Background

Atlas: local graph exploration in a global context[1]

Given a graph, decompose its edge set into fixed points by iterative edge peeling. Higher fixed points usually has higher density (|E| / (|V| choose 2)) than lower ones. For visualization, they first pre-compute a 2D layout of the input graph, and then add the third coordinate corresponding to the fixed point value. They are able to handle millions-edge graph, and provide a visualization on their local fixed points with a context of the global layout. However, when the input graph scales up to billions, both 1). the pre-computation of 2D layout and 2). The visualization of each fixed point cannot handle.

Graph Waves[2]

Given a fixed point, further decompose its edge set into waves and edge fragments via a BFS traversal with degree conditions. As long as edge fragment are not too large to fit in the visualization threshold, the second issue of Atlas is solved. However, this paper does not provide a context of the global structure of the input graph.

Graph Cities: Their Buildings, Waves, and Fragments[3]

Given a graph, decompose its edge set into fixed points and then further waves and edge fragments. A visualization metaphor that each building corresponding to a connected fixed point is applied. Road networks represent shared vertices between connected fixed points. Floors in a building encodes its wave decomposition. A building interior view shows Meta-DAG that represents the macrostructure of edge fragments, and users may click meta-nodes to check details. The application can scale up to 2 billion edges, and provides interactively exploration with global contexts. However, the decomposition is too powerful that users may be overwhelmed by such many subgraphs. A computer-aided exploration plug-in is required.

Scalable K-Core Decomposition for Static Graphs Using a Dynamic Graph Data Structure[4]

Given a graph, use Hornet data structure to maintain the peeling information and utilize thousands of threads from GPU results in faster fixed point decomposition than ParK algorithm which is implemented for CPU. However, the scan they used is not linear compared with the backbone of Park, which should be further optimized.

Sketch Recognition

Deep Sketch Hashing: Fast Free-hand Sketch-Based Image Retrieval[5]

Given a sketch and a set of natural pictures, first generate sketch-tokens from natural pictures by extracting edges. A Cross-weight Late-fusion Net takes natural pictures and sketch-tokens and outputs their encoding, and a Shared-weight Sketch Net takes the input sketch and outputs its encoding, which enables comparison between the sketch and natural pictures. Cross-view Pairwise Loss and Semantic Factorization Loss are applied to ensure the encoded sketch is as closed as the corresponding natural images, or at least the similar-classes natural images. This paper provides an end-to-end sketch retrieval, but it requires the input only contain a single main object.

SketchParse: Towards Rich Descriptions for Poorly Drawn Sketches using Multi-Task Hierarchical Deep Networks[6]

Given an input sketch, a two-level network is applied to segment it into areas. The first level is multi-scale version of ResNet-101 to capture category-agnostic low-level information, and the second level is a collection of experts networks to parse the sketch within its domain. A Router Layer takes the first level outputs and predicts its corresponding domain to forward to the second level. An auxiliary task of 2D pose estimation is added to help training. This network provides a region-based segmentation, but the sketch lines (strokes) are not taken into consideration.

Fast Sketch Segmentation and Labeling With Deep Learning[7]

Given a sketch, a U-net is applied to segment strokes, and a post-processing with label sampling predicts the label for strokes. As a complement of SketchParse, it takes care of strokes, but becasue the absence of the auxiliary task, it may produce flipped segmentations.

Multi-Scale Attention with Dense Encoder for Handwritten Mathematical Expression Recognition[8]

Given a handwritten mathematical expression, a variant of CNN, DenseNet, that takes all the outputs of previous layers is applied to encode the input, and a multi-scale branch further provides both low-resolution features and high-resolution features for recognition. A GRU decoder reconstructs the LaTeX expression of the input. A more intelligence way could utilize the grammar constrains of mathematical expression, i.e., a symbolic expression trees, on the decoder.

Graph Regression

IsoNN: Isomorphic Neural Network for Graph Representation Learning and Classification[9]

Given a query graph, a CNN based network is applied to extract the subgraph patterns and classify the input graph. To deal with the non-ordering of vertices, they use either a transpose layer to extract all possible permutations, which gives a huge computational complexity, or the optimal permutation matrix, which decreases the accuracy slightly.

On the equivalence between graph isomorphism testing and function approximation with GNNs[10]

Given two query graph, a GNN based network with considering power graphs predicts if two graphs are isomorphic or not. Though it is highly efficient, the result is binary, namely, either isomorphic or not. How to predict the similarity between two graphs are under investigation.

---

1. Abello, James et al. "Atlas: local graph exploration in a global context." *Proceedings of the 24th International Conference on Intelligent User Interfaces* (2019): n. pag. ↩

2. Abello, James and Daniel Nakhimovich. “Graph Waves.” *EDBT/ICDT Workshops* (2020). ↩
3. Abello, James et al. “Graph Cities: Their Buildings, Waves, and Fragments.” *EDBT/ICDT Workshops* (2021). ↩
4. Tripathy, Alok et al. “Scalable K-Core Decomposition for Static Graphs Using a Dynamic Graph Data Structure.” *2018 IEEE International Conference on Big Data (Big Data)* (2018): 1134-1141. ↩
5. Liu, Li et al. “Deep Sketch Hashing: Fast Free-Hand Sketch-Based Image Retrieval.” *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017): 2298-2307. ↩
6. Sarvadevabhatla, Ravi Kiran et al. “SketchParse: Towards Rich Descriptions for Poorly Drawn Sketches using Multi-Task Hierarchical Deep Networks.” *Proceedings of the 25th ACM international conference on Multimedia* (2017): n. pag. ↩
7. Li, Lei et al. “Fast Sketch Segmentation and Labeling With Deep Learning.” *IEEE Computer Graphics and Applications* 39 (2019): 38-51. ↩
8. Zhang, Jian-shu et al. “Multi-Scale Attention with Dense Encoder for Handwritten Mathematical Expression Recognition.” *2018 24th International Conference on Pattern Recognition (ICPR)* (2018): 2245-2250. ↩
9. Meng, Lin and Jiawei Zhang. “IsoNN: Isomorphic Neural Network for Graph Representation Learning and Classification.” *ArXiv* abs/1907.09495 (2019): n. pag. ↩
10. Chen, Zhengdao et al. “On the equivalence between graph isomorphism testing and function approximation with GNNs.” *NeurIPS* (2019). ↩