# A APPENDIX

All necessary material like experimental, supplementary illustration, and code is located in https://www.dropbox.com/sh/khyk8oxmswrikr6/AADA6No1RYm-nXNNGVIjzJNya?dl=0, anonymously.

## A.1 NOTATION SUMMARIZATION

The notation and symbols used in the paper are summarized in Table 4.

Table 4: Table of Notation

| Symbol | Definition and Description |
|---|---|
| $\widetilde{\mathcal{G}}$ | temporal graph set $\{(\mathcal{G}_0, y_0), (\mathcal{G}_1, y_1), \ldots, (\mathcal{G}_n, y_n)\}$ |
| $\mathcal{G}_j$ | $j$-th temporal graph with snapshots $\{\mathcal{S}_j^{(t_s)}\}_{t_s=0}^{T_s}$ |
| $\mathcal{S}_j^{(t_s)}$ | $t_s$ timestamped snapshot of the $j$-th temporal graph $\mathcal{G}_j$ |
| $t_e, t_s$ | edge timestamp, snapshot timestamp |
| $\mathcal{T}_i \sim P(\mathcal{T})$ | graph metric learning tasks sampled from distribution $P(\mathcal{T})$ |
| $\theta_i$ | all learnable parameters of task $\mathcal{T}_i$ |
| $\Theta$ | meta-learner over all graph metric learning tasks during meta-training |
| $\widetilde{\mathcal{G}}_{support}^{train}$ | set of support temporal graphs in meta-training |
| $\widetilde{\mathcal{G}}_{query}^{train}$ | set of query temporal graphs in meta-training |
| $\widetilde{\mathcal{G}}_{support}^{test}$ | set of support temporal graphs in meta-testing |
| $\widetilde{\mathcal{G}}_{query}^{test}$ | set of query temporal graphs in meta-testing |

## A.2 DATA PREPROCESSING OF BIOLOGICAL DATA

As shown in Table 1, there naturally exists one kind timestamp at the edge level. Therefore, we adopt it as $t_e$ in our streaming-snapshot model. To generate $t_s$, we set a time window as a snapshot for each $t_e$ as $[t_e - \Delta, t_e + \Delta]$ for the snapshot reconstruction to further capture the episodic evolution pattern. In the experiment, we set $\Delta$ as one unit of $t_e$. For each temporal graph, 36 edge timestamps together describe three consecutive metabolic cycles. In each graph, we take a subgraph by extracting a interval of 5 edge timestamps every 3 edge timestamps. The subgraph shares the same class label with its original entire graph. Therefore, we have 11 temporal subgraphs per class. And our task is to classify these temporal subgraphs using our METATAG model comparing with selected baseline algorithms. Moreover, for static graph kernel and graph representation learning algorithms, we use Reduced Graph Representation (Oettershagen et al., 2020) to map temporal graphs into dynamics-preserving static graphs, such that static algorithms could take them as input.

The dataset mentioned in the paper is publicly available. The four cross-validation groups are listed as follows.

1. $\widetilde{G}^{train}$ = {Babu, Breitkreutz, Gavin, Hazbun, Ho, Ito, Krogan-LCMS, Krogan-MALDI}, $\widetilde{G}^{test}$ = {Lambert, Tarassov, Uetz, Yu}
2. $\widetilde{G}^{train}$ = {Breitkreutz, Ho, Krogan-MALDI, Tarassov, Yu, Gavin, Ito, Babu,}, $\widetilde{G}^{test}$ = {Krogan-LCMS, Hazbun, Lambert, Uetz}
3. $\widetilde{G}^{train}$ = {Babu, Breitkreutz, Ho, Hazbun, Krogan-MALDI, Lambert, Tarassov, Uetz,}, $\widetilde{G}^{test}$ = {Yu, Gavin, Ito, Krogan-LCMS}
4. $\widetilde{G}^{train}$ = {Babu, Hazbun, Lambert, Tarassov, Yu, Gavin, Uetz, Krogan-LCMS}, $\widetilde{G}^{test}$ = {Breitkreutz, Ho, Krogan-MALDI, Ito}

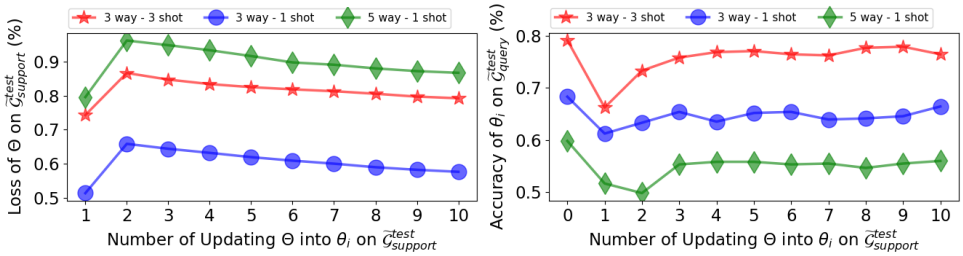
For the graph kernel methods, we insert the graph kernel into the SVM classifier, and the implementation can be found here [2]. For the graph metric learning and graph representation learning algorithms,

[2] https://ysig.github.io/GraKeL/0.1a8/documentation/introduction.html

we set the dimension of graph representation vectors as 64, and the implementation can be found here [3]. In our METATAG method, we set the learning rate $\alpha$ and $\beta$ as 0.001, and the weight for the snapshot reconstruction loss $\gamma = 0.7$. The experiments are performed on a Linux machine with and a single NVIDIA Tesla V100 32GB GPU.

### A.3 CONVERGENCE OF METATAG DURING META-TESTING

Following the same experimental setting mentioned above, we vary the number of shots and number of ways during the meta-training [4], to investigate the convergence of our METATAG model in the meta-testing phase. We report the loss of $\Theta$ on $\widetilde{\mathcal{G}}^{test}_{support}$ and the accuracy of $\theta_i$ on $\widetilde{\mathcal{G}}^{test}_{query}$ by every update of $\Theta$ on $\widetilde{\mathcal{G}}^{test}_{support}$. As shown in Figure 4, we observe that our METATAG could fast converge to the unseen tasks with only a few updates (e.g. 3 or 4 times) of the meta-learner. Interestingly, the ideal case occurs that the meta-training and meta-testing tasks share the global knowledge, $\Theta$ could perform well even without parameters update.




(a) Loss of $\Theta$ on $\widetilde{\mathcal{G}}^{test}_{support}$ before Different Number of Updates.

(b) Accuracy of $\theta_i$ on $\widetilde{\mathcal{G}}^{test}_{query}$ after Different Number of Updates.

Figure 4: Convergence of METATAG with Different Number of Updates.

### A.4 ABLATION STUDY

In our METATAG model, we design the multi-scale time attention mechanism to encode the temporal graph evolution pattern into the representation vector, which has three components: node-level time attention, intra-snapshot time attention, and inter-snapshot time attention. In the ablation study, we aim to remove them individually to investigate the effectiveness of each part.

To ablate the node-level time attention, we remove Eq. 3 directly, such that each node could only look for its previous neighbors but omit the corresponding interval time information. To ablate the intra-snapshot time attention, we eliminate the snapshot reconstruction loss function (i.e., Eq. 5) just by setting $\gamma = 0$. Then, there will be no constraints on the intra-snapshot level. To remove the inter-snapshot time attention, we replace Eq. 7 with the average pooling scheme, such that each snapshot will contribute equally to form the temporal graph representation vector. After we remove each component respectively, in Figure 5, we report the dynamic protein-protein interaction networks classification accuracy in the 3-way 3-shot setting with same hyperparameters.

As shown in Figure 5, we learn that each proposed component plays its own role in improving the metric learning ability to help the temporal graph classification problem. Interestingly, when we ablate the intra-snapshot attention scheme, METATAG achieves the worst performance with larger variances. It suggests that the snapshot reconstruction loss (i.e., capturing episodic patterns in the streaming graph) is very important in identifying the property of temporal graphs for the accurate and stable classification performance, which means adding the snapshot reconstruction loss $\ell$ is vital, but to what extent should we pay attention to the snapshot reconstruction loss $\ell$ in the streaming graph is not clear. Next, we design the parameter analysis in terms of different values of $\gamma$.

[3] https://github.com/benedekrozemberczki/karateclub and https://github.com/moranbel/tdGraphEmbed

[4] We randomly pick one class from $\widetilde{\mathcal{G}}^{train}$ and add it to $\widetilde{\mathcal{G}}^{test}$ for the 5-way 1-shot setting.

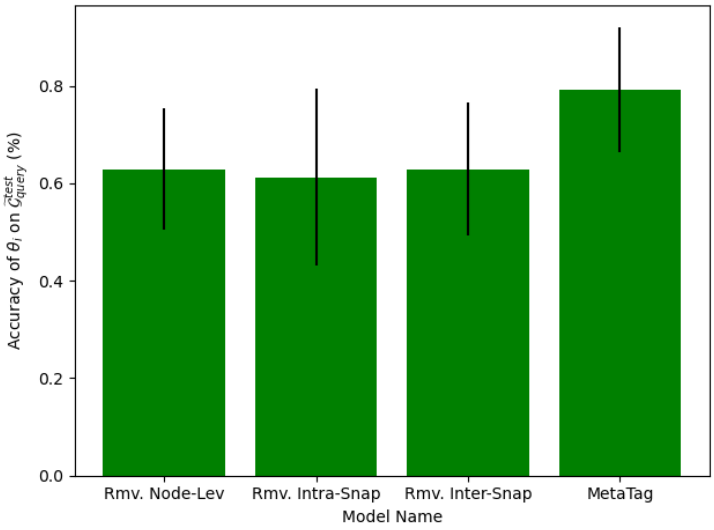



Figure 5: Variants of METATAG for Temporal Graph Classifications in the 3 Way - 3 Shot Setting.

### A.5 PARAMETER SENSITIVITY

Here, we change the weight of the snapshot reconstruction loss $\ell$ by changing the value of $\gamma$, to investigate the effect of capturing episodic patterns in the streaming graph. We show the temporal graph classification accuracy in the 3 way - 1 shot setting with different $\gamma$ in Figure 6.

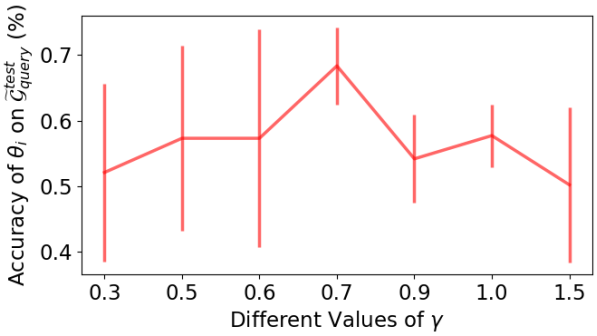



Figure 6: Effectiveness of METATAG with Different $\gamma$ in the 3 Way - 1 Shot Setting.

As shown in Figure 6, we learn that neither adjective nor dominant snapshot reconstruction loss $\ell$ is ideal such like $\gamma = 0.3$ or $\gamma = 1.5$. When the proportion of loss $\ell$ is too small or too large, the accuracy of METATAG is not that high and stable. Also, we can see that when the loss $\ell$ has considerable attention weights like $\gamma = 0.7$, METATAG could classify accurately and robustly.

### A.6 TEMPORAL GRAPH CLASSIFICATION ON SOCIAL NETWORK DATA

Except for the dynamic protein-protein interaction networks study with our METATAG, we also prepare extra datasets from the social network and human-contact perspectives. As shown in Table 2, the amount of offline graph data is less than online data. In this study, we aim to investigate whether sufficient online temporal graph data could provide transferable and shareable knowledge to help the offline temporal graph classification. To this end, we select online networks (i.e., Facebook, Tumblr, and DBLP) for $\widetilde{\mathcal{G}}^{train}$ and offline networks (i.e., Infectious, HighSchool, and MIT) for $\widetilde{\mathcal{G}}^{test}$. Also, we select NetLSD and tdGraphEmbed algorithms (that achieve competitive performance in the above experiments) along with other metric learning baselines (i.e., GL2Vec, TGAT, and CAW) to be

Table 5: Temporal Graph Classification Accuracy on Social Temporal Graphs

| Methods | Few-Shot Setting | | | |
|---|---|---|---|---|
| | 3 way - 5 shot | 3 way - 3 shot | 3 way - 2 shot | 3 way - 1 shot |
| GL2Vec + ProtoNet | **0.6250 ±0.0249** | 0.5810±0.0271 | 0.5633±0.0184 | 0.4887±0.0206 |
| NetLSD + ProtoNet | 0.3296 ±0.0307 | 0.3327±0.0179 | 0.3273±0.0068 | 0.3220±0.0297 |
| TGAT + ProtoNet | 0.3227 ±0.0171 | 0.3293±0.0156 | 0.3243±0.0110 | 0.3363±0.0010 |
| CAW + ProtoNet | 0.3340 ±0.0113 | 0.3333±0.0229 | 0.3380±0.0155 | 0.3270±0.0189 |
| tdGraphEmbed + ProtoNet | 0.5083 ±0.0121 | 0.4523±0.0353 | 0.4670±0.0199 | 0.3973±0.0164 |
| MetaTag | 0.6161 ±0.0139 | **0.5931±0.0148** | **0.6074±0.0164** | **0.5605±0.0201** |

responsible for the temporal graph classification. By comparing the performance shown in Table 5, we can learn that our METATAGachieves the best performance frequently (i.e., three of four times). Even in the second place, our METATAG only performs $1.42\%$ less than the first place (i.e., GL2Vec) in the 3 way - 5 shot setting. An intuitive explanation is that exploring online social network patterns could help identify offline human-interact patterns. To be specific, the prototypical temporal graph encoder and the meta-learner of our METATAG successfully finish the graph metric learning tasks.