Self-Attention Graph Pooling

Paper ID: 2233  Project page: github.com/in yeoplee77/SAGPool

Junhyun Lee†
Inyeop Lee†
Jaewoo Kang

†Joint-first authors
Research background & Motivation

- Generalizing convolution operation to graphs.
- Growing interest in graph pooling methods.
- Graph pooling methods that can learn hierarchical representations of graphs.
• Task: Graph classification.

• Key Idea: Utilize GNNs as a graph pooling module.
Related Work

- Global pooling methods: use summation or neural networks to pool all the representations of nodes in each layer (Set2Set\(^1\) and SortPool\(^2\)).

- Hierarchical pooling methods: obtain intermediate graphs (adjacency, features) and pass them to the next layer (DiffPool\(^3\) and gPool\(^4\)).

---


Self-Attention Graph Pooling

\[ Z = \sigma(\text{GNN}(X, A)) \quad \text{idx} = \text{top-rank}(Z, \lceil kN \rceil), \quad Z_{\text{mask}} = Z_{\text{idx}} \]

\[ X' = X_{\text{idx}}; \quad X_{\text{out}} = X' \odot Z_{\text{mask}}, \quad A_{\text{out}} = A_{\text{idx}, \text{idx}} \]
Evaluation

Global pooling methods

- Graph Convolution
- Graph Convolution
- Graph Convolution

- Concatenate
- Graph Pooling

- Readout

- MLP

- Classification

Hierarchical pooling methods

- Graph Convolution
- Graph Pooling
- Readout

- Graph Convolution
- Graph Pooling
- Readout

- Graph Convolution
- Graph Pooling
- Readout

- MLP

- Classification

Global pooling methods

Hierarchical pooling methods

MLP
Evaluation

• Graph benchmark datasets.

• the same early stopping criterion and hyper-parameter selection strategy for a fair comparison

• 20 random seeds to split each dataset.

• 10-fold cross validation for evaluations (a total of 200 testing results for each evaluation).

• pytorch_geometric\[^1\] for implementation.

## Results

<table>
<thead>
<tr>
<th></th>
<th>D&amp;D</th>
<th>PROTEINS</th>
<th>NCI1</th>
<th>NCI109</th>
<th>FRANKENSTEIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set2Set</td>
<td>71.27±0.84</td>
<td>66.06±1.66</td>
<td>68.55±1.92</td>
<td>69.78±1.16</td>
<td>61.92±0.73</td>
</tr>
<tr>
<td>SortPool</td>
<td>72.53±1.19</td>
<td>66.72±3.56</td>
<td>73.82±0.96</td>
<td>74.02±1.18</td>
<td>60.61±0.77</td>
</tr>
<tr>
<td>SAGPool</td>
<td>76.19±0.944</td>
<td>70.04±1.47</td>
<td>74.18±1.20</td>
<td>74.06±0.78</td>
<td>62.57±0.60</td>
</tr>
<tr>
<td>DiffPool</td>
<td>66.95±2.41</td>
<td>68.20±2.02</td>
<td>62.32±1.90</td>
<td>61.98±1.98</td>
<td>60.60±1.62</td>
</tr>
<tr>
<td>gPool</td>
<td>75.01±0.86</td>
<td>71.10±0.90</td>
<td>67.02±2.25</td>
<td>66.12±1.60</td>
<td>61.46±0.84</td>
</tr>
<tr>
<td>SAGPool</td>
<td>76.45±0.97</td>
<td>71.86±0.97</td>
<td>67.45±1.11</td>
<td>67.86±1.41</td>
<td>61.73±0.76</td>
</tr>
</tbody>
</table>
Self-Attention Graph Pooling

Paper ID: 2233  Project page: github.com/inyeoplee77/SAGPool

• Additional details and discussion at the poster (Pacific Ballroom #8).