling cadence, and forecast horizons. Hyperparameters are selected using the validation split.

4.4 Evaluation Metrics

Trajectory forecasting is evaluated using position RMSE and velocity RMSE. Position RMSE at horizon H is defined as

$$\mathrm{RMSE}_r@H = \sqrt{\frac{1}{N_{\mathrm{test}}} \sum_{i,t} \left\| \mathbf{r}_i^{\mathrm{ref}}(t+H) - \widehat{\mathbf{r}}_i(t+H) \right\|_2^2}. \tag{42}$$

Velocity RMSE is computed in the same way using velocity vectors. Lower RMSE indicates more accurate trajectory prediction.

Conjunction triage is evaluated using Recall@50, NDCG@50, and PR-AUC. Recall@50 measures whether high-risk pairs appear in the top 50 ranked candidates. NDCG@50 measures whether high-risk pairs are ranked near the top of the list. PR-AUC measures the precision-recall tradeoff under class imbalance, which is important because close-approach candidates are rare compared with all possible object pairs.

For distributed evaluation, we report Recall@50 and NDCG@50 under message delay and packet loss. Communication cost is measured as the mean number of kilobytes exchanged per update per node.

4.5 Implementation Details

The proposed model uses two edge-aware graph attention layers followed by a two-layer causal Transformer encoder. The hidden dimension is set to 128, with four attention heads and dropout of 0.1. The probabilistic decoder predicts residual corrections over SGP4 as well as a diagonal covariance for each future state. The model is trained using AdamW with a learning rate of 10^{-4}, batch size 128, and early stopping based on validation Recall@50 and position RMSE. Unless otherwise stated, the graph neighborhood size is $K_g = 32$, and the uncertainty risk-aversion parameter is set to $\gamma = 1.0$.

4.6 Trajectory Forecasting Results

Table 4 reports trajectory forecasting performance on the test split. The proposed ST-Graph model achieves the lowest position and velocity RMSE across all horizons. At the 60-minute horizon, it reduces position RMSE from 2.84 km with SGP4 to 1.91 km, corresponding to a 33% reduction. The improvement becomes more visible at longer horizons, where errors from catalog propagation accumulate. At 180 minutes, the proposed model reduces position RMSE from 6.73 km to 4.58 km.

Table 4 – Trajectory forecasting performance on the test split.

Method	RMSE$_r$@30m (km)	RMSE$_r$@60m (km)	RMSE$_r$@180m (km)	RMSE$_v$@60m (m/s)
SGP4 [3]	1.41	2.84	6.73	0.92
dSGP4 [4]	1.28	2.55	6.20	0.84
LSTM [5]	1.22	2.35	5.70	0.78
Transformer [6]	1.15	2.24	5.45	0.74
Ours (ST-Graph)	**0.96**	**1.91**	**4.58**	**0.61**

The improvement over the per-object Transformer shows the value of interaction-aware modeling. The Transformer baseline can learn temporal error patterns, but it does not explicitly condition each object on nearby risk-relevant objects. In contrast, the proposed graph model uses relative-motion neighborhoods, allowing trajectory corrections to be informed by local conjunction geometry.

4.7 Conjunction Triage Results

Table 5 reports conjunction triage performance at the 60-minute horizon. The proposed method improves Recall@50 from 0.62 with SGP4 risk ranking to 0.79. NDCG@50 improves from 0.55 to 0.70, indicating that high-risk pairs are not only recovered more often but are also ranked closer to the top. PR-AUC improves from 0.41 to 0.57, showing better ranking quality under class imbalance.

Table 5 – Conjunction triage performance at $H = 60$ minutes.

Method	Recall@50	NDCG@50	PR-AUC	Recall@50 $(\tau = 3, p = 0.3)$	Comm.
SGP4 risk ranking [3]	0.62	0.55	0.41	0.58	–
AI screening [7]	0.70	0.61	0.49	0.62	–
Ours (ST-Graph + uncertainty)	**0.79**	**0.70**	**0.57**	**0.73**	**7.6**

These results show that lower trajectory error translates into better decision quality. More importantly, the uncertainty-aware score improves top-K triage by preventing uncertain close-approach pairs from being confidently discarded. This is important in operational SSA because operators often inspect only a limited number of candidates.

In addition to aggregate ranking metrics, we visualize one representative triage decision to illustrate how the proposed method changes the decision-making process. The baseline ranks candidate pairs only by the predicted closest approach distance obtained from SGP4 propagation. In contrast, the proposed method combines the corrected CPA with predictive uncertainty, so a pair with moderate CPA but elevated uncertainty can be promoted in the top-K list rather than being prematurely discarded.

Figure 3 shows that the proposed method does not simply output a lower trajectory error; it changes the operational triage behavior. The uncertainty-aware score makes the ranking more conservative by keeping uncertain close-approach candidates visible to the operator or to a downstream high-fidelity screening module. This behavior is especially important in distributed SSA, where stale TLEs, delayed boundary summaries, or missing cross-node context can make point predictions overconfident.

4.8 Robustness to Delay and Packet Loss

Table 6 evaluates the proposed method under distributed message delay and packet loss. Performance decreases gradually as delay and loss increase, but the ranking remains stable. Under the strongest tested impairment, $\tau = 3$ and $p = 0.3$, Recall@50 remains 0.73 and NDCG@50 remains 0.65. This shows that the model does not depend on perfectly synchronized cross-node information. The degradation is moderate because the model is trained with delayed messages, neighbor dropout, and missing-context masks. When boundary information becomes stale or unavailable, the model increases uncertainty and ranks potentially risky pairs more conservatively.

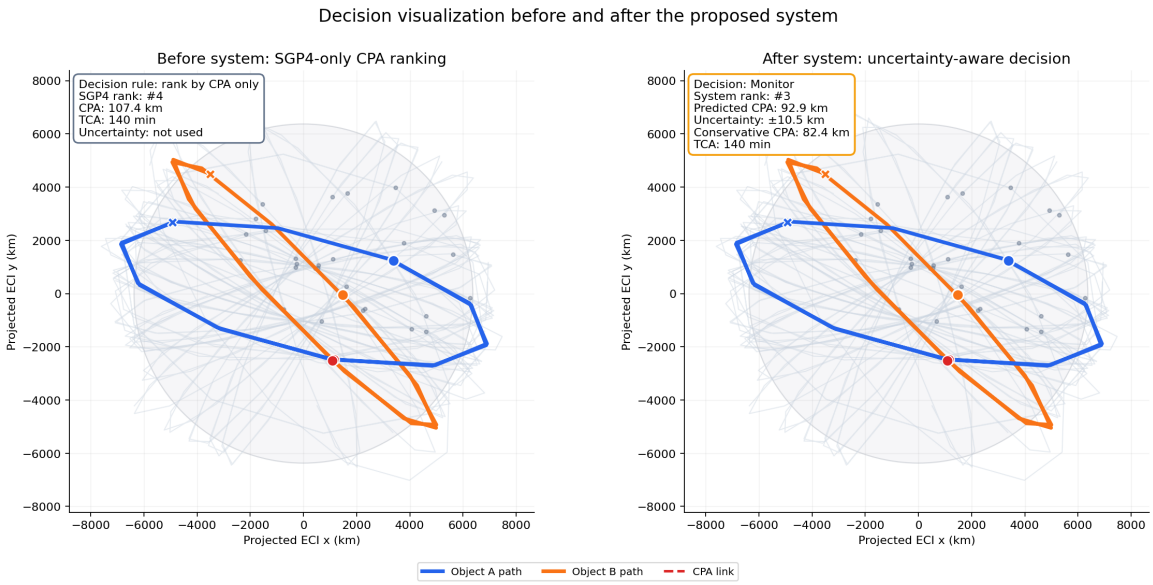


Figure 3 – Qualitative decision visualization comparing the baseline SGP4-only CPA ranking with the proposed uncertainty-aware triage. The left panel shows the baseline decision, where candidate pairs are ranked only by the SGP4-predicted closest approach distance. The right panel shows the proposed system, where the predicted CPA is combined with predictive uncertainty to produce an uncertainty-adjusted CPA score. In this example, the pair has an SGP4 CPA of 107.4 km and is ranked fourth by the baseline. After uncertainty-aware scoring, the predicted CPA becomes 92.9 km, the uncertainty-adjusted CPA score becomes 82.4 km, and the pair is promoted to rank three with a monitor decision. The gray background trajectories indicate neighboring objects in the local risk-aware graph, the blue and orange trajectories denote the two candidate objects, and the dashed red segment denotes the CPA link.

Table 6 – Robustness of uncertainty-aware triage under distributed delay and packet loss.

Delay τ	Loss p	Recall@50	NDCG@50	Comm. (kB/node)
0	0.0	0.79	0.70	7.6
1	0.0	0.78	0.69	7.6
1	0.1	0.77	0.68	7.6
1	0.3	0.75	0.66	7.6
3	0.0	0.76	0.67	7.6
3	0.1	0.75	0.66	7.6
3	0.3	0.73	0.65	7.6

4.9 Uncertainty Calibration

Table 7 reports uncertainty calibration at different forecast horizons. The empirical coverage of the nominal 90% prediction interval remains close to the target level, ranging from 91% at 30 minutes to 87% at 180 minutes. The small decrease at longer horizons is expected because uncertainty grows with forecast length and TLE errors accumulate over time.

Table 7 – Predictive uncertainty calibration.

Horizon	Nominal coverage	Empirical coverage	Mean interval width (km)
30 min	90%	91%	2.4
60 min	90%	89%	4.8
180 min	90%	87%	10.9

These results support the use of uncertainty in the triage score. The model is not only producing point forecasts; it also provides a meaningful estimate of when predictions should be trusted less. This is especially useful under delayed or missing distributed information.

4.10 Ablation Study

Table 8 studies the contribution of the main design components.

Table 8 – Ablation study at the 60-minute horizon.

Variant	RMSE_r@60m	Recall@50	NDCG@50	Recall@50 $(\tau = 3, p = 0.3)$
Full model	**1.91**	**0.79**	**0.70**	**0.73**
w/o graph interactions	2.24	0.68	0.60	0.62
w/o CPA/TCA edge features	2.02	0.74	0.65	0.69
w/o uncertainty-aware ranking	1.91	0.73	0.64	0.67
w/o robustness training	1.96	0.77	0.68	0.66
Local-only distributed inference	1.93	0.69	0.61	0.69

Removing graph interactions reduces Recall@50 from 0.79 to 0.68 and increases RMSE_r@60m from 1.91 km to 2.24 km. This confirms that risk-aware neighbor context is important for both forecasting and triage. Removing CPA/TCA edge features also reduces ranking quality, showing that closest-approach information is useful for graph construction and message passing.

Removing uncertainty-aware ranking does not change trajectory RMSE, but it reduces Recall@50 and NDCG@50. This indicates that uncertainty mainly improves decision quality rather than point prediction accuracy. Removing robustness training has the largest effect under distributed impairment, where Recall@50 drops from 0.73 to 0.66. Finally, using only local information without boundary exchange reduces communication to zero, but it also reduces Recall@50 because cross-partition risk-relevant neighbors are missing.

The ablation confirms that the proposed performance comes from the combination of interaction-aware forecasting, CPA/TCA-informed edge features, uncertainty-aware ranking, and robustness-aware training.

4.11 Maneuver-Like Perturbation Analysis

To evaluate maneuver robustness, we test the model under maneuver-like perturbations. Since confirmed maneuver labels are not always available in public TLE data, we inject small velocity impulses into selected trajectories and evaluate whether the model can maintain useful forecasts and uncertainty-aware triage. This experiment tests whether the method becomes confidently wrong after abrupt trajectory changes.

Table 9 compares performance on normal intervals and maneuver-like intervals. As expected, maneuver-like perturbations increase forecasting error for all methods. However, the proposed model maintains higher Recall@50 and better uncertainty coverage than the version trained without maneuver-like perturbations. This indicates that residual forecasting and uncertainty inflation help the triage system remain cautious when motion deviates from the nominal SGP4 rollout.

Table 9 – Performance under maneuver-like perturbations.

Scenario	RMSE_r@60m (km)	Recall@50	90% coverage
Normal intervals, full model	1.91	0.79	89%
Maneuver-like intervals, SGP4	3.42	0.55	–
Maneuver-like intervals, w/o robustness training	2.71	0.63	78%
Maneuver-like intervals, full model	2.36	0.70	86%

The goal of this experiment is not to estimate the exact maneuver command. Instead, it verifies that the system does not over-trust nominal propagation when sudden deviations occur. The proposed model either partially corrects the forecast or increases uncertainty, allowing affected pairs to remain visible in the top-K triage list.

4.12 Communication and Runtime Cost

Table 10 reports the average runtime and communication cost per update per node. The proposed distributed protocol exchanges only boundary summaries containing predicted states, uncertainty, timestamps, and TLE age. The average communication cost is 7.6 kB per update per node, which is much smaller than exchanging raw TLE histories or full local catalogs.

Table 10 – Average runtime and communication cost per update per node.

Component	Time/update (ms)	Communication
SGP4 propagation	84	–
Risk-aware graph construction	210	–
ST-Graph inference	145	–
CPA/TCA scoring and top-K merge	56	–
Boundary-summary exchange	–	7.6 kB/node
Total local processing	495	7.6 kB/node

The runtime results show that the framework can support frequent short-horizon updates. Most of the computation is spent on risk-aware graph construction, which is expected because candidate neighbor selection requires short-horizon pairwise screening. The communication cost grows with the number of boundary objects rather than the full catalog size, making the method suitable for distributed SSA settings where bandwidth and data-sharing constraints are important.

4.13 Discussion

The results show three main trends. First, interaction-aware graph forecasting improves trajectory accuracy compared with SGP4 and independent sequence models. Second, uncertainty-aware triage improves the stability of the top-K risk list, especially when predictions become less reliable. Third, compact boundary-summary exchange allows distributed nodes to maintain cross-partition conjunction awareness with low communication cost.

The most important result is not only the reduction in average RMSE, but the improvement in top-K triage quality. In operational SSA, a method is useful only if it keeps important candidate pairs visible to operators or downstream high-fidelity screening modules. The proposed framework directly addresses this requirement by combining relative-motion graph modeling, calibrated uncertainty, and distributed inference under delay and packet loss.

5. Conclusion

This paper presented a distributed SSA framework for short-horizon trajectory forecasting and uncertainty-aware conjunction triage. The proposed method represents resident space objects as a dynamic risk-aware graph, where edges capture relative motion, CPA, and TCA information. A spatio-temporal graph forecaster then combines spatial interaction modeling, temporal sequence modeling, and residual correction over SGP4 to predict future states and their uncertainty. The framework supports distributed execution by allowing nodes to exchange compact boundary summaries instead of full catalogs. This enables cross-partition conjunction awareness while reducing communication cost. The method also accounts for delayed messages, packet loss, stale TLEs, and missing neighbor context through robustness-aware training and uncertainty-aware ranking. Experiments using catalog-driven TLE data show that the proposed approach improves trajectory forecasting and top-K conjunction triage compared with SGP4-based and learning-based baselines. The results suggest that interaction-aware graph forecasting can improve not only orbit prediction, but also risk prioritization under incomplete and delayed SSA information.

Future work will extend the framework to stronger maneuver modeling, high-fidelity covariance propagation, operational ephemeris validation, and larger-scale distributed deployment.

Acknowledgment

This work was supported by Japan's Ministry of Internal Affairs and Communications (MIC) "Research and Development for Expansion of Radio Wave Resources" program (Grant Number JPJ000254) and

by the Japan Aerospace Exploration Agency (JAXA) Space Strategy Fund (SSF) for the project “High accuracy satellite formation flight technology” (ID: JPJXSSF24MS09003).

6. Contact Author Email Address

The contact author email address is: hamada_rizk@ist.osaka-u.ac.jp

7. Copyright Statement

The authors confirm that they, and/or their company or organization, hold copyright on all of the original material included in this paper. The authors also confirm that they have obtained permission, from the copyright holder, of any third-party material included in this paper, to publish it as part of this paper. The authors confirm that they give permission, or have obtained permission from the copyright holder of this paper, for the publication and distribution of this paper as part of the ICAS proceedings or as individual off-prints from the proceedings.

References

[1] Yui Maruyama, Ryohei Tsuji, Akira Uchiyama, Hamada Rizk, Hirozumi Yamaguchi, Sumio Morioka, and Takahiro Inagawa. When satellites work as one: Adaptive beamforming for picosat swarms. In *Companion Proceedings of the 27th International Conference on Distributed Computing and Networking*, pages 26–27, 2026.

[2] Hamada Rizk, Hirozumi Yamaguchi, Sumio Morioka, and Takahiro Inagawa. Surrogate-guided graph policy learning for real-time coordination of mobile agents. In *IEEE International Conference on Mobile Data Management*, 2026.

[3] David A. Vallado, Paul Crawford, Richard Hujsak, and T. S. Kelso. Revisiting spacetrack report #3. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Keystone, Colorado, 2006.

[4] Giacomo Acciarini, Atılım Güneş Baydin, and Dario Izzo. Closing the gap between SGP4 and high-precision propagation via differentiable programming. *Acta Astronautica*, 226:694–701, 2025.

[5] Wei Zhang, Keke Zhang, Xingxing Li, and Jiande Huang. Improving LEO short-term orbit prediction using LSTM neural network. *Advances in Space Research*, 76(1):481–496, 2025.

[6] Seungwon Jeong and Youjin Shin. LTE: Lightweight transformer encoder for orbit prediction. *Electronics*, 13(22):4371, 2024.

[7] Emma Stevenson, Victor Rodriguez-Fernandez, Hodei Urrutxua, and David Camacho. Benchmarking deep learning approaches for all-vs-all conjunction screening. *Advances in Space Research*, 72(7):2660–2675, 2023.

[8] Maruthi R. Akella and Kyle T. Alfriend. Probability of collision between space objects. *Journal of Guidance, Control, and Dynamics*, 23(5):769–772, 2000.

[9] Russell P. Patera. General method for calculating satellite collision probability. *Journal of Guidance, Control, and Dynamics*, 24(4):716–722, 2001.

[10] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[11] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 3634–3640, 2018.

[12] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian 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.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.