Characterizing Massive Activations of Attention Mechanism in Graph Neural Networks

Supplementary Material

1 Dataset Composition

This section provides additional details on the used datasets throughout the experiments.

The ZINC dataset [1] is a benchmark collection for evaluating GNNs in molecular chemistry, where molecules are represented as graphs with atoms as nodes and chemical bonds as edges. Contents include:

- Graphs: The dataset includes over $250,000$ molecular graphs. Each molecule is represented by a graph with nodes (atoms) and edges (bonds), incorporating various bond types (e.g., single, double, triple).
- Node Features: Atoms are described by features that capture their chemical properties, such as atom types, hybridization states, and other atomic attributes.
- Edge Features: Bonds between atoms are characterized by features representing bond types and additional chemical information.
- Task: The primary task is **graph regression**, where the goal is to predict continuous values associated with each molecule. This often involves predicting molecular properties such as solubility or biological activity.

ZINC is useful for evaluating GNNs' performance in learning molecular representations and predicting continuous chemical properties, providing insights into the model's ability to generalize across diverse chemical compounds.

The TOX21 dataset [2, 3] is designed for toxicity prediction and focuses on classifying chemical compounds based on their potential toxicity. It is part of the Toxicology Data Challenge and features molecular graphs with associated toxicity labels. Contents include:

- Graphs: The dataset consists of molecular graphs where nodes represent atoms and edges represent chemical bonds. It includes thousands of molecules with toxicity annotations, and it consists of $7,831$ graphs with each graph representing a molecular structure with associated toxicity labels.
- Node Features: Atoms are encoded with features representing their types, hybridization states, and other chemical properties.
- Edge Features: Bonds are detailed with features indicating bond types and additional chemical attributes.
- Task: The main task is **multi-label graph classification**, where each molecule is classified into multiple toxicity categories. This allows for the prediction of various toxicity endpoints simultaneously.

TOX21 is valuable for assessing GNN models in predicting toxicity from molecular structures, which is crucial for drug discovery and safety evaluation, providing a benchmark for multi-label classification tasks.

The OGBN-PROTEINS dataset, part of the Open Graph Benchmark (OGB) [4], focuses on protein function prediction. It contains one large graph representing protein structures, with nodes corresponding to amino acids and edges to their interactions. Contents include:

- One Large Graph: OGBN-PROTEINS contains $54,879$ nodes and $89,724$ edges. These nodes represent amino acids in protein structures, and edges represent interactions or bonds between these amino acids. It includes various protein structures used for functional prediction.
- Node Features: Amino acids are described by features capturing biochemical properties, such as amino acid type, secondary structure, and other relevant attributes.
- Edge Features: Edges denote interactions between amino acids and include features reflecting the nature of these interactions or spatial relationships.
- Task: The task is **multi-label node classification**, where the goal is to predict multiple functional categories for each amino acid node in the protein graph. This involves classifying nodes into various functional classes based on their role in the protein's functionality.

OGBN-PROTEINS is suitable for evaluating GNNs on biological data, specifically in predicting protein functions based on structural information. It provides insights into how well models can handle multi-label node classification tasks in a complex biological context.

2 Model Architecture

This section provides additional details on the models' architecture used throughout all the experiments, namely GT [5], GraphiT [6] and SAN [7]. These graph-transformer architectures integrate the principles of both GNNs and transformers, leveraging the strengths of attention mechanisms to capture intricate relationships within graph-structured data. Graph transformers extend the transformer structure, typically used for sequence data, to graphs, operating by embedding nodes and edges into higher-dimensional spaces and then applying multi-head self-attention mechanisms to capture dependencies between nodes.

Mathematically, let $G = (V, E)$ be a graph where $V = \{v_1, ..., v_n\}$ is the set of nodes and $E \subseteq V \times V$ is the set of edges. Each node v_i is associated with a feature vector $x_i \in \mathbb{R}^d$, and each edge (v_i, v_j) may have an edge feature $e_{ij} \in \mathbb{R}^k$. Therefore, graph transformer models are designed as follows.

Input Embedding

The initial node features $X = [x_1, ..., x_n]^T \in \mathbb{R}^{n \times d}$ are typically projected to a higher-dimensional space:

$$H^{(0)} = XW_{in} + b_{in} \tag{1}$$

where $W_{in} \in \mathbb{R}^{d \times d'}$ is a learnable weight matrix and $b_{in} \in \mathbb{R}^{d'}$ is a bias vector.

Positional Encoding

To capture structural information, positional encodings $P \in \mathbb{R}^{n \times d'}$ are often added:

$$H^{(0)} = H^{(0)} + P \tag{2}$$

Multi-Head Attention Layer

The core of a graph transformer is the multi-head attention mechanism. For each attention head i (out of h heads) there are also:

1. Query, Key, and Value Projections:

$$Q_i = H^{(l)} W_i^Q \tag{3}$$
$$K_i = H^{(l)} W_i^K \tag{4}$$
$$V_i = H^{(l)} W_i^V \tag{5}$$

   where $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d' \times d_k}$ are learnable weight matrices, and $d_k = d'/h$.

2. Attentions Scores (node features only):

$$A_i = \text{softmax}\left(\frac{Q_i K_i^T}{\sqrt{d_k}} + M\right), \tag{6}$$

where $M \in \mathbb{R}^{n \times n}$ is a mask matrix to enforce the graph structure:

$$M_{ij} = \begin{cases} 0 & \text{if } (v_i, v_j) \in E \text{ or } i = j \\ -\infty & \text{otherwise.} \end{cases} \tag{7}$$

3. Output of each head:

$$\text{head}_i = A_i V_i. \tag{8}$$

4. Concatenation and Projection:

$$H' = \text{Concat}(\text{head}_1, ..., \text{head}_h) W^O, \tag{9}$$

where $W^O \in \mathbb{R}^{d' \times d'}$ is a learnable weight matrix.

Feed-Forward Network (FFN)

Each attention layer is typically followed by a position-wise feed-forward network:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{10}$$

where $W_1 \in \mathbb{R}^{d' \times d_{ff}}$, $W_2 \in \mathbb{R}^{d_{ff} \times d'}$, $b_1 \in \mathbb{R}^{d_{ff}}$, and $b_2 \in \mathbb{R}^{d'}$ are learnable parameters.

Layer Normalization and Residual Connections

Each sub-layer (attention and FFN) employs a residual connection followed by layer normalization:

$$H^{(l+1)} = \text{LayerNorm}(H^{(l)} + \text{Sublayer}(H^{(l)})) \tag{11}$$

where Sublayer is either the multi-head attention or the FFN.

Edge Feature Integration

GraphTransformer, GraphiT and SAN incorporate edge features:

1. In attention computation:

$$A_{ij} = \text{softmax}\left(\frac{q_i^T k_j + f(e_{ij})}{\sqrt{d_k}}\right) \tag{12}$$

where f is a learnable function (e.g., a small neural network) that projects edge features.

2. In value computation:

$$v_{ij} = V_i + g(e_{ij}) \tag{13}$$

where g is another learnable function.

Global Node

Some architectures introduce a global node v_g connected to all other nodes to capture graph-level information:

$$h_g^{(l+1)} = \text{Attention}(h_g^{(l)}, H^{(l)}) \tag{14}$$

Output Layer

The final layer depends on the task:

- For node classification: $Y_{node} = \text{softmax}(H^{(L)} W_{out} + b_{out})$
- For graph classification: $y_{graph} = \text{MLP}(\text{Pool}(H^{(L)}))$

where Pool is a pooling operation (e.g., mean, sum, or attention-based pooling).

Training

The model is typically trained end-to-end using backpropagation to minimize a task-specific loss function, such as cross-entropy for classification or mean squared error for regression.

3 Kolmogorov-Smirnov Test

This section provides additional details on the Kolmogorv-Smirnov (KS) test [8] used to analyze the distribution of activations. The KS test is a non-parametric test that compares the cumulative distribution functions of two samples. It is used to compare a sample with a reference probability distribution (one-sample KS test) or to compare two samples (two-sample KS test) with each other.

In our study, we utilized the KS statistic to compare the distribution of activation values before and after training (i.e. base against trained model), identifying Massive Activations (MAs). We primarily used the one-sample KS test to assess the goodness of fit between our observed activation distributions and a theoretical gamma distribution.

3.1 One-Sample Kolmogorov-Smirnov Test

The one-sample KS test can typically be formulated as follows:

3.1.1 Null Hypothesis

The null hypothesis for the one-sample KS test is:

H_0: The sample data follows the specified distribution (in our case, a gamma distribution).

3.1.2 Test Statistic

The KS statistic D_n is defined as the supremum of the absolute difference between the empirical cumulative distribution function (ECDF) $F_n(x)$ of the sample and the cumulative distribution function (CDF) $F(x)$ of the reference distribution:

$$D_n = \sup_x |F_n(x) - F(x)| \tag{15}$$

where $\sup_x$ denotes the supremum of the set of distances.

3.1.3 Empirical Cumulative Distribution Function

For a given sample $x_1, x_2, ..., x_n$, the ECDF is defined as:

$$F_n(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{x_i \leq x} \tag{16}$$

where $\mathbf{1}_{x_i \leq x}$ is the indicator function, equal to 1 if $x_i \leq x$ and 0 otherwise.

3.1.4 Critical Values and p-value

The distribution of the KS test statistic under the null hypothesis can be calculated, which allows us to obtain critical values and p-values. The null hypothesis is rejected if the test statistic D_n is greater than the critical value at a chosen significance level α, or equivalently if the p-value is less than α.

3.2 Application to MAs Detection

In our experiments, we used the KS statistic to assess whether the distribution of activation ratios in our GNNs follows a gamma distribution. The process is as follows:

1. We computed the activation ratios for each layer of our models, as defined in Equation (1) of the main paper.
2. We took the negative logarithm of these ratios to transform the distribution.
3. We fit a gamma distribution to this transformed data using maximum likelihood estimation.
4. We performed a one-sample KS test to compare our sample data to the fitted gamma distribution.

The KS test statistic provides a measure of the discrepancy between the observed distribution of activation ratios and the theoretical gamma distribution. A lower KS statistic indicates a better fit, suggesting that the activation ratios more closely follow the expected distribution.

3.3 Interpretation in the Context of MAs

Following the described procedure in Section 3.2, we employed the KS statistic as quantitative/statistical measure to detect the presence of MAs:

- For untrained (base) models, we typically observed low KS statistics, indicating that the activation ratios closely follow a gamma distribution.
- For trained models exhibiting MAs, we often saw higher KS statistics. This indicates a departure from the gamma distribution, which we interpret as evidence of MAs.
- The magnitude of the KS statistic provided a quantitative measure of how significantly the presence of MAs distorts the expected distribution of activation ratios.

Moreover, we complemented our KS statistic results with visual inspections of the distributions and other analyses as described in the main paper.

References

[1] John J Irwin, Teague Sterling, Michael M Mysinger, Erin S Bolstad, and Ryan G Coleman. Zinc: a free tool to discover chemistry for biology. *Journal of chemical information and modeling*, 52(7):1757–1768, 2012.

[2] Andreas Mayr, Günter Klambauer, Thomas Unterthiner, and Sepp Hochreiter. Deeptox: toxicity prediction using deep learning. *Frontiers in Environmental Science*, 3:80, 2016.

[3] Ruili Huang, Menghang Xia, Dac-Trung Nguyen, Tongan Zhao, Srilatha Sakamuru, Jinghua Zhao, Sampada A Shahane, Anna Rossoshek, and Anton Simeonov. Tox21challenge to build predictive models of nuclear receptor and stress response pathways as mediated by exposure to environmental chemicals and drugs. *Frontiers in Environmental Science*, 3:85, 2016.

[4] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

[5] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs, 2021.

[6] Grégoire Mialon, Dexiong Chen, Margot Selosse, and Julien Mairal. Graphit: Encoding graph structure in transformers. *arXiv preprint arXiv:2106.05667*, 2021.

[7] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021.

[8] Indra Mohan Chakravarti, Radha Govira Laha, and Jogabrata Roy. Handbook of methods of applied statistics. *Wiley Series in Probability and Mathematical Statistics (USA) eng*, 1967.