Appendix

Notations

Table 4 lists our frequently used notations.

Prominence Modeling

In this section, we present the intuition for introducing the prominence Modeling in the node-scale and feature-scale graph structures learning process in detail.

We first select 10% observations of the original Dutch-Wind (2023) dataset as the missing values by MCAR mechanism (Bohannon et al. 2005). Then we use cross-temporal self-attention to capture the temporal dependencies and then use cross-feature self-attention to capture the spatial dependencies, based on the setup of the "TemporalFeatureRL" scenario in the Ablation Study section. After that, we use the above architecture to impute missing values. Furthermore, we extract the attention map obtained from cross-feature self-attention after training. Note that this setup is consistent with the survey addressed in Figure 1(c) and Figure 1(d) in the Introduction section.

According to the attention map from the cross-feature self-attention mechanism, we first compute the average attention scores of different stations of each feature. As shown in Figure 4, the average attention score for different stations is different. This suggests the influence for the overall imputation task is different for different stations in the node-scale meta-graph corresponding to each feature.

Then we compute the average attention scores of different features in a station. According to Figure 5, we can find that the average attention score for different features is different. Therefore, the influence for the overall imputation task should be different in the feature-scale meta-graph structure.

The above observation inspires us to perform prominence modeling for the node-scale graph structure learning and feature-scale graph structure learning, as presented in Equation 1, Equation 2, Equation 6, and Equation 7. The process first obtains the corresponding prominence vector self-adaptively based on the input source embedding. And then apply the prominence vector to the source embedding through the Hadamard product to make it enhanced or weakened.

Proofs

Proof of Proposition 1 The canonical graph diffusion convolution treats all features on a node as a uniform node embedding. Thus, its convolution operation can be expressed as:

$$\mathbf{R^{DC}} = \Sigma_{k=0}^{K} \left[\dot{\mathbf{A}}^{\mathbf{\Omega}} \mathbf{R} \mathbf{\Theta}_k^{\mathbf{\Omega 1}} + \left(\mathbf{D^{O}}^{-1} \mathbf{A} \right)^k \mathbf{R} \mathbf{\Theta}_k^{\mathbf{\Omega 2}} + \left(\mathbf{D^{I}}^{-1} \mathbf{A}^\top \right)^k \mathbf{R} \mathbf{\Theta}_k^{\mathbf{\Omega 3}} \right],$$

where $\dot{\mathbf{A}}^{\mathbf{\Omega}} \in \mathbb{R}^{N \times N}$ is the learned global graph structure, $\mathbf{R} \in \mathbb{R}^{N \times T \times FC}$ is the input graph signal. $\mathbf{\Theta}_k^{\mathbf{\Omega 1}}, \mathbf{\Theta}_k^{\mathbf{\Omega 2}}, \mathbf{\Theta}_k^{\mathbf{\Omega 3}} \in \mathbb{R}^{FC \times FC}$ are graph convolution kernels.

Symbol	Description
$\mathbf{X}$	incomplete temporal signal of spatial-temporal data with N nodes, T timestamps and F features
$\mathcal{G}$	spatial graph of spatial-temporal data
$\mathbf{M}$	mask matrix indicating the missing status of temporal signal $\mathbf{X}$
$\mathbf{A}$	adjacency matrix of the graph $\mathcal{G}$
$\mathbf{R}$	output of the cross-temporal representation learning module
$\mathbf{R^{NL}}$	output of the node-scale spatial learning module
$\mathbf{R^{FL}}$	output of the feature-scale spatial learning module
$\overline{\mathbf{X}}$	output of GSLI framework
$\mathbf{\Omega_f^1}, \mathbf{\Omega_f^2}$	meta node embeddings of feature f
$\dot{\mathbf{A}}_f^{\mathbf{\Omega}}$	adjacency matrix of node-scale meta graph $\dot{\mathcal{G}}_f^{\mathbf{\Omega}}$ for feature f
$\mathbf{\Phi^1}, \mathbf{\Phi^2}$	meta feature embeddings across all nodes
$\dot{\mathbf{A}}^{\mathbf{\Phi}}$	adjacency matrix of feature-scale meta graph $\dot{\mathcal{G}}^{\mathbf{\Phi}}$

Table 4: Notations

Considering the feature heterogeneity, then there exists at least one node i satisfying the correlation weight of feature f_1 from node j to node i is x, and the correlation weight of feature f_2 from node j to node i is y, where $x \neq y$. The expected result $\mathbf{B}$ obtained by multiplying $\dot{\mathbf{A}}^{\mathbf{\Omega}}$ with $\mathbf{R}$ in the first term of the first order graph diffusion convolution, the corresponding result $b_{i,t,f_2,c} \in \mathbf{B}$ of the channel c for f_2 feature at timestamp t for the node i is given by:

$$(a_{i1}^{\mathbf{\Omega}} r_{1,f_2,c} + \cdots + y r_{j,f_2,c} + \cdots + a_{iN}^{\mathbf{\Omega}} r_{N,f_2,c}),$$

where $a_{i1}^{\mathbf{\Omega}}, a_{iN}^{\mathbf{\Omega}}, \in \dot{\mathbf{A}}^{\mathbf{\Omega}}$, $r_{1,f_2,c}, r_{N,f_2,c}, r_{j,f_2,c} \in \mathbf{R}$, y is the expected value for $a_{ij}^{\mathbf{\Omega}} \in \dot{\mathbf{A}}^{\mathbf{\Omega}}$.

Since the canonical graph diffusion convolution operation has only one learned graph structure, if we assume that $\dot{\mathbf{A}}^{\mathbf{\Omega}}$ satisfies the global correlation for f_1, then the actual information flow to form the $b_{i,t,f_2,c} \in \mathbf{B}$ is:

$$b_{i,t,f_2,c} = a_{i1}^{\mathbf{\Omega}} r_{1,f_2,c} + \cdots + x r_{j,f_2,c} + \cdots + a_{iN}^{\mathbf{\Omega}} r_{N,f_2,c},$$

which is in conflict with our expected result. Similarly, if we assume that $\dot{\mathbf{A}}^{\mathbf{\Omega}}$ satisfies the global correlation for f_2, the information flow to form the $b_{i,t,f_1,c} \in \mathbf{B}$ will conflict with our expected result. Following the same line, we can show that neither the second nor the third term of the graph diffusion convolution addresses the feature heterogeneity.

Proof of Proposition 2 Similar to Proposition 1, we assume that there exists at least one node i satisfying the correlation weight of feature f_1 from node j to node i is x, and the correlation weight of feature f_2 from node j to node i is y for the feature heterogeneity problem, where $x \neq y$.

According to Equation 3, we learn independent graph structures for each heterogeneous feature. Thus, we suppose that $a_{f_1,i,j}^{\mathbf{\Omega}} = x$, $a_{f_2,i,j}^{\mathbf{\Omega}} = y$, where $a_{f_1,i,j}^{\mathbf{\Omega}} \in \dot{\mathbf{A}}_{f_1}^{\mathbf{\Omega}}$ and $a_{f_2,i,j}^{\mathbf{\Omega}} \in \dot{\mathbf{A}}_{f_2}^{\mathbf{\Omega}}$. If the first term of the first order graph

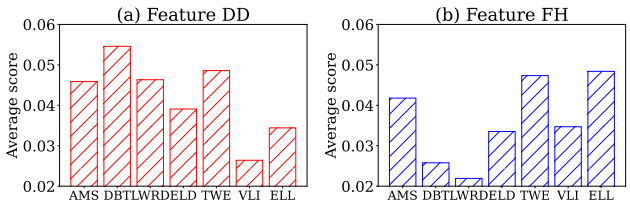


Figure 4: Average attention scores of different stations from cross-feature self-attention mechanism

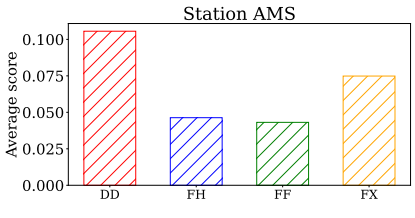


Figure 5: Average attention scores of different features from cross-feature self-attention mechanism

convolution operation shown in Equation 3 operates on the channel c of feature f_2, the result obtained by multiplying $\dot{\mathbf{A}}^{\mathbf{\Omega}}_{f_2}$ with $\mathbf{R}_{f2}$ is $\mathbf{B}_{f2}$. The corresponding result of $\mathbf{B}_{f_2}$ at timestamp t for the node i can be obtained by:

$$b_{f_2,i,t} = a^{\mathbf{\Omega}}_{f_2,i,1} r_{f_2,1} + \cdots + y r_{f_2,j} + \cdots + a^{\mathbf{\Omega}}_{f_2,i,N} r_{f_2,N}.$$

On the other hand, the corresponding result of feature f_1 at timestamp t for the node i can be obtained by:

$$b_{f_1,i,t} = a^{\mathbf{\Omega}}_{f_1,i,1} r_{f_1,1} + \cdots + x r_{f_1,j} + \cdots + a^{\mathbf{\Omega}}_{f_1,i,N} r_{f_1,N}.$$

It can be seen that the heterogeneity of f_1 and f_2 does not affect the correct information flow in our node-scale spatial learning process for the timestamp t. According to the above process, we can generalize the above result for all heterogeneous feature pairs and all timestamps.

Complexity Analysis

Time complexity of Node-scale Spatial Learning Consider the node-scale spatial learning for feature f, the complexity to get the prominence vector $\mathbf{P}^{\mathbf{\Omega}}_f$ is $\mathcal{O}(Nd^2)$. Then the complexity to obtain the meta-graph $\dot{\mathcal{G}}^{\mathbf{\Omega}}_f$ is $\mathcal{O}(N^2d + Nd^2)$. For the graph diffusion convolution shown in Equation 4, the complexity of the three convolution terms is both $\mathcal{O}(N^2TC + NTC^2)$. Since the graph diffusion convolution step K is a smaller hyperparameter, the overall time complexity of Equation 4 is $\mathcal{O}(N^2TC + NTC^2 + N^2d + Nd^2)$. Finally, Equation 5 concatenates the results obtained on the F features, so the overall time complexity is $\mathcal{O}(FN^2TC + FNTC^2 + FN^2d + FNd^2)$.

Space complexity of Node-scale Spatial Learning For the spatial learning on feature f, the complexity of Equation 3 is $\mathcal{O}(Nd + d^2)$, thus the complexity of obtaining the meta-graph is $\mathcal{O}(N^2 + Nd + d^2)$. And the graph diffusion convolution brings a complexity of $\mathcal{O}(NTC + C^2)$. Since we need to concatenate all the features, the overall space complexity is $\mathcal{O}(FNTC + FC^2 + FN^2 + FNd + Fd^2)$.

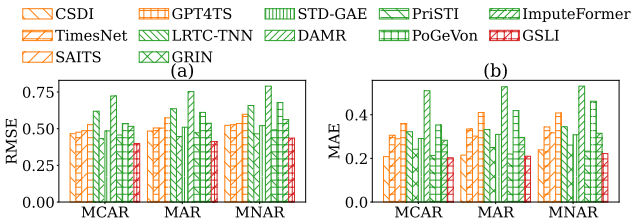


Figure 6: Varying the missing mechanism over BeijingMEO dataset with 10% missing values

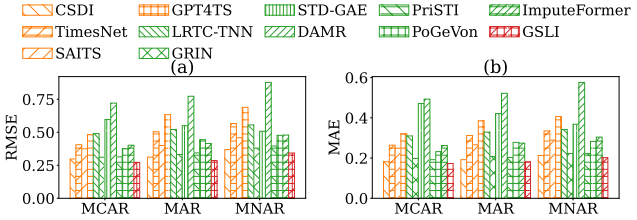


Figure 7: Varying the missing mechanism over LondonAQ dataset with 10% missing values

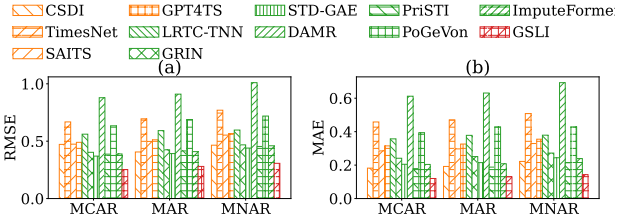


Figure 8: Varying the missing mechanism over CN dataset with 10% missing values

Time complexity of Feature-scale Spatial Learning First, we need to get the adjacency matrix $\dot{\mathbf{A}}^{\mathbf{\Phi}}$ of the meta-graph which represents the spatial correlation of different features, the time complexity of this operation is $\mathcal{O}(F^2d + Fd^2)$. Then the time complexity of the diffusion convolution layer shown in Equation 10 is $\mathcal{O}(F^2NTC + FNTC^2)$. Therefore, the overall time complexity is $\mathcal{O}(F^2NTC + FNTC^2 + F^2d + Fd^2)$.

Space complexity of Feature-scale Spatial Learning Similar to the node-scale spatial learning, the complexity of obtaining the meta-graph $\dot{\mathcal{G}}^{\mathbf{\Phi}}$ is $\mathcal{O}(F^2 + Fd + d^2)$. The space complexity of graph diffusion convolution is $\mathcal{O}(FNTC + C^2)$. The overall space complexity is $\mathcal{O}(FNTC + C^2 + F^2 + Fd + d^2)$.

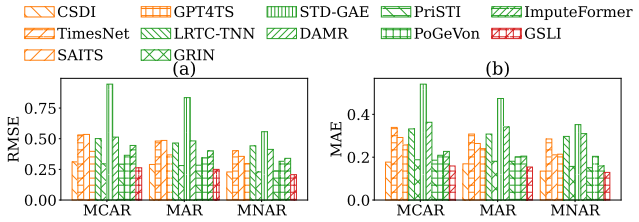


Figure 9: Varying the missing mechanism over Los dataset with 10% missing values

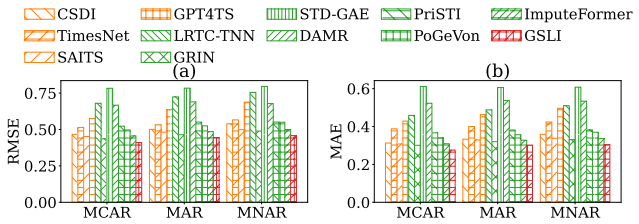


Figure 10: Varying the missing mechanism over LuohuTaxi dataset with 10% missing values

Supplementary of experiments

Dataset Details DutchWind (2023) gathers wind speed and direction data from 7 stations in the Netherlands hourly from January 1 to December 28, 2023. BeijingMEO (2018) contains meteorological data recorded at 18 locations in Beijing from January 30, 2017, to January 31, 2018, collected hourly. LondonAQ (2018) collects hourly air quality readings from January 1, 2017 to March 31, 2018 at 13 locations in London. CN (2014) contains hourly air quality data at 140 stations in China from October 1, 2014, to December. 31, 2014. Each station records six features: PM2.5, PM10, NO2, CO, O3, and SO2. Los (2020) contains average traffic speeds per five-minute period from March 1, 2012 to March 7, 2012 at different locations on Los Angeles highways. LuohuTaxi (2020) records the average speed of taxis every 15 minutes on different major roads in Luohu District, Shenzhen in January 2015.

Missing Mechanisms In this section, we provide the imputation performance with various missing mechanisms for all datasets.

The occurrence of missing data in real-world scenarios is usually related to the external environment or the sensors themselves. Therefore, we consider three missing mechanisms: missing completely at random (MCAR) (Bohannon et al. 2005), missing at random (MAR) (Xia et al. 2017) and missing not at random (MNAR) (Twala 2009). For MCAR, the missing value is not related to other attributes, and each observation has an equal chance of being missing. MAR potentially implies that the missing status of all features depends on the frequency of a particular feature. For example, missing records of traffic flow data may be related to rush hour and activities in public places. For MNAR, the missing observation depends on the feature itself, for example, the reliability of recording devices. As in the Imputation Comparison section, we repeat each experiment 5 times for each experiment and report the average results. It should be noted that in all experiments we used different random seeds in our 5 replications, i.e. from 3407 to 3411. Since the random seeds are different, the imputation labels selected in the five replications will also be different.

As shown in Figure 3, Figure 6, Figure 7, Figure 8, Figure 9, and Figure 10, our GSLI method consistently delivers optimal results across different missing mechanisms. It demonstrates that our proposed GSLI is effectively adapted to various missing scenarios in reality. Furthermore, it is important to note that when it comes to the MNAR mechanism, most imputation methods tend to perform slightly worse than the other mechanisms across the majority of datasets. This is because MNAR tends to remove observations in unconventional statuses.

Mask Strategies of Training Label For training our GSLI framework, we randomly select some observations as the training label. In this process, the mask ratio and mask pattern to get the training label directly determine the effectiveness of training. Thus, in this section, we explore the performance with different mask ratios and mask patterns.

We first mask different ratios of observations as the training label, the performance of GSLI with different mask ratios is shown in Table 5. We can find that the model performs better when the mask ratio is set to 0.2 or 0.3 in most cases. The training will be easy to converge when the mask ratio is too low, and therefore the model will not be able to accurately learn the the required dependencies for imputation. On the contrary, when the mask ratio is too large, it is difficult for the model to mine valid dependencies from training. Therefore, we set the mask ratio to 0.2 by default.

Then we explore the performance when using different mask patterns. Following CSDI (Tashiro et al. 2021) and PriSTI (Liu et al. 2023a), we consider three mask pattern strategies: (1) Block missing (2) Historical missing (3) Random missing. For the "Historical missing" scenario, the mask at the current timestamp has a half probability of being the same as the previous timestamp, and a half probability of performing Random missing. As shown in Table 6, the Block missing or Historical missing strategy is not directly comparable to Random missing in most cases. This is because there is a possibility that the mask pattern may not accurately correspond to the actual missing scenario. Therefore, we default to using the Random missing mask pattern. It should be noted that by comparing the results presented in Table 2, it can be found that benefiting from the learning of multi-scale graph structures, our method consistently outperforms other methods when adjusting the mask strategies in all of them.

Mask Ratio	DutchWind		BeijingMEO		LondonAQ		CN		Los		LuohuTaxi	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
10%	0.4143	0.2080	0.4114	0.2153	0.3034	0.1963	0.2582	0.1233	0.2697	0.1622	0.4186	0.2841
20%	**0.4101**	0.2051	**0.3986**	**0.2034**	0.2720	0.1730	0.2534	**0.1202**	**0.2632**	**0.1592**	**0.4102**	**0.2761**
30%	0.4114	**0.2038**	**0.3986**	**0.2034**	0.2718	0.1731	**0.2520**	0.1203	0.2635	0.1612	0.4147	0.2819
40%	0.4226	0.2124	0.3995	0.2051	**0.2708**	**0.1728**	0.2530	0.1207	0.2651	0.1616	0.4174	0.2836
50%	0.4318	0.2179	0.4022	0.2142	0.2800	0.1752	0.2563	0.1227	0.2704	0.1649	0.4183	0.2851

Table 5: Varying the mask ratio of the training label for various datasets with 10% missing values

Mask Pattern	DutchWind		BeijingMEO		LondonAQ		CN		Los		LuohuTaxi	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
Block missing	0.4150	0.2057	0.4089	0.2117	0.3328	0.2197	0.2887	0.1447	0.2759	0.1674	0.4748	0.3209
Historical missing	**0.4101**	**0.2041**	0.4013	0.2057	0.3068	0.1989	0.2786	0.1385	0.2636	0.1596	0.4621	0.3080
Random misssing	**0.4101**	0.2051	**0.3986**	**0.2034**	**0.2720**	**0.1730**	**0.2534**	**0.1202**	**0.2632**	**0.1592**	**0.4102**	**0.2761**

Table 6: Varying the mask pattern of the training label for various datasets with 10% missing values

Methods	Parameters	GPU Memory Usage (MiB)	Time Cost(s)
CSDI	413441	1888	87.78
TimesNet	713692	604	26.25
SAITS	1362896	1044	1.42
GPT4TS	60736540	1386	89.11
LRTC-TNN	-	-	4.96
GRIN	24101	1950	126.01
STD-GAE	3803	432	212.38
DAMR	2977	23080	5518.04
PriSTI	729202	2266	1129.58
PoGeVon	329616	2868	67.09
ImputeFormer	798585	2934	23.70
w/o Feature-Split&Scale	4281781	2354	47.70
GSLI	4478913	2464	47.90

Table 7: Resource consumption over DutchWind dataset with 10% missing values

Resource Consumption In this section, we report the resource consumption in Table 7 for our GSLI and existing methods. In this table, we also consider another ablation scenario "w/o FeatureSplit&Feature-scale". As described in Ablation Study Section, we replace our node-scale Spatial Learning and feature-scale Spatial Learning with the canonical graph diffusion convolution (Li et al. 2018; Wu et al. 2019).

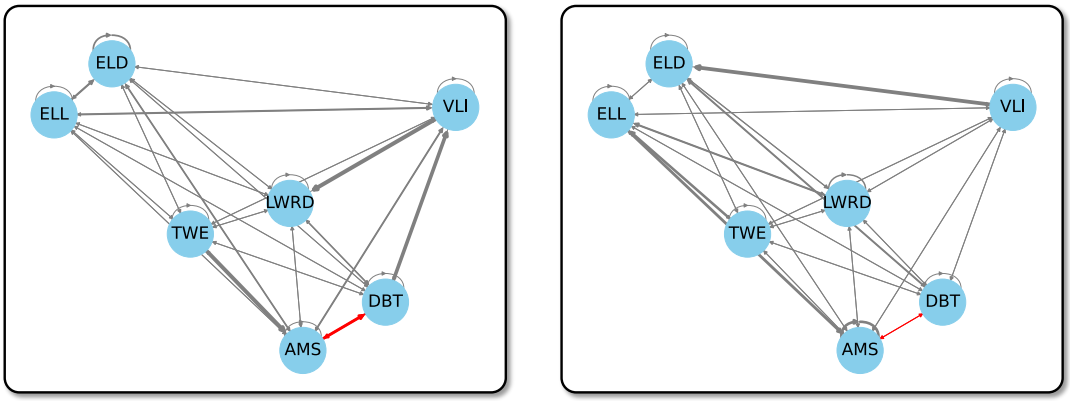
We can find that the overall resource consumption of our method is in the same order of magnitude as the end-to-end deep learning imputation methods. Note that LRTC-TNN is based on low-rank tensor completion with no learnable parameters and no need to utilize GPU training. Since the canonical graph diffusion convolution treats all features on a node as a uniform node embedding, the time complexity is $\mathcal{O}(FN^2TC + F^NTC^2 + Nd^2)$, the space complexity is $\mathcal{O}(FNTC + F^2C^2)$. Thus, our resource consumption is slightly higher than the canonical graph diffusion convolution. However, as illustrated in Table 3, our method can produce better imputation results than the above ablation scenario. Therefore, we believe that it is acceptable to pay a small amount of additional resource consumption.

Graph Structure Visualisation In this section, we visualize the graph structures we have learned over the DutchWind dataset with 10% missing rate.

First, we present the learned node-scale meta-graphs for heterogeneous features from our proposed GSLI model, as illustrated in Figure 11. In the presented graphs, thicker edges represent higher edge weights, i.e., more spatial correlations are learned. As mentioned in the Introduction section, feature DD captures wind direction data and FH is related to wind speed. As shown in Figure 1(a), AMS and DBT stations are geographically close to each other. Therefore, there is a stronger spatial correlation between the nodes corresponding to AMS and DBT in the node-scale meta-graph corresponding to the feature DD that records the wind direction. In contrast, due to the large difference in emptiness degrees in the vicinity of AMS and DBT stations, the node-scale meta-graph corresponding to the feature FH for recording wind speeds does not have a strong correlation between the two nodes. This result shows that our GSLI can learn different global node-scale graph structures for features from different domains in response to feature heterogeneity, which also provides an empirical evidence for Proposition 1.

Then we present the learned feature-scale meta-graph, which represents the common spatial correlation of different features over all nodes learned by GSLI. As shown in Figure 12, there is a stronger spatial correlation between FH and FF in the feature-scale meta-graph. Given the strong correlation between features FH and FF, i.e. FH records the hourly average wind speed and FF records the average wind speed in the last 10 minutes of the past hour, it can be shown that GSLI can capture the common spatial correlations of different features over all nodes through the feature-scale graph structure learning.

Application Study In this section, we validate different imputation methods for the downstream tasks on the original incomplete LondonAQ dataset. We first impute the missing values through various imputation methods. Then, we eval-



(a) Learned node-scale meta-graph for Feature DD (b) Learned node-scale meta-graph for Feature FH

Figure 11: Visualisation of the learned node-scale meta-graphs

Method	LondonAQ		LuohuTaxi	
	RMSE	MAE	RMSE	MAE
TimesNet	0.886	0.648	0.372	0.272
GPT4TS	0.885	0.666	0.371	0.266
DCRNN	0.948	0.738	0.764	0.596
GWN	0.967	0.739	0.402	0.300
GTS	0.866	0.665	0.490	0.377
MegaCRN	1.075	0.783	0.373	0.274
CrossGNN	1.052	0.736	0.566	0.423
GSLI	**0.806**	**0.600**	**0.368**	**0.267**

Table 8: Forecasting performance of GSLI compared to existing methods over real missing datasets

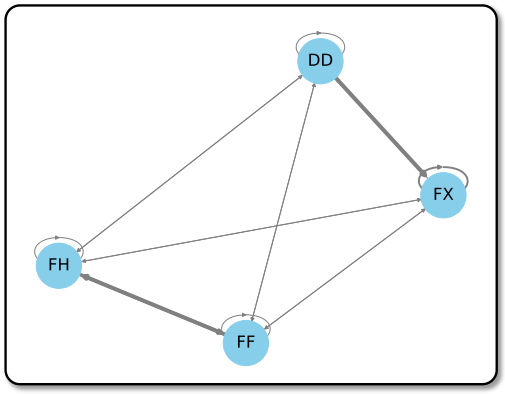


Figure 12: Visualisation of the learned feature-scale meta-graph

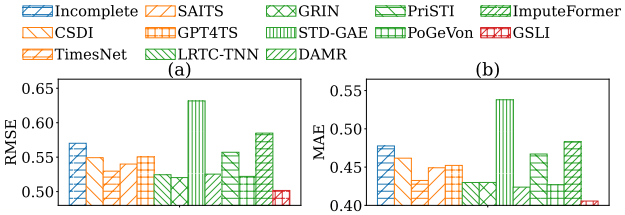


Figure 13: Downstream application results of air quality forecasting over LondonAQ dataset

uate the accuracy of air quality forecasts following the same line of existing study (Luo et al. 2019). To be more specific, we use the Adaboost implementation (Pedregosa et al. 2011) to forecast the average PM2.5 concentration at the CD1 station for the next six hours data according to the data from all stations during a 12-hour period. According to Figure 13, most methods with higher imputation accuracies tend to have better air quality forecasting performance. Meanwhile, our method still obtains the best forecasting result, validating its applicability.

Spatial-Temporal Forecasting In this section, we directly apply GSLI for downstream spatial-temporal forecasting task on datasets with missing values. We utilize 96 historical timestamps to predict 48 future timestamps for the LondonAQ and LuohuTaxi datasets. To meet the prediction requirements, we introduce an additional linear layer that adjusts the output dimensions to match the forecasting length.

As demonstrated in Table 8, GSLI outperforms existing state-of-the-art methods for spatial-temporal forecasting in situations with missing data. This advantage is due to GSLI's ability to model complex spatial relationships and adapt its learning of node-scale and feature-scale graph structures to account for feature heterogeneity. In missing scenarios, temporal dependencies are modelled less accurately, but other methods cannot accurately model spatial dependencies, which affects their forecasting performance. The results confirm GSLI's superior performance and highlight its robustness in spatial-temporal forecasting when dealing with missing data scenarios.

Statistical Analysis In this section, we first conduct a t-test based on the experimental results shown in Table 2. As shown in Table 9, GSLI performs significantly better than various baselines in $521/528 \approx 98.77\%$ cases under the t-test threshold ($p < 0.05$). Then we report the the average improvement percentage of the imputation performance between existing methods and GSLI based on Table 2, and the results are shown in Table 10. The results show that GSLI shows at least an average improvement of **10.81%** in performance metrics compared to the second-best results.

Dataset	Missing rate	Metric	CSDI	TimesNet	SAITS	GPT4TS	LRTC-TNN	GRIN	STD-GAE	DAMR	PriSTI	PoGeVon	ImputeFormer	GSLI
DutchWind	10%	RMSE	**3.4E-05**	**4.1E-09**	**7.4E-07**	**1.0E-10**	**2.2E-08**	**1.6E-04**	**5.4E-07**	**4.8E-10**	**6.6E-07**	**1.3E-02**	**2.4E-08**	-
		MAE	**4.5E-05**	**2.2E-12**	**3.0E-08**	**2.7E-13**	**1.2E-10**	**8.2E-08**	**3.5E-10**	**3.7E-10**	**5.5E-05**	**3.0E-02**	**1.0E-10**	-
	20%	RMSE	**1.6E-04**	**3.9E-11**	**2.1E-05**	**1.5E-12**	**3.6E-09**	**7.4E-04**	**1.4E-07**	**1.6E-08**	**1.1E-07**	**1.5E-02**	**3.2E-07**	-
		MAE	**3.2E-04**	**2.7E-14**	**1.7E-07**	**8.2E-17**	**2.7E-11**	**2.5E-08**	**6.8E-12**	**3.8E-10**	**2.4E-05**	**1.6E-02**	**5.8E-09**	-
	30%	RMSE	**8.3E-06**	**1.3E-12**	**1.2E-06**	**1.3E-12**	**5.2E-11**	**1.2E-02**	**2.6E-08**	**1.3E-08**	**1.9E-06**	**3.8E-02**	**1.5E-07**	-
		MAE	**1.0E-04**	**1.4E-15**	**1.1E-10**	**1.0E-15**	**4.6E-11**	**2.8E-06**	**3.6E-11**	**2.4E-11**	**3.4E-05**	**4.4E-02**	**1.3E-09**	-
	40%	RMSE	**2.9E-04**	**5.0E-13**	**1.8E-06**	**4.0E-13**	**2.8E-11**	1.5E-01	**4.8E-09**	**3.4E-08**	**6.5E-05**	7.1E-02	**3.2E-07**	-
		MAE	**8.2E-04**	**2.8E-14**	**1.2E-07**	**1.6E-14**	**3.7E-11**	**2.2E-03**	**3.8E-10**	**2.2E-10**	**3.4E-03**	6.2E-02	**1.5E-08**	-
BeijingMEO	10%	RMSE	**6.5E-08**	**2.9E-08**	**8.3E-09**	**6.6E-10**	**2.8E-11**	**1.9E-05**	**9.3E-09**	**1.1E-10**	**6.0E-07**	**1.0E-07**	**2.1E-08**	-
		MAE	2.2E-01	**1.7E-09**	**1.4E-08**	**2.2E-10**	**8.7E-10**	**6.2E-06**	**5.4E-09**	**5.1E-09**	**4.7E-02**	**2.7E-07**	**9.0E-08**	-
	20%	RMSE	**9.3E-09**	**2.3E-10**	**1.8E-09**	**4.9E-12**	**8.4E-12**	**3.8E-06**	**1.5E-09**	**3.8E-14**	**2.1E-07**	**2.5E-06**	**9.1E-10**	-
		MAE	**5.7E-02**	**3.4E-11**	**6.0E-09**	**2.3E-12**	**1.9E-10**	**3.4E-06**	**2.8E-09**	**1.3E-11**	**4.5E-03**	**5.4E-05**	**2.1E-08**	-
	30%	RMSE	**2.1E-08**	**2.9E-11**	**1.4E-08**	**2.0E-12**	**2.1E-12**	**2.4E-05**	**8.0E-09**	**3.4E-12**	**9.3E-06**	**4.3E-06**	**1.9E-08**	-
		MAE	**2.5E-02**	**9.3E-12**	**2.5E-08**	**9.1E-13**	**2.3E-10**	**1.3E-05**	**1.7E-08**	**1.3E-10**	**1.1E-03**	**5.4E-05**	**1.0E-07**	-
	40%	RMSE	**8.8E-09**	**6.9E-13**	**4.4E-09**	**1.9E-13**	**1.7E-13**	**1.1E-05**	**2.8E-09**	**5.2E-11**	**5.6E-03**	**4.3E-04**	**5.6E-08**	-
		MAE	**8.5E-04**	**2.1E-13**	**5.3E-09**	**4.5E-14**	**1.5E-11**	**2.0E-06**	**3.6E-09**	**8.0E-10**	**2.5E-04**	**4.7E-03**	**5.5E-08**	-
LondonAQ	10%	RMSE	**2.0E-03**	**2.1E-08**	**1.2E-07**	**2.0E-09**	**1.5E-07**	**1.8E-04**	**1.2E-11**	**4.7E-10**	**3.8E-04**	**8.8E-07**	**7.3E-08**	-
		MAE	**3.1E-02**	**4.0E-09**	**4.3E-08**	**2.6E-10**	**3.0E-10**	**8.7E-05**	**1.2E-12**	**3.3E-12**	**1.3E-03**	**2.3E-07**	**2.0E-08**	-
	20%	RMSE	2.1E-01	**4.9E-09**	**4.0E-05**	**4.3E-10**	**1.8E-08**	**3.3E-02**	**3.6E-10**	**4.0E-11**	**1.1E-02**	**3.5E-05**	**9.9E-06**	-
		MAE	5.6E-01	**4.3E-11**	**6.5E-07**	**1.9E-12**	**1.1E-10**	**1.2E-03**	**9.2E-13**	**1.6E-10**	**5.7E-03**	**3.0E-07**	**1.4E-07**	-
	30%	RMSE	**2.4E-02**	**7.6E-12**	**4.8E-07**	**2.7E-12**	**2.3E-11**	**1.7E-03**	**3.1E-05**	**2.5E-10**	**3.8E-02**	**3.8E-06**	**7.7E-07**	-
		MAE	**1.2E-01**	**7.7E-12**	**1.6E-07**	**1.5E-12**	**2.0E-10**	**1.1E-03**	**9.1E-07**	**7.6E-10**	**7.8E-03**	**1.9E-06**	**9.4E-08**	-
	40%	RMSE	**8.6E-02**	**1.5E-12**	**1.8E-07**	**6.2E-13**	**1.1E-10**	**3.0E-03**	**8.5E-12**	**2.7E-09**	**1.1E-02**	**1.3E-06**	**4.6E-07**	-
		MAE	2.2E-01	**2.3E-12**	**4.0E-07**	**6.6E-13**	**2.9E-10**	**3.4E-03**	**7.8E-12**	**2.8E-09**	**5.8E-04**	**2.4E-06**	**4.2E-07**	-
CN	10%	RMSE	**5.6E-02**	**2.2E-13**	**5.2E-11**	**2.2E-11**	**2.6E-12**	**7.3E-09**	**4.8E-09**	**4.9E-14**	**2.5E-09**	**4.0E-10**	**3.3E-09**	-
		MAE	**2.3E-12**	**1.8E-16**	**2.1E-12**	**2.2E-14**	**2.3E-17**	**1.3E-11**	**8.9E-14**	**1.0E-15**	**1.3E-12**	**3.5E-10**	**1.9E-10**	-
	20%	RMSE	**5.7E-03**	**2.1E-14**	**4.1E-12**	**1.3E-13**	**1.4E-14**	**3.2E-10**	**5.8E-11**	**2.1E-12**	**5.8E-11**	**7.3E-09**	**1.6E-10**	-
		MAE	**5.2E-13**	**5.4E-16**	**2.2E-12**	**2.8E-15**	**2.3E-17**	**2.9E-12**	**9.3E-14**	**1.1E-13**	**9.3E-10**	**8.6E-08**	**1.0E-10**	-
	30%	RMSE	**3.1E-05**	**1.2E-15**	**1.4E-12**	**2.7E-14**	**2.6E-14**	**1.7E-10**	**4.4E-12**	**7.2E-12**	**2.2E-09**	**2.3E-06**	**3.7E-12**	-
		MAE	**8.5E-12**	**2.1E-17**	**5.9E-12**	**2.3E-16**	**2.6E-15**	**6.5E-12**	**3.4E-15**	**1.6E-13**	**1.1E-08**	**1.2E-05**	**2.2E-13**	-
	40%	RMSE	**7.3E-09**	**6.6E-17**	**6.7E-13**	**1.1E-15**	**2.3E-13**	**2.3E-10**	**1.0E-13**	**7.5E-13**	**9.0E-10**	**7.9E-14**	**3.2E-12**	-
		MAE	**1.4E-10**	**1.1E-18**	**1.5E-10**	**4.6E-16**	**2.0E-14**	**1.1E-10**	**2.0E-15**	**2.4E-14**	**4.0E-07**	**1.3E-13**	**5.6E-13**	-
Los	10%	RMSE	**1.6E-07**	**2.3E-14**	**7.3E-13**	**3.7E-11**	**2.8E-13**	**1.4E-06**	**2.1E-09**	**3.1E-11**	**1.0E-05**	**3.2E-11**	**5.1E-09**	-
		MAE	**1.3E-08**	**1.3E-13**	**9.5E-14**	**1.5E-12**	**4.2E-15**	**8.7E-07**	**3.0E-04**	**1.7E-13**	**1.6E-07**	**7.0E-13**	**4.8E-08**	-
	20%	RMSE	**7.2E-09**	**2.2E-14**	**1.1E-12**	**2.7E-12**	**2.1E-15**	**9.9E-07**	**2.9E-09**	**4.3E-11**	**8.7E-06**	**8.9E-12**	**9.3E-13**	-
		MAE	**1.9E-10**	**7.5E-15**	**3.4E-13**	**1.4E-13**	**1.9E-17**	**7.8E-07**	**3.5E-04**	**7.7E-12**	**8.9E-06**	**1.2E-12**	**1.0E-10**	-
	30%	RMSE	**7.9E-10**	**1.0E-15**	**9.9E-13**	**3.3E-10**	**1.0E-14**	**1.4E-07**	**2.8E-09**	**9.9E-15**	**4.6E-05**	**2.0E-12**	**3.7E-10**	-
		MAE	**1.8E-11**	**1.5E-15**	**8.2E-14**	**7.5E-12**	**1.0E-15**	**3.6E-07**	**3.5E-04**	**7.7E-13**	**5.1E-05**	**2.0E-12**	**2.7E-07**	-
	40%	RMSE	**7.7E-05**	**4.2E-17**	**1.8E-13**	**1.7E-10**	**3.8E-15**	**3.3E-08**	**3.1E-09**	**1.3E-12**	**1.9E-05**	**1.8E-10**	**4.9E-11**	-
		MAE	**3.9E-05**	**3.8E-16**	**2.9E-12**	**6.7E-12**	**1.8E-16**	**2.5E-08**	**3.8E-04**	**2.1E-13**	**1.2E-05**	**9.5E-12**	**7.4E-08**	-
LuohuTaxi	10%	RMSE	**2.0E-09**	**1.2E-11**	**1.1E-08**	**1.7E-08**	**1.8E-11**	**8.1E-06**	**4.7E-04**	**2.2E-05**	**4.8E-08**	**3.4E-09**	**2.2E-05**	-
		MAE	**9.7E-10**	**9.5E-14**	**2.3E-09**	**1.6E-09**	**3.5E-12**	**1.5E-06**	**8.4E-05**	**6.7E-07**	**2.0E-06**	**2.3E-10**	**2.0E-05**	-
	20%	RMSE	**4.0E-10**	**1.1E-14**	**3.7E-11**	**4.3E-11**	**6.6E-13**	**2.1E-08**	**4.5E-04**	**1.5E-05**	**1.0E-10**	**5.8E-11**	**1.1E-11**	-
		MAE	**6.5E-11**	**3.7E-16**	**4.5E-10**	**3.0E-10**	**3.8E-14**	**7.6E-08**	**7.7E-05**	**3.7E-07**	**6.4E-08**	**1.1E-12**	**4.4E-09**	-
	30%	RMSE	**2.3E-10**	**1.3E-14**	**2.0E-09**	**2.8E-14**	**3.8E-14**	**1.8E-08**	**4.6E-04**	**3.0E-05**	**1.0E-11**	**1.8E-10**	**2.7E-08**	-
		MAE	**1.4E-09**	**2.7E-16**	**1.2E-10**	**2.8E-12**	**9.9E-16**	**1.1E-08**	**7.8E-05**	**1.3E-06**	**1.8E-09**	**1.6E-12**	**1.6E-08**	-
	40%	RMSE	**1.4E-10**	**2.7E-16**	**2.0E-08**	**1.1E-14**	**1.3E-12**	**4.0E-08**	**5.1E-04**	**1.6E-05**	**4.5E-11**	**5.7E-12**	**5.8E-10**	-
		MAE	**9.7E-10**	**7.1E-19**	**1.0E-09**	**3.1E-13**	**9.8E-14**	**1.1E-08**	**8.5E-05**	**1.0E-06**	**3.7E-10**	**1.1E-13**	**2.6E-08**	-

Table 9: T-test P-value (bolding for significant with p=0.05) of the imputation performance between GSLI and existing methods with various missing rates

Dataset	Missing rate	Metric	CSDI	TimesNet	SAITS	GPT4TS	LRTC-TNN	GRIN	STD-GAE	DAMR	PriSTI	PoGeVon	ImputeFormer	GSLI
DutchWind	10%	RMSE	11.67%	19.52%	13.38%	26.71%	29.99%	6.20%	13.30%	32.69%	15.03%	16.86%	22.08%	-
		MAE	7.94%	35.56%	22.47%	44.02%	30.06%	10.59%	18.45%	48.51%	5.30%	28.52%	31.51%	-
	20%	RMSE	13.94%	32.05%	12.59%	36.97%	31.05%	4.47%	14.01%	31.02%	13.74%	10.69%	22.23%	-
		MAE	11.53%	50.18%	21.73%	54.51%	32.49%	8.82%	20.01%	47.25%	6.08%	16.68%	31.94%	-
	30%	RMSE	13.71%	39.26%	12.58%	42.26%	33.89%	3.12%	15.83%	29.15%	14.11%	12.33%	20.96%	-
		MAE	13.72%	56.69%	21.78%	59.01%	36.14%	6.94%	23.45%	45.23%	7.84%	20.82%	30.46%	-
	40%	RMSE	14.81%	43.05%	11.12%	45.11%	36.37%	1.54%	20.50%	27.76%	16.51%	21.18%	20.58%	-
		MAE	17.62%	59.48%	20.16%	60.96%	40.26%	5.11%	30.52%	43.02%	12.90%	35.63%	28.97%	-
BeijingMEO	10%	RMSE	14.49%	16.22%	17.94%	24.38%	35.65%	7.69%	17.88%	44.89%	12.80%	25.30%	22.80%	-
		MAE	2.17%	33.62%	29.85%	43.22%	36.83%	15.80%	30.17%	60.15%	4.52%	42.40%	28.12%	-
	20%	RMSE	14.83%	22.93%	17.59%	35.84%	38.15%	7.17%	17.55%	44.22%	13.83%	24.76%	22.86%	-
		MAE	3.16%	42.39%	30.64%	56.50%	38.75%	15.11%	29.27%	59.35%	8.02%	39.36%	28.68%	-
	30%	RMSE	15.38%	30.83%	16.71%	42.42%	40.20%	6.79%	17.03%	41.82%	17.77%	22.45%	23.33%	-
		MAE	4.57%	50.67%	29.18%	62.00%	40.54%	14.54%	27.98%	56.90%	17.24%	34.83%	29.42%	-
	40%	RMSE	16.37%	37.15%	16.26%	46.28%	42.32%	6.58%	16.68%	40.85%	24.66%	24.61%	21.78%	-
		MAE	6.80%	56.36%	29.16%	64.75%	42.54%	14.60%	26.83%	55.94%	25.46%	38.39%	27.87%	-
LondonAQ	10%	RMSE	8.83%	32.94%	27.55%	43.45%	44.49%	12.68%	54.46%	62.28%	13.40%	27.55%	32.34%	-
		MAE	5.11%	34.52%	30.52%	46.17%	44.19%	12.49%	63.31%	64.94%	9.86%	25.42%	34.00%	-
	20%	RMSE	4.89%	43.35%	23.42%	51.82%	42.66%	8.14%	49.88%	58.98%	23.91%	22.74%	26.01%	-
		MAE	1.16%	47.59%	27.48%	56.26%	43.80%	7.96%	60.47%	62.38%	8.90%	21.68%	29.33%	-
	30%	RMSE	5.73%	51.36%	22.48%	56.75%	45.20%	8.64%	51.82%	59.28%	36.20%	22.89%	26.50%	-
		MAE	3.73%	57.34%	27.73%	62.28%	47.89%	9.99%	62.44%	63.34%	18.08%	23.42%	30.48%	-
	40%	RMSE	10.89%	55.21%	21.92%	58.76%	47.87%	7.06%	47.01%	57.19%	46.35%	21.82%	22.82%	-
		MAE	3.29%	60.48%	26.07%	63.58%	49.29%	8.07%	58.54%	60.62%	26.76%	21.92%	26.46%	-
CN	10%	RMSE	46.28%	62.08%	46.60%	48.29%	54.82%	37.08%	31.50%	71.16%	34.53%	60.03%	34.80%	-
		MAE	33.84%	73.76%	57.84%	61.84%	66.36%	50.08%	41.40%	80.37%	32.72%	69.52%	41.07%	-
	20%	RMSE	38.78%	61.75%	44.43%	47.56%	55.14%	35.88%	31.19%	69.77%	36.77%	57.79%	33.31%	-
		MAE	33.81%	73.46%	55.48%	61.30%	66.75%	48.69%	40.42%	79.31%	34.79%	67.34%	38.64%	-
	30%	RMSE	36.15%	61.46%	42.43%	48.11%	56.16%	34.91%	30.23%	68.83%	40.76%	55.30%	32.49%	-
		MAE	34.33%	73.06%	53.33%	61.96%	67.41%	47.35%	39.20%	77.92%	38.64%	65.21%	37.78%	-
	40%	RMSE	35.60%	60.91%	39.98%	49.53%	57.61%	33.69%	29.64%	66.92%	41.56%	48.55%	32.52%	-
		MAE	35.36%	72.39%	50.56%	63.26%	68.08%	45.64%	38.07%	76.07%	42.51%	56.87%	36.43%	-
Los	10%	RMSE	15.50%	50.45%	50.78%	33.68%	47.47%	10.85%	72.16%	48.72%	10.15%	27.95%	40.81%	-
		MAE	10.20%	53.01%	45.56%	38.04%	51.98%	15.19%	70.56%	56.14%	14.61%	23.86%	29.90%	-
	20%	RMSE	17.94%	51.20%	49.18%	39.56%	50.88%	10.72%	70.98%	46.92%	12.73%	27.42%	38.17%	-
		MAE	11.56%	55.19%	44.77%	44.99%	56.01%	15.60%	69.67%	54.76%	18.34%	24.11%	26.60%	-
	30%	RMSE	20.59%	53.24%	48.28%	48.16%	54.45%	11.70%	70.19%	45.99%	20.17%	27.84%	38.70%	-
		MAE	13.27%	58.19%	43.93%	52.90%	59.63%	16.61%	68.99%	54.44%	25.42%	24.80%	27.07%	-
	40%	RMSE	26.19%	54.73%	47.35%	54.91%	55.95%	11.66%	68.89%	44.09%	28.10%	26.86%	36.32%	-
		MAE	17.51%	60.69%	43.38%	58.99%	61.92%	16.83%	68.00%	52.37%	35.08%	24.85%	25.10%	-
LuohuTaxi	10%	RMSE	12.19%	20.24%	9.22%	28.82%	39.67%	6.01%	47.59%	38.52%	21.59%	17.40%	10.00%	-
		MAE	11.70%	28.78%	10.46%	35.65%	39.79%	8.28%	54.91%	47.17%	24.70%	18.80%	10.73%	-
	20%	RMSE	11.52%	23.29%	8.89%	40.17%	42.32%	5.70%	46.94%	38.29%	21.17%	16.85%	8.50%	-
		MAE	11.38%	32.70%	10.45%	44.50%	42.91%	8.09%	54.33%	46.96%	22.98%	18.41%	9.33%	-
	30%	RMSE	11.32%	27.38%	8.09%	45.80%	45.35%	5.52%	46.35%	38.77%	21.18%	16.73%	8.22%	-
		MAE	11.38%	37.09%	9.46%	49.74%	46.59%	7.91%	53.79%	47.45%	23.54%	18.21%	9.13%	-
	40%	RMSE	10.58%	31.76%	8.00%	48.81%	45.67%	5.31%	45.75%	38.04%	21.02%	17.00%	7.16%	-
		MAE	11.20%	41.23%	9.62%	52.84%	47.64%	7.75%	53.16%	46.80%	23.20%	18.26%	7.64%	-

Table 10: Improvement percentage of the imputation performance between existing methods and GSLI with various missing rates