Effective Receptive Field of Graph Neural Networks

Andrei Nicolicioiu[*]
Bitdefender
anicolicioiu@bitdefender.com

Iulia Duta[*]
Bitdefender
iduta@bitdefender.com

Abstract

Long range interactions in the scene represent key elements in solving numerous visual tasks. While convolutional models have been extensively studied, newer methods like graph neural networks have been gaining track in recent time. We investigate the capacities of current models to capture distant connections by studying theoretically and empirically the effective receptive field of graph models.

We prove that the outputs of Graph Convolutional Networks are influenced by regions where it is highly probable to arrive in several steps, while Self-Attention depends more on the regions that are strongly correlated with the considered location. We show that graph models achieve superior results when distant connections are essential and we empirically confirm our theoretically findings.

1. Introduction

Visual tasks requires a complex analysis of the scene consisting of different elements at multiple scales, with different spatial relationships and various levels of semantic meaning. Recently, graph neural networks have been used with success in computer vision tasks, as they can better model relationships, even at large distances, and are able to connect entities that are semantically correlated.

Different spatial modeling are needed in various vision tasks, with some tasks being solvable with local information while most of them need a larger view of the scene. For example, in classification the entire object must be covered, in tracking both the object and some background information must be considered, while in activity recognition the tasks could be dependent on the relationships between multiple objects.

A model should be capable of looking at multiple regions in the scene in order to incorporate relevant interactions into its representation. But having a global view should not cripple the capabilities of the model to learn meaningful representations. The model should have enough capacity while still being easily optimizable.

We analyse the *effective receptive field* (ERF) [25] of different graph based models, representing the regions in the input that have a large influences for an output location p of the model. The ERF of convolutional models is shown to be Gaussian distributed [25], meaning that the produced features mainly depend on its close neighbourhood.

As stated above, there are cases in computer vision where a view of the entire scene is necessary, and such thing could be achieved by using a large number of convolutional layers that results in an ERF distributed by a Gaussian with higher standard deviation. This comes at the cost of increasing computational resources and making the model harder to optimize. Such issues could be alleviated by using Graph Neural Networks (GNNs), with long range connections, that have proven good results in many vision tasks such as visual question answering [28], video action recognition [32, 33, 26] and tracking [13].

In this work, we focus on Graph Convolutional Networks (GCNs) and Self-Attention methods and observe that their effective receptive field is more flexible and could cover the entire input space.

We prove that GCNs have effective receptive field that depends on the fixed connections in the graph, being able to see the entire input, but having a fixed structure, without the possibility to adapt to the input. On the other hand, Self-Attention models could cover the same large zones while being more influenced by correlated regions.

Main contributions:

- We prove that the effective receptive field of a Graph Convolutional Network depends on the probability of reaching every input starting from the considered node, as given by A^N.
- We prove that the effective receptive field of a Self-Attention layer is proportional with the correlations of the considered node with every input node.
- We empirically show that the effective receptive field of graph models are consistent with our theoretically

[*]equal contribution

results and we experimentally show that they obtain superior results in tasks where long-range connections are essential.

2. Related Work

Convolutional neural networks have become ubiquitous in computer vision tasks, from classification[18, 8, 6] to segmentation[7], detection [16, 27] or tracking [10, 29] for both image and video tasks.

We are interested in the receptive field [2] of neural models, representing the region in the input that affects in any way the output. In convolutional layers, the receptive field grows linear with the number of layers with stride one and multiplicatively by using layers with stride higher than 1.

The receptive field can be increased by using different types of convolutional layers. Dilated convolutions [36] use a sparse kernel, effectively aggregating more distant points, in turn expanding the field of view. Similarly, in deformable convolutions [9], the kernel of the convolution is sparse but the location of each point in the kernel is learned.

The effective receptive field [25], representing the region in input image that has a non-negligible impact for an output location, have been studied for convolutional models showing that the effective receptive fields is Gaussian distributed. We study the behaviour of graph neural networks in regards to the effective receptive field.

Graph neural networks have been recently used in many domains where the data has a non-uniform structure [4, 14, 5]. They have been applied with superior results on tasks such as molecule generation [24], document categorization [19, 11, 23, 31], physical interactions [3], motion forecasting [21], video question answering [28], action recognition [32, 33].

There have been a number of works that generalize convolutional neural networks for non-uniform structured data, by learning in the spectral domain of a graph [5, 19, 11]. Such methods are approximated by Graph Convolutional Networks [23] that send linear projection of node features as messages between all connected nodes.

The general framework used in graph neural networks [14] consists of processing data arranged in a graph structure by creating messages between each pair of connected nodes, aggregating the messages from a neighbourhood and updating the state of each node using the received information. Different aggregation methods have been proposed, such as mean or average pooling [34] or pooling by recurrent networks[15]. In order to selectively attend to some subset of nodes, attention mechanisms have been used to aggregate the neighbourhood [31].

The idea of forming relations from visual elements given by convolutional features appears in [28] where they process pairs of features from every location in order to capture distant interactions in the scene.

Current language tasks largely adopted graph methods [12] in the form of Self-Attention [30] models. The Transformer model [30] uses Self-Attention implemented as dot product between projected features to attend differently to word features. This can be seen as sending messages in a fully connected graph, with the messages weighted by a similarity measure. For vision tasks, the Non-Local [32] method creates a graph from convolutional features and uses Self-Attention mechanism in order to have long range connection between entities in the video.

The authors of JK-Net [35] suggest that different samples need to inspect a neighbourhood of various size. In order to do this, they analyse the *influence score* of the nodes in a Graph Convolutional Network, showing that the influence of nodes in expectation is the random walk distribution. We obtain similar results for GCN, and also extend the analysis for Self-Attention layers.

3. Effective receptive field of GNNs

The *effective receptive field* (ERF) measures how much each input location $i = (h_i, w_i)$ impacts the features for the output location $p = (h_p, w_p)$. Formally, following [25], ERF is the derivative of output feature y_p with respect to each input features x_i. A large value of the effective receptive field indicates that a small change in the input value has a big impact on the output results.

We are interested in the importance of each location $i = (h_i, w_i)$ in the input $X \in \mathbb{R}^{H_1 \times W_1 \times C_1}$ for a certain location $p = (h_p, w_p)$ in the output $Y \in \mathbb{R}^{H_2 \times W_2 \times C_2}$. We sometimes denote the location p as the central location, but it can be any fixed position in the output. The exact influence of the channels is not relevant for our analysis, we are only concerned with the spatial locations. Thus we add the influence of all the channels of each input X_i to every channels of the central output Y_p as in Equation 19.

$$\text{erf}(p, i) = \sum_{r=1}^{c} \sum_{s=1}^{c} \frac{\partial Y_{pr}}{\partial X_{is}} \in \mathbb{R} \tag{1}$$

By setting $\mathcal{L} = \sum_r Y_{pr}$, we obtain $\nabla \mathcal{L}_Y = \frac{d\mathcal{L}}{dY}$ a $n \times c$ matrix with ones on the p-th line and zeros otherwise and we can write the ERF as:

$$\text{erf}(p, i) = \sum_{s=1}^{c} (\nabla \mathcal{L}_X)_{i,s} \in \mathbb{R} \tag{2}$$

We will now analyse the effective receptive field of Graph Convolutional Networks and Self-Attention layers and observe their behaviour. We give the main results regarding the effective receptive field in the following sections, and leave in the Appendix the complete derivations.

We study graph methods for vision tasks, where each node contains visual information from a specific region in the scene. We use a convolutional backbone model to extract feature volumes $X \in \mathbb{R}^{h \times w \times c}$ and use it to create $n = h \times w$ nodes $n_i \in \mathbb{R}^c$, each corresponding to a location. For video processing, we could integrate the time dimension in a complex way, but as we are only interested in the influence of spatial locations, we simplify the method by treating them independently.

We create the structure of the graph by linking the nodes in various ways, such as connecting only the neighbouring nodes from the grid structure of the convolutional features, or connecting each pair of nodes.

3.1. ERF of Graph Convolutional Networks

Given a graph $G = (V, E)$, where all nodes n_i, arranged in a matrix $X \in \mathbb{R}^{n \times c}$, are connected by an adjacency matrix $A \in \mathbb{R}^{n \times n}$, a Graph Convolutional Network [23] with L layers is defined as:

$$\begin{aligned} X^{(l+1)} &= \sigma\big(AX^{(l)}W_l\big) \\ Y = X^{(L+1)} &= AX^{(L)}W_L, \end{aligned} \tag{3}$$

where $W_l \in \mathbb{R}^{c \times c}$ is a linear projection at layer l, $Y \in \mathbb{R}^{n \times c}$ is the output nodes features of the GCN and σ is the ReLU activation.

We begin by studying the effective receptive field of GCN models without ReLU activations, then show how the properties generalises for models with ReLU activations. In order to observe the general behaviour of the model, we compute the expected value of the ERF for a given model, with fixed weights, by varying only the input, or for all possible model configurations by varying both the input and the weights.

Theorem 1. *In a Graph Convolutional Network with L layers, without ReLU non-liniarities, the effective receptive field of an output node Y_p with respect to every input node X_i depends on the probability of reaching the X_i node starting from Y_p as in Equation 23.*

We obtain the ERF by summing the partial derivatives $\frac{\partial Y_p}{\partial X_i}$ according to Equation 19. As we do for the rest of the paper, we give the final result and leave for the Appendix the complete derivations.

$$\frac{\partial Y_p}{\partial X_i} = (A^L)_{p,i}(\prod_{l=1}^{L} W_l)^T \in \mathbb{R}^{c \times c} \tag{4}$$

Alternatively, using Eq. 20 we can obtain the ERF from:

$$\nabla \mathcal{L}_X = (A^L)^T \nabla \mathcal{L}_Y (\prod_{l=1}^{L} W_l)^T \in \mathbb{R}^{n \times c} \tag{5}$$

Constant weights. We study the behaviour of the model for various inputs by computing the expected effective receptive field for a given set of parameters, by averaging over all possible inputs. For this case we set $W = J_c$, a $c \times c$ all-ones matrix, obtaining:

$$\mathbb{E}_X[\text{erf}(p, i)] = c^{(L+1)}(A^L)_{p,i} \tag{6}$$

Since the term $c^{(L+1)}$ is constant with respect to the position in the input, we are only interested in the second term $(A^L)_{p,i}$, giving us an order of the importance of each location. Thus the location i is proportional to the probability of reaching i from p in L steps by a random walk. Thus the most impactful locations are the direct neighbours then comes the higher order ones in decreasing order. This is similar to the behaviour of convolutions, but in GCN case, we can explicitly set the connections, being preferred when a good prior on the structure is known.

Random weights. In order to consider the bias of the GCN architecture, regardless of the weights, we compute the expected ERF by varying both the input and the weights. We consider a GCN with random weights, sampled independently from a standard Gaussian $W_l \sim \mathcal{N}(0, I)$.

We compute the expected values of the ERF over all the inputs and all possible weights. Using the fact that the weights in different layers are independent and all of them are centered in zero we obtain:

$$\mathbb{E}_{X,W}[\text{erf}(p, i)] = \sum_{r=1}^{c}\sum_{s=1}^{c}(A^L)_{p,i}(\prod_{l=1}^{L}\mathbb{E}[W_l])^T_{r,s} = 0 \tag{7}$$

For every ERF given by a set of weights W, we have another model with weights $-W$ that has the same ERF with opposite sign, adding zero to the expectation, but their absolute values are still of interest. Thus it is relevant to observe the magnitude of the ERF variations, given by its variance.

$$\mathbb{V}_{X,W}[\text{erf}(p, i)] = c^{(L+1)}[(A^L)_{p,i}]^2 \tag{8}$$

We observe that the ERF variance in the preceding Equation is also proportional to $(A^L)_{i,j}$ similar with the constant case in Equation 37.

We now analyse effective receptive field for GCN models with ReLU activations and obtain similar observations.

Theorem 2. *In a Graph Convolutional Network with L layers with ReLU non-linearities, the effective receptive field of an output node y_p with respect to every input node x_i depends on the probability of reaching the x_i node starting from y_p as in Equation 51.*

By summation as in Equation 20 we obtain the ERF from:

$$\begin{aligned}
\nabla\mathcal{L}_X^{(l)} &= A^T(\nabla\mathcal{L}_X^{(l+1)} \odot Z^{(l)})W_l^T && \in \mathbb{R}^{n\times c} \qquad (9)\\
\nabla\mathcal{L}_X^{(L)} &= A^T\nabla\mathcal{L}_Y W_L^T && \in \mathbb{R}^{n\times c}\\
Z^{(l)} &= \mathbb{I}(AX^{(l)}W_l > 0) && \in \mathbb{R}^{n\times c}
\end{aligned}$$

where $\nabla\mathcal{L}_X^{(l)} = \nabla\mathcal{L}_{X^{(l)}}$

At every layer, the maximum possible effective receptive field is the same as in the case without activations, but the impact of every input could be blocked by the ReLU non-linearity, lowering the importance of certain inputs.

In the forward pass, the activations of the model given by ReLU are $Z \odot X$, where Z is a binary gate being equal to one for positive features and zero otherwise. The same Z gates the impact of the input features in the ERF.

In the following, we assume that the ReLU gates $Z^{(l_1)}$ and $Z^{(l_2)}$ are independent for every two layers $l_1 \neq l_2$. Based on the assumption in [17] that the activations $X_i^{(l)}$ have a symmetric distribution around zero at every layer, the $Z^{(l)}$ gates follows a Bernoulli distribution with equal probability. Thus at each layer, the expected value $\mathbb{E}[Z]$ of each channel is $\frac{1}{2}$, with the same expected values gateing the ERF.

For simplicity, we only consider the case of a GCN with two layers with single channels, corresponding to scalar weights. We analyse the same cases as before, constant weight and variable weights. We note that, for scalar case $\text{erf}(p,i) = (\nabla\mathcal{L}_X)_i = \frac{\partial Y_p}{\partial X_i}$.

Scalar weights. By considering a single channel $c = 1$ in Eq. 51, the ERF of the GCN in the ReLU case is given by:

$$\begin{aligned}
\nabla\mathcal{L}_X^{(l)} &= A^T(\nabla\mathcal{L}_X^{(l+1)} \odot Z^{(l)})w_l && \in \mathbb{R}^{n\times 1} \qquad (10)\\
\nabla\mathcal{L}_X^{(L)} &= A^T\nabla\mathcal{L}_Y w_L && \in \mathbb{R}^{n\times 1}\\
\nabla\mathcal{L}_Y &= (I_n)_{:p} && \in \mathbb{R}^{n\times 1}
\end{aligned}$$

Constant Weights. For a GCN with two layers, single channel, with input X that is symmetrically distributed around zero, we compute the expected value of the ERF as follows:

$$\mathbb{E}_X[\text{erf}(p,i)] = \mathbb{E}_X[\frac{\partial Y_p}{\partial X_i}] = \frac{1}{2}(A^2)_{p,i}\prod_{l=2}^{1} w_l \qquad (11)$$

We can extend this to L layers where we obtain $(\frac{1}{2})^{L-1}(A^L)_{p,i}\prod_{l=L}^{1} w_l$ and observe that this is the same ERF as in Equation 37, but it is weighted by $\frac{1}{2}^{L-1}$ equally reducing the importance of each location exponentially in the number of layers. Thus the same order of importance is maintained between node locations.

Random Weights. Same as in the case without ReLU, we analyse the ERF for all possible weights w_l sampled from a standard Gaussian $\mathcal{N}(0,1)$ in order to observe the bias of the architecture. Firstly, we compute the expected value of the ERF as follows:

$$\begin{aligned}
\mathbb{E}_{X,W}[\text{erf}(p,i)] &= \mathbb{E}_{X,W}[\frac{\partial Y_p}{\partial X_i}]\\
&= \mathbb{E}_{X,W}[A_p(Z \odot A_i^T)w_1w_2)]\\
&= \sum_k A_{pk}A_{ki}\mathbb{E}[Z_k w_1]\mathbb{E}[w_2] = 0 \qquad (12)
\end{aligned}$$

This way, we see that each component k in the expected values of the effective receptive field is proportional to the probability of reaching node i from node p in two steps, by passing through each node k. Because of the symmetric form of w around zero, same as before, the expected values is zero, and we are interested in computing the variance.

The variance of the ERF depends on the probability of reaching node i from node p in two steps $A_{p,u}A_{u,i}$ and also on the covariance $\text{Cov}(Z_u, Z_v)$ between the gates of each pair of nodes u, v. The exact equations can be found in the Appendix.

We have shown that the effective receptive field of Graph Convolutional Networks depends on the input only by the gating induces by the ReLU activation, and it is strongly influence by the probability of reaching the locations i starting from p in L steps, as given by A^L matrix. Thus the most important aspect for GCN regarding the ERF is the structure of the graph.

3.2. ERF of Self-Attention layer

Given a fully connected graph $G = (V, E)$, with nodes n_i arranged in a matrix $X \in \mathbb{R}^{n\times c}$, Self-Attention layer is defined as:

$$Y = \text{softmax}\left(\frac{(XW_q)(XW_k)^T}{\sqrt{c}}\right)(XW_v) \qquad (13)$$

where $Y \in \mathbb{R}^{n\times c}$, and $W_q, W_k, W_v \in \mathbb{R}^{c\times c}$.

In computer vision tasks it has been shown [32] that the softmax activation is not essential in Self-Attention layers, thus we do not use it in our formulation. The dot product similarity is only scaled by the number of nodes but, for clarity, we omit it as it does not affect our results, since it only uniformly scales every position in the ERF.

We follow the same steps as in the Graph Convolutional Network case, firstly we compute the effective receptive field, then investigate the architectural bias by observing the behaviour of model while varying inputs and weights.

Theorem 3. *In a Self-Attention layer, without softmax, the effective receptive field of a node Y_p with respect to every* *other node X_i depends on the similarity between linear projections of the two nodes $(X_pW_q)(X_iW_k)^T$ and the outer product $(X_iW_v)^T(X_pW_q)$.*

By summation as in Equation 19 we obtain the ERF from:

$$\frac{\partial Y_p}{\partial X_i} = W_v^T(X_pW_q)(X_iW_k)^T + (X_iW_v)^T(X_pW_q)W_k^T, \quad \forall i \neq p \quad (14)$$

$$\frac{\partial Y_p}{\partial X_p} = W_v^T(X_pW_q)(X_pW_k)^T + (X_pW_v)^T(X_pW_q)W_k^T + \sum_{j=1}^{n}(X_jW_v)^T(X_jW_k)W_q^T \quad (15)$$

This way much higher importance is given to the self location p, as the Equation 73 contains the two terms of Equation 72 and also an outer product for each node in the graph. This suggest that a more carefully normalisation of the central location p should be made. We also observe this discrepancy between the magnitude of the ERF of the central node and the others in our experiments in Section 4.

In the following we consider the case of scalar weights to analyse the expected effective receptive field when varying the input and also the weights.

Constant Weights. For a given model with parameters w_q, w_k, w_v as scalar constant, we obtain the ERF from:

$$\begin{aligned} \mathbb{E}_X[\text{erf}(p,i)] &= 2w_qw_kw_v\text{Cov}(X_p, X_i), \qquad \forall i \neq p \\ \mathbb{E}_X[\text{erf}(p,p)] &= (n+2)w_qw_kw_v\mathbb{V}[X_p] \end{aligned} \quad (16)$$

The expected receptive fields, for each pair of nodes (p, i) becomes proportional with the covariance between their features. Because of this, although it can see all the input, when the model is computing features for position p, it takes more into account the nodes that are highly correlated with the p node. This suggest that Self-Attention models does indeed take into account regions in the input that are relevant for the current location, capturing complex interactions in the scene more easily.

The position p in the input receives more importance, linear in the number of nodes, causing a discrepancy between the central node p and other nodes i. In order to have the same contribution to the output p, larger changes should be made at input i compared to input p. This makes the model somehow insensitive to locations in the input different from the considered node.

Random weights. We will compute the expectation of the effective receptive field with respect to the input and the parameters. In order to do this, we will make the same assumptions as before, that the weights are randomly sampled independently from a standard Gaussian $w_q, w_k, w_v \sim \mathcal{N}(0, I)$.

$$\begin{aligned} \mathbb{E}[\text{erf}(p,i)] &= 2\mathbb{E}_{X,W}[w_qw_kw_v]\text{Cov}(X_p, X_i) \\ &= 0, \qquad \forall p \neq i \\ \mathbb{E}[\text{erf}(p,p)] &= (n+2)\mathbb{E}_{X,W}[w_qw_kw_v]\mathbb{V}[X_p] = 0 \end{aligned} \quad (17)$$

The expected derivatives are again proportional with the covariances between the nodes but, because the weights are symmetric around zero, the final expected values is zero in both cases. Thus, to be able to measure the expected receptive field when varying the weights of the model, we compute the variance.

$$\begin{aligned} \mathbb{V}_{X,W}[\text{erf}(p,i)] &= 4\mathbb{V}[w_qw_kw_v](\text{Cov}(X_p^2, X_i^2) + \mathbb{V}^2[X_p]) \\ \mathbb{V}_{X,W}[\text{erf}(p,p)] &= (n+8)\mathbb{V}[w_qw_kw_v](\mathbb{V}[X_j^2] + \mathbb{V}^2[X_j]) + \\ &+ \mathbb{V}[w_qw_kw_v]\sum_{j,l;l\neq p}\sum b_jb_l(\text{Cov}(X_j^2, X_l^2) + \mathbb{V}^2[X_j]) \end{aligned}$$

where we denote $b_i = 1, \forall i \neq p$ and $b_p = 3$ (18)

For the case when $p \neq i$, the variance depends on the covariance between the squared features of the nodes p and i and the variance of the central node p. Thus the order of importance for input nodes i is given by their covariance with the input node p. Similar, the importance of the input node p depends on its variance, but also on the covariance between all pairs of input nodes.

In this section we have proved that, for Graph Convolutional Networks, the effective receptive field depends on the probability of reaching each node in L steps, starting from the considered node p. For the Self-Attention case, the effective receptive field for the node p depends on the covariance between all the input features i and the input p.

Large effective field can be achieved in both graph models. Self-Attention has by default connections between all locations, weighted by the correlations between them whereas, for the GCN, we can capture long range influences by connecting distant nodes in the adjacency matrix. Self-Attention could be made more local, by masking the similarity matrix $(XW_q)(XW_k)^T$, thus only considering a subset of connections.

Convolutional networks have local effective receptive field, Gaussian distributed [25], thus they are appropriate for cases where all the needed information is concentrated in a local neighborhood. In contrast, graph methods are suited for cases where there are non-local interactions that should be modeled. GCN is preferred in cases where a structure is known a priori, while Self-Attention is more flexible and appropriate when the similarity between node features are relevant.

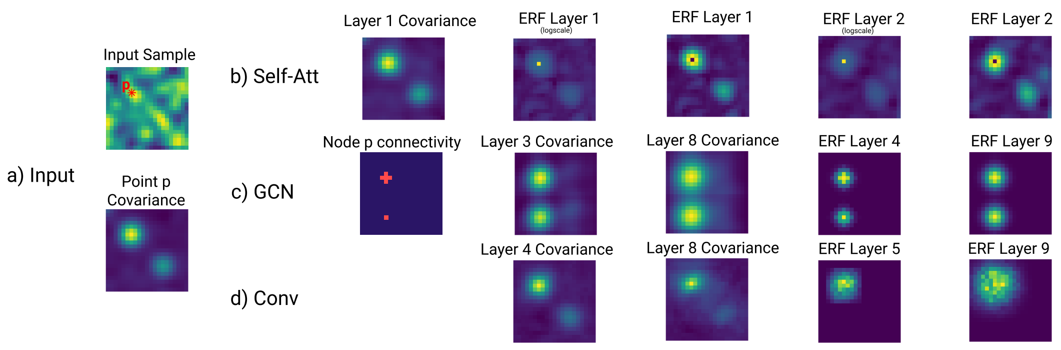

Figure 1. For the synthetic data experiment, we show the covariance of the input data and the ERF and covariance of intermediate features of the three models at different layers. We always consider the same point (situated in the upper left corder) for computing both the ERF and the covariances. We observe that for Self-Attention the effective receptive field follows the correlations in the input and that for GCN model it represents the probability of reaching each node according to its adjacency matrix.

4. Experiments

We compared the performance and the effective receptive field for GCN, Self-Attention and convolutional network in three sets of experiments. Firstly, we generate random data with a certain correlation to empirically confirm our theoretically result that Self-Attention layers have effective receptive fields that follows the correlation of the input. Then, we measure the performance and the ERF on two datasets, where relations are essentials, one for video classification task and the other for image question answering.

4.1. Syntetic correlations

We designed a set of experiments that tests our theoretically proven claim that the Self-Attention layer has effective receptive field influenced by the correlations in the input features. We also want to see if our claim generalized for multiple layers.

In natural images, there is a strong bias towards locality, with high correlations between neighbouring pixels. We created a synthetic dataset with random pixels, where the covariance of each pixels with all of the others follows a mixture of two Gaussian, one centered in the pixel location and the other centered in a distant point, as seen in Figure 1 for a considered pixel. This way we could investigate if the models are able to capture such long range influences in the input.

Each sample is a $H \times W$ map, with C independently channels, where $H = W = 21$ and $C = 32$. The correlation between a pixel $p = (h_0, w_0)$ and every other pixel is the same at every channel.

In order to observe the bias of the architectures, we compute the effective receptive field of different randomly initialised models. We obtain the ERF by computing the gradients of the output feature y_p with respect to all the input features x_i and summing all the channels as in Equation 20. Because we are interested to observe the relative magnitudes of the ERF, we normalise it by its sum.

We compare three types of models: convolutional network, Self-Attention and graph convolutional networks. We create one convolutional neural network with 9 layers, with stride 1 and 24 filters, a Self-Attention network with 2 layers 256 hidden channels and one GCN with 9 layers, all of them receiving as input the raw data.

For each model, at each layer, we sum the activation maps and compute the covariance between the considered node p and every other location of the resulting map. For self attention and convolutional network, at the forward pass, the covariance is mainly preserved in all the layers as shown in Figure 1. For better visualization, we show the ERF of Self-Attention model twice, once in the log-scale and once by ignoring the considered node used for the ERF.

We compute the effective receptive field for every sample in the dataset and calculate the average across the dataset. For the convolutional network, the computed ERF is Gaussian, in agreement with the proven claim in [25]. We computed the ERF at the forth and the ninth layer of the network and, as expected, we observed that the ERF increased with the number of convolutional layers.

For the Self-Attention model, we also compute the effective receptive field and observe that it agrees with our theoretical findings, more specific it follows the covariance of the input. Also, the Equations 84, 18, are reflected in the experimental results, as the ERF has a much larger value in the central p position. We observed that the same conclusions maintain for two layers of Self-Attention.

We create a GCN model, with a special adjacency matrix, that connects each node to its 4 neighbours in the grid and also with a single distant pixel. Because there are two separate regions connected by the graph, the features produced at their locations become correlated, as shown in Fig-

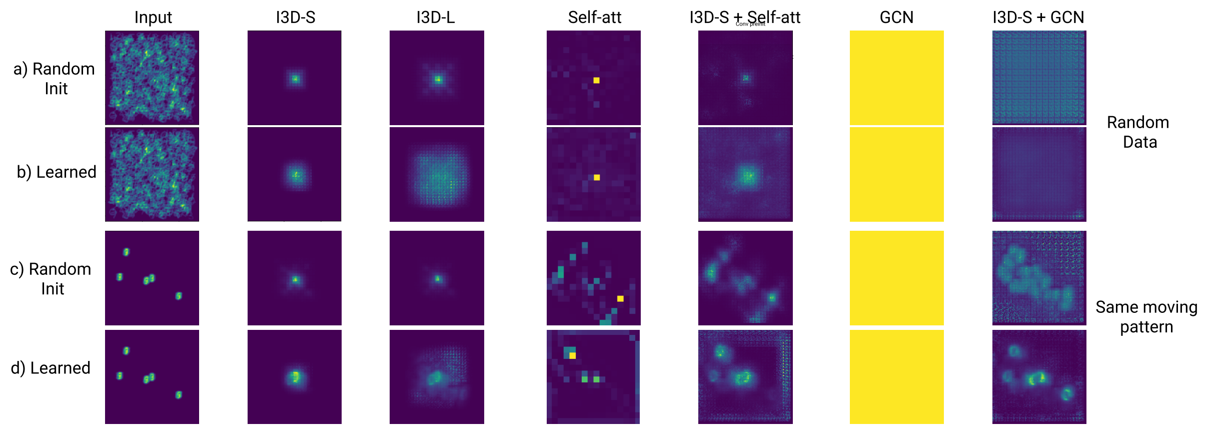

Figure 2. We present the effective receptive field for different models. Each column represents a model, and we consider input data from two subsets of the dataset, one with digits moving different in every video and one with different digits following the same pattern in every video. We observe that Self-Attention models and GCN have larger ERF.

ure 1 b). In agreement with Equation 57, the ERF at layer L represents the probability to reach each position, starting from the considered node. In this case, our artificial structured imposed in the graph connectivity is observed in the ERF, ignoring the real distribution of the data.

4.2. SyncMNIST experiments

In this set of experiments we want to compare the behaviour of convolutional methods and graph methods on a simple video dataset, where interactions between entities at various locations play a crucial role. We want to observe the distribution of effective receptive fields of these models, and how it affects their final performance.

For this, we use the SyncMNIST [26] dataset and adapt it in order to create multiple tasks of increasing difficulty. It contains videos of moving digits in a scene, and the task is to find a pairs of digits that moves synchronously, resulting in a classification task with 46 classes. Each video consists of 10 frames, of size 128×128, created by moving digits of size 14×14 on a uniform background. We use videos with 5 moving digits, with two of them moving synchronous.

We create 4 datasets of $600K$ videos, each of them having a maximum distance of $20, 60, 80$ or 100 pixels between synchronously digits . We analyse the results on each datasets and see that graph methods maintain their performance regardless of the distance between digits, while convolutional models decrease their performance with the distance between digits.

We experiment with three main architectures. The first one is a 3D convolutional model, I3D [6] adapted from a smaller ResNet [18] model in order to fit with our task. The others are graph methods, a Self-Attention model and a GCN, that use the I3D model as a backbone.

I3D We create two variants of I3D-ResNet. The first one, denoted as I3D-S has 3 stages, each having 2 blocks resulting in 4.1M parameters while the second one, denoted as I3D-L has 3 stages, each with 3 blocks for a total of 6.1M parameters.

Self-Attention We use as a backbone model the I3D-S model defined above and Self-Attention layers to process its intermediate layers. Following the Non-Local [32] method, we insert Self-Attention layers in the I3D, after the second stage, using a residual connection. As we are only interested in the spatial processing of the Self-Attention, we process each time step of the features independently, but with the same Self-Attention parameters.

The *res3* $\in \mathbb{R}^{5\times16\times16\times128}$ features from I3D are projected into a smaller 32 dimensions and normalized using BatchNorm [20]. For each time step, the features are processed independently, using a Self-Attention model with two layers and 32 channels, projected back to the initial 128 dimension and normalized before adding them with a residual connection to the *res3* I3D features. The resulting features are passed through the rest of the I3D model.

GCN We use the same pipeline as above, with the same I3D-S used as backbone, but replace Self-Attention with GCN layers. We use a GCN with two layers with 32 features to aggregate the I3D intermediate features and use a fully-connected adjacency matrix.

4.2.1 Results

We show the performance of our models in Figure 3, each line representing the accuracy of a model on all datasets. We observe that both convolutional I3D models have a drop in performance when the synchronous digits are far apart, with

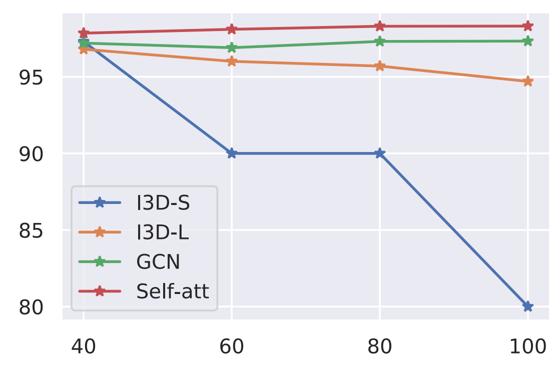

Figure 3. Results on SyncMNIST datasets. We observe that the I3D models have a drop in performance on the datasets with longer distance between synchronous digits, while graph methods are much more stable, being unaffected by the distance.

I3D-S having a larger decrease due to its smaller receptive field. On the other hand, the two graph models are more stable in their performance, without being affected by the distance between digits.

In Figure 2 we present the effective receptive field for our models, always computed from the center of the frame in the middle of the video. They are calculated with respect to the input image, with the exception of Self-Attention and GCN columns, where we show the ERF with respect to the input received by the graph layers. We average the maps on two subsets of our datasets, one being the evaluation set and the other where various digits are moving using the exact same patterns in all videos. On the random subset, we observe the general reach of the ERF, whereas on the same moving patterns we see more clearly the influence of specific regions for each model.

For the convolutional models, represented by I3D-S and I3D-L in our experiments, the effective receptive field is Gaussian and become spreader after training. The GCN has constant uniform ERF, since it has connections between each locations.

We can see that the Self-Attention also covers the entire input, with its ERF. For the random subset, the ERF is higher in the considered node, since on average it correlates more with itself. It is interesting to note that Self-Attention gives a higher importance to the location of the digit that is indeed in sync with the digit in the center of the video.

4.3. Sort-of-CLEVR

We conduct experiments on Sort-of-CLEVR dataset, to observe the differences in performance between the Self-Attention and the convolutional models, on a task based on long distance relations between entities.

Sort-of-CLEVR dataset [28] is a simplified version of the CLEVR[22] dataset, used in visual question answering task. It contains 10000 images. each one with 6 objects of different colors assigned with a randomly chosen shape: square or circle. For each image, a set of 20 questions, splitted in 10 relational questions and 10 non-relational questions, are annotated with the appropriate answer. The relational set of questions are especially designed such that, in order to be able to infer the right answer, the model should be able to understand the relationship between the queried object and the rest of the scene.

For our experiments, we create a backbone model, consisting of 2 convolutional layers, each one followed by a Batch Normalization. For the convolutional neural network, the features extracted from the backbone model are concatenated with the embedding of the question and used as input for two additional convolutional layers. The resulting map are linearized and used to predict the answer. We trained two versions of convolutional networks using different stride in the convolutional layers. In this way, the to models have significantly different receptive field. In Table 1 we denoted by Conv-S the model with smaller effective receptive field, and by Conv-L the other one.

For the Self-Attention model, we create a completely connected graph from the features extracted using the backbone model. Each node receives information from a specific location of the activation map, together with the embedding of the question. The graph is processed using one Self-Attention layer, and the representation of the nodes are then concatenated and used to obtain the prediction.

We trained all models in Tensorflow [1], with Adam optimizer, using both the relational and non-relational dataset for training. We compare only the accuracy for the relational split, as we are interested to study the capabilities of the models to capture long range interactions.

In Table 1 we show the results for the 2 versions of convolutional networks and for the Self-Attention network. We observe that the Self-Attention model obtain the best results with an improvement of 7.5%, since it is the only model that covers the entire image from the first layer. Between the two convolutional network, the one with the broader receptive field obtain better results even if the number of weights are smaller due to the reduced activation map used in the final fully connected layer. This results suggests a larger ERF is essential in tasks involving relations between entities.

5. Conclusion

In this paper we analysed the effective receptive field for two types of graph neural networks Graph Convolutional Network, and Self-Attention. We have shown that the two models achieve good performance on tasks where distant connections are essential. We proved that for Graph Convolutional Networks the effective receptive field of a node is dependent on the graph structure, more specific on the

Model	Accuracy - relational
Conv-S	54.4
Conv-L	73.3
Self-Attention	81.8

Table 1. Comparison between Self-Attention and convolutional networks on the relational split of the Sort-of-CLEVR dataset. The order of the models correlates with the ranking given by the dimension of the effective receptive field.

probability of reaching each input node starting from the considered one. For Self-Attention layers, we proved that the effective receptive field follows the correlation between the considered node and all the others and we empirically validate our findings.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] André Araujo, Wade Norris, and Jack Sim. Computing receptive fields of convolutional neural networks. *Distill*, 2019. https://distill.pub/2019/computing-receptive-fields.

[3] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In *Advances in neural information processing systems*, pages 4502–4510, 2016.

[4] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[5] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *CoRR*, abs/1312.6203, 2013.

[6] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 4724–4733. IEEE, 2017.

[7] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[8] François Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint*, pages 1610–02357, 2017.

[9] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017.

[10] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *CVPR*.

[11] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[13] Junyu Gao, Tianzhu Zhang, and Changsheng Xu. Graph convolutional tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4649–4659, 2019.

[14] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272, 2017.

[15] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 1024–1034. Curran Associates, Inc., 2017.

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[19] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *CoRR*, abs/1506.05163, 2015.

[20] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine*

Learning, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR.

[21] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5308–5317, 2016.

[22] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Fei Fei Li, C. Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. pages 1988–1997, 07 2017.

[23] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[24] Yujia Li, Oriol Vinyals, Chris Dyer, Razvan Pascanu, and Peter Battaglia. Learning deep generative models of graphs, 2018.

[25] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *Advances in neural information processing systems*, pages 4898–4906, 2016.

[26] Andrei Nicolicioiu, Iulia Duta, and Marius Leordeanu. Recurrent space-time graph neural networks. In *Advances in Neural Information Processing Systems 32*, pages 12818–12830. 2019.

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 91–99, Cambridge, MA, USA, 2015. MIT Press.

[28] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4967–4976. Curran Associates, Inc., 2017.

[29] Ran Tao, Efstratios Gavves, and Arnold WM Smeulders. Siamese instance search for tracking. In *CVPR*.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[31] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[32] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, page 4, 2018.

[33] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 399–417, 2018.

[34] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[35] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5453–5462, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[36] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

Appendix: Effective Receptive Field of Graph Neural Networks

In the following, we give complete proofs of all the results presented in the main paper.

A. Notations

Throughout our paper we use the following notations:

- M_i is the i-th line of matrix M.
  $M_{:i}$ is the i-th column of matrix M.
- $vec(M)$ is the vectorized form of the matrix M:
$$M = [M_1^T; M_2^T; ..M_n^T] \in \mathbb{R}^{n\times m}$$
$$vec(M) = [M_1^T, M_2^T, ..M_n^T] \in \mathbb{R}^{nm}$$
- unless otherwise specified $X^{(1)} := X$, $X^{(L+1)} := Y$
- $\nabla\mathcal{L}_M \in \mathbb{R}^{n\times c}$ is the gradient of a scalar loss function w.r.t. matrix $M \in \mathbb{R}^{n\times c}$ such that $(\nabla\mathcal{L}_M)_{ij} = \frac{\partial\mathcal{L}}{\partial M_{ij}}$
- $\nabla\mathcal{L}_X^{(l)} := \nabla\mathcal{L}_{X^{(l)}}$
- $\mathbb{V}[W]$ represents the matrix formed from the variance of each element:
$$(\mathbb{V}[W])_{i,j} := \mathbb{V}[W_{ij}]$$

B. Effective receptive field of Graph Neural Networks

$$\text{erf}(p,i) = \sum_{r=1}^{c}\sum_{s=1}^{c}\frac{\partial Y_{pr}}{\partial X_{is}} \tag{19}$$

By setting $\mathcal{L} = \sum_r Y_{pr}$, we obtain $\nabla\mathcal{L}_Y$ a matrix with the p-th line full of ones and zeros otherwise and we can write the ERF as:

$$\text{erf}(p,i) = \sum_{s=1}^{c}(\nabla\mathcal{L}_X)_{is} \tag{20}$$

B.1. ERF of Graph Convolutional Networks

GCN is defined by:

$$X^{(l+1)} = \sigma\big(AX^{(l)}W_l\big) \tag{21}$$
$$Y = AX^{(L)}W_L \tag{22}$$

where σ denote the ReLU non-linearity.

Theorem 4. *In a Graph Convolutional Network with L layers, without ReLU non-liniarities, the effective receptive field of an output node Y_p with respect to every input node X_i depends on the probability of reaching the X_i nodes starting from Y_p as in Equation 23.*

We obtain the ERF using Equation 19 from:

$$\frac{\partial Y_p}{\partial X_i} = (A^L)_{pi}(\prod_{l=1}^{L}W_l)^T \tag{23}$$

Proof. We know that for any matrices M_1, X, M_2 we have:

$$\frac{\partial vec(M_1XM_2)}{\partial vec(X)} = M_2^T \otimes M_1 \tag{24}$$

where $M_1 \otimes M_2$ is the Kronecker product between the two matrices.

A GCN without ReLU non-liniarities has the output:

$$Y = A^L X \prod_{l=1}^{L} W_l \tag{25}$$

Thus, by setting $M_1 := A^L$ and $M_2 := (\prod_{l=1}^{L} W_l)$

$$\frac{\partial vec(Y)}{\partial vec(X)} = (\prod_{l=1}^{L}W_l)^T \otimes A^L \tag{26}$$

By selecting an element from the derivative, in turn we select a block from the Kronecker product:

$$\frac{\partial Y_p}{\partial X_i} = (\prod_{l=1}^{L}W_l)^T \otimes (A^L)_{pi} \tag{27}$$
$$\frac{\partial Y_p}{\partial X_i} = (A^L)_{pi}(\prod_{l=1}^{L}W_l)^T \tag{28}$$

□

Corollary 1. *Alternatively, in the same conditions as Theorem 4, we can obtain the ERF from:*

$$\nabla\mathcal{L}_X = (A^L)^T\nabla\mathcal{L}_Y(\prod_{l=1}^{L}W_l)^T \tag{29}$$

Proof. We begin by giving some helping remarks.

Remark 1. *For $Y = XW$, where $X \in \mathbb{R}^{n\times c}$ and $W \in \mathbb{R}^{c\times c}$, with given $\nabla\mathcal{L}_Y \in \mathbb{R}^{n\times c}$:*

$$\nabla\mathcal{L}_X = \nabla\mathcal{L}_Y W^T \tag{30}$$

Proof.

$$\frac{\partial \mathcal{L}}{\partial X_{is}} = \sum_{q=1}^{n}\sum_{r=1}^{c} \frac{\partial \mathcal{L}}{\partial Y_{qr}} \frac{\partial Y_{qr}}{\partial X_{is}} \tag{31}$$

Since, in this case, every line Y_i of output is a linear projection of the corresponding line X_i of the input, we have $\frac{\partial Y_{qr}}{\partial X_{is}}$ is zero for $q \neq i$, resulting in:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial X_{is}} &= \sum_{r=1}^{c} \frac{\partial \mathcal{L}}{\partial Y_{ir}} \frac{\partial Y_{ir}}{\partial X_{is}} \\ &= \sum_{r=1}^{c} \frac{\partial \mathcal{L}}{\partial Y_{ir}} \frac{\partial (\sum_k X_{ik} W_{kr})}{\partial X_{is}} = \sum_{r=1}^{c} \frac{\partial \mathcal{L}}{\partial Y_{ir}} W_{sr} \\ &= \sum_{r=1}^{c} (\nabla \mathcal{L}_Y)_{ir} W^T_{rs} = (\nabla \mathcal{L}_Y W^T)_{is}\end{aligned}$$

□

Remark 2. *For $Y = AX$, where $X \in \mathbb{R}^{n\times c}$ and $A \in \mathbb{R}^{n\times n}$, with given $\nabla \mathcal{L}_Y \in \mathbb{R}^{n\times c}$:*

$$\nabla \mathcal{L}_X = A^T \nabla \mathcal{L}_Y \tag{32}$$

Proof.

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial X_{is}} &= \sum_{q=1}^{n}\sum_{r=1}^{c} \frac{\partial \mathcal{L}}{\partial Y_{qr}} \frac{\partial Y_{qr}}{\partial X_{is}} \\ &= \sum_{q=1}^{c} \frac{\partial \mathcal{L}}{\partial Y_{qs}} \frac{\partial Y_{qs}}{\partial X_{is}} = \sum_{q=1}^{c} \frac{\partial \mathcal{L}}{\partial Y_{qs}} A_{qi} \\ &= \sum_{q=1}^{c} (A^T)_{iq} (\nabla \mathcal{L}_Y)_{qs} = (A^T \nabla \mathcal{L}_Y)_{is}\end{aligned}$$

□

Remark 3. *For $Y = AXW$, where $X \in \mathbb{R}^{n\times c}$ and $W \in \mathbb{R}^{c\times c}$, $A \in \mathbb{R}^{n\times n}$, with given $\nabla \mathcal{L}_Y \in \mathbb{R}^{n\times c}$:*

$$\nabla \mathcal{L}_X = A^T \nabla \mathcal{L}_Y W^T \tag{33}$$

Proof. Let $X_2 = AX$, resulting in $Y = X_2 W$ and from Remark 2:

$$\nabla \mathcal{L}_{X_2} = \nabla \mathcal{L}_Y W^T \tag{34}$$

From $X_2 = AX$ and Remark 1 we have:

$$\nabla \mathcal{L}_X = A^T \nabla \mathcal{L}_{X_2} = A^T \nabla \mathcal{L}_Y W^T \tag{35}$$

□

For $Y = A^L X \prod_{l=1}^{L} W_l$, by using the previous remark with $A := A^L$ and $W := \prod_{l=1}^{L} W_l$ we obtain:

$$\nabla \mathcal{L}_X = (A^T)^L \nabla \mathcal{L}_Y (\prod_{l=1}^{L} W_l)^T \tag{36}$$

□

Constant weights. For a GCN with constant weights $W = J_c$, a $c \times c$ all-ones matrix, the expected values of the ERF, by varying the input, is given by:

$$\mathbb{E}_X[\text{erf}(p,i)] = c^{(L+1)} (A^L)_{p,i} \tag{37}$$

Proof. By setting $W_l = J_c$ Equation.23 becomes:

$$\frac{\partial Y_p}{\partial X_i} = (A^L)_{p,i} (\prod_{l=1}^{L} J_c)^T = (A^L)_{p,i} (c^{(L-1)} J_c) \tag{38}$$

$$\text{erf}(p,i) = \sum_{r=1}^{c}\sum_{s=1}^{c} (\frac{\partial Y_p}{\partial X_i})_{r,s} = (A^L)_{p,i} c^{(L+1)} \tag{39}$$

The $\text{erf}(p,i)$ is independent of the input X, obtaining:

$$E_X[\text{erf}(p,i)] = (A^L)_{p,i} c^{(L+1)} \tag{40}$$

□

Random weights. We consider a GCN with random weights, sampled independently from a standard Gaussian $W_l \sim \mathcal{N}(0, I)$. We compute the expected values of the ERF over all the inputs and all possible weights.

$$\mathbb{E}_{X,W}[\text{erf}(p,i)] = 0 \tag{41}$$

$$\mathbb{V}_{X,W}[\text{erf}(p,i)] = c^{(L+1)} [(A^L)_{p,i}]^2 \tag{42}$$

Proof.

$$\mathbb{E}_{X,W}[\text{erf}(p,i)] = \sum_{r=1}^{c}\sum_{s=1}^{c} (A^L)_{p,i} \mathbb{E}[\prod_{l=L}^{1} W_l^T]_{r,s} \tag{43}$$

Since W_{ij} is independent from W_{lk} $\forall i,j,k,l$ and are all centered in zero, $\mathbb{E}_W[W_l^T] = 0$ we get:

$$\mathbb{E}[\prod_{l=L}^{1} W_l^T] = \prod_{l=L}^{1} \mathbb{E}_W[W_l^T])] = 0 \tag{44}$$

Thus,

$$\mathbb{E}_{X,W}[\text{erf}(p,i)] = \sum_{r=1}^{c}\sum_{s=1}^{c} (A^L)_{p,i} \cdot 0 = 0 \tag{45}$$

For computing the variance, we demonstrate the following lemmas.

Lemma 1. *Given two scalar independent random variables $w_1, w_2 \in \mathbb{R}$, $w_1 \perp\!\!\!\perp w_2$ with $\mathbb{E}[w_1] = \mathbb{E}[w_2] = 0$, we have $\mathbb{V}[w_1 w_2] = \mathbb{V}[w_1]\mathbb{V}[w_2]$.*

Proof.

$$\mathbb{V}[w_1 w_2] = \mathbb{E}[w_1^2 w_2^2] - E^2[w_1 w_2] = \mathbb{E}[w_1^2]\mathbb{E}[w_2^2]$$

Since we assume $\mathbb{E}[w_i] = 0$ we obtain $\mathbb{V}[w_1] = E[w_1^2]$:

$$\mathbb{V}[w_1 w_2] = \mathbb{V}[w_1]\mathbb{V}[w_2] \tag{46}$$

□

Lemma 2. *Given two random variables $A, B \in \mathbb{R}^{n\times n}$ with independent elements and with $A_{ij} \perp\!\!\!\perp B_{ab} \quad \forall(i,j,a,b)$ and $\mathbb{E}[A] = \mathbb{E}[B] = 0$, then the elementwise variance is:*

$$\mathbb{V}[AB] = \mathbb{V}[A]\mathbb{V}[B] \tag{47}$$

Proof. For all indices (i, j) we have that:

$$(AB)_{i,j} = \sum_s (A_{is}B_{sj})$$

From the hypothesis $(A_{is}B_{sj}) \perp\!\!\!\perp (A_{ir}B_{rj}) \quad \forall s \neq r$

$$\begin{aligned}\mathbb{V}[(AB)_{i,j}] &= \sum_s \mathbb{V}[A_{is}B_{sj}] = \sum_s \mathbb{V}[A_{is}]\mathbb{V}[B_{sj}] \\ &= (\mathbb{V}[A]\mathbb{V}[B])_{i,j}\end{aligned} \tag{48}$$

□

To finish the proof, we need to show that if (W_i) are independent matrices, each one having independent elements, then $\prod_{k=1}^{L} W_k$ has independent elements.

Lemma 3. *Given two random variables $A, B \in \mathbb{R}^{n\times n}$ with independent elements and with $A_{ij} \perp\!\!\!\perp B_{ab} \quad \forall(i,j,a,b)$ then $(AB)_{ij} \perp\!\!\!\perp (AB)_{a,b}\ \forall(i,j) \neq (a,b)$*

Proof. We want to prove that $\text{Cov}((AB)_{ij}, (AB)_{ab}) = 0$, $\forall(i,j) \neq (a,b)$. By symmetry, we only consider the case $j \neq b$. Since the covariance is biliniar:

$$\begin{aligned}\text{Cov}((AB)_{ij}, (AB)_{ab}) &= \sum_s\sum_r \text{Cov}(A_{is}B_{sj}, A_{ar}B_{rb}) \\ &= \mathbb{E}[A_{is}B_{sj}A_{ar}B_{rb}] - \mathbb{E}[A_{is}B_{sj}]\mathbb{E}[A_{ar}B_{rb}] \\ &= \mathbb{E}[A_{is}B_{sj}A_{ar}]\mathbb{E}[B_{rb}] - \mathbb{E}[A_{is}B_{sj}]\mathbb{E}[A_{ar}]\mathbb{E}[B_{rb}] \\ &= 0 - 0 = 0\end{aligned}$$

□

From Lemma 3 the elements of $\prod_{k=1}^{L} W_k$ are independent. Because $\mathbb{V}(a+b) = \mathbb{V}[a] + \mathbb{V}[b]$ for $a \perp\!\!\!\perp b$ we obtain:

$$\begin{aligned}\mathbb{V}[\text{erf}(p,i)] &= \mathbb{V}\big[\sum_s\sum_r (A^L)_{p,i}(\prod_{k=1}^{L} W_k)_{s,r}\big] \\ &= ((A^L)_{p,i})^2 \sum_s\sum_r \mathbb{V}\big[(\prod_{k=1}^{L} W_k)\big]_{s,r}\end{aligned} \tag{49}$$

Based on Lemma 2, if W_i, W_j are independent matrices with each pair of elements $(W_i)_{ab} \perp\!\!\!\perp (W_j)_{cd}$, all of them centered in zero, we have that:

$$\mathbb{V}\big[\prod_{i=1}^{L} W_i\big] = \prod_{i=1}^{L} \mathbb{V}[W_i] \tag{50}$$

$$\begin{aligned}\mathbb{V}[\text{erf}(p,i)] &= ((A^L)_{p,i})^2 \sum_s\sum_r (\prod_{k=1}^{L} \mathbb{V}[W_k])_{s,r} \\ &= ((A^L)_{p,i})^2 \sum_s\sum_r (\prod_{k=1}^{L} J_c)_{s,r} \\ &= ((A^L)_{p,i})^2 \sum_s\sum_r c^{(L-1)} \\ &= c^{(L+1)}((A^L)_{p,i})^2\end{aligned}$$

□

Theorem 5. *In a Graph Convolutional Network with L layers with ReLU non-linearities, the effective receptive field of an output node Y_p with respect to every input node X_i depends on the probability of reaching the X_i nodes starting from Y_p as in Equation 51.*

We obtain the ERF using Equation. 20 from:

$$\begin{aligned}\nabla\mathcal{L}_X^{(l)} &= A^T(\nabla\mathcal{L}_X^{(l+1)} \odot Z^{(l)})W_l^T \\ \nabla\mathcal{L}_X^{(L)} &= A^T\nabla\mathcal{L}_Y W_L^T \\ Z^{(l)} &= \mathbb{I}(AX^{(l)}W_l > 0)\end{aligned} \tag{51}$$

Proof. At forward, GCN is defined as:

$$\begin{aligned}X^{(l+1)} &= \sigma\big(AX^{(l)}W_l\big) = \big(AX^{(l)}W_l\big) \odot Z^{(l)} \\ Y &= AX^{(L)}W_L,\end{aligned} \tag{52}$$

Let $D^{(l)} := AX^{(l)}W_l$.

$$\frac{\partial\mathcal{L}}{\partial D_{is}^{(l)}} = \sum_{q=1}^{n}\sum_{r=1}^{c} \frac{\partial\mathcal{L}}{\partial X_{qr}^{(l+1)}} \frac{\partial X_{qr}^{(l+1)}}{\partial D_{is}^{(l)}} \tag{53}$$

Since $D^{(l)}$ is formed by applying elementwise the ReLU non-linearity on $X^{(l)}$, in the equation above all the terms with $p \neq i$ or $r \neq s$ are zero.

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial D_{is}^{(l)}} &= \frac{\partial\mathcal{L}}{\partial X_{is}^{(l+1)}} \frac{\partial X_{is}^{(l+1)}}{\partial D_{is}^{(l)}} \\ &= \frac{\partial\mathcal{L}}{\partial X_{is}^{(l+1)}} Z_{is}^{(l)} = (\frac{\partial\mathcal{L}}{\partial X^{(l+1)}} \odot Z^{(l)})_{is}\end{aligned}$$

$$\nabla\mathcal{L}_D^{(l)} = \nabla\mathcal{L}_X^{(l+1)} \odot Z^{(l)} \tag{54}$$

From Remark 3 and the definition of $D^{(l)}$, we have:

$$\nabla\mathcal{L}_X^{(l)} = A^T\nabla\mathcal{L}_D^{(l)}W^T = A^T(\nabla\mathcal{L}_X^{(l+1)} \odot Z^{(l)})W^T$$

From Remark 3 and Equation 52 we obtain:

$$\nabla\mathcal{L}_X^{(L)} = A^T\nabla\mathcal{L}_Y W_L^T$$

□

Scalar weights. For a single channel $c = 1$, the effective receptive field of the GCN in the ReLU case is given by:

$$\nabla\mathcal{L}_X^{(l)} = A^T(\nabla\mathcal{L}_X^{(l+1)} \odot Z^{(l)})w_l \tag{55}$$
$$\nabla\mathcal{L}_X^{(L)} = A^T\nabla\mathcal{L}_Y w_L$$
$$\nabla\mathcal{L}_Y = (I_n)_{:p}$$

In the case of scalar weights, $\mathcal{L} = Y_p$, thus

$$(\nabla\mathcal{L}_X)_i = \frac{\partial Y_p}{\partial X_i} = \text{erf}(p, i) \tag{56}$$

Constant Weights. For a GCN with two layers, single channel, with input X that is symmetrically distributed around zero, we compute the expected value of the ERF by varying only the input and keeping the weights fixed, as follows:

$$\mathbb{E}_X[\text{erf}(p, i)] = \mathbb{E}_X[\frac{\partial Y_p}{\partial X_i}] = \frac{1}{2}(A^2)_{p,i}\prod_{l=2}^{1} w_l \tag{57}$$

Proof. From Equation 55 we have:

$$\text{erf}(p, i) = (\nabla\mathcal{L}_X)_i = (A^T\big((A^T\nabla\mathcal{L}_Y w_2) \odot Z\big)w_1)_i, \tag{58}$$

We use that $\nabla\mathcal{L}_Y = (I_n)_{:p}$:

$$\begin{aligned}\text{erf}(p, i) &= A_i^T\big(A_p \odot Z\big)w_2w_1,\\ &= \sum_k A_{ik}^T\big(A_p \odot Z\big)_k w_2w_1\\ &= \sum_k A_{ki}A_{pk}Z_k w_2w_1 \end{aligned} \tag{59}$$

We assume that for all k, X_k is symmetric and zero centered, thus $(AXW)_k$ is symmetric and zero centered for all k This results in Z_k being Bernoulli distributed, with equal probability thus $\mathbb{E}[Z_k] = \frac{1}{2}$.

We obtain that:

$$\begin{aligned}\mathbb{E}[\text{erf}(p, i)] &= \mathbb{E}[\sum_k A_{ki}A_{pk}Z_k w_2w_1]\\ &= \sum_k A_{ki}A_{pk}w_2w_1\mathbb{E}[Z_k] = \frac{1}{2}(A^2)_{p,i}w_2w_1\end{aligned} \tag{60}$$

□

Constant Weights Multiple Layers. We now give a generalisation of the previous result for the case of GCN with ReLU and L single channel layers, with input X that is symmetrically distributed around zero and assuming ReLU gates $Z^{(l_1)}$ and $Z^{(l_2)}$ are independent for every two layers $l_1 \neq l_2$.

$$\mathbb{E}_X[\text{erf}(p, i)] = (\frac{1}{2})^{(L-1)}(A^L)_{p,i}\prod_{l=L}^{1} w_l \tag{61}$$

Proof. From Equation 55 we have:

$$\begin{aligned}\mathbb{E}[\nabla\mathcal{L}_X^{(l)}] &= \mathbb{E}[A^T(\nabla\mathcal{L}_X^{(l+1)} \odot Z^{(l)})w_l]\\ &= A^T\mathbb{E}[(\nabla\mathcal{L}_X^{(l+1)} \odot Z^{(l)})]w_l\end{aligned}$$

We assume that at each layer the input is symmetric and zero centered and, like previously stated, this results in $\mathbb{E}[Z_l] = \frac{1}{2}$. We also make the additional assumption that $Z^{(l_1)}$ and $Z^{(l_2)}$ are independent for every two layers $l_1 \neq l_2$, thus Z_l is independent of the gradient $\nabla\mathcal{L}_X^{(l+1)}$, obtaining:

$$\begin{aligned}\mathbb{E}[\nabla\mathcal{L}_X^{(l)}] &= A^T\mathbb{E}[(\nabla\mathcal{L}_X^{(l+1)}] \odot \mathbb{E}[Z^{(l)})]w_l\\ &= \frac{1}{2}A^T\mathbb{E}[(\nabla\mathcal{L}_X^{(l+1)}]w_l\end{aligned} \tag{62}$$

Based on this recursion and using Equation 55 we obtain:

$$\mathbb{E}[\nabla\mathcal{L}_X] = (\frac{1}{2})^{(L-1)}(A^T)^{(L-1)}\mathbb{E}[(\nabla\mathcal{L}_X^{(L)}]\prod_{l=L-1}^{1} w_l \tag{63}$$

From Equation 55:

$$\mathbb{E}[\nabla\mathcal{L}_X^{(L)}] = A^T(I_n)_{:p}w_L$$

Using Equation 20, we obtain:

$$\begin{aligned}\mathbb{E}[\text{erf}(p, i)] = \mathbb{E}[\nabla\mathcal{L}_X]_i &= (\frac{1}{2})^{(L-1)}(A^T)_i^L(I_n)_{:p}\prod_{l=L}^{1} w_l\\ &= (\frac{1}{2})^{(L-1)}(A^L)_{p,i}\prod_{l=L}^{1} w_l\end{aligned}$$

□

Random Weights. For a GCN with two layers, single channel, with w_l sampled from a standard Gaussian $\mathcal{N}(0, 1)$, assuming that the weights from the first layer w_1 are independent from the ReLU gates on the first layer Z, and the input X_k is symmetric and zero centered $\forall k$, we have:

$$\mathbb{E}_{X,W}[\text{erf}(p, i)] = \sum_{k=1}^{n} A_{pk}A_{ki}\mathbb{E}[Z_k w_1]\mathbb{E}[w_2] = 0 \tag{64}$$

$$\mathbb{V}_{X,W}[\text{erf}(p, i)] = \sum_k^n\sum_l^n A_{pk}A_{ki}A_{pl}A_{li}(\text{Cov}(Z_k, Z_l) + \frac{1}{4}) \tag{65}$$

Proof. We use the Equation 55 and get:

$$\nabla\mathcal{L}_X = A^T((A^T\nabla\mathcal{L}_Y w_2) \odot Z)w_1$$
$$A^T\nabla\mathcal{L}_Y = (A^T)(I_n)_{:p} = (A^T)_{:p} = A_p$$

Using Equation 56 we obtain:

$$\begin{aligned}\mathbb{E}_{X,W}[\text{erf}(p,i)] &= \mathbb{E}_{X,W}[(\nabla\mathcal{L}_X)_i] \\ &= \mathbb{E}[(A^T((A_p w_2)\odot Z)w_1)_i] \\ &= \mathbb{E}[(A^T)_i((A_p w_2)\odot Z)w_1] \\ &= \mathbb{E}[\sum_{k=1}^{n}(A^T)_{ik}(A_p\odot Z)_k w_2 w_1] \\ &= \mathbb{E}[\sum_{k=1}^{n}(A^T)_{ik}A_{pk}Z_k w_2 w_1] \\ &= \mathbb{E}[\sum_{k=1}^{n}A_{ki}A_{pk}Z_k w_2 w_1] \\ &= \sum_{k=1}^{n}A_{pk}A_{ki}\mathbb{E}[Z_k w_2 w_1] \end{aligned} \tag{66}$$

Since the weights of the second layer w_2 are zero centered and are independent from the ReLU gates on the first later Z, we obtain:

$$\mathbb{E}_{X,W}[\text{erf}(p,i)] = \sum_{k=1}^{n}A_{pk}A_{ki}\mathbb{E}[Z_k w_1]\mathbb{E}[w_2] = 0 \tag{67}$$

□

We obtain the variance of the ERF in the same way:

$$\begin{aligned}\mathbb{V}_{X,W}[\text{erf}(p,i)] &= \mathbb{V}[\sum_{k=1}^{n}A_{pk}A_{ki}Z_k w_2 w_1] \\ &= \sum_{k=1}^{n}\mathbb{V}[A_{pk}A_{ki}Z_k w_2 w_1]+ \\ &+ \sum_{k,l;k\neq l}^{n}\sum^{n}\text{Cov}(A_{pk}A_{ki}Z_k w_2 w_1, A_{pl}A_{li}Z_l w_2 w_1) \\ &= \sum_{k=1}^{n}(A_{pk}A_{ki})^2\mathbb{V}[Z_k w_2 w_1]+ \\ &+ \sum_{k,l;k\neq l}^{n}\sum^{n}A_{pk}A_{ki}A_{pl}A_{li}\text{Cov}(Z_k w_2 w_1, Z_l w_2 w_1)\end{aligned} \tag{68}$$

We use the fact that weights from the second layer w_2 are independent from weights from the first layer multiplied by the ReLU gates from the first layer $Z_k w_1$.

We also assume that the weights from the first layer w_1 are independent from the ReLU gates on the first layer Z but this assumption usually does not hold.

$$\begin{aligned}\mathbb{V}[Z_k w_2 w_1] &= \mathbb{E}[(Z_k w_2 w_1)^2] - \mathbb{E}^2[Z_k w_2 w_1] \\ &= \mathbb{E}[(Z_k w_1)^2]\mathbb{E}[w_2^2] - \mathbb{E}^2[Z_k w_1]\mathbb{E}^2[w_2] \\ &= \mathbb{E}[Z_k^2 w_1^2]\mathbb{V}[w_2] - 0 = \mathbb{E}[Z_k^2]\mathbb{E}[w_1^2]\mathbb{V}[w_2] \\ &= (\mathbb{V}[Z_k]+\mathbb{E}^2[Z_k])\mathbb{V}[w_1]\mathbb{V}[w_2] \\ &= \frac{1}{2}\end{aligned} \tag{69}$$

We assume that for all k, X_k is symmetric and zero centered, thus $(AXW)_k$ is symmetric and zero centered for all k, resulting in Z_k being Bernoulli distributed, with equal probability thus $\mathbb{E}[Z_k] = \frac{1}{2}$. Also the weights are sampled from a Gaussian with variance equal to 1 and using the previous assumptions we have:

$$\begin{aligned}&\text{Cov}(Z_k w_2 w_1, Z_l w_2 w_1) = \\ &= \mathbb{E}[Z_k Z_l w_2^2 w_1^2] - \mathbb{E}[Z_k w_2 w_1]\mathbb{E}[Z_l w_2 w_1] \\ &= \mathbb{E}[Z_k Z_l]\mathbb{E}[w_2^2]\mathbb{E}[w_1^2] - \mathbb{E}[Z_k w_1]\mathbb{E}[w_2]\mathbb{E}[Z_l w_1]\mathbb{E}[w_2] \\ &= \mathbb{E}[Z_k Z_l]\mathbb{V}[w_2]\mathbb{V}[w_1] \\ &= \mathbb{E}[Z_k]\mathbb{E}[Z_l] + \text{Cov}(Z_k, Z_l) \\ &= \frac{1}{4} + \text{Cov}(Z_k, Z_l)\end{aligned} \tag{70}$$

Combining the previous three results, given in Equations 68 , 69 and 70 we obtain:

$$\begin{aligned}\mathbb{V}_{X,W}[\text{erf}(p,i)] &= \frac{1}{2}\sum_{k=1}^{n}(A_{pk}A_{ki})^2 + \\ &+ \sum_{k,l;k\neq l}^{n}\sum^{n}A_{pk}A_{ki}A_{pl}A_{li}(\text{Cov}(Z_k, Z_l) + \frac{1}{4}) \\ &= \sum_{k}^{n}\sum_{l}^{n}A_{pk}A_{ki}A_{pl}A_{li}(\text{Cov}(Z_k, Z_l) + \frac{1}{4})\end{aligned}$$

B.2. ERF of self-attention layer

We use the following form of self-attention layer:

$$Y = (XW_q)(XW_k)^T(XW_v) \tag{71}$$

Theorem 6. *In a self-attention layer, without softmax, the effective receptive field of a node Y_p with respect to every other node X_i depends on the similarity between linear projections of the two nodes $(X_pW_q)(X_iW_k)^T$ and the outer product $(X_iW_v)^T(X_pW_q)$.*

We obtain the ERF using Equation 19 from:

$$\frac{\partial Y_p}{\partial X_i} = W_v^T(X_pW_q)(X_iW_k)^T + (X_iW_v)^T(X_pW_q)W_k^T, \quad \forall i \neq p \tag{72}$$

$$\frac{\partial Y_p}{\partial X_p} = W_v^T(X_pW_q)(X_pW_k)^T + (X_pW_v)^T(X_pW_q)W_k^T + \sum_{j=1}^{n}(X_jW_v)^T(X_jW_k)W_q^T \quad (73)$$

Proof. For this proof, we will use the following matrix derivatives:

$$\frac{\partial Ax^T}{\partial x} = A \quad (74)$$

$$\frac{\partial f(x)A}{\partial x} = A^T\frac{\partial f(x)}{\partial x} \quad (75)$$

$$\frac{\partial ax^Tx}{\partial x} = x^T\frac{\partial Ax^T}{\partial x} + ax^T\frac{\partial x}{\partial x} = x^Ta + ax^TI \quad (76)$$

where $A \in \mathbb{R}^{n\times n}$ and $x, a \in \mathbb{R}^{1\times n}$.
Case 1: $p \neq i$:

$$\frac{\partial Y_p}{\partial X_i} = \frac{\partial (X_pW_q)(X_iW_k)^T(X_iW_v)}{\partial X_i} \quad (77)$$

Using Equations 75 and 76:

$$\begin{aligned}\frac{\partial Y_p}{\partial X_i} &= \frac{\partial (X_pW_qW_k^TX_i^TX_i)W_v}{\partial X_i}\\ &= W_v^T\frac{\partial (X_pW_qW_k^T)X_i^TX_i}{\partial X_i}\\ &= W_v^T(X_pW_qW_k^T)X_i^T + W_v^TX_i^T(X_pW_qW_k)\end{aligned}$$

Case 2: $p = i$:

$$\frac{\partial Y_p}{\partial X_p} = \frac{\partial (X_pW_qW_kX_p^TX_p)W_v}{\partial X_p} + S \quad (78)$$

where $S = \sum_{j\neq p}\frac{\partial (X_pW_qW_kX_j^TX_j)W_v}{\partial X_p}$ (79)

Using the product rule we obtain : (80)

$$\begin{aligned}\frac{\partial Y_p}{\partial X_p} &= W_v^T\frac{\partial (X_pW_qW_k^TX_p^T)X_p}{\partial X_p} + S\\ &= W_v^T(X_pW_qW_k^TX_p^T)\frac{\partial X_p}{\partial X_p})+\\ &\quad + (W_v^TX_p^T\frac{\partial X_p(W_qW_k^TX_p^T)}{\partial X_p} + S\\ &= W_v^TX_pW_qW_k^TX_p^T+\\ &\quad + (W_v^TX_p^T(W_qW_k^TX_p^T)^T\frac{\partial X_p}{\partial X_p}+\\ &\quad + W_v^TX_p^TX_p\frac{\partial W_qW_k^TX_p^T}{\partial X_p}) + S\\ &= W_v^T(X_pW_q)(X_pW_k)^T+\\ &\quad + (X_pW_v)^T(X_pW_k)W_q^T+\\ &\quad + (X_pW_v)^T(X_pW_q)W_k^T + S \quad (81)\end{aligned}$$

To compute the last term we use Equation 75:

$$\begin{aligned}S &= \sum_{j\neq p}\frac{\partial X_p(W_qW_kX_j^TX_jW_v)}{\partial X_p}\\ &= \sum_{j\neq p}(W_qW_k^TX_j^TX_jW_v)^T\\ &= \sum_{j\neq p}((X_jW_v)^T(X_jW_k)W_q^T) \quad (82)\end{aligned}$$

From Equations 81 and 82 we have:

$$\begin{aligned}\frac{\partial Y_p}{\partial X_p} &= W_v^T(X_pW_q)(X_pW_k)^T+\\ &\quad + (X_pW_v)^T(X_pW_q)W_k^T+\\ &\quad + \sum_j((X_jW_v)^T(X_jW_k)W_q^T) \quad (83)\end{aligned}$$

□

Constant Weights. For a self-attention layer, with parameters w_q, w_k, w_v as scalar constant and input X centered in zero, having the same variance $\mathbb{V}[X_i]$ for all i, we have:

$$\begin{aligned}\mathbb{E}_X[\text{erf}(p,i)] &= 2w_qw_kw_v\text{Cov}(X_p,X_i), \qquad \forall i\neq p\\ \mathbb{E}_X[\text{erf}(p,p)] &= (n+2)w_qw_kw_v\mathbb{V}(X_p) \quad (84)\end{aligned}$$

Proof. Case 1: $p \neq i$
From Equation 72 we have:

$$\begin{aligned}\mathbb{E}_X[\text{erf}(p,i)] &= \mathbb{E}_X[2w_qw_kw_vX_pX_i]\\ &= 2w_qw_kw_v\mathbb{E}[X_pX_i]\\ &= 2w_qw_kw_v[\mathbb{E}[X_p]\mathbb{E}[X_i] + \text{Cov}(X_p,X_i)]\\ &= 2w_qw_kw_v\text{Cov}(X_p,X_i) \quad (85)\end{aligned}$$

We have used the fact that the the inputs X is zero centered, meaning $\mathbb{E}[X_i] = 0, \forall i$.
Case 2: $p = i$
From Equation 73 we have:

$$\begin{aligned}\mathbb{E}_X[\text{erf}(p,p)] &= \mathbb{E}[\sum_{k\neq i}^{n}w_qw_kw_vX_k^2 + 3w_qw_kw_vX_p^2]\\ &= w_qw_kw_v\sum_{k\neq i}^{n}\mathbb{E}[X_k^2] + 3w_qw_kw_v\mathbb{E}[X_p^2]\\ &= w_qw_kw_v(n+2)\mathbb{E}[X_p^2]\\ &= w_qw_kw_v(n+2)(\mathbb{E}^2[X_p] + \mathbb{V}[X_p])\\ &= w_qw_kw_v(n+2)\mathbb{V}[X_p] \quad (86)\end{aligned}$$

Where we have used the fact that all the nodes have the same expected $E[X_i] = 0$ values and variance $\mathbb{V}[X_i]$.

□

We note that the following result is slightly different from the one in the main article by having a different constant scaling of some of the terms in Equation 88. This doesn't affect in any way our claims. We changed $n-1$ into $n+8$ and we added the $b_j b_l$ constants.

Random weights. For a self-attention layer with weights randomly sampled independently from a standard Gaussian $w_q, w_k, w_v \sim \mathcal{N}(0,1)$ and with input X centered in zero, having the same variance $\mathbb{V}[X_i]$ for all i, we have:

$$\begin{aligned}\mathbb{E}_{X,W}[\text{erf}(p,i)] &= 2\mathbb{E}_{X,W}[w_q w_k w_v]\text{Cov}(X_p, X_i)\\ &= 0, \qquad \forall p \neq i\\ \mathbb{E}_{X,W}[\text{erf}(p,p)] &= (n+2)\mathbb{E}_{X,W}[w_q w_k w_v]\mathbb{V}[X_p] = 0\end{aligned} \tag{87}$$

$$\begin{aligned}\mathbb{V}_{X,W}[\text{erf}(p,i)] &= 4\mathbb{V}[w_q w_k w_v](\text{Cov}(X_p^2, X_i^2) + \mathbb{V}^2[X_p])\\ &\qquad \forall p \neq i\\ \mathbb{V}_{X,W}[\text{erf}(p,p)] &= (n+8)\mathbb{V}[w_q w_k w_v](\mathbb{V}[X_j^2] + \mathbb{V}^2[X_j]) +\\ &+ \mathbb{V}[w_q w_k w_v]\sum_{j,l;l\neq p}\sum b_j b_l(\text{Cov}(X_j^2, X_l^2) + \mathbb{V}^2[X_j])\end{aligned} \tag{88}$$

where we denote $b_i = 1, \forall i \neq p$ and $b_p = 3$.

Proof. Case 1: $p \neq i$

From Equation 72:

$$\mathbb{E}_{X,W}[\text{erf}(p,i)] = \mathbb{E}[2 w_q w_k w_v X_p X_i]$$

Since the weights W are independent from the inputs X, we have:

$$\begin{aligned}\mathbb{E}_{X,W}[\text{erf}(p,i)] &= 2\mathbb{E}[w_q w_k w_v]\mathbb{E}[X_p X_i]\\ &= 2\mathbb{E}[w_q w_k w_v](\text{Cov}(X_p, X_i) + \mathbb{E}_X[X_p]\mathbb{E}_X[X_i])\\ &= 2\mathbb{E}[w_q w_k w_v]\text{Cov}(X_p, X_i)\end{aligned} \tag{89}$$

Case 2: $p = i$

From Equation 73, we have:

$$\begin{aligned}\mathbb{E}_{X,W}[\text{erf}(p,p)] &= \mathbb{E}[\sum_{k\neq i}^{n} w_q w_k w_v X_k^2 + 3 w_q w_k w_v X_p^2]\\ &= \sum_{k\neq i}^{n}\mathbb{E}[w_q w_k w_v X_k^2] + 3\mathbb{E}[w_q w_k w_v X_p^2]\\ &= \sum_{k\neq i}^{n}\mathbb{E}[w_q w_k w_v]\mathbb{E}[X_k^2] + 3\mathbb{E}[w_q w_k w_v]\mathbb{E}[X_p^2]\end{aligned}$$

Since we assume that X_i is centered in zero, $\mathbb{E}[X_i] = 0$, for all i we have:

$$\mathbb{E}[X_i^2] = \mathbb{V}[X_i] + \mathbb{E}^2[X_i] = \mathbb{V}[X_i] \tag{90}$$

Thus we obtain:

$$\begin{aligned}&\mathbb{E}_{X,W}[\text{erf}(p,p)] =\\ &\quad = \sum_{k\neq i}^{n}\mathbb{E}[w_q w_k w_v]\mathbb{V}[X_k] + 3\mathbb{E}[w_q w_k w_v]\mathbb{V}[X_p]\\ &\quad = (n+2)\mathbb{E}[w_q w_k w_v]\mathbb{V}[X_p]\end{aligned} \tag{91}$$

Case 1: $p \neq i$

Using the fact that the weights $w_q w_k w_v$ and the inputs $X_p X_i$ are independent, from Equation 72 we obtain:

$$\begin{aligned}&\mathbb{V}_{X,W}[\text{erf}(p,i)] = \mathbb{V}[2 w_q w_k w_v X_p X_i]\\ &= (\mathbb{V}[2 w_q w_k w_v] + \mathbb{E}^2[2 w_q w_k w_v])(\mathbb{V}[X_p X_i] + \mathbb{E}^2[X_p X_i])\\ &- \mathbb{E}^2[2 w_q w_k w_v]\mathbb{E}^2[X_p X_i]\\ &= 4\mathbb{V}[w_q w_k w_v](\mathbb{V}[X_p X_i] + \mathbb{E}^2[X_p X_i])\end{aligned} \tag{92}$$

Using the fact that X_p and X_i have the same expected values $\mathbb{E}[X_p] = \mathbb{E}[X_i] = 0$ and variance $\mathbb{V}[X_p] = \mathbb{V}[X_i]$, we obtain:

$$\begin{aligned}\mathbb{V}[X_p X_i] &= \text{Cov}(X_p^2, X_i^2) +\\ &+ (\mathbb{V}[X_p] + \mathbb{E}^2[X_p])(\mathbb{V}[X_i] + \mathbb{E}^2[X_i]) - \text{Cov}^2(X_p, X_i)\\ &= \text{Cov}(X_p^2, X_i^2) + \mathbb{V}^2[X_p] - \text{Cov}^2(X_p, X_i)\end{aligned} \tag{93}$$

We also get:

$$\begin{aligned}\mathbb{E}^2[X_p X_i] &= (\mathbb{E}[X_p]\mathbb{E}[X_i] + \text{Cov}(X_p, X_i))^2\\ &= \text{Cov}^2(X_p, X_i)\end{aligned} \tag{94}$$

Using the previous three Equations we obtain:

$$\begin{aligned}\mathbb{V}_{X,W}[\text{erf}(p,i)] &= 4\mathbb{V}[w_q w_k w_v](\text{Cov}(X_p^2, X_i^2) + \mathbb{V}^2[X_p]\\ &\quad - \text{Cov}^2(X_p, X_i) + \text{Cov}^2(X_p, X_i))\\ &= 4\mathbb{V}[w_q w_k w_v](\text{Cov}(X_p^2, X_i^2) + \mathbb{V}^2[X_p])\end{aligned} \tag{95}$$

Case 2: $p = i$

From Equation 73 we obtain:

$$\mathbb{V}_{X,W}[\text{erf}(p,p)] = \mathbb{V}[2 w_q w_k w_v X_p^2 + \sum_j w_q w_k w_v X_j^2]$$

Lets denote $b_i = 1, \forall i \neq p$ and $b_p = 3$, and $w_q w_k w_v = w_a$ thus the previous Equation becomes:

$$\begin{aligned}\mathbb{V}_{X,W}[\text{erf}(p,p)] &= \mathbb{V}[\sum_j b_j w_a X_j^2]\\ &= \sum_j \mathbb{V}[b_j w_a X_j^2] + \sum_{j,l;l\neq p}\sum \text{Cov}(b_j w_a X_j^2, b_l w_a X_l^2)\\ &= \sum_j b_j^2(\mathbb{E}[w_a^2 X_j^4] - E^2[w_a X_j^2]) +\\ &+ \sum_{j,l;l\neq p}\sum b_j b_l(\mathbb{E}[w_a^2 X_j^2 X_l^2] - \mathbb{E}[w_a X_j^2]\mathbb{E}[w_a X_l^2])\end{aligned} \tag{96}$$

Because the weights w_q, w_k, w_v and the inputs X_i are all independent, we have:

$$\mathbb{E}[w_a] = \mathbb{E}[w_q w_k w_v] = \mathbb{E}[w_q]\mathbb{E}[w_k]\mathbb{E}[w_v] = 0 \tag{97}$$

$$\mathbb{E}[w_a X_i^2] = \mathbb{E}[w_a]\mathbb{E}[X_i^2] = 0, \qquad \forall i \tag{98}$$

Thus, we obtain:

$$\begin{aligned}
\mathbb{V}_{X,W}[\text{erf}(p,p)] &= \sum_j b_j^2 \mathbb{E}[w_a^2 X_j^4] + \sum_{j,l;l\neq p}\sum b_j b_l \mathbb{E}[w_a^2 X_j^2 X_l^2] \\
&= \sum_j b_j^2 \mathbb{E}[w_a^2]\mathbb{E}[X_j^4] + \sum_{j,l;l\neq p}\sum b_j b_l \mathbb{E}[w_a^2]\mathbb{E}[X_j^2 X_l^2] \\
&= \mathbb{E}[w_a^2]\sum_j b_j(\mathbb{V}[X_j^2] + \mathbb{E}^2[X_j^2]) + \\
&\quad + \mathbb{E}[w_a^2]\sum_{j,l;l\neq p}\sum b_j b_l(\text{Cov}(X_j^2, X_l^2) + \mathbb{E}[X_j^2]\mathbb{E}[X_l^2]) \\
&= \mathbb{E}[w_a^2]\sum_j b_j(\mathbb{V}[X_j^2] + \mathbb{V}^2[X_j]) + \\
&\quad + \mathbb{E}[w_a^2]\sum_{j,l;l\neq p}\sum b_j b_l(\text{Cov}(X_j^2, X_l^2) + \mathbb{V}[X_j]\mathbb{V}[X_l])
\end{aligned} \tag{99}$$

Finally we have:

$$\begin{aligned}
\mathbb{V}_{X,W}[\text{erf}(p,p)] &= (n+8)\mathbb{E}[w_a^2](\mathbb{V}[X_p^2] + \mathbb{V}^2[X_p]) + \\
&\quad + \mathbb{E}[w_a^2]\sum_{j,l;l\neq p}\sum b_j b_l(\text{Cov}(X_j^2, X_l^2) + \mathbb{V}^2[X_j])
\end{aligned}$$

$$\begin{aligned}
\mathbb{V}_{X,W}[\text{erf}(p,p)] &= (n+8)\mathbb{E}[w_q^2 w_k^2 w_v^2](\mathbb{V}[X_p^2] + \mathbb{V}^2[X_p]) + \\
&\quad + \mathbb{E}[w_q^2 w_k^2 w_v^2]\sum_{j,l;l\neq p}\sum b_j b_l(\text{Cov}(X_j^2, X_l^2) + \mathbb{V}^2[X_j])
\end{aligned} \tag{100}$$

$$\begin{aligned}
\mathbb{V}_{X,W}[\text{erf}(p,p)] &= (n+8)\mathbb{V}[w_q w_k w_v](\mathbb{V}[X_p^2] + \mathbb{V}^2[X_p]) + \\
&\quad + \mathbb{V}[w_q w_k w_v]\sum_{j,l;l\neq p}\sum b_j b_l(\text{Cov}(X_j^2, X_l^2) + \mathbb{V}^2[X_j])
\end{aligned} \tag{101}$$

Where $b_i = 1, \forall i \neq p$, $b_p = 3$.

$\square$