Optimizing the Efficiency of Heterogeneous Fusion of Multi-Source Information Nodes in Smart Government Platforms Using GraphSAGE

Mingyue Wang
Development Service Center, Qingdao Guzhenkou Core Area Management Committee, Qingdao 266400, Shandong, China
Corresponding author: Mingyue Wang (e-mail: 18560621750@163.com)

ABSTRACT To address low fusion efficiency in multi-source heterogeneous data and the difficulty of processing large-scale graph data in smart government platforms, this paper proposes a GraphSAGE-based optimization method. A heterogeneous graph structure with multiple node and edge types is first constructed, abstracting government entities into heterogeneous nodes and distinguishing business-related edge semantics. The GraphSAGE framework is improved using type-aware neighbor sampling and a multi-head attention aggregation function to preserve heterogeneous semantic features. Considering the temporal characteristics of government data, an LSTM dynamic modeling module is introduced to enhance the temporal evolution of node representations. A graph partitioning strategy combined with distributed training is further used to improve large-scale computation by optimizing communication and synchronization processes. Experiments show that the proposed method achieves node embedding accuracy of at least 0.95 and NMI of 0.88 in complex government graph scenarios, indicating high fusion quality. For 100,000 nodes, the training time is 12 seconds, with convergence achieved in 140 rounds. The results demonstrate that the method can effectively support large-scale heterogeneous graph representation, dynamic information fusion, and efficient decision-support modeling in smart platform environments.

INDEX TERMS smart government platforms, heterogeneous information fusion, graphsage, multi-head attention, distributed training

I. INTRODUCTION

SMART government platforms and "smart factories" both rely on integrating multi-source, heterogeneous data—across departments, systems, and supply chains—to enable intelligent, interconnected decision-making [1,2]. Government data, involving users, processes, policies, and approvals, is highly heterogeneous and relationally complex [3]. Graph-based modeling suits this structure, yet traditional methods fail to balance semantic fidelity with efficiency [4–6], and lack systematic mechanisms for cross-type semantic alignment and information transfer, limiting generalization [7,8]. Thus, preserving heterogeneous semantics while improving dynamic modeling and large-scale training efficiency remains a critical challenge.

The research's key innovations are:

(1) constructing a heterogeneous graph structure tailored to government scenarios that accurately captures multi-type node and edge relationships;

(2) optimizing information aggregation by integrating type awareness with attention mechanisms to enhance heterogeneous data fusion;

(3) incorporating LSTM modules to model temporal dynamics in node representations; and

(4) applying distributed training techniques to accelerate large-scale graph model training.

Experiments show the method outperforms existing approaches in accuracy, efficiency, and adaptability across multiple government graph scenarios, indicating strong potential for adaptation in the industry to analyze complex supply chain and manufacturing graphs with superior performance.

II. RELATED WORK

In recent years, graph neural networks have advanced heterogeneous information fusion and graph modeling for smart government platforms. Graph Convolutional Network (GCN) aggregates neighbors but ignores node/edge type distinctions in heterogeneous graphs [9,10]. Graph Attention Network (GAT) improves aggregation flexibility via attention, yet struggles with multi-type semantics [11,12].

Graph Transformer enhances global modeling but suffers high complexity, limiting scalability [13,14]. GraphGAN uses adversarial learning to refine embeddings and has been applied in dynamic scheduling and fuzzy detection, but its training is unstable and lacks systematic heterogeneous modeling [15,16]. GraphSAGE supports largescale representation learning but originally overlooks node-type differences and semantic heterogeneity [17]. To address these, some studies adopt meta-path mechanisms to enhance semantics, but rely on manual path design, limiting adaptability to dynamic government graphs [18]. Li M proposed HEAL, a meta-path-based social recommendation model using high-order relationships and dual attention to tackle sparsity and influence heterogeneity [19]. Others use heterogeneous GNNs for fine-grained aggregation but face efficiency bottlenecks on ultra-large graphs [20,21]. Overall, existing work still suffers from inadequate semantic modeling, low training efficiency, and weak dynamic adaptability. This paper proposes a GraphSAGE-based method integrating a type-aware mechanism, multi-head attention aggregation, an LSTM dynamic module, and distributed training to enhance fusion efficiency and expressiveness of multi-source heterogeneous nodes in smart government platforms.

III. METHODS

Figure 1 depicts a GraphSAGE-based architecture for heterogeneous information fusion in smart government. Government entities are modeled as multiple node types in a semantically rich heterogeneous graph. A type-aware neighbor sampling strategy prioritizes critical business links, while multi-head attention assigns edge-type-specific weights to preserve semantic heterogeneity. An LSTM captures dynamic signals like approval status and policy validity. To handle data scale, the graph is partitioned by business domain, and a parameter server enables efficient distributed training with optimized cross-subgraph communication. The fused representation supports node classification, heterogeneous fusion, and time-series prediction, jointly addressing government data's heterogeneity, dynamics, and scale.

A. CONSTRUCTING A HETEROGENEOUS GRAPH STRUCTURE WITH MULTI-TYPE NODES AND EDGES

1) Heterogeneous Node Abstraction of Government Entities

The core of building a graph structure for a smart government platform lies in the precise abstraction and modeling of various government entities. Within the government platform, various entities exist, including users, transactions, departments, and policies. These entities play different roles in the service process and have complex and diverse relationships with each other. To model these entities, node abstraction is used to preserve and clarify the semantic characteristics of each entity, and each entity is represented differently through node features. User nodes store user personal information, such as identity and registration time; transaction nodes store business attributes, such as task type and processing time; department nodes store organizational structure information; and policy nodes store rules and regulations. This fine-grained node feature design avoids simplistic, uniform modeling, allowing each entity to retain its own semantic characteristics within the graph structure. To enable semantic alignment across node types, all original node features are mapped to a unified dimension S = 128 via type-specific linear projection layers. Each node type maintains its own projection matrix, ensuring semantic characteristics are preserved in the shared embedded space.

The node design fully leverages the specific characteristics of government services. User and transaction nodes emphasize temporal relationships, while department and policy nodes emphasize hierarchical relationships. Each node is modeled specifically based on its business role. Therefore, multidimensional node abstraction effectively preserves the complex relationships within government data, providing a high-quality graph structure for subsequent in-depth analysis and information fusion.

2) Semantic Differences of Multiple Edges

The diversity of relationships within government platforms is also astonishing. Various nodes are connected by different types of edges, each of which carries specific business meanings. A user and an item might have an "initiate" or "approve" relationship, while a department and a policy might have an "execute" or "enact" relationship. These different edge types directly reflect the operational logic of the government system. When constructing edge relationships, the type of each connection should be clearly defined and assigned corresponding characteristic attributes. Therefore, edge feature design is closely related to actual business scenarios. For example, if an edge between a user and an item exists, it can record the number of interactions or urgency of the item on that edge; if an edge between a department and a policy exists, it can record the policy's execution priority or the scope of influence of the department's execution. This fine-grained edge feature design enables the graph structure to directly reflect the interactions between different types of entities in the government system.

For edge weight calculation:

$$w_{ij} = \sum_{k=1}^{S} \alpha_k \cdot x_{ik} \cdot y_{jk} \tag{1}$$

w_{ij} represents the weight of the edge between nodes, x_{ik} and y_{jk} represent the kth feature of different nodes, α_k is the weight coefficient of each feature, and S is the total number of features. This formula allows the edge weight to be adjusted based on different node characteristics, accurately expressing the semantics of the relationship.

The edge type in the graph structure influences the information transfer between nodes and also has a significant impact on subsequent information fusion. By distinguishing different edge types, the graph structure can more accurately capture the heterogeneous relationships between different

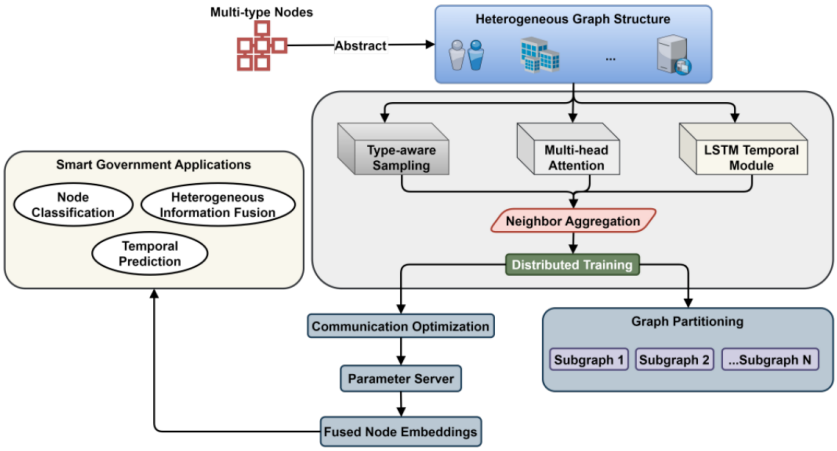

FIGURE 1. Optimizing Heterogeneous Information Fusion in Smart Government Based on GraphSAGE

types of nodes. In certain business scenarios, the "execution" relationships between departments and matters, and between departments and policies, may be assigned different weights and transmission mechanisms, thus affecting the speed and direction of information flow.

Regarding the complex multi-type edges within the government affairs platform, strict semantic differentiation and edge feature design are employed to ensure that each edge plays an appropriate semantic role within the graph model. This strategy effectively supports the deep integration of information within the platform and provides a structural guarantee for the efficient processing of subsequent tasks.

B. IMPROVING GRAPHSAGE'S NEIGHBOR SAMPLING AND AGGREGATION STRATEGY

1) Type-Aware Neighbor Sampling Strategy

When processing heterogeneous graphs, the relationships between nodes and neighbors are often not simple connections. Different types of neighbors have varying degrees of influence on nodes. Effectively selecting neighbors associated with each node is particularly crucial in diverse data scenarios like smart government platforms. To address this, it improved the neighbor sampling mechanism to enable differentiated sampling of neighbors based on node type. This strategy was developed through observing the imbalance in node roles and information sensitivity within government network diagrams. By prioritizing sampling neighbors with high semantic relevance, it effectively mitigates information overload caused by dominance of highly connected nodes like "departments" and "policies". This approach ensures privacy-sensitive nodes such as "users" can focus on their core business interactions, thereby enhancing semantic accuracy and security during the aggregation process.

Specifically, during the sampling process, the sampling probability is adjusted based on the type of neighboring node. For example, if a node is of the "user" type, the sampling probability of its neighboring nodes can be different than for a node of the "event" type. Considering the close interaction between users and issues, the sampling probability of user nodes for issue nodes can be appropriately increased, while the sampling probability of issue nodes for department or policy nodes can be relatively low. This sampling mechanism ensures that the feature aggregation process of each node aligns with its actual business logic, thereby more accurately reflecting the heterogeneous relationships between nodes.

To implement a type-aware sampling mechanism, a weighted sampling approach is adopted, with weights determined by the relationship between node type and neighbor types. When setting the sampling probability, the similarity between node types is first calculated and used as an adjustment factor for weighting. The specific sampling probability formula is as follows:

$$P_{i,j} = \frac{\exp\left(\alpha_{i,j} \cdot \mathrm{sim}(i,j)\right)}{\sum_{k \in N_i} \exp\left(\alpha_{i,k} \cdot \mathrm{sim}(i,k)\right)} \tag{2}$$

$P_{i,j}$ represents the probability of node i sampling from neighbor node j, $\alpha_{i,j}$ is the weight coefficient between a node and its neighbors, $\mathrm{sim}(i,j)$ is a measure of the type similarity between the node and its neighbors, and N_i is the set of neighbors of node i. By adjusting the sampling probability, different types of neighbor nodes are assigned different sampling weights, further strengthening the representation of relationships between different types of nodes in the graph structure.

This type-aware neighbor sampling method effectively avoids the limitations of traditional static sampling strategies and improves the model's ability to model relationships between heterogeneous nodes. This is particularly true for complex data types found in government platforms. The sampling strategy can be dynamically adjusted based on node characteristics, more accurately capturing important node interaction information.

Figure 2 shows the distribution of sampling weights for various neighboring nodes for different target node types, revealing the practical effectiveness of type-aware neighbor sampling strategies in heterogeneous graphs. The data clearly demonstrates a preference for sampling neighbors

of the same type, with significantly higher weights given to similar neighbors than to heterogeneous neighbors. This sampling preference effectively ensures that key semantic information is retained during the node feature fusion process, avoiding information dilution and confusion. The high sampling weight (0.7) of the User node for user neighbors reflects the closeness of information connections between users, making it easier to capture user behavior patterns and preferences. The Event and Department nodes also show a significant dependence on similar neighbors, which is consistent with the strong business connections between similar matters and departments in government processes. The policy node's emphasis on policy Neighbor emphasizes the importance of consistency and hierarchy between policies, which is conducive to understanding policy evolution and impact chains. Although the sampling weight of heterogeneous neighbors is relatively low, it cannot be ignored. Their existence provides the model with a channel for cross-type information transmission and enriches the semantic expression of the node.

2) Multi-Head Attention Aggregation Function

In heterogeneous graphs, the relationships between nodes and their neighbors vary not only in type but also in the strength of the interactions between different edge types. To more accurately capture the information exchange between a node and its neighbors, a multi-head attention mechanism is employed to aggregate neighbor information. The input to this attention mechanism not only includes the features of neighboring nodes, but also incorporates the features of heterogeneous edge types that represent relational semantics defined in the graph construction phase, thus taking into account the heterogeneity of node attributes and connection relationships during the aggregation process.

The multi-head attention mechanism aggregates neighbor information using multiple independent attention heads. Each attention head corresponds to an independent attention calculation process. Neighbor node features are weighted and summed according to different weights. The result is simply to connect all the heads and weighted average them to get the final representation of the nodes. Attention aggregation can be expressed as follows:

$$h_i' = \sum_{j \in N_i} \beta_{i,j} \cdot W_j h_j \tag{3}$$

h_i' represents the new feature representation after node aggregation, W_j is the feature transformation matrix of neighbor node j, and h_j represents the features of neighbor node j. Different from the single-attention model, multiple attentions learn different aggregation ways and compose a richer representation of node. Each attention head analyzes the feature of neighbor node from different view and thus makes the model more comprehensive to aggregate information and avoids that certain relationship has too much influence on the aggregation results. This mechanism is also suitable for handling heterogeneous information, as the final representation of node can contains more semantic information and represents richer semantics.

C. INTRODUCING A DYNAMIC TIME SERIES MODELING MECHANISM

1) Time Series Modeling of Node Features

Government data has a very obvious temporal evolutionary characteristic. The modeling method not only reflects the state of node at a certain moment, but also reflects the state change of feature of node . Traditional static graph neural networks (GNN) are limited in that they cannot effectively model the dynamic evolution of node state. This paper uses time-series-aware module to solve this problem. This module uses time series modeling technology to continuously and dynamically update the representation of node. The module uses long short-term memory network (LSTM) to handle node time series feature data. Compared with traditional RNN (Recurrent Neural Networks), LSTM has a gating mechanism that can effectively avoid the gradient disappearance or explosion problem when handling long sequence. For each historical feature sequence of node, LSTM intelligently selects which historical information to retain and which to discard based on the current input and past states, which makes the current state update of LSTM more accurately reflects the temporal evolution of feature of node . The state update of time series modeling unit obeys the following calculation rules:

$$f_t = \sigma\left(W_f \cdot [y_{t-1}, x_t] + b_f\right) \tag{4}$$

$$g_t = \tanh\left(W_g \cdot [y_{t-1}, x_t] + b_g\right) \tag{5}$$

f_t represents the output of the forget gate, which controls the influence of the previous state on the current state. g_t represents the candidate memory cell. σ is the sigmoid function. W_f and W_g are the weight matrices for the forget gate and candidate memory cell, respectively. x_t is the input feature at the current moment, y_{t-1} is the hidden state at the previous moment, and b_f and b_g are bias terms.

In government graph scenarios, not all nodes possess high-frequency or regular time series. Therefore, the input time series for the LSTM module is constructed based on discrete update events of key node states. For policy nodes, the sequence consists of timestamps of different version releases and their corresponding core parameters; for approval item nodes, the sequence comprises records of processing status and time stamps at each stage of their lifecycle; for department and user nodes, the statistical characteristics of their associated transactions (such as monthly processing volume and average processing time) are aggregated to form a low-frequency but semantically clear time series. This design ensures that even in sparse update scenarios, LSTM can effectively capture the evolution patterns of node core attributes.

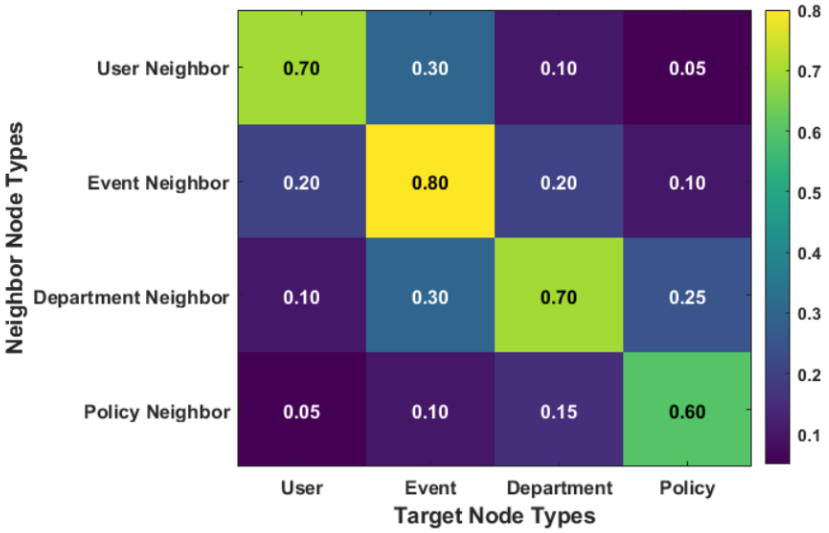

FIGURE 2. Neighborhood Sampling Weights by Node Type (data-driven estimation results based on node type similarity and edge weights)

2) The Impact of Time Series Features on Node Representation

By adding a time series modeling mechanism, nodes' expression is no longer bound to be in a static graph, but can be updated dynamically with time and status change. Such capability of modeling time change greatly improve the ability of graph neural network to handle time series data in government. It can not only express the status of node at current moment, but also remember the change at past and analyze the trend of development. It's very important in government application scenario. For example, evaluate the effectiveness of implementation of policy, trace the whole process of approval, etc.

When facing the data with time information, such as "different versions of policies", "approval process of different years", etc., the model can correctly distinguish and use the time sequence and context of events' development, just like managing the archives. Take LSTM as an example, it can generate a new representation of node representation by combining the historical information and current input information of the node at certain moment.

Combined with temporal continuity feature of government data, update formula of node representation can be expressed as:

$$h_t = f(h_{t-1}, x_t) \tag{6}$$

h_t represents the node representation at time t, and h_{t-1} represents the node representation at time $t-1$. This update process ensures that the node representation comprehensively considers information from the time series, thereby more accurately reflecting the dynamic changes in government data.

This mechanism can capture the changes of node's status in real time. In government system, information such as "the process of approval of project" or "period of implementation of regulation" can also be updated continuously along with the change of policy and market. By adding time dimension to model, it can make the system more dynamic and adapt to the current situation like flowing water. Traditional models are very rigid and difficult to adapt to the changes, but the graph neural network can model the time-sensitive data in smart government application more flexibly by introducing such capability, so as to improve the accuracy of situation analysis and prediction of future trend.

D. DISTRIBUTED TRAINING AND GRAPH PARTITION OPTIMIZATION

1) Graph Data Partitioning and Subgraph Training

Single-machine computing for processing large-scale government graph data often struggles to meet resource requirements and can easily become a performance bottleneck. To address this issue, this study employs a community detection algorithm based on node type and edge weight for graph partitioning, prioritizing the clustering of nodes with high connection density and strong semantic associations (such as nodes with the same or related business types) into the same subgraph. During the training phase, each subgraph performs independent computations using a local sampling strategy, eliminating the need to access global data. This achieves true parallel processing and improves computing resource utilization. This reduced reliance on central nodes or remote data makes the overall training process more efficient and stable. Once all subgraphs have completed their training, the system integrates the learned node representations across subgraphs through an intelligent fusion mechanism, ultimately generating a consistent and semantically complete global graph representation. This "divide and conquer" approach not only fully preserves the rich information in the original graph data, but also effectively breaks through the limitations of single-machine processing capabilities, providing a practical technical path

for large-scale graph data analysis in the government field. This strategy synergizes with the type-aware neighbor sampling mechanism proposed in this paper. Since the sampling process prioritizes semantically relevant high-weight neighbors, the node connections within the subgraphs after graph partitioning are more semantically coherent, thereby reducing the need for critical information transmission across subgraphs. Therefore, this combined approach not only inherits the inherent scalability of GraphSAGE but also significantly reduces communication overhead in distributed environments by leveraging the semantic structure of heterogeneous graphs, providing a more efficient solution than the standard GraphSAGE framework for processing large-scale heterogeneous government affairs graphs.

2) Optimizing Communication and Synchronization Mechanisms

This strategy combines local computation with global synchronization. During each subgraph computation, only necessary neighbor node information is transmitted, and during node feature updates, only essential features are synchronized. Two key operations are defined to manage inter-node communication:

$$\mathbf{h}_t^{(v)} = \sum_{u \in N(v)} w_{vu} \cdot \mathbf{h}_t^{(u)} \tag{7}$$

$\mathbf{h}_t^{(v)}$ represents the feature of node v at time t, $N(v)$ represents the set of neighbor nodes of node v, and w_{vu} is the weight between node v and neighbor node u. Formula (7) shows the aggregation process of local neighbor information. The feature of the current node is updated by weighted summation of neighbor node information.

Then, the parameter server architecture employs an embedded consistency synchronization mechanism. After local computation, each subgraph embeds the updated node into shared storage and pulls the latest embedded data in subsequent iterations, ensuring consistency of node representations during distributed training:

$$\mathbf{H}_t = \frac{1}{N} \sum_{v \in V} \mathbf{h}_t^{(v)} \tag{8}$$

Here, $\mathbf{H}_t$ denotes the node representations at time t in the global graph, and N is the total number of nodes. Equation (8) computes a global average aggregation of node representations to form a graph-level embedding. Distributed training consistency is handled separately via the parameter server–based embedding update mechanism described above.

IV. METHOD EVALUATION

A. EXPERIMENTAL DATA

The data used in the experiment comes from a smart government platform, covering information from multiple government departments, policy issues, users, and services, and is significantly heterogeneous. The dataset includes government graphs of varying sizes: basic business graph, policy service graph, extended service graph, full government graph, and ultra-large-scale government graph. Each graph scenario contains a varying number of nodes and edges. The dataset size ranges from 10,000 to 100,000 nodes, with the number of edges and graph complexity increasing with the number of nodes. The dataset also includes time series information for testing time series modeling. The features of each node contain multi-dimensional information, including user behavior data, service requirements, and policy implementation status. In the experiment, all data was preprocessed, node features were standardized, and time windows were partitioned for time series modeling. This data provided a realistic context for testing the model's performance in processing multi-source heterogeneous information and time series prediction tasks, effectively verifying the model's feasibility and superiority.

Table 1 summarizes the key parameter configurations of the heterogeneous information fusion model for the smart government platform, optimized using GraphSAGE. The neighbor sampling strategy uses a fixed sampling depth of 4 layers, sampling 20 neighbors per layer, and dynamically setting node type weights to balance heterogeneous semantics. The multi-head attention aggregation layer uses 8 attention heads, a hidden layer dimension of 128, and the activation function is LeakyReLU (Leaky Rectified Linear Unit). A 30% dropout is introduced to prevent overfitting. The training configuration uses the Adam optimizer (learning rate = 0.002), a batch size of 256, and The maximum number of training rounds is 200. This configuration serves as a baseline setting in the experiment, balancing model expressiveness and computational efficiency, and is suitable for processing large-scale heterogeneous graph data in government scenarios.

TABLE 1. Key Parameter Configurations for GraphSAGE Heterogeneous Information Fusion

Parameter Category	Parameter Name	Configuration Value	Remarks
Neighbor Sampling	Sampling Depth	4	Fixed 4-hop neighbor sampling
	Samples per Layer	20	Sample 20 neighbors per layer
	Node Type Weight	0–1.0	Type similarity coefficient
Multi-Head Attention	Attention Heads	8	8 independent attention heads
	Hidden Dimension	128	Feature dimension per head
	Activation Function	LeakyReLU	Negative slope = 0.2
	Dropout Rate	0.3	30% neuron dropout during training
Training Config	Learning Rate	0.002	Adam optimizer initial rate
	Batch Size	256	Process 256 nodes per batch
	Maximum Training Epochs	100	The maximum number of training rounds is 200.

B. NODE DEGREE DISTRIBUTION AND EDGE WEIGHTS IN MULTI-SOURCE HETEROGENEOUS DATA

Node analysis of multi-source heterogeneous data focuses on two aspects: node degree distribution and edge weight distribution. By analyzing the degree distribution of different node types, the strength of their interactions with other node types is assessed. The degree of each node is calculated to study the connection frequency and interaction patterns of nodes in the graph. Regarding the edge weight distribution, the importance and interaction strength of various relationships in the graph are assessed by measuring the weights of different edge types. During the weight calculation process, the weight of each edge is quantified based on the connection strength between nodes. These assessments can reveal the roles and functions of different nodes and edges in the smart government platform, providing a theoretical basis for optimizing information fusion strategies.

Figure 3 shows the degree distribution of node types ("User," "Event," "Department," "Policy") and the weight distribution of edge types in the smart government platform. Department (degree $\geq$ 24) and Policy (degree $\geq$

30) nodes exhibit higher degrees, reflecting their central roles in multi-party collaboration and information flow. In contrast, User and Event nodes have lower degrees, as their interactions rely more on Department and Policy intermediaries than on direct connections. Among edge types, Department–Policy edges have the highest weights ($\geq$ 0.8), underscoring their tight coupling in policy execution and feedback. User–Event and User–Policy edges have lower weights, indicating limited direct interaction and greater dependence on bridging nodes.

A contribution matrix, comprised of a multi-head attention mechanism, is constructed, recording the contributions of different numbers of attention heads (1 to 8) to four types of neighbors. The contribution value is defined as the average of the normalized attention weights assigned by each attention head to a certain type of neighbor. A type-aware neighbor sampling strategy is applied to differentiate information transfer and aggregation methods for different neighbor types. Each attention head performs weighted aggregation of semantic features from different neighbor types to generate a node representation. Figure 4 shows a heatmap of neighbor-type contributions to node representations under the multi-head attention mechanism: the x-axis is the number of attention heads (1–8), the y-axis lists neighbor types, and color intensity indicates contribution value. As head count increases, User ($0.5 \rightarrow 0.63$) and Event ($0.3 \rightarrow 0.52$) contributions rise steadily, reflecting enhanced semantic aggregation and finer-grained modeling of user behavior and event dynamics. In contrast, Department and Policy contributions grow slowly, suggesting their semantic signals are more dispersed and less task-relevant, making them harder for attention to exploit. This highlights the heterogeneity in information propagation and underscores the need for type-aware aggregation. While more heads enrich representations, gains saturate beyond a point—excessive heads increase complexity with diminishing returns, illustrating a clear trade-off between expressiveness and efficiency. The heatmap thus reveals the hierarchical, dynamic nature of heterogeneous information fusion in the model.

C. NODE INFORMATION REPRESENTATION CAPABILITIES UNDER OPTIMIZATION STRATEGIES

Several government graph scenarios were constructed with varying numbers of node types, simulating scenarios ranging from basic government services to highly complex, full-scale business systems. For each graph, the paper compared three strategies: sampling-only, type-aware sampling, and multi-head attention aggregation. Node information coverage is calculated by counting the proportion of neighbor types covered during aggregation, reflecting the model's ability to absorb multi-source information. Semantic preservation uses cosine similarity to measure the consistency of nodes in the semantic space before and after updates, assessing whether the aggregation process retains the original semantic features. Each strategy was tested using the same data structure and number of training rounds to ensure a fair comparison.

Figure 5 shows how node information coverage and semantic preservation vary with node type count, reflecting the impact of sampling and aggregation strategies on heterogeneous fusion. In Figure 5(a), sampling alone—ignoring type differences—leads to slow coverage growth in highly heterogeneous graphs. Type-aware sampling balances neighbor types, significantly boosting coverage as complexity rises. Multi-head attention further refines aggregation with type-specific weights, achieving near-0.9 coverage at 15 node types. In Figure 5(b), sampling-only causes semantic preservation to drop with more types, indicating poor semantic

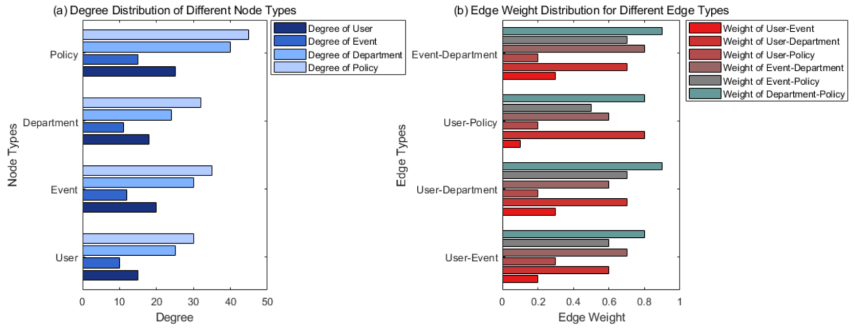

FIGURE 3. Degree Distribution of Different Node Types and Weight Distribution of Different Edge Types

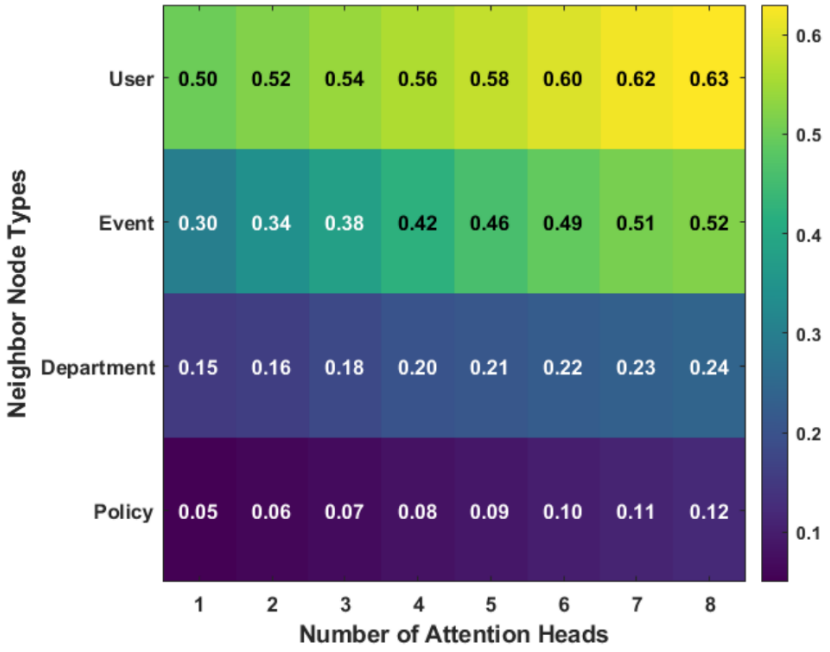

FIGURE 4. Multi-Head Attention Contribution Heatmap

retention. Type-aware sampling mitigates this by aligning representations with type semantics, and multihead attention maintains stable preservation through nonlinear cross-type modeling. Together, these strategies enhance the model's adaptability and expressiveness on complex government graphs.

D. NODE EMBEDDING ACCURACY AND F1-SCORE

The experimental task involves multi-category node classification, with labels derived from real-world business types of government entities (e.g., user roles, item categories, departmental levels, policy areas), and the categories are evenly distributed (the largest category accounts for $< 35\%$). The model's node embedding accuracy was tested on a government data set, and the fusion efficiency was evaluated using accuracy and F1-score as evaluation metrics. The GraphSAGE model, based on this paper, was compared with the popular GCN, GAT, Graph Transformer, GraphGAN (Graph Representation Learning with Generative Adversarial Nets), and the native GraphSAGE model under varying node complexity. Data is partitioned by timestamp: 70% of the earliest data is used for training, 15% of the intermediate time period is used for validation, and 15% of the latest data is used for testing, ensuring no time leakage; all samples from the same entity appear in only one set to avoid leakage within the same entity. These varying node complexity types were quantitatively represented using five levels of government graph scenarios: basic business graph, policy service graph, extended service graph, full government graph, and large-scale government graph. Figure 6 shows the accuracy and F1-score of different models across five graph scenarios of increasing complexity. Performance generally declines as node and edge heterogeneity grows, due to heightened learning and computational demands. The basic business graph—being simplest—yields the highest scores. In large-scale government graphs, massive data volume and complex temporal dependencies challenge most models. The proposed GraphSAGE excels, especially in complex

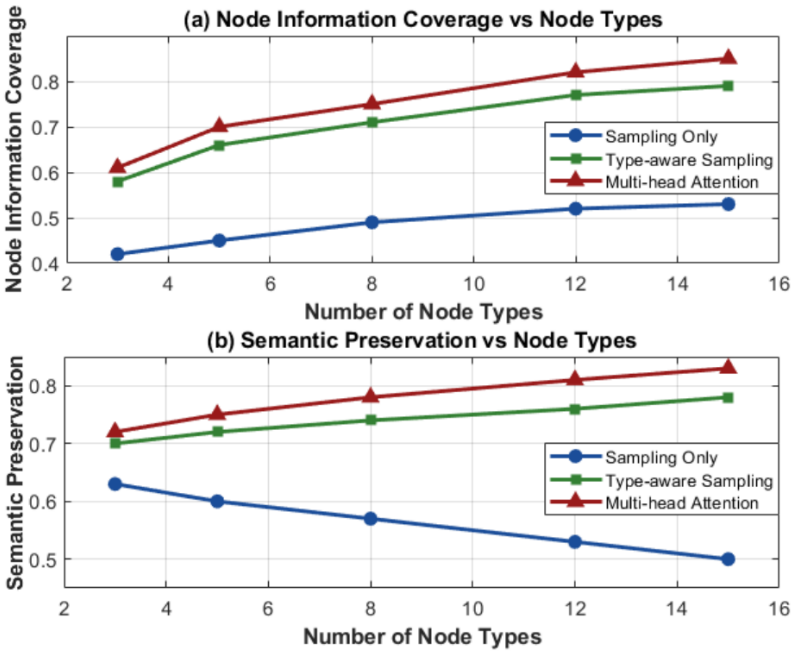

FIGURE 5. Information Coverage and Semantic Preservation

settings, achieving ≥0.95 accuracy and ≥0.94 F1-score—substantially outperforming native GraphSAGE. Other models perform well on simple graphs but degrade on large-scale data due to computational bottlenecks.

E. RECALL AND NORMALIZED MUTUAL INFORMATION (NMI) METRICS FOR HETEROGENEOUS INFORMATION FUSION

Recall and Normalized Mutual Information (NMI) are used as core evaluation metrics to assess the performance of various models in multi-source heterogeneous information fusion. Recall measures the proportion of relevant information correctly identified by the model, while NMI examines the consistency between the clustering results generated by the model and the ground-truth labels. A higher recall indicates a model's ability to extract relevant content from fused information; NMI is used to assess a model's ability to capture relationships between information; a larger value indicates a better model's ability to fuse heterogeneous information. The GraphSAGE model proposed in this paper and other popular comparative methods (GCN, GAT, Graph Transformer, GraphGAN, and native GraphSAGE) were evaluated on a unified dataset. The recall and NMI values for each model were calculated separately to compare the different models' clustering capabilities and information fusion performance when handling heterogeneous information.

Table 2 shows the performance of different graph neural network models using the recall and normalized mutual information (NMI) metrics, and also calculates the standard deviation of each metric. GraphSAGE performs excellently in both recall and NMI, with a mean recall of 0.91 (standard deviation 0.02) and a mean NMI of 0.88 (standard deviation 0.03). This demonstrates its high stability in heterogeneous information fusion tasks and its ability to maintain consistency across diverse datasets. Although Graph Transformer performed poorly in terms of recall and NMI, its standard deviation was relatively small, indicating that the model was able to perform relatively stable feature aggregation and information transfer when fusing heterogeneous information. In comparison, GCN and GAT, while demonstrating some performance in the task, had slightly larger standard deviations, particularly in NMI, reflecting some instability in the representation and fusion of heterogeneous information. GraphGAN and native GraphSAGE performed poorly in recall and NMI, with larger standard deviations, indicating high performance fluctuations when fusing heterogeneous information, likely due to the instability of their generative adversarial model training. GraphSAGE in this paper stands out for its efficient and stable feature fusion capabilities.

TABLE 2. Recall and Normalized Mutual Information

Model	Recall	Recall Standard Deviation	NMI	NMI Standard Deviation
Proposed GraphSAGE	0.91	0.02	0.88	0.03
GCN	0.85	0.04	0.81	0.05
GAT	0.87	0.03	0.83	0.04
Graph Transformer	0.89	0.02	0.85	0.02
GraphGAN	0.84	0.05	0.8	0.06
Native GraphSAGE	0.83	0.06	0.79	0.07

F. COMPUTATIONAL EFFICIENCY OF THE FUSION PROCESS

The computational efficiency of the fusion process was evaluated by recording the time required for each model to complete training and the number of training rounds

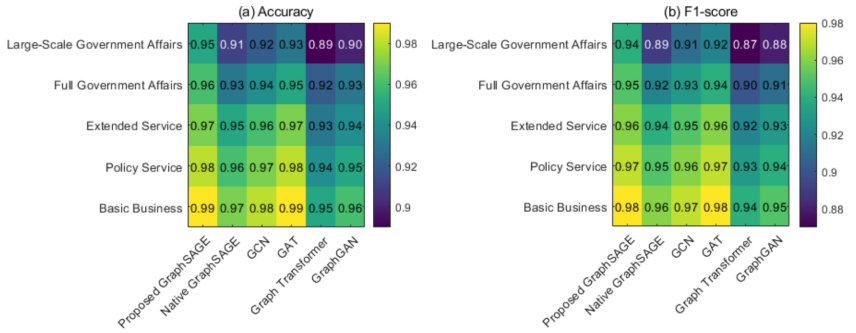

FIGURE 6. Node Embedding Accuracy and F1-score

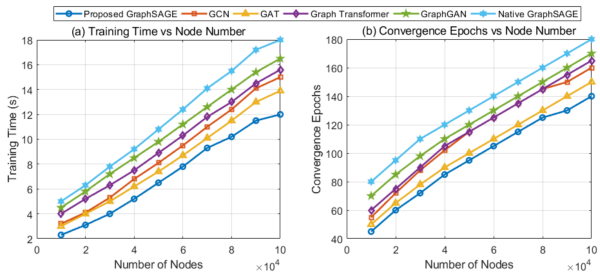

FIGURE 7. Computational Efficiency of the Fusion Process

required to reach convergence. The comparison method used GraphSAGE in this paper, along with the currently popular GCN, GAT, Graph Transformer, GraphGAN, and native GraphSAGE models, for testing at 10 different graph sizes (10k, 20k, 30k,

40k, 50k, 60k, 70k, 80k, 90k, and 100k nodes, respectively). All models were tested on a distributed cluster equipped with 8 NVIDIA V100 GPUs to ensure fair comparison. All models use the same graph partitioning strategy and parameter server architecture, and the training time includes communication and synchronization overhead.

Figure 7 shows that the proposed GraphSAGE achieves the shortest training time—just 12 seconds on 100k nodes—and converges in only 140 rounds, significantly outperforming other models in both efficiency and convergence speed. In contrast, GraphSAGE (Native), GraphGAN, and Graph Transformer suffer from much longer training times and slower convergence, especially at larger scales, highlighting their higher computational demands. The proposed model is thus well-suited for large-scale heterogeneous government graphs.

G. FORECAST ERROR AND CORRELATION COEFFICIENT FOR TIME SERIES MODELING

Mean Squared Error (MSE) and Pearson Correlation Coefficient were used to measure model performance across different time window lengths. During the evaluation process, the models were tested before and after incorporating the dynamic time series modeling mechanism to compare their performance across different time windows. The MSE and Pearson Correlation Coefficient for each model were calculated for several selected time windows, and their

TABLE 3. Dynamic Time Series Modeling Results.

Time Window (Steps)	MSE (w/o Dynamic Modeling)	MSE (with Dynamic Modeling)	Pearson Correlation (w/o Dynamic Modeling)	Pearson Correlation (with Dynamic Modeling)
5	0.021	0.016	0.912	0.946
10	0.034	0.027	0.876	0.922
15	0.045	0.036	0.854	0.909
20	0.054	0.042	0.832	0.898
25	0.061	0.048	0.811	0.887
30	0.069	0.053	0.792	0.875

[Note: "w/o" is an abbreviation of "without."]

trends over time window length were analyzed to assess the impact of introducing the dynamic time series modeling mechanism on forecast accuracy.

Table 3 shows that the dynamic time series modeling mechanism consistently reduces MSE and improves Pearson correlation across all time window lengths. MSE increases with longer horizons, but the mechanism lowers it to 0.053 at 30 steps, confirming its effectiveness in long-term forecasting. It achieves the best performance at 5 steps, with the lowest error and highest correlation, demonstrating strong short-term gains. Even as errors grow over longer horizons, the model consistently outperforms the baseline in both accuracy and correlation, highlighting the mechanism's sustained benefit in time series modeling.

V. CONCLUSIONS

This paper proposes an optimized GraphSAGE method to enhance fusion efficiency and time series modeling of multi-source heterogeneous nodes in smart government platforms—highly relevant to smart manufacturing in the industry, where it could fuse real-time sensor data with production schedules to predict bottlenecks. By constructing a government-adapted heterogeneous graph and improving neighbor sampling and aggregation, the model preserves semantic characteristics and boosts fusion quality. A dynamic time series mechanism enables evolving node representations, capturing temporal dependencies and improving prediction accuracy. Experiments show node embedding accuracy ≥ 0.95 and F1-score ≥ 0.94 on complex government graphs; with 100k nodes, training takes 12s over 140 convergence rounds. The method demonstrates strong scalability, efficiency, and stability, especially for large-scale, dynamic, heterogeneous data.

AUTHOR CONTRIBUTIONS

Mingyue Wang designed, collected and analyzed the data, and drafted the manuscript. Mingyue Wang conducted the study, critically revised the manuscript for important intellectual content, and gave final approval of the version to be published. Mingyue Wang participated fully in the work, take public responsibility for appropriate portions of the content, and agreed to be accountable for all aspects of the work in ensuring that questions related to the accuracy or integrity of any part of the work are appropriately investigated and resolved.

CONFLICTS OF INTEREST

The author declares no conflict of interest.

FUNDING

This research received no external funding.

ACKNOWLEDGEMENTS

Not applicable.

REFERENCES

[1] I. S. Kalganov, "Assessment of the results of the functioning of e-government and digitalization of public services," Intellect. Innovations. Investments, vol. (1), pp. 29-41, 2024, doi: 10.25198/2077-7175-2024-1-29.

[2] M. Di Giulio and G. Vecchi, "Implementing digitalization in the public sector," Technologies, agency, and governance. Public Policy and Administration, vol. 38, no. 2, pp. 133-158, 2023, doi: 10.1177/09520767211023283.

[3] D. Agostino, I. Saliterer, and I. Steccolini, "Digitalization, accounting and accountability: A literature review and reflections on future research in public services," Financial Accountability & Management, vol. 38, no. 2, pp. 152-176, 2022, doi: 10.1111/faam.12301.

[4] A. Androniceanu, I. Georgescu, and J. Kinnunen, "Public administration digitalization and corruption in the EU member states," A comparative and correlative research analysis. Transylvanian Review of Administrative Sciences, vol. 18, no. 65, pp. 5-22, 2022, doi: 10.24193/tras.65E.1.

[5] S. R. Chohan and G. Hu, "Strengthening digital inclusion through e-government: Cohesive ICT training programs to intensify digital competency," Information technology for development, vol. 28, no. 1, pp. 16-38, 2022, doi: 10.1080/02681102.2020.1841713.

[6] I. Dhaoui, "E-government for sustainable development: Evidence from MENA countries," Journal of the Knowledge Economy, vol. 13, no. 3, pp. 2070-2099, 2022, doi: 10.1007/s13132-021-00791-0.

[7] J. Lin, L. Carter, and D. Liu, "Privacy concerns and digital government: exploring citizen willingness to adopt the COVIDSafe app," European Journal of Information Systems, vol. 30, no. 4, pp. 389-402, 2021, doi: 10.1080/0960085X.2021.1920857.

[8] D. Špaček and Z. Špačková, "Issues of e-government services quality in the digital-by-default era–the case of the national e-procurement platform in Czechia," Journal of Public Procurement, vol. 23, no. 1, pp. 1-34, 2023, doi: 10.1108/JOPP-02-2022-0004.

[9] Y. Ma, Y. Qiu, W. Zhao, G. Li, M. Wu, and T. Jia, "DCIM-GCN: Digital computing-in-memory accelerator for graph convolutional network," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 71, no. 6, pp. 2735-2748, 2024, doi: 10.1109/TCSI.2024.3384748.

[10] P. Do and P. Pham, "Heterogeneous graph convolutional network pre-training as side information for improving recommendation," Neural Computing and Applications, vol. 34, no. 18, pp. 15945-15961, 2022, doi: 10.1007/s00521-022-07251-z.

[11] Y. Yang, "Research on the Innovation of "Last Kilometer" Delivery Mode of Rural E-commerce in Henan-Based on Dynamic Path Planning Technology (GAT)," European Journal of Business, Economics & Management, vol. 1, no. 1, pp. 8-16, 2025, doi: 10.71222/xqfpe028.

[12] W. Zhou, Z. Xia, P. Dou, T. Su, and H. Hu, "Double attention based on graph attention network for image multilabel classification," ACM Transactions on Multimedia Computing, Communications and Applications, vol. 19, no. 1, pp. 1-23, 2023, doi: 10.1145/3519030.

[13] S. Dhanasekaran, D. Gopal, J. Logeshwaran, N. Ramya, and A. O. Salau, "Multi-model traffic forecasting in smart cities using graph neural networks and transformer-based multi-source visual fusion for intelligent transportation management," International Journal of Intelligent Transportation Systems Research, vol. 22, no. 3, pp. 518-541, 2024, doi: 10.1007/s13177-024-00413-4.

[14] Y. Tang, Y. Huang, J. Hou, and Z. Liu, "Type-adaptive graph Transformer for heterogeneous information networks," Applied Intelligence, vol. 54, no. 22, pp. 11496-11509, 2024, doi: 10.1007/s10489-024-05793-4.

[15] J. Chen, T. Yu, Z. Pan, M. Zhang, G. Lu, and K. Zhu, "Stochastic dynamic power dispatch with human knowledge transfer using graph-GAN assisted inverse reinforcement learning," IEEE Transactions on Smart Grid, vol. 15, no. 3, pp. 3303-3315, 2023, doi: 10.1109/TSG.2023.3329459.

[16] Z. Guo, K. Yu, A. Jolfaei, A. K. Bashir, A. O. Almagrabi, and N. Kumar, "Fuzzy detection system for rumors through explainable adaptive learning," IEEE Transactions on Fuzzy Systems, vol. 29, no. 12, pp. 3650-3664, 2021, doi: 10.1109/TFUZZ.2021.3052109.

[17] S. Li, N. A. Zaidi, M. Du, Z. Zhou, H. Zhang, and G. Li, "Property graph representation learning for node classification," Knowledge and Information Systems, vol. 66, no. 1, pp. 237-265, 2024, doi: 10.1007/s10115-023-01963-x.

[18] Z. Zhong, C. T. Li, and J. Pang, "Personalised meta-path generation for heterogeneous graph neural networks," Data Mining and Knowledge Discovery, vol. 36, no. 6, pp. 2299-2333, 2022, doi: 10.1007/s10618-022-00862-z.

[19] M. Li, K. Liu, H. Liu, Z. Zhao, T. E. Ward, and X. Wu, "Heterogeneous meta-path graph learning for higher-order social recommendation," ACM Transactions on Knowledge Discovery from Data, vol. 18, no. 8, pp. 1-25, 2024, doi: 10.1145/3673658.

[20] R. Bing, G. Yuan, M. Zhu, F. Meng, H. Ma, and S. Qiao, "Heterogeneous graph neural networks analysis: a survey of techniques, evaluations and applications," Artificial Intelligence Review, vol. 56, no. 8, pp. 8003-8042, 2023, doi: 10.1007/s10462-022-10375-2.

[21] D. Cai, S. Qian, Q. Fang, J. Hu, and C. Xu, "User cold-start recommendation via inductive heterogeneous graph neural network," ACM Transactions on Information Systems, vol. 41, no. 3, pp. 1-27, 2023, doi: 10.1145/3560487.