Presentation

Sike Fu

sk6@mail.ustc.edu.cn

September 24, 2024

Existing Graph Foundation Model

GNN-based Models

Model	Backbone Architecture	citations	year
All in One		90	2023
PRODIGY		33	2023
DGI		776	2018
GRACE		874	2020
VGAE		3070	2016
MAGCL		36	2022
MultiGPrompt		15	2023
IGAP		6	2024
HGPROMPT		17	2023
GraphMAE	GAT	438	2022
GraphMAE2		67	2023
GPPT		119	2022
VPGNN		3	2023
GPT-GNN		557	2020
PT-HGNN		44	2021
CPT-HG		45	2021

Model	Backbone Architecture	citations	year
Graph-Prompt		113	2023
Graph-Prompt+		13	2023
GCC		920	2020
GraphCL		2002	2020
AdapterGNN		3	2023
AAGOD		15	2023
GPF		48	2022
SGL-PT		17	2023
FOTOM		2	2023
GraphControl		9	2023
G-TUNING		1	2023
Graph-BERT		286	2020
GROVER	GraphTransformer	738	2020
G-Adapter		10	2023

GraphMAE: Self-Supervised Masked Graph Autoencoders

What's GraphMAE

masked graph autoencoder for generative self-supervised graph pre-training

focus on **graph feature reconstruction** with both a *masking strategy* and *scaled cosine error*

self-supervised

- constructive
- generative
  - autoregressive
  - autoencoding

background

- CV&NLP: contrastive SSL has experienced emergence, generative SSL has been gaining steadily significance.
- graph learning: constractive SSL has been the dominant approach, especially for two tasks- -node and graph classification.
  - complicated training strategy
  - need negativa samples
  - rely on high qulity data augmentation

self-supervised GAE avoid the issues in constractive learning as its learning strategy is to directly reconstruct the input graph data.
But no GAEs succeed to achieve a comprehensive outperformance over contrastive SSL methods, especially on node and graph classifications

four challenges that none of the previous GAE collectively deals with:

- the structure information may be over-emphasized
- feature reconstruction without corruption may not be robust
- the mean square error (MSE) can be sensitive and unstable
- the decoder architectures(usually MLP) are of little expressiveness

GraphMAE's design:

- reconstruct **feature** with masking
  - different from most GAEs' structure reconstruction
- scaled cosine error
- re-mask decoding
  - re-masks the encoder's output embeddings of masked nodes before they are fed into the decoder

Feature reconstruction as the objective

simple MLPs distilled from trained GNN teachers can work comparably to advanced GNNs on node classification, indicating the vital role of features in such tasks

Masked feature reconstruction

to solve problems: when hidden's dimesion size is larger than input's, autoencoders learn the notorious "identity function"—the trivial solution—that makes the learned code useless.

denoise or *mask* autoencoder. mask succeeds in CV&NLP.

uniform random sampling, prevent bias center.

relatively large mask ratio, reduce redundancy.

random-substitution ranther than leave unchanged to masked nodes.

GNN decoder with re-mask decoding

a more expressive **single layer GNN** as decoder rather than no decoder or MLP in traditional GAE.

to further encourage the encoder to learn compressed representations, replace hidden feature on masked node indices again with another mask token [DMASK]

Scaled cosine error as the criterion

MSE suffer from the issues of *sensitivity* and *low selectivity*

scale: improve selectivity, down-weight easy samples' contribution

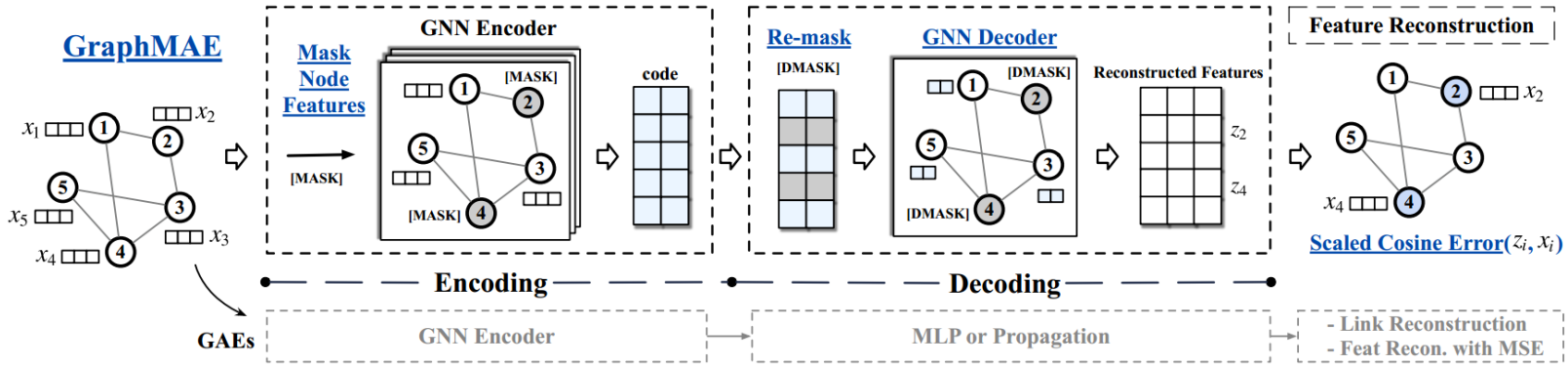

Figure: Illustration of GraphMAE and the comparison with GAE

GROVER: Self-Supervised Graph Transformer on Large-Scale Molecular Data

- GROVER means: Graph Representation frOm self-superVised mEssage passing tRansformer
- transformer-based neural network with tailored GNNs as the self-attention building block
- solves two issues
  - insufficient labeled molecules for supervised training;
  - poor generalization capability to new-synthesized molecules.

Architecture:

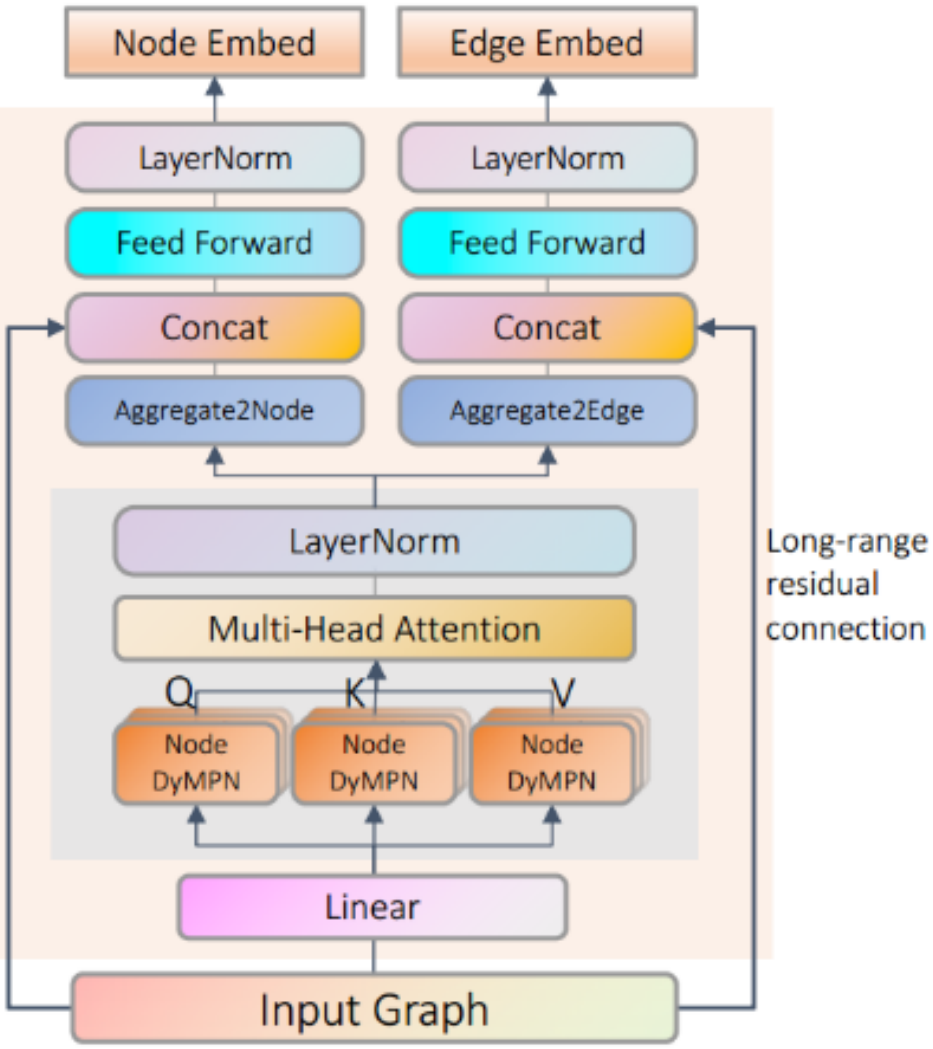

Figure: Gtransformer

a single long-range residual connection

- like ordinary residual connections, it improves the training process by alleviating the vanishing gradient problem
- long-range residual connection can alleviate the over-smoothing problem in the message passing process.

Dynamic Message Passing Network (dyMPN)

- two hyperparameters: number of iterations/layers L and number of hops $K_l, l = 1, 2, ..., L$ within each iteration
- dynamic: choose K_l from some random distribution for layer l. Two choices: random distribution $U(a, b)$ and truncated normal distribution $\phi(\mu, \sigma, a, b)$

$$m_v^{l,k} = \text{AGGREGATE}^l(\{(h_v^{l,k-1}, h_u^{l,k-1}, e_{uv}) \mid u \in \mathcal{N}_v\}), \quad (1)$$

$$h_v^{l,k} = \sigma(W^l m_v^{l,k} + b^l), \quad (2)$$

- **pre-train**
  - contextual property prediction
    - node-edge-counts C_N-DOUBLE1_O-SINGLE1
  - graph-level motif prediction
    - Motifs are recurrent sub-graphs among the input graph data, e.g. function group in molecules
- **fine-tune**: node, edge, graph level

LLM-based Models

Model	Backbone Architecture	citations	year
Gimnet	Graph-to-token+Transformer	20	2023
InstructGLM		13	2024
LLMtoGraph		23	2023
NLGraph	Graph-to-text+GPTs	98	2023
GraphText		41	2023
LLM4Mol		41	2023
TextForGraph		2	2023
When&Why		33	2023
GraphWiz		8	2024
CGForLLM		4	2023
LLM4DYG		96	2023
GPT4Graph		96	2023
Graph-LLM	BERT, DeBERTa, Sentence-BERT, GPTs, LLaMA	178	2023

GIMLET: Graph Instruction based MolecuLe zEro-shoT learning

- natural language instructions
- molecule related tasks
- zero-shot

- construct a molecule dataset consisting of two thousand tasks with instructions derived from task descriptions
- propose GIMLET, which extends language models to handle graph and text data
  - applying the transformer mechanism with generalized position embedding and decoupled attention
  - instruction-based pretraining

individual pre-encoding modules present problems

- dense vectors encoded by GNN have a limited capacity to carry *structure information*
- training the additional module is difficult due to the increased layers
- additional modules increase parameters and training costs

GIMLET not only directly unifies the standard language model for graph and text **without** introducing additional graph encoder module, but also remains the decoupled graph encoding for better generalization.

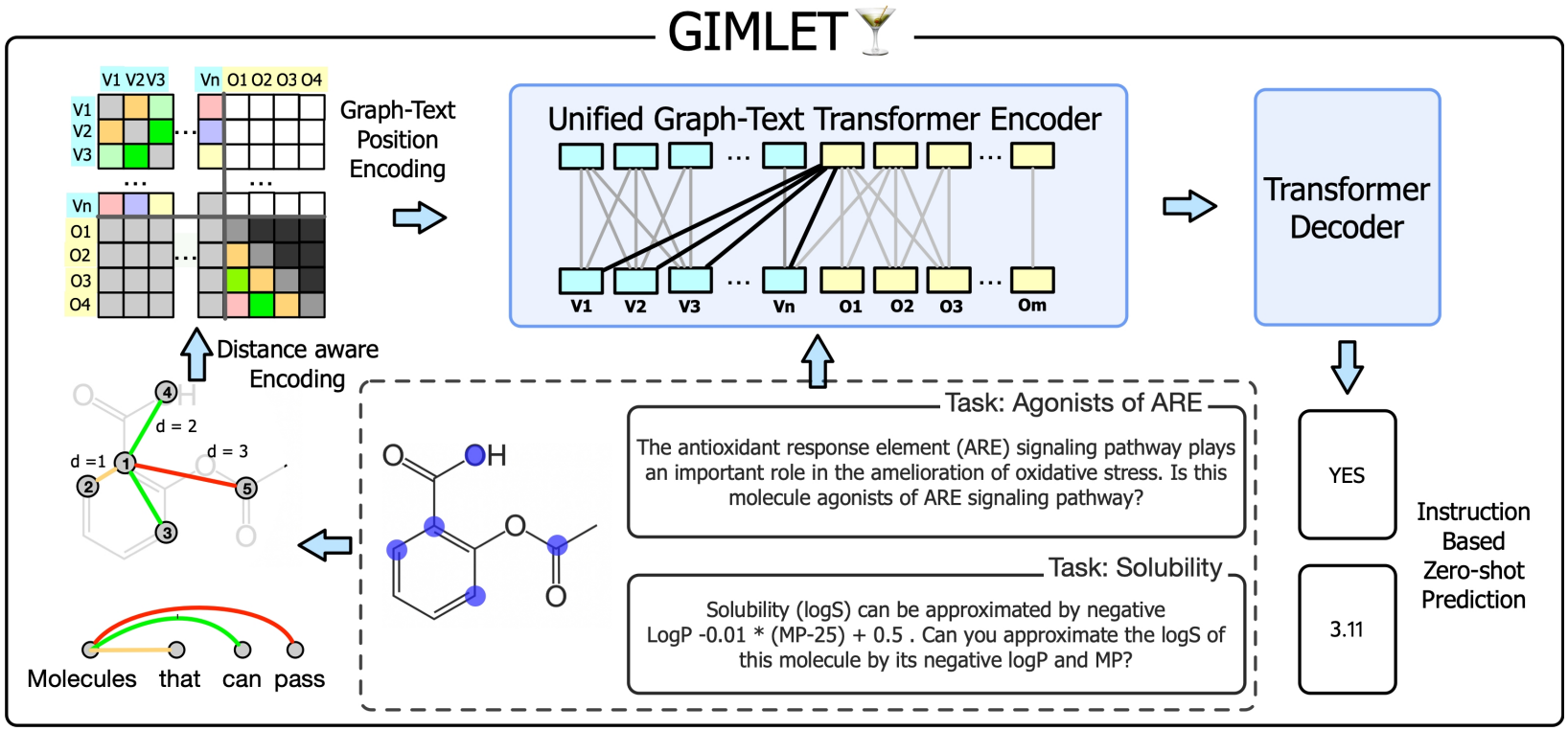

Figure: GIMLET

The instruction of task τ is T^{τ}, a sentence $[o_1, ..., o_m]$ describing the task

$\hat{y}_{str} = GIMLET(G, T^{\tau})$

backbone language model: **T5**, encoder-decoder architecture

represent the **graph nodes** and **text tokens** as tokens, resulting hidden states are denoted as $H = [h_1, ..., h_n, h_{n+1}, ..., h_{n+m}]^T \in R^{(n+m)\times d_h}$ for corresponding n graph nodes and m text tokens.

relative position embedding for i-th and j-th token:

$$\hat{A}_{ij} = \frac{\left(h_i W^Q\right)\left(h_j W^K\right)^T}{\sqrt{d_k}} + b(i,j)$$

$$A = \text{softmax}(\hat{A})$$

$$\text{Attn}(H) = AHW^V W^O$$

$b(i,j)$

is embedding of the relative distance between i and j

- for sequence, i − j
- For the relative position of graph nodes, the graph shortest distance
- For graph-text joint data, conjunction of different types of distances...

$$b(i,j) = b^{D}_{\mathrm{POS}(i,j)} + b^{M}_{i,j} + \underset{k \in \mathrm{SP}(i,j)}{\mathrm{Mean}} b^{E}_{e_k},$$

$$\mathrm{POS}\,(i,j) = \begin{cases} i-j & \text{if } n+1 \le i,j \le n \\ \textsc{Graph Shortest Distance}(i,j) & 1 \le i,j \le n \\ <\textsc{Cross}> & \text{otherwise} \end{cases}$$

$<\textsc{Cross}>$: a special distance token for cross distance between graph and text tokens

$$b(i,j) = b^D_{\mathrm{POS}(i,j)} + b^M_{i,j} + \underset{k\in \mathrm{SP}(i,j)}{\mathrm{Mean}} b^E_{e_k},$$

$b^M_{i,j}$: the cross mask used to decompose graph encoding from text instructions.

$$b^M_{i,j} = -\infty \text{ if } i \leq n \text{ and } j > n \quad \text{otherwise } 0$$

a *unidirectional* constraint from the graph to the text. allows to separate the encoding of graphs from instructions, enabling instructions to selectively utilize graph features for various downstream tasks.

$$b(i,j) = b^D_{\mathrm{POS}(i,j)} + b^M_{i,j} + \underset{k\in \mathrm{SP}(i,j)}{\mathrm{Mean}}\ b^E_{e_k},$$

$\mathrm{Mean}_{k\in \mathrm{SP}(i,j)}\ b^E_{e_k}$: the mean pooling of the edge features $b^E_{e_k}$ in the shortest path $\mathrm{SP}(i,j)$ between node i and j

only defined between graph node tokens

GNN+LLM-based Models

Grap  ph Instruction Tuning for Large Language Mode

a graph-oriented LLM capable of exceptional generalization across various datasets and tasks **without relying on downstream graph data**

integrates LLMs with graph structural knowledge through graph instruction tuning.

includes a text-graph grounding component to link textual and graph structures and a dual-stage instruction tuning approach with a lightweight graph-text alignment projector.

limitation of many GNN approaches: heavy reliance on supervised learning
$\rightarrow$ self-supervised learning, but often require a fine-tuning process using labels specific to the downstream graph learning scenarios

limitations associated with directly prompting LLMs using *purely text-based prompts* for graph structure modeling; text-based structural prompts leads to an increase in token size

Structural Information Encoding with Text-Graph Grounding

graph encoder $f_{\mathbf{G}}$: graph transformer
text encoder $f_{\mathbf{T}}$: vanilla transformer

$$\mathbf{H} = f_{\mathbf{G}}(\mathbf{X}), \mathbf{T} = f_{\mathbf{T}}(\mathbf{C}), \hat{\mathbf{H}} = \text{norm}(\mathbf{H}), \hat{\mathbf{T}} = \text{norm}(\mathbf{T})$$

$$\Gamma_1 = (\hat{\mathbf{H}}\hat{\mathbf{T}}^\top) \cdot \exp(\tau), \Gamma_2 = (\hat{\mathbf{H}}\hat{\mathbf{T}}'^\top) \cdot \exp(\tau), \Gamma_3 = (\hat{\mathbf{T}}^\top\hat{\mathbf{T}}'^\top) \cdot \exp(\tau$$

$$\mathcal{L} = \sum_{i=1}^{3} \frac{1}{2}\lambda_i(\text{CE}(\Gamma_i, \mathbf{y}) + \text{CE}(\Gamma_i^\top, \mathbf{y}))$$

where $\hat{\mathbf{T}}' = \{\frac{1}{|\mathcal{N}_i|}\sum_{j\in\mathcal{N}_i}\hat{\mathbf{T}}_j, 1 \le i \le N\}$ and N is the number of nodes. label $\mathbf{y} = (0, 1, \cdots, n-1)^\top$ for the contrastive alignment objective.

Dual-Stage Graph Instruction Tuning

1. Self-Supervised Instruction Tuning
   - Instruction Design: i) graph information, ii) human question, and iii) GraphGPT response.
   - Tuning Strategy: During training, optimize the parameters of the Lightweight Alignment Projector, while keeping the parameters of both the LLM and the graph encoder fixed
   - domain-specific structural knowledge related to graphs
2. Task-Specific Instruction Tuning
   - task-specific graph instructions

Graph Information: <graph>: **Central Node:** 68442, **Edge index:** [[…src node…],[…dst node…]], **Node list:** […] **Graph Matching**
Human Question: Given a sequence of graph tokens <graph> that constitute a subgraph of a citation graph, …. Here is a list of paper titles: 1. … 2. …, please reorder the list of papers according to the order of graph tokens.
GraphGPT Response: Based on the given graph tokens and the list of paper titles, we obtain the matching of graph tokens and papers: Graph token 1 corresponds to smt based induction methods for timed systems. Graph token 2 corresponds to …

Graph Information: <graph>: **Central Node:** 2, **Edge index:** [[…src node…],[…dst node…]], **Node list:** […] **Node Classification**
Human Question: Given a citation graph: <graph> where the 0th node is the target paper, with the following information: Abstract: … Title: … Question: Which arXiv CS sub-category does this paper belong to? …
GraphGPT Response: cs.IT, cs.LG, cs.SP, cs.CV, cs.NA. The paper discusses the Restricted Isometry …. So, it is likely to belong to cs.IT…

Graph Information: <graph>: **Central Node 1:** 8471, **Edge index 1:** [[…src node…],[…dst node…]], **Node list 1:** […] **Link Prediction**
<graph>: **Central Node 2:** 19368, **Edge index 2:** [[…src node…],[…dst node…]], **Node list 2:** […]
Human Question: Given a sequence of graph tokens: <graph> that constitute a subgraph of a citation graph, …. Abstract: … Titile: … and the other sequence of graph tokens: <graph>, … Abstract: … Title: …, are these two central nodes connected? Give me an answer of "yes" or "no".
GraphGPT Response: Yes, they are connected. Based on the first paper, …. And the second paper proposes ….

Figure: instruction designs for graph matching task (upper), node classification (middle) and link prediction (lower)

Chain-of-Thought (CoT) Distillation

- to equip GraphGPT with step-by-step reasoning abilities
- By extracting valuable knowledge from a closed-source, powerful language model like ChatGPT
- generate high-quality COT instructions and enhance our model's COT reasoning capabilities without increasing the parameter count

integrate the chatGPT generated COT instruction data with previously designed instructions for task-specific instruction tuning.

Methods of Adversarial Attack on Graph Structured Data

$$\begin{aligned} \max_{\tilde{G}} \quad & \mathbb{I}(f(\tilde{G}, c) \neq y) \\ s.t. \quad & \tilde{G} = g(f, (G, c, y)) \\ & \mathcal{I}(G, \tilde{G}, c) = 1. \end{aligned}$$

the graph adversarial attacker $g(\cdot, \cdot) : \mathcal{G} \times \mathcal{D} \mapsto \mathcal{G}$
$\mathcal{I}(\cdot, \cdot, \cdot) : \mathcal{G} \times \mathcal{G} \times V \mapsto \{0, 1\}$ is an equivalency indicator that tells whether two graphs G and $\tilde{G}$ are equivalent under the classification semantics
$\mathbb{I}(\cdot) \in \{0, 1\}$ is an indicator function

Small modifications: ask the attacker to make as few modifications as possible within a neighborhood graph

$$\mathcal{I}(G, \tilde{G}, c) = \mathbb{I}(|(E - \tilde{E}) \cup (\tilde{E} - E)| < m) \cdot \mathbb{I}(\tilde{E} \subseteq \mathcal{N}(G, b))). \quad (3)$$

In the above equation, m is the maximum number of edges that allowed to modify, and $\mathcal{N}(G, b) = \{(u, v) : u, v \in V, d^{(G)}(u, v) <= b\}$ defines the b-hop neighborhood graph, where $d^{(G)}(u, v) \in \{1, 2, \ldots\}$ is the distance between two nodes in graph G.

RL-S2V

Finite Horizon Markov Decision Process $\mathcal{M}^{(m)}(f, G, c, y)$

- **Action** single action at time step t is $a_t \in \mathcal{A} \subseteq V \times V$
- **State** The state s_t at time t is represented by the tuple $(\hat{G}_t, c)$, where $\hat{G}_t$ is a partially modified graph with some of the edges added/deleted from G.
- **Reward** the non-zero reward is only received at the end of the MDP, with reward being

$$r((\tilde{G}, c)) = \begin{cases} 1 : f(\tilde{G}, c) \neq y \\ -1 : f(\tilde{G}, c) = y \end{cases}$$

- **Terminal** Once the agent modifies m edges, the process stops

use Q-learning to learn the MDPs.

RANDOM SAMPLING

- simplest attack method that randomly adds or deletes edges from graph G
- requires the least information for attack

GradArgmax

$$\hat{G}_{t+1} = \begin{cases} (\hat{V}_t, \hat{E}_t \setminus (u_t, v_t)) : \frac{\partial \mathcal{L}}{\partial \alpha_{u_t, v_t}} < 0 \\ (\hat{V}_t, \hat{E}_t \cup \{(u_t, v_t)\}) : \frac{\partial \mathcal{L}}{\partial \alpha_{u_t, v_t}} > 0 \end{cases}$$

the gradient considers all pairs of nodes in a graph, the computation cost is at least $O(|V|^2)$
cannot scale to large graphs.

GeneticAlg

- Population
- Fitnees
- Selection
- Crossover
- Mutation

computation cost is $O(|V| + |E|)$