GEOMETRIC ALGEBRA TRANSFORMER (GATR)

Ali Ghasemi Goudarzi
Sapienza University of Rome
Rome
ghasemigoudarzi.2053156@studenti.uniroma1.it

Yusupha Juwara
Sapienza University of Rome
Rome
juwara.1936515@studenti.uniroma1.it

ABSTRACT

Projective geometric algebra (PGA)[1] provides a powerful algebraic framework for representing geometric data in a manner that captures relevant **symmetries** and **structural properties**. In this work, we explore several deep learning architectures based on PGA for the task of classifying 3D vascular geometries. We implement two linear baselines using **SVM**[2] with linear kernels and **Logistic Regression**, both achieving **optimal F1 scores**. We then implement the recently proposed **Geometric Algebra Transformer (GATr)**[3] architecture, also attaining top performance. To analyze which components are most critical, we implement **ablated versions** using only the **EquiLinear** and **BiEquiLinear layers**, finding that even these simpler equivariant models approximately match the optimal results. Additionally, we propose novel **EquiLSTM** and **BiEquiLSTM** architectures that integrate PGA representations with LSTM[4] networks and show these models also achieve optimal F1 scores on the task. Our findings demonstrate the effectiveness of incorporating geometric algebra[5] representations and equivariant operations into deep learning models for geometric processing tasks.

***K*eywords** Geometric Algebra · Projective Geometric Algebra · Equivariant Deep Learning · Geometric Algebra Transformer (GATr) · EquiLSTM · BiEquiLSTM

1 Projective Geometric Algebra (PGA)

Projective Geometric Algebra (PGA)[1], denoted as $\mathcal{G}_{3,0,1}$, is a 16-dimensional algebra that provides a powerful framework for representing and manipulating geometric objects and transformations in 3D space. It extends traditional vector algebra to include **geometric primitives** like points, lines, planes, and volumes, as well as **geometric operations** such as rotations, translations, and reflection by incorporating a fourth homogeneous coordinate, e_0, resulting in a vector space spanned by the basis $\{e_0, e_1, e_2, e_3\}$.

1.1 Key Properties and Operations

Representation of Geometric Objects: PGA can represent scalars, points, lines, planes, and pseudoscalars in 3D space as multivectors of different **grades**. The 16-dimensional basis vectors' space summary is as follows:

- **Scalar** $\{1\}$ – Grade 0
- **Vectors (Planes)** $\{e_0, e_1, e_2, e_3\}$ – Grade 1
- **Bivector (Lines)** $\{e_{01}, e_{02}, e_{03}, e_{12}, e_{13}, e_{23}\}$ – Grade 2
- **Trivectors (Points)** $\{e_{012}, e_{013}, e_{023}, e_{123}\}$ – Grade 3
- **Pseudoscalar** $\{e_{0123}\}$: – Grade 4

[1]This part is heavily inspired by [6]

Representation of Transformations: PGA can represent rotations, reflections, and translations as **versors** (products of unit vectors) in the algebra. These transformations form the **Pin** and **Spin groups**, which are double covers of the Euclidean group $E(3)$ [2] and the special Euclidean group $SE(3)$ [3], respectively.

Geometric Product : The **geometric product**, denoted by $\wedge$, combines the inner and outer products of vectors, allowing for the composition of geometric objects and transformations.

$$xy = \overbrace{\langle x, y \rangle}^{\textbf{inner prod}} + \overbrace{x \wedge y}^{\textbf{outer prod}}$$

Reverse: The reverse is defined by reversing the sequence of the basis vectors in each blade. For a k-blade $A = a_1 \wedge a_2 \wedge \ldots \wedge a_k$ where $\wedge$ represents the outer product, the reverse $\tilde{A}$ is given by:

$$\tilde{A} = a_k \wedge a_{k-1} \wedge \ldots \wedge a_1 = (-1)^{k(k-1)/2}(a_1 \wedge a_2 \wedge \cdots \wedge a_k)$$

Dual: The dual operator essentially reflects geometric elements across the duality relationship between the primal and dual spaces. It acts on basis elements by swapping "empty" and "full" dimensions.

Join: The **join** was designed to be dual to the **meet**. In PGA, where $\wedge$ (**meet**) is an intersection, it is customary to denote the **join** by a $\vee$:

$$A \vee B = (A^* \wedge B^*)^*$$

Grade involution $\hat{x}$: Flips the sign of odd-grade elements such as vectors and trivectors, while leaving even-grade elements unchanged.

Sandwich Product: To apply a versor u to any element x, the sandwich product is used and it is **grade-preserving**:

$$\rho_u(x) = \begin{cases} uxu^{-1} & \text{if } u \text{ is even} \\ u\hat{x}u^{-1} & \text{if } u \text{ is odd} \end{cases}$$

Pin and Spin Groups: The **Pin group**, denoted as $\text{Pin}(n)$, is the group of all versors (products of unit vectors), both even and odd, under geometric multiplication. It is a double cover of the Euclidean group $E(n)$. The **Spin group**, denoted as $\text{Spin}(n)$, is a subgroup of the Pin group, consisting of only the even versors. It is a double cover of the special Euclidean group $SE(n)$.

Checking Equivariance: Equivariance, in this context, means that applying a transformation before the operation yields the same result as applying the operation and then the transformation.

We first compute the transformation on the two multivectors and then apply the operation (layer) on them. Then, we apply the operation (layer) on the multivectors and then compute the transformation. If the distance between these two values is near 0, we have high equivariance.

1.2 Embedding of Geometric Objects

In PGA, **scalars** and **multivectors** (elements built from the basis) can represent various geometric entities, including points, lines, planes, volumes, rotations, reflections, and translations in $\mathbb{R}^3$. These objects are embedded into PGA through a specific mapping, as shown in the figure below.

Scalar: A scalar $\lambda \in R$ becomes a multivector given by $[\lambda, 0, ..., 0]$

Plane: Given a plane with unit normal vector **n** at directed distance δ from the origin (measured in the direction **n**, so δ**n** is a location on the plane), we pick as the homogeneous representative the vector

$$\mathbf{n} + \delta e_0 = \delta e_0 + n_1 e_1 + n_2 e_2 + n_3 e_3$$

In our case, given a face with 3 points P, Q, R the normal is $\mathbf{n} = \vec{PQ} \times \vec{PR}$ and the shift is $\delta = \mathbf{n} \cdot P$.

[2] E(3) encompasses all possible transformations (rotations, translations, reflections, and combinations) that preserve distances and angles in Euclidean space.

[3] SE(3) is a subgroup of E(3) that represents only proper rigid-body transformations, i.e., rotations and translations, but excluding reflections.

Object / operator	Scalar	Vector		Bivector		Trivector		PS
	1	e_0	e_i	e_{0i}	e_{ij}	e_{0ij}	e_{123}	e_{0123}
Scalar $\lambda \in \mathbb{R}$	λ	0	0	0	0	0	0	0
Plane w/ normal $n \in \mathbb{R}^3$, origin shift $d \in \mathbb{R}$	0	d	n	0	0	0	0	0
Line w/ direction $n \in \mathbb{R}^3$, orthogonal shift $s \in \mathbb{R}^3$	0	0	0	s	n	0	0	0
Point $p \in \mathbb{R}^3$	0	0	0	0	0	p	1	0
Pseudoscalar $\mu \in \mathbb{R}$	0	0	0	0	0	0	0	μ
Reflection through plane w/ normal $n \in \mathbb{R}^3$, origin shift $d \in \mathbb{R}$	0	d	n	0	0	0	0	0
Translation $t \in \mathbb{R}^3$	1	0	0	$\frac{1}{2}t$	0	0	0	0
Rotation expressed as quaternion $q \in \mathbb{R}^4$	q_0	0	0	0	q_i	0	0	0
Point reflection through $p \in \mathbb{R}^3$	0	0	0	0	0	p	1	0

Table 1: Embeddings of common geometric objects and transformations into the projective geometric algebra $\mathbb{G}_{3,0,1}$. The columns show different components of the multivectors with the corresponding basis elements, with $i, j \in \{1, 2, 3\}, j \neq i$, i.e. $ij \in \{12, 13, 23\}$. For simplicity, we fix gauge ambiguities (the weight of the multivectors) and leave out signs (which depend on the ordering of indices in the basis elements).

Figure 1: Embedding of geometric objects and transformations into PGA.

Line: We can represent a line when we have been given its direction **u** and some location **p** as

$$L = (p_3u_2 - p_2u_3)\mathbf{e}_{01} + (p_1u_3 - p_3u_1)\mathbf{e}_{02} + (p_2u_1 - p_1u_2)\mathbf{e}_{03} + u_3\mathbf{e}_{12} - u_2\mathbf{e}_{13} + u_1\mathbf{e}_{23}$$

Point: In PGA, points are trivectors. We would of course expect the Euclidean coordinates of a point to show up as the coefficients of the 3-blade that represents it. That parametrization is most easily seen when we construct a point at location $\mathbf{x} = x_1\mathbf{e}_1 + x_2\mathbf{e}_2 + x_3\mathbf{e}_3$ by intersecting three orthogonal planes p_i parallel to the coordinate planes $\mathbf{e}_i$ at the properly signed distances $x_i = \mathbf{x} \cdot \mathbf{e}_i$.

$$X = p_1 \wedge p_2 \wedge p_3 = (\mathbf{e}_1 + x_1e_0) \wedge (\mathbf{e}_2 + x_2e_0) \wedge (\mathbf{e}_3 + x_3e_0) = x_3\mathbf{e}_{012} - x_2\mathbf{e}_{013} + x_1\mathbf{e}_{023} + \mathbf{e}_{123}$$

2 Dataset and Processing

Dataset:

- The first class includes idealized arteries with a single outlet and randomly located stenoses.
- The second class contains bifurcating arteries.

(a) Single Outlet (b) Bifurcation

Figure 2: (Left) Idealized arteries with a single outlet and randomly located stenoses. (Right) Bifurcating arteries as shown.

The dataset comprises 3D vascular geometries, which are preprocessed and embedded into multivectors of the PGA $\mathcal{G}_{3,0,1}$ before being input into a model. Geometric objects are embedded as described in Table 1, such as embedding a point $(x, y, z) \in \mathbb{R}^3$ as $1 + xe_1 + ye_2 + ze_3$, and a plane with normal (a, b, c) and origin shift d as $d + ae_1 + be_2 + ce_3$.

Given that each feature in each sample has varying sizes, we experimented with these three preprocessing steps:

1. **Feature Averaging**: Each feature in a sample is averaged into a single float value. This preprocessed data was used in training only the simple linear baselines: SVM with a linear kernel and Logistics Regression.
2. **Feature Splitting**: Each feature of a sample is split into N lists, each list is averaged, resulting in N averaged values per feature.
3. **Truncation**: We kept only the first N items of each feature, while discarding the rest.

The preprocessed and embedded data in 2 and 3 are then used to train our other models for further processing and classification.

3 Geometric Algebra Transformer (GATr) Architecture

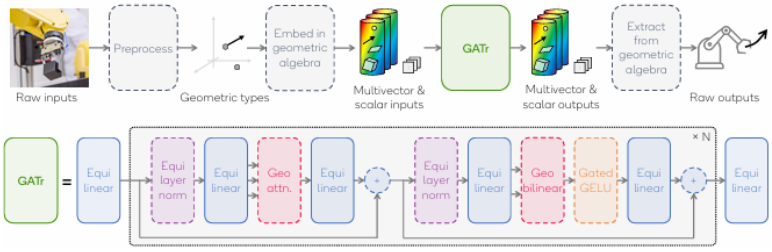

Figure 1: Overview over the GATr architecture. Boxes with solid lines are learnable components, those with dashed lines are fixed.

The components of the **GATr** Architecture:

Equivariant Layer Norm: This component performs layer normalization on the multivector inputs, ensuring equivariance by normalizing each multivector separately using the invariant inner product of $\mathbb{G}_{3,0,1}$:

$$\text{LayerNorm}(x) = \frac{x}{\sqrt{\mathbb{E}_c\langle x, x\rangle}} \tag{1}$$

Gated GELU[7]: This component applies the gated GELU activation function to the scalar component of the multivector outputs to control the activation of the entire multivector:

$$\text{GatedGELU}(x) = \text{GELU}(x_1)x \tag{2}$$

where x_1 is the scalar component of the multivector x.

We can use the same **gated nonlinearity** properties for other activations, such as **ReLU**, **Sigmoid**, and **Tanh**. For exmple, for **ReLU**:

$$\text{GatedReLU}(x) = \text{ReLU}(x_1)x \tag{3}$$

Equivariant Linear Transformation: This component performs an equivariant linear transformation on multivector inputs

$$\phi(x) = \sum_{k=0}^{d} w_k\langle x\rangle_k + \sum_{k=0}^{d+1} v_k e_0\langle x\rangle_k \tag{4}$$

where x is the input multivector, $\langle x\rangle_k$ is the blade projection of x to grade k, w_k and v_k are learnable parameters, and e_0 is the homogeneous basis vector.

Additionally, if auxilary scalars are provided, such as **pressure** in our case, then performs another linear transformation on it, not necessarily equivariant.

Geometric Bilinear Transformation: This component performs geometric bilinear operations, that is two operations: **geometric product** and the **join**:

$$\text{Geometric}(x, y; z) = \text{Concatenate}_{\text{channels}}(xy, \text{EquiJoin}(x, y; z)) \tag{5}$$

where $\text{EquiJoin}(x, y; z) = z_{0123}(x^* \wedge y^*)^*$, and z_{0123} is the **pseudoscalar** component of a reference multivector z.

Geometric Attenetion[8]: This component represents the geometric attention mechanism, which is a crucial part of the GATr architecture and is responsible for capturing geometric relationships between multivectors. In the implementation, though, either **Geometric Attenetion** or **Dot Product Attention** can be used. However, it must be noted that they must respect the **equivariace properties of PGA**, such as **symmetries (rotation, translation, reflection)**.

$$\textbf{Attention(q,k,v)} = \sum_i \textbf{Softmax}_i \frac{\sum_c \langle q_i, k_i\rangle}{\sqrt{8n_c}} v_i \tag{6}$$

where the expectation is over the channel dimension

4 Results

The performance of various models on the dataset is summarized below. For each model, we report the **F1 score**, the **memory footprint**, and the **number of parameters**.

Table 1: Performance of SVM and Logistic Regression models

Model	Non-normalized data F1 score	Normalized data F1 score
SVM	100%	98.625%
Logistic Regression	100%	98%

Table 2: Performance of Deep Learning models

Model	F1 score	Memory Footprint	Parameters
GATr	100%	0.1 MB	25.1K
EquiLinear	98.85%	0.008 MB	2.1K
BiEquiLinear	96.47%	0.074 MB	18.4K
EquiLSTM	100%	0.009 MB	2.3K
BiEquiLSTM	$\approx 100\%$	0.014 MB	3.4K

4.1 Conclusion

In this study, we evaluated the performance of several **Geometric Deep Learning** architectures based on PGA for classifying 3D vascular geometries. The SVM with linear kernel and Logistic Regression models, serving as linear baselines, achieved perfect accuracy on non-normalized data, with a slight decrease when the data was normalized.

The **GATr** architecture also attained optimal performance, demonstrating the efficacy of leveraging PGA representations. To further analyze the critical components of this architecture, we conducted ablation studies using only the EquiLinear and BiEquiLinear layers. Despite their simplicity, these models achieved high F1 scores of 98.85% and 96.47%, respectively, highlighting the importance of equivariant operations within the PGA framework.

Furthermore, we introduced the novel **EquiLSTM** and **BiEquiLSTM** architectures, which integrate PGA representations with LSTM networks achieved optimal F1 scores of 100% and $\approx 100\%$ respectively, showcasing the efficiency and effectiveness of our proposed approach.

Our findings demonstrate the effectiveness of incorporating geometric algebra representations and equivariant operations into deep learning models for geometric processing tasks. The EquiLinear and BiEquiLinear models, despite being simpler, show that even basic equivariant transformations can achieve near-optimal results. The effectiveness of the EquiLSTM and BiEquiLSTM models, both in terms of memory footprint and runtime in O(N), highlights the promise of combining geometric algebra with LSTM.

References

[1] C. G. Gunn. Projective geometric algebra: A new framework for doing euclidean geometry. *ArXiv. /abs/1901.05873*, 2019.

[2] M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, 1998.

[3] J. Brehmer, P. De Haan, S. Behrends, and T. Cohen. Geometric algebra transformer. *ArXiv. /abs/2305.18415*, 2023.

[4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[5] E. Chisolm. Geometric algebra. *ArXiv. /abs/1205.5935*, 2012.

[6] Leo Dorst. *A Guided Tour to the Plane-Based Geometric Algebra PGA*. University of Amsterdam, 2020.

[7] D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus). *ArXiv. /abs/1606.08415*, 2016.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *ArXiv. /abs/1706.03762*, 2017.