CW-CNN & CW-AN: Convolutional Networks and Attention Networks for CW-Complexes

Anonymous authors
Paper under double-blind review

Abstract

We present a novel framework for learning on CW-complex structured data points. Recent advances have discussed CW-complexes as ideal learning representations for problems in cheminformatics. However, there is a lack of available machine learning methods suitable for learning on CW-complexes. In this paper we develop notions of convolution and attention that are well defined for CW-complexes. These notions enable us to create the first Hodge-Laplacian informed neural network that can receive a CW-complex as input. We illustrate and interpret this framework in the context of supervised prediction.

1 Introduction

1.1 Complexes

Succinctly, a cell complex is an object in a category obtained by successively gluing together cells using pushouts. More formally, Whitehead (1949) defined them in the following way.

Definition 1.1. A cell complex K, or alternatively a *complex*, is a Hausdorff space which is the union of disjoint open cells e, e^n, e^n_i subject to the condition that the closure $\bar{e}^n$ of each n-cell, $e^n \in K$ is the image of a fixed n-simplex in a map $f : \sigma^n \to \bar{e}^n$ such that

(1) $f|\sigma^n - \partial\sigma^n$ is a homeomorphism onto e^n

(2) $\partial e^n \subset K^{n-1}$, where $\partial e^n = f\partial\sigma^n = \bar{e}^n - e^n$ and K^{n-1} is the $(n-1)$-section of K consisting of all the cells whose dimensionalities do not exceed $n-1$.

A CW-complex is a cell complex that has the weak topology and is closure finite. A complex K is said to be closure finite if and only if $K(e)$ is a finite subcomplex, for every cell $e \in K$. We say K has the weak-topology if and only if a subset $X \subset K$ is closed provided $X \cap \bar{e}$ is closed for each cell $e \in K$. To construct a CW-complex, we inductively glue cells together. More formally, Hatcher (2002) describes how we construct a finite CW-complex X as follows. Initially, we start with a collection of zero cells $X^0 = \{e^0_i\}^N_{i=0}$. X^0 is called the 0-skeleton. Then, for all $j \in \{1, \dots, n\}$ we take a collection of j-cells $\{e^j_i\}^N_{i=0}$ and glue their boundaries to points in the $j-1$ skeleton using continuous attaching maps $\phi^j_i : \partial e^j_i \to X^{j-1}$. Each j-cell is a topological space. Essentially, a CW-complex is constructed by taking a union of a sequence of topological spaces $\varnothing = X^{-1} \subset X^0 \subset X^1 \subset \cdots$ such that each X^j is obtained by attaching j-cells to X^{j-1}. In the language of category theory, we often think of the topology on finite CW-complex X as the direct limit of the diagram $X^{-1} \hookrightarrow X^0 \hookrightarrow X^1 \hookrightarrow \cdots \hookrightarrow X^k$ for some $k \in \mathbb{N}$. CW-complexes generalize the notion of a graph. A 1-dimensional CW-complex is a regular graph without loops. Moreover, every topological space is weakly homotopy equivalent to a CW-complex.

1.2 Learning on CW-complexes

Consider the following learning problem. Suppose we are presented with a dataset $\mathcal{D} = \{(x_i, y_i)\}^n_{i=1}$ where x_i is a CW-complex, $y_i \in \mathbb{R}^d$, and $n, d \in \mathbb{N}$. Then, the task of learning a function $\mathcal{F}$ such that $y_i = \mathcal{F}(x_i) + \epsilon$,

where ϵ is some error, naturally arises. We tackle this problem by developing a convolutional layer and attention mechanism for a CW-complex x_i. Essentially, we extend the work of Kipf & Welling (2017) to define a notion of convolution for CW-complexes. Additionally, we extend the work of Veličković et al. (2018) to develop a notion of attention for CW-complexes.

2 Related Work

2.1 Graph Neural Networks

Kipf & Welling (2017) develop a semi-supervised learning framework for graphs. The authors consider a graph-based semi-supervised learning approach, where label information is smoothed over the graph by applying a form of explicit graph-based regularization.

Definition 2.1. Let $\mathcal{G} = (V, E)$ be a graph. Let f be a neural network. Let λ be a weighting factor. Let X be a matrix of node feature vectors. Let A be the adjacency matrix for a graph $\mathcal{G}$. Let D be the degree matrix for $\mathcal{G}$. Finally, let $\Delta = D - A$ be the unnormalized graph Laplacian. Then, $\mathcal{L} = \mathcal{L}_0 + \lambda\mathcal{L}_{reg}$ where $\mathcal{L}_{reg} = \sum_{i,j} A_{ij}\|f(X_i) - f(X_j)\|^2 = f(X)^\top \Delta f(X)$. Note that $\mathcal{L}_0$ is the supervised loss with respect to the labeled part of the graph (Kipf & Welling, 2017).

The authors introduce a well behaved layer-wise propagation rule, and demonstrate semi-supervised classification of graphs.

Definition 2.2. Let $\mathcal{G}$ be a graph. Let X be a matrix of node feature vectors. Let A be the adjacency matrix for a graph $\mathcal{G}$. Let D be the degree matrix for $\mathcal{G}$. Let $\Delta = D - A$ be the unnormalized graph Laplacian. Additionally, define $\tilde{A} = A + I_N$ to be the adjacency matrix of $\mathcal{G}$ with self-connections, $\tilde{D}_{ii} = \sum_j \tilde{A}_{jj}$, $W^{(\ell)}$ be a weight matrix, and σ a nonlinearity. Then, we can consider a neural network $f(X, A)$ which follows the layer-wise propagation rule: $H^{(\ell+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(\ell)}W^{(\ell)}\right)$. Note that $H^{(\ell)} \in \mathbb{R}^{N\times M}$ is a matrix of activations in the ℓ-th layer and $H^{(0)} = X$. Neural networks of the form $f(X, A)$ which are composed by stacking hidden layers of the form H^ℓ are called graph convolutional networks (GCN) (Kipf & Welling, 2017).

2.2 Graph Attention Networks

Veličković et al. (2018) develop a notion of attention for Graphs. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ contains nodes $\mathcal{V} = \{1, \ldots, n\}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, where $(j, i) \in \mathcal{E}$ denotes an edge from a node j to a node i. We assume that every node $i \in \mathcal{V}$ has an initial representation $h_i^0 \in \mathbb{R}^{d_0}$.

Definition 2.3. A Graph Neural Network take in a set of node representations $\{h_i \in \mathbb{R}^d \mid i \in \mathcal{V}\}$ and the set of edges $\mathcal{E}$ as input. The layer outputs a new set of node representations $\{h_i^{'} \in \mathbb{R}^{d'} \mid i \in \mathcal{V}\}$, where the same parametric function is applied to every node given its neighbors $\mathcal{N}_i = \{j \in \mathcal{V} \mid (j, i) \in \mathcal{E}\}$: $h_i^{'} = f_\theta(h_i, \mathsf{AGGREGATE}(\{h_j \mid j \in \mathcal{N}_i\}))$. The design of f and $\mathsf{AGGREGATE}$ is what distinguishes Graph Neural Networks (Brody et al., 2022).

Definition 2.4. A Graph Attention Network computes a learned weighted average of the representations of $\mathcal{N}_i$. A scoring function $e : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ computes a score for every edge (j, i), which indicates the importance of the features of the neighbor j to node i. $e(h_i, h_j) = \text{LeakyReLU}(a^\top \cdot [Wh_i \,\|\, Wh_j])$ where $a \in \mathbb{R}^{2d'}$, $W \in \mathbb{R}^{d'\times d}$ are learned and $\|$ denotes vector concatenation. These attention scores are normalized across all neighbors $j \in \mathcal{N}_i$ using softmax and the attention function is defined as: $\alpha_{ij} = \text{softmax}_j(e(h_i, h_j)) = \frac{\exp(e(h_i,h_j))}{\sum_{j'\in\mathcal{N}_i}\exp(e(h_i,h_{j'}))}$. Then the Graph Attention Network computes a new node representation using nonlinearity σ as $h_i' = \sigma\left(\sum_{j\in\mathcal{N}_i}\alpha_{ij}\cdot Wh_j\right)$ (Veličković et al., 2018; Brody et al., 2022).

2.3 Gaussian Processes on Cellular Complexes

Alain et al. (2023) define the first Gaussian process on cell complexes. In doing so, the authors define the Hodge Laplacian and introduce important notation. We provide their definitions and re-use some of their notation. In order to define a path and function over a finite cellular complex or CW-complex X, one has to define a notion of chains and cochains.

Definition 2.5. Suppose X is an n-dimensional complex. Then, a k-chain c_k for $0 \leq k \leq n$ is simply a sum over the cells: $c_k = \sum_{i=1}^{N_k} \eta_i e_i^k, \quad \eta_i \in \mathbb{Z}$. The authors show this generalizes the notion of directed paths on a graph. The set of all k-chains on X is denoted by $C_k(X)$, which has the algebraic structure of a free Abelian group with basis $\{e_i^k\}_{i=1}^{N_k}$.

Definition 2.6. Given definition 2.5, the boundary operator naturally follows as $\partial_k : C_k(X) \to C_{k-1}(X)$. The operator ∂_k maps the boundary of a k-chain to a $k-1$-chain. This map is linear, thereby leading to the equation: $\partial_k \left(\sum_{i=1}^{N_k} \eta_i e_i^k \right) = \sum_{i=1}^{N_k} \eta_i \partial(e_i^k)$.

The authors then define the k-cochain (the dual notion of the k-chain) and the coboundary operator (the dual notion of the boundary operator).

Definition 2.7. Suppose X is an n-dimensional complex. Then, a k-cochain on X is a linear map $f : C_k(X) \to \mathbb{R}$ where $0 \leq k \leq n$. $f\left(\sum_{i=1}^{N_k} \eta_i e_i^k\right) = \sum_{i=1}^{N_k} \eta_i f(e_i^k)$ where $f(e_i^k) \in \mathbb{R}$ is the value of f at cell e_i^k. The space of k-cochains is defined as $C^k(X)$, which forms a real vector space with the dual basis $\{(e_i^k)^*\}_{i=1}^{N_k}$ such that $(e_i^k)^*(e_j^k) = \delta_{ij}$.

Definition 2.8. Given definition 2.7, the coboundary operator naturally follows as $d_k : C^k(X) \to C^{k+1}(X)$ which, for $0 \leq k \leq n$, is defined as $d_k = f(\partial_{k+1}(c))$ for all $f \in C^k(X)$ and $c \in C_{k+1}(X)$. Note that for $k \in \{-1, n\}$ $d_k f \equiv 0$.

Using these definitions, Alain et al. (2023) formally introduce a generalization of the Laplacian for graphs. They further prove that for $k = 0$ and identity weights, the Hodge Laplacian is the graph Laplacian. In essence, the authors prove $W_0 = I \implies \Delta_0 = B_1 W_1 B_1^\top$ and $W_1 = I \implies \Delta_0 = B_1 B_1^\top$, which is the standard graph Laplacian.

Definition 2.9. Let X be a finite complex. Then, we define a set of weights for every k. Namely, let $\{w_i^k\}_{i=1}^{N_k}$ be a set of real valued weights. Then, $\forall f, g \in C^k(X)$, one can write the weighted L^2 inner product as:$\langle f, g \rangle_{L^2(w^k)} := \sum_{i=1}^{N_k} w_i^k f(e_i^k) g(e_i^k)$. This inner product induces an adjoint of the coboundary operator $d_k^* : C^{k+1}(X) \to C^k(X)$. Namely, $\langle d_k^* f, g \rangle = \langle f, d_k g \rangle$ for all $f \in C^{k+1}(X)$ and $g \in C^k(X)$.

Definition 2.10. Using the previous definitions, the Hodge Laplacian $\Delta_k : C^k(X) \to C^k(X)$ on the space of k-cochains is then $\Delta_k := d_{k-1} \circ d_{k-1}^* + d_k^* \circ d_k$. The matrix representation is then $\Delta_k := B_k^\top W_{k-1}^{-1} B_k W_k + W_k^{-1} B_{k+1} W_{k+1} B_{k+1}^\top$. Here, $W_k = \text{diag}(w_1^k, \ldots, w_{N_k}^k)$ is the diagonal matrix of cell weights and B_k is the order k incidence matrix, whose j-th column corresponds to a vector representation of the cell boundary ∂e_j^k viewed as a $k-1$ chain.

2.4 CW-Complex Networks

Recent advances have proposed neural networks for CW-complexes. We discuss these methods and describe key distinctions between our method. Hajij et al. (2021) propose an inter-cellular message-passing scheme on cell complexes. Under the proposed scheme, the propagation algorithm then performs a sequence of message passing executed between cells in X defined as

$$h_{c^{n-1}}^{(k)} := \alpha_{n-1}^{(k)} \left(h_{c^{n-1}}^{(k-1)}, E_{a^{n-1} \in \mathcal{N}_{adj}(c^{n-1})} \left(\phi_{n-1}^{(k)} (h_{c^{n-1}}^{(k-1)}, h_{a^{n-1}}^{(k-1)}, F_{e^n \in \mathcal{CO}[a^{n-1}, c^{n-1}]} (h_{e^n}^{(0)})) \right) \right) \tag{1}$$

where $h_{e^m}^{(k)}$, $h_{a^m}^{(k)}$, $h_{c^m}^{(k)} \in \mathbb{R}^{\ell_m^k}$, E, and F are permutation invariant differentiable functions and $\alpha_j^{(k)}$, $\phi_j^{(k)}$ are trainable differentiable functions. Hajij et al. (2021) then utilize this message passing framework to define a convolutional message passing scheme $H^{(k)} := \text{ReLU}(\hat{A}_{adj} H^{(k-1)} W^{k-1})$. We improve upon this framework in numerous ways. First we leverage the Hodge-Laplacian thereby making our network Hodge-aware. This Hodge-awareness allows for effective learning on CW-complexes when compared to message-passing networks (Bodnar et al., 2021b). Additionally, we rely on boundary operators $B_{k+1}^\top$ and B_{k+1} as opposed to using message passing or permutation invariant differentiable functions. This enables a more time and space efficient way in which to process the CW-complex. We show in Lemma 3.2 that $\dim(H^{(k)}) = N_k \times N_k$. Moreover, our form of convolution is defined for any non-linearity not just ReLU. Finally, as stated by Hajij et al. (2021), one may wish to train a CCXN for every k-cells adjacency matrix individually. Consequently, one has to

train $n-1$ many networks Contrastingly, our architecture requires training only one network for every k-cells adjacency matrix.
Similarly Bodnar et al. (2021a) extend a message-passing algorithm to cell complexes. Bodnar et al. (2021a) essentially define lifting transformations, $f : G \to X$, augmenting a graph with higher-dimensional constructs. This results in a multi-dimensional and hierarchical message passing procedure over the input graph (Bodnar et al., 2021a). The authors specify this procedure over the space of chemical graphs in section 4, defining message passing from atoms to bonds, and bonds to rings. In this manuscript we propose a network that receives as input a CW-complex of dimension $n \geq 1$, which need not be a graph. Additionally we do not rely on lifting transformations or an explicit message passing scheme. In Appendix D, Bodnar et al. (2021a) discuss how one can define a convolutional operator on cochains. However, the authors leave development of an actual convolutional network as future work and do not mention attention (Bodnar et al., 2021a). We proceed in this direction by developing a Hodge-Laplacian informed convolutional network that receives a CW-complex as input.
Giusti et al. (2023) introduce a neural architecture operating on data defined over the vertices of a graph. The approach described leverages a lifting algorithm that learns edge features from node features, then applies a cellular attention mechanism, and finally applies pooling. In particular, Giusti et al. (2023) define a cellular lifting map as a skeleton-preserving function $s : G \to C_G$ incorporating graph G into regular cell complex C_G. Using the cellular lifting map the authors define attentional lift, giving way to their attention mechanism. The procedure computes F^0 many attention heads such that when given input graph $G = (\mathcal{V}, \mathcal{E})$, for vertices $i, j \in \mathcal{V}$ connected by edge $e \in \mathcal{E}$, edge features $x_e \in \mathbb{R}^{F^0}$ are computed by concatenating attention scores. Mathematically written as

$$x_e = g(x_i, x_j) = \|_{k=1}^{F^0} a_n^k(x_i, x_j), \ \forall \ e \in \mathcal{E} \tag{2}$$

Giusti et al. (2023) utilize equation (2) above to define their layer propagation scheme by combining the attentional lift with message passing and aggregating using any permutation invariant operator, see equation 6 in (Giusti et al., 2023). We improve on this approach in numerous ways. First we leverage the Hodge-Laplacian thereby making our network Hodge-aware. This Hodge-awareness allows for effective learning on CW-complexes when compared to message-passing networks (Bodnar et al., 2021b). In contrast to the work of Giusti et al. (2023), our network does not rely on incorporating an input graph G into a regular cell complex C_G or attentional lift. Consequently, our method saves computations. By propagating over cells via the boundary operator, our method can account for the topology of the individual cell and its open neighborhoods. Finally, our method is leveraged to develop multi-head attention and a transformer-like architecture.

3 Hodge-Laplacian informed CW-Complex Networks

We extend the work of Kipf & Welling (2017) and Veličković et al. (2018) by developing a notion of convolution and attention for CW-complexes that is informed by the Hodge-Laplacian.

3.1 Convolutional CW-complex Layer

Definition 3.1. Let X be a finite n-dimensional CW-complex. Then for $k \in [0, n]$ let $A_k \in \mathbb{R}^{N_k \times N_k}$ be a matrix of cell feature vectors and let Δ_k be the Hodge Laplacian. Then we can define a **Convolutional CW-complex Network (CW-CNN)** $f(X)$ as being composed by stacking hidden layers $H^{(k)}$ according to the following layer-wise propagation rule:

$$H^{(k+1)} = \sigma\left(B_{k+1}^{\top}\left(\Delta_k A_k H^{(k)}\right) B_{k+1}\right) \tag{1}$$

Initially, we set $H^{(0)} = X^0 \in \mathbb{R}^{N_0 \times N_0}$, which is the matrix representation of the zero-skeleton of CW-complex X. Recall the definitions of the boundary operator (definition 2.6), coboundary operator (definition 2.8), and Hodge-Laplacian (definition 2.10). By definition 2.10, $\Delta_k = B_k^{\top} W_{k-1}^{-1} B_k W_k + W_k^{-1} B_{k+1} W_{k+1} B_{k+1}^{\top}$. Additionally, by definitions 2.6 and 2.8, $B_k \in \mathbb{Z}^{N_{k-1} \times N_k}$, $B_{k+1}^{\top} \in \mathbb{Z}^{N_{k+1} \times N_k}$, and the weight matrix $W_k = \mathrm{diag}(w_1^k, \ldots, w_{N_k}^k) \in \mathbb{R}^{N_k \times N_k}$.

Checking the dimensions we can see that $\Delta_k \in \mathbb{R}^{N_k \times N_k}$:

$$\dim(B_k^\top W_{k-1}^{-1} B_k W_k) = (N_k \times N_{k-1})(N_{k-1} \times N_{k-1})(N_{k-1} \times N_k)(N_k \times N_k) = N_k \times N_k \tag{2}$$

$$\dim(W_k^{-1} B_{k+1} W_{k+1} B_{k+1}^\top) = (N_k \times N_k)(N_k \times N_{k+1})(N_{k+1} \times N_{k+1})(N_{k+1} \times N_k) = N_k \times N_k \tag{3}$$

Therefore $\Delta_k \in \mathbb{R}^{N_k \times N_k} \implies \dim(\Delta_k A_k) = N_k \times N_k$. Additionally, we know from above $\dim(B_{k+1}) = N_k \times N_{k+1}$ and $\dim(B_{k+1}^\top) = N_{k+1} \times N_k$. Therefore, by induction, we can show $\dim(H^{(k)}) = N_k \times N_k$ (Lemma 3.2). Formally, we call B_k the order k incidence matrix, and let σ be any nonlinearity. Thus, using the layer-wise propagation rule from equation (1), we can define a neural network $f(X)$ by stacking the hidden layers $H^{(k)}$. We call such a network a Convolutional CW-complex Network, or CW-CNN for short.

Lemma 3.2. *The dimension of hidden layer k in a CW-CNN is* $\dim(H^{(k)}) = N_k \times N_k$.

Proof. We want to show that $\dim(H^{(k)}) = N_k \times N_k$. For the base case ($k = 0$) we define $H^{(0)} = X^0 \in \mathbb{R}^{N_0 \times N_0}$. Let the inductive hypothesis $P(j)$ be that $\forall j \in \{0, 1, \ldots, k-1\}$ $\dim(H^{(j)}) = N_j \times N_j$. Then, we can show $P(j) \implies P(j+1)$ using equation (1). We know $\dim(H^{(j+1)}) = \dim(\sigma(B_{j+1}^\top(\Delta_j A_j H^{(j)}) B_{j+1})) = (N_{j+1} \times N_j)(N_j \times N_j)(N_j \times N_j)(N_j \times N_j)(N_j \times N_{j+1}) = N_{j+1} \times N_{j+1}$. Thus we see $\dim(H^{(k)}) = N_k \times N_k$. □

The weight matrices W_k can be randomly initialized by choosing w_i^k randomly or by adopting a similar strategy to He Initialization (He et al., 2015). In order to train this network, one would update the weight matrices W_k with gradient descent and define a loss function $\mathcal{L}$. One can replace W_k^{-1} by $W_k^\dagger$, the Moore-Penrose pseudoinverse of W_k, in the expression for Δ_k. Then, let $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^M$ be a dataset where $\mathbf{x}_i$ is a CW-complex and $\mathbf{y}_i \in \mathbb{R}^d$ where $d \in \mathbb{N}$. Our learning paradigm then becomes $\mathbf{y}_i = f(\mathbf{x}_i) + \varepsilon$, where f is a CW-CNN. If we let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_M]$ and $\mathbf{y} = [\mathbf{y}_1, \ldots, \mathbf{y}_M]$ we want to choose the weight matrices W_k for each CW-complex $\mathbf{x}_i$ such that we solve $\min_W \|f(\mathbf{X}) - \mathbf{y}\|_2^2$.

3.2 CW-complex Attention

In order to define, a CW-complex Attention Network (CW-AN) we must first develop a notion of connectedness or adjacency for CW-complexes. The analogue of adjacency for CW-complexes is termed incidence. We collect these incidence values, which are integers, in a matrix. This matrix is defined below as B_k in equation (4).

3.2.1 Incidence Matrices

Let the relation $\prec$ denote incidence. If two cells e_j^{k-1} and e_i^k are incident, we write $e_j^{k-1} \prec e_i^k$. Similarly, if two cells e_j^{k-1} and e_i^k are not incident, we write $e_j^{k-1} \not\prec e_i^k$. Additionally, let the relation $\sim$ denote orientation. We write $e_j^{k-1} \sim e_i^k$ if the cells have the same orientation. If two cells have the opposite orientation we write $e_j^{k-1} \nsim e_i^k$. This enables us to define the classical incidence matrix (Sardellitti & Barbarossa, 2024). Traditionally we define for indices i and j the value of $(B_k)_{j,i}$ as:

$$(B_k)_{j,i} = \begin{cases} 0 & \text{if } e_j^{k-1} \not\prec e_i^k \\ 1 & \text{if } e_j^{k-1} \prec e_i^k \text{ and } e_j^{k-1} \sim e_i^k \\ -1 & \text{if } e_j^{k-1} \prec e_i^k \text{ and } e_j^{k-1} \nsim e_i^k \end{cases} \tag{4}$$

The matrix B_k establishes which k-cells are incident to which $k-1$-cells. We know from above that $B_k \in \mathbb{Z}^{N_{k-1} \times N_k}$. Let $\mathrm{Col}_j(B_k)$ denote the j-th column of B_k. We know that $\mathrm{Col}_j(B_k) = \mathbb{Z}^{N_{k-1} \times 1}$ which corresponds to a vector representation of the cell boundary ∂e_j^k viewed as a $k-1$ chain (Alain et al., 2023).

Using the definition of an incidence matrix and the relation $\prec$ we can now define a CW-AN.

Definition 3.3. Let X be a finite n-dimensional CW-complex. Then for $k \in [0, n]$ we can define a **CW-complex Attention Network (CW-AN)**. A CW-AN computes a learned weighted average of the

representations of each skeleton. We start by initializing for all i the starting hidden state $h^{(0)}_{e^0_i} = \left[e^0_1, \ldots, e^0_{N_0}\right]^\top \left(\mathrm{Col}_i(B_0^\top)\right)^\top \in \mathbb{R}^{N_0 \times N_0}$. We then define a scoring function $\mathcal{S}$

$$\mathcal{S}(h^{(k)}_{e^k_i}, h^{(k-1)}_{e^{k-1}_j}) = \text{LeakyReLU}\left(\left[W_k h^{(k)}_{e^k_i} \parallel B_k^\top W_{k-1} h^{(k-1)}_{e^{k-1}_j} B_k\right] a^\top\right) \tag{5}$$

where $W_k \in \mathbb{R}^{N_k \times N_k}$, $W_{k-1} \in \mathbb{R}^{N_{k-1} \times N_{k-1}}$, $B_k \in \mathbb{R}^{N_{k-1} \times N_k}$, $a^\top \in \mathbb{R}^{2N_k \times N_k}$, and $\parallel$ denotes vector concatenation. Checking dimensions, we see that:

$$\dim(W_k h^{(k)}_{e^k_i}) = (N_k \times N_k)(N_k \times N_k) = N_k \times N_k \tag{6}$$

Additionally, for e^{k-1}_j we see:

$$\dim(B_k^\top W_{k-1} h^{(k-1)}_{e^{k-1}_j} B_k) = (N_k \times N_{k-1})(N_{k-1} \times N_{k-1})(N_{k-1} \times N_{k-1})(N_{k-1} \times N_k) = N_k \times N_k \tag{7}$$

By equations (6) and (7) we know:

$$\dim\left(\left[W_k h^{(k)}_{e^k_i} \parallel B_k^\top W_{k-1} h^{(k-1)}_{e^{k-1}_j} B_k\right]\right) = N_k \times 2N_k \tag{8}$$

Equations (5) and (8) imply:

$$\dim\left(\mathcal{S}(h^{(k)}_{e^k_i}, h^{(k-1)}_{e^{k-1}_j})\right) = \dim\left(\text{LeakyReLU}\left(\left[W_k h^{(k)}_{e^k_i} \parallel B_k^\top W_{k-1} h^{(k-1)}_{e^{k-1}_j} B_k\right] a^\top\right)\right) = N_k \times N_k \tag{9}$$

Therefore, we can compute attention scores $\alpha_{e^k_i, e^k_j}$, which are normalized over all incident cells. Let the set $\mathcal{N}_{e^k_i} := \left\{e^{k-1}_{j'} \mid e^{k-1}_{j'} \prec e^k_i \wedge (e^{k-1}_{j'} \sim e^k_i \vee e^{k-1}_{j'} \nsim e^k_i)\right\}$ contain all cells incident to e^k_i. We define a variation of the standard Softmax function called Cell-Softmax, which converts cells to a probability distribution.

$$\alpha_{e^k_i, e^{k-1}_j} = \text{Cell-Softmax}\left(\mathcal{S}(h^{(k)}_{e^k_i}, h^{(k-1)}_{e^{k-1}_j})\right) = \frac{\exp\left(\mathcal{S}(h^{(k)}_{e^k_i}, h^{(k-1)}_{e^{k-1}_j})\right)}{\sum\limits_{e^{k-1}_{j'} \in \mathcal{N}_{e^k_i}} \exp\left(\mathcal{S}(h^{(k)}_{e^k_i}, h^{(k-1)}_{e^{k-1}_{j'}})\right)} \in \mathbb{R}^{N_k \times N_k} \tag{10}$$

This enables us to define our update rule for computing later hidden states:

$$h^{(k)}_{e^k_i} = \sigma\left(\sum_{e^{k-1}_{j'} \in \mathcal{N}_{e^k_i}} \alpha_{e^k_i, e^{k-1}_{j'}} B_k^\top W_{k-1} h^{(k-1)}_{e^{k-1}_{j'}} B_k\right) \tag{11}$$

We know from equation (7) that

$$\dim(h^{(k)}_{e^k_i}) = \dim\left(\alpha_{e^k_i, e^{k-1}_{j'}} B_k^\top W_{k-1} h^{(k-1)}_{e^{k-1}_{j'}} B_k\right) = (N_k \times N_k)(N_k \times N_k) = N_k \times N_k \tag{12}$$

Thus, we have defined a notion of self-attention. A CW-AN is then composed by stacking numerous $h^{(k)}_{e^k_i}$ for all $k \in [0, n]$ and all cells.

We note that all W_i and $a^\top$ are learned. To stabilize the learning process of this self-attention mechanism, we can extend definition 3.3 to develop multi-head attention. Just as in Veličković et al. (2018), we can concatenate K independent self-attention mechanisms to execute equation (11). This results in the following representation

$$h_i^{'} = \Big\|_{\ell=1}^{K} \sigma\left(\sum_{e^{k-1}_{j'} \in \mathcal{N}_{e^k_i}} \alpha^{(\ell)}_{e^k_i, e^{k-1}_{j'}} B_k^\top W^{(\ell)}_{k-1} h^{(k-1)}_{e^{k-1}_{j'}} B_k\right) \tag{13}$$

where $\alpha^{(\ell)}_{e^k_i, e^{k-1}_{j'}}$ are normalized attention coefficients computed by the ℓ-th attention mechanism and $W^{(\ell)}_{k-1}$ is the corresponding weight matrix. In practice, the operation of the self-attention layer described in definition 3.3 can be parallelized. One can develop a transformer based architecture by combining the multi-head attention layer described in equation (13) with modified Add&Layer Norm as well as Feed Forward networks, equivalent to those developed by Vaswani et al. (2017). The weight matrices W_k can be randomly initialized or one may adopt a different strategy.

4 Model Architecture

We develop two distinct networks for one synthetic task.

4.1 CW-CNN architecture

Graph convolutional networks are thought of as models which efficiently propagate information on graphs (Kipf & Welling, 2017). Convolutional networks traditionally stack multiple convolutional, pooling, and fully-connected layers to encode image-specific features for image-focused tasks (O'Shea & Nash, 2015). The CW-CNN is motivated by similar principles. In particular, a CW-CNN efficiently propagates information on CW-complexes and encodes topological features for tasks in which geometry plays a pivotal role.

There are numerous potential use cases for such an architecture. In the areas of drug design, molecular modeling, and protein informatics the three dimensional structure of molecules plays a pivotal role and numerous approaches in these areas have attempted to meaningfully extract or utilize geometric information (Kuang et al., 2024; Gebauer et al., 2022; Zhung et al., 2024; Isert et al., 2023).

A CW-CNN is composed of a stack of Convolutional CW-complex layers as described above in definition 3.1. One may additionally add pooling, linear, or dropout layers. In our experiment we stack Convolutional CW-complex layers, and follow up with a Linear layer and GELU activation. The architecture is pictured below.

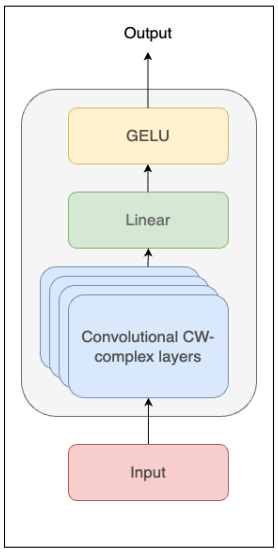

Figure 1: CW-CNN architecture.

4.2 CW-AT architecture

Initially, Graph Attention Networks were developed as a method to perform node classification on graph-structured data (Veličković et al., 2018). Transformers are neural sequence transduction models that maintain an encoder decoder structure, wherein an input sequence $(x_1, \ldots, x_n)$ is mapped to a sequence of continuous

representations $(z_1, \ldots, z_n)$. Then $(z_1, \ldots, z_n)$ are passed to the decoder for generating an output sequence $(y_1, \ldots, y_m)$ in an auto-regressive fashion (Vaswani et al., 2017). The CW-AT is motivated by similar principles. In particular, a CW-AT leverages a different kind of attention mechanism, which can perform classification or regression on CW-complex structured data. While one can theoretically setup a sequence-to-sequence learning problem utilizing CW-complexes, we do not venture into such problems. However, we develop an architecture that vaguely resembles the classic Transformer with far fewer parameters.

There are numerous potential use cases for such an architecture. One can attempt language translation, image captioning and sequence transformation tasks (Sutskever et al., 2014). Doing so would require viewing a word or entity as a CW-complex. This is somewhat reasonable since the 1-skeleton of a CW-complex is a graph without loops (Whitehead, 1949). Graphs have appeared in numerous contexts within Natural language processing. Classically vertices can encode text units of various sizes and characteristics such as words, collocations, word senses, sentences and documents (Nastase et al., 2015). Edges may represent relationships such as co-occurrence (Nastase et al., 2015). One can replace the notion of vertex with cell and edge with a kind of gluing map and extend these ideas to CW-complexes. One can represent co-occurence for instance by scaling the matrix B_k.

A CW-AT is composed of two separate networks, with each receiving as input a CW-complex. In each network the input is processed by a Multi-Cellular Attention mechanism as described by equation (13). Afterwards one may apply dropout, feed forward, and layer norm. The outputs from the first network are combined with output from the second network through addition and applying SELU activation. Finally, a linear layer is used to get the desired output shape. One may apply a Softmax if the output is to be viewed as probabilities.

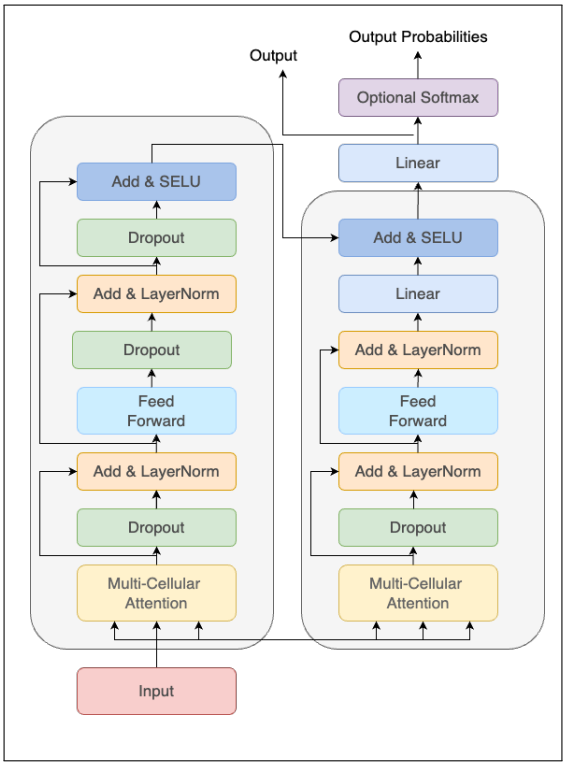

Figure 2: CW-AT architecture.

5 Experiment

This section describes our experimental setup and training regime.

Table 1: CW-CNN and CW-AT attain low test-set RMSE.

Model	RMSE	Parameters
CW-AT	0.02533091887133196	310
CW-CNN	$1.1487752999528311 \times 10^{-5}$	30

5.1 Dataset

We construct a synthetic dataset consisting of CW-complexes and attempt to predict the number of cells in each complex. This task is intended to be a proof of concept. Let $\mathcal{D} = \{x_i, y_i\}_{i=1}^{N}$ be our dataset. Here we choose $N = 500$ and let x_i be a CW-complex and let $y_i \in \mathbb{R}$ be the number of cells found in complex x_i.

5.2 Hardware and Wall clock time

We train both of our models on one AWS m7g.2xlarge instance. For both models, each training step took approximately 0.1 seconds. We trained both models for only 400 steps. Therefore, total training took approximately 40 seconds for each model.

5.3 Optimizer

We utilized SGD in order to train both models. For our CW-CNN we utilized learning rate $\eta = 0.001$ and momentum $\alpha = 0.9$. For our CW-AT we utilized learning rate $\eta = 0.001$, momentum $\alpha = 0.7$ and weight decay $\lambda = 0.02$.

5.4 Regularization

We utilized regularization during training the CW-AT. We apply dropout in the CW-AT with $P_{drop} = 0.1$ for every dropout layer shown in Figure 2.

6 Results

Using an 80/20 train/test split we report test set accuracy for both models. We summarize our results in Table 1.

On our synthetic task, we demonstrate low test-set RMSE. At the time of experiment, there are no other existing neural networks with which to draw comparison. We interpret our experiment as a proof of concept demonstrating that such an architecture is feasible, compute-efficient, and well-defined. In the event that one wishes to develop deeper architectures, stacking more layers or increasing model dimensions are well defined.

7 Conclusion

In this work, we presented the CW-CNN and CW-AT, the first types of neural network that can receive CW-complexes as input. We demonstrate high accuracy with relatively few parameters on a synthetic task. These results have implications for machine learning tasks in which geometric information or three dimensional structure plays a pivotal role. These areas include, but are not limited to, molecular design, cheminformatics and drug discovery. Additionally one may view tasks involving graphs in natural language as good candidates for a CW-complex representation and correspondingly a CW-AT. CW-complexes capture interactions between higher-order cells enabling the one to model polyadic relations effectively. Our neural networks enable learning on CW-complexes thereby facilitating the learning of polyadic relations. We are excited about the future of these models and plan to apply them to other tasks in cheminformatics and natural language processing.

References

Mathieu Alain, So Takao, Brooks Paige, and Marc Peter Deisenroth. Gaussian processes on cellular complexes. *arXiv preprint arXiv:2311.01198*, 2023.

Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yuguang Wang, Pietro Liò, Guido F Montufar, and Michael Bronstein. Weisfeiler and lehman go cellular: Cw networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 2625–2640. Curran Associates, Inc., 2021a. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/157792e4abb490f99dbd738483e0d2d4-Paper.pdf.

Cristian Bodnar, Fabrizio Frasca, Yu Guang Wang, Nina Otter, Guido Montúfar, Pietro Liò, and Michael Bronstein. Weisfeiler and lehman go topological: Message passing simplicial networks, 2021b. URL https://arxiv.org/abs/2103.03212.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=F72ximsx7C1.

Niklas WA Gebauer, Michael Gastegger, Stefaan SP Hessmann, Klaus-Robert Müller, and Kristof T Schütt. Inverse design of 3d molecular structures with conditional generative neural networks. *Nature communications*, 13(1):973, 2022.

Lorenzo Giusti, Claudio Battiloro, Lucia Testa, Paolo Di Lorenzo, Stefania Sardellitti, and Sergio Barbarossa. Cell attention networks. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2023.

Mustafa Hajij, Kyle Istvan, and Ghada Zamzmi. Cell complex neural networks, 2021. URL https://arxiv.org/abs/2010.00743.

Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015. URL http://arxiv.org/abs/1502.01852.

Clemens Isert, Kenneth Atz, and Gisbert Schneider. Structure-based drug design with geometric deep learning. *Current Opinion in Structural Biology*, 79:102548, 2023. ISSN 0959-440X. doi: https://doi.org/10.1016/j.sbi.2023.102548. URL https://www.sciencedirect.com/science/article/pii/S0959440X23000222.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Taojie Kuang, Yiming Ren, and Zhixiang Ren. 3d-mol: A novel contrastive learning framework for molecular property prediction with 3d information, 2024. URL https://arxiv.org/abs/2309.17366.

Vivi Nastase, Rada Mihalcea, and Dragomir R. Radev. A survey of graphs in natural language processing. *Natural Language Engineering*, 21(5):665–698, 2015. doi: 10.1017/S1351324915000340.

Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

Stefania Sardellitti and Sergio Barbarossa. Topological signal processing over generalized cell complexes. *IEEE Transactions on Signal Processing*, 2024.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

John HC Whitehead. Combinatorial homotopy. ii. *Bulletin of the American Mathematical Society*, 55:213–245, 1949.

Wonho Zhung, Hyeongwoo Kim, and Woo Youn Kim. 3d molecular generative framework for interaction-guided drug design. *Nature Communications*, 15(1):2688, 2024.