Geometric Algebra Transformer

Deep Learning

Master's Degree in Artificial Intelligence and Robotics

Kimia Hafezi **Atra Hossein Tafreshi** **Matteo Silvestri**

(1908578,1908579,1872908)

Academic Year 2023/2024

SAPIENZA
UNIVERSITÀ DI ROMA

In recent years, transformer models have revolutionized the field of deep learning, demonstrating unparalleled performance across a wide range of tasks. However, the representation of data in these models often lacks the geometric and algebraic structure inherent to many real-world phenomena, potentially limiting the depth of understanding and inference capabilities of the models.

This research is based on the Geometric Algebra Transformer (GATr) architecture, an approach that integrates geometric algebra into the transformer framework to leverage the rich mathematical framework for representing complex geometric structures and operations within deep learning models.

Reference: GATr article

Table of Contents

Dataset

The type of data we worked with is a three-dimensional representation of two classes of arteries: single arteries with stenosis and bifurcating arteries. Below we illustrate two representatives of the two classes using the ParaView software.

Figure: Bifurcating artery

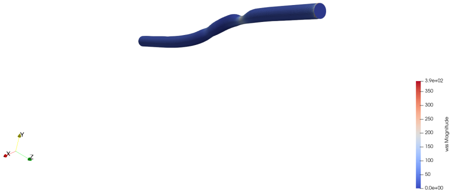


Figure: Single artery with stenosis

Dataset

The dataset consists of 4000 samples, 2000 for each class. Each sample consists of the following variables:

- *pos*: positions of the points that make up the artery
- *wss*: wall shear stress vector for each point
- *pressure*: pressure value for each point
- *face*: triples of numbers corresponding to the points forming a triangle in the mesh
- *inlet idcs*: location(in terms of points) of inlets to other arteries

The problem we faced is a standard *binary classification*.

Moreover, it is only fair to point out that in this case the task is very simple. This is because the two classes are very distinct from each other quantitatively.
For example, if we only consider mean arterial pressure, a simple linear classifier would perform well, as can be seen in the following figure.

Task

1 Dataset and task

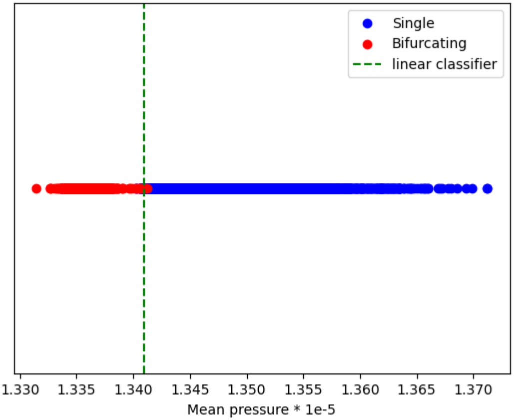


Figure: Scatter plot of the rescaled mean pressure

Q: How do we represent geometric objects such as planes, lines, etc. of $\mathbb{R}^3$ space?

With the Clifford Algebra $\mathbb{G}_{3,0,1}$ also known as 3-D Projective Geometric Algebra.

Reference: PGA article

$\mathbb{G}_{3,0,1}$ is a 16-dimensional space where the basis elements are

- $\{1\}$ Scalar - grade 0
- $\{e_0, e_1, e_2, e_3\}$ Vectors - grade 1
- $\{e_{01}, e_{02}, e_{03}, e_{12}, e_{13}, e_{23}\}$ Bivectors - grade 2
- $\{e_{012}, e_{013}, e_{023}, e_{123}\}$ Trivectors - grade 3
- $\{e_{0123}\}$ Pseudoscalar - grade 4

A standard element of this algebra is called a *multivector*. Notice that vectors are representing planes, bivectors lines and trivectors points.

Geometric product

The bivector e_{01}, for instance, is the combination of the vectors e_0, e_1 through the so-called *geometric product*. This corresponds to the main operation between multivectors and is composed of a symmetrical part, the *inner product* $\langle \cdot, \cdot \rangle$, and an antisymmetrical part, the *outer product* $\wedge$. The former calculates the similarity, while the latter generates the subspace generated by the two multivectors. With an abuse of notation, we can represent the three operations as follows

$$xy = \langle x, y \rangle + x \wedge y \tag{1}$$

In a very crude way, one can think of the inner product as a dot product and the outer product as a cross product. Indeed, we have $\langle e_i, e_j \rangle = \delta_{ij}$ and $e_i \wedge e_i = 0$ as we are normally used to.

Geometric product

Main results for this operation:

- $vv = v^2 = \langle v, v \rangle$
- $e_i e_j = -e_j e_i$
- $e_0^2 = 0$ and $e_1^2 = e_2^2 = e_3^2 = 1$

With these results and a lot of goodwill, you can calculate the following table.

Geometric product

2 Projective Geometric Algebra

1	e_{0}	e_{1}	e_{2}	e_{3}	e_{01}	e_{02}	e_{03}	e_{12}	e_{31}	e_{23}	e_{021}	e_{013}	e_{032}	e_{123}	e_{0123}
e_{0}	0	e_{01}	e_{02}	e_{03}	0	0	0	$-e_{021}$	$-e_{013}$	$-e_{032}$	0	0	0	e_{0123}	0
e_{1}	$-e_{01}$	1	e_{12}	$-e_{31}$	$-e_{0}$	e_{021}	$-e_{013}$	e_{2}	$-e_{3}$	e_{123}	e_{02}	$-e_{03}$	e_{0123}	e_{23}	e_{032}
e_{2}	$-e_{02}$	$-e_{12}$	1	e_{23}	$-e_{021}$	$-e_{0}$	e_{032}	$-e_{1}$	e_{123}	e_{3}	$-e_{01}$	e_{0123}	e_{03}	e_{31}	e_{013}
e_{3}	$-e_{03}$	e_{31}	$-e_{23}$	1	e_{013}	$-e_{032}$	$-e_{0}$	e_{123}	e_{1}	$-e_{2}$	e_{0123}	e_{01}	$-e_{02}$	e_{12}	e_{021}
e_{01}	0	e_{0}	$-e_{021}$	e_{013}	0	0	0	e_{02}	$-e_{03}$	e_{0123}	0	0	0	$-e_{032}$	0
e_{02}	0	e_{021}	e_{0}	$-e_{032}$	0	0	0	$-e_{01}$	e_{0123}	e_{03}	0	0	0	$-e_{013}$	0
e_{03}	0	$-e_{013}$	e_{032}	e_{0}	0	0	0	e_{0123}	e_{01}	$-e_{02}$	0	0	0	$-e_{021}$	0
e_{12}	$-e_{021}$	$-e_{2}$	e_{1}	e_{123}	$-e_{02}$	e_{01}	e_{0123}	-1	e_{23}	$-e_{31}$	e_{0}	e_{032}	$-e_{013}$	$-e_{3}$	$-e_{03}$
e_{31}	$-e_{013}$	e_{3}	e_{123}	$-e_{1}$	e_{03}	e_{0123}	$-e_{01}$	$-e_{23}$	-1	e_{12}	$-e_{032}$	e_{0}	e_{021}	$-e_{2}$	$-e_{02}$
e_{23}	$-e_{032}$	e_{123}	$-e_{3}$	e_{2}	e_{0123}	$-e_{03}$	e_{02}	e_{31}	$-e_{12}$	-1	e_{013}	$-e_{021}$	e_{0}	$-e_{1}$	$-e_{01}$
e_{021}	0	e_{02}	$-e_{01}$	$-e_{0123}$	0	0	0	e_{0}	e_{032}	$-e_{013}$	0	0	0	e_{03}	0
e_{013}	0	$-e_{03}$	$-e_{0123}$	e_{01}	0	0	0	$-e_{032}$	e_{0}	e_{021}	0	0	0	e_{02}	0
e_{032}	0	$-e_{0123}$	e_{03}	$-e_{02}$	0	0	0	e_{013}	$-e_{021}$	e_{0}	0	0	0	e_{01}	0
e_{123}	$-e_{0123}$	e_{23}	e_{31}	e_{12}	e_{032}	e_{013}	e_{021}	$-e_{3}$	$-e_{2}$	$-e_{1}$	$-e_{03}$	$-e_{02}$	$-e_{01}$	-1	e_{0}
e_{0123}	0	$-e_{032}$	$-e_{013}$	$-e_{021}$	0	0	0	$-e_{03}$	$-e_{02}$	$-e_{01}$	0	0	0	$-e_{0}$	0

Figure: GP table

There are other two important operations within $\mathbb{G}_{3,0,1}$ algebra :

- *dual* operator $*$
- *join* operator $\vee$

The former exchanges ”empty” dimensions for ”full” dimensions. For instance, $*e_0 = e_{123}$ or $*e_{01} = e_{23}$. The latter is given by

$$x \vee y = (x^* \wedge y^*)^* \tag{2}$$

Embeddings

- A *scalar* $\lambda \in \mathbb{R}$ becomes a multivector given by $[\lambda, 0, ..., 0]$
- A *point* $x = x_1e_1 + x_2e_2 + x_3e_3 \in \mathbb{R}^3$ becomes a multivector X given by

$$X = (1 + e_0x)e_{123} = e_{123} + x_1e_{023} - x_2e_{013} + x_3e_{012}$$

- A *plane* in space is determined by an equation $ax + by + cz + d = 0$, where $n = (a, b, c)$ it's the normal and d it's the distance from the origin (dn is a location on the plane). In algebra $\mathbb{G}_{3,0,1}$, this plane X becomes a multivector given by

$$X = de_0 + ae_1 + be_2 + ce_3$$

Embeddings

- A *line* X defined by a direction $v \in \mathbb{R}^3$ and a location $p \in \mathbb{R}^3$ is given by

$$\begin{aligned} X &= (v + e_0(p \wedge v))e_{123} = ve_{123} + e_0(p \wedge v)e_{123} \\ &= \Big[v_1 e_{23} - v_2 e_{13} + v_3 e_{12}\Big] + \\ &+ \Big[(p_3 v_2 - p_2 v_3)e_{01} + (p_1 v_3 - p_3 v_1)e_{02} + (p_2 v_1 - p_1 v_2)e_{03}\Big] \end{aligned}$$

We embed *pos* as points, *wss* as lines, *pressure* as scalar. About the variable *face*, given three points P,Q,R the normal will be $n = \vec{PQ} \times \vec{PR}$ while the distance from the origin will be $d = n \cdot P$. We didn't use *inlet idcs* for the task.

Equivariance

The main objective of the GATr model is to extract geometric features from the input data, such that the model is stable with respect to Euclidean transformations $\mathbf{E}(3)$. To test this property, we work with versors $u = u_1 ... u_k$ where u_i is a reflection, since any euclidean transformation is equal to a sequence of reflections. These elements form a group called $Pin(3, 0, 1)$, that cover $\mathbf{E}(3)$ within $\mathbb{G}_{3,0,1}$ algebra. If we consider only versors defined by an even number of reflections, then we have the group $Spin(3, 0, 1)$ that is related to $\mathbf{SE}(3)$. But what is the difference between the two groups of transformations?

Equivariance

The *Spin* group includes:

- translations
- linear reflections
- rotations about a line
- screw motions(i.e. rotation about a line and a translation along the same line)

While the *Pin* group includes also:

- planar reflections
- rotoreflections
- point reflections

Equivariance

We will say that a function $f : \mathbb{G}_{3,0,1} \to \mathbb{G}_{3,0,1}$ is *Pin-equivariant* if $\forall x \in \mathbb{G}_{3,0,1}$ and $\forall u \in Pin(3,0,1)$ it holds that

$$f(\rho_u(x)) = \rho_u(f(x)) \tag{3}$$

where ρ_u is the so-called *sandwich product*. This operation is given by

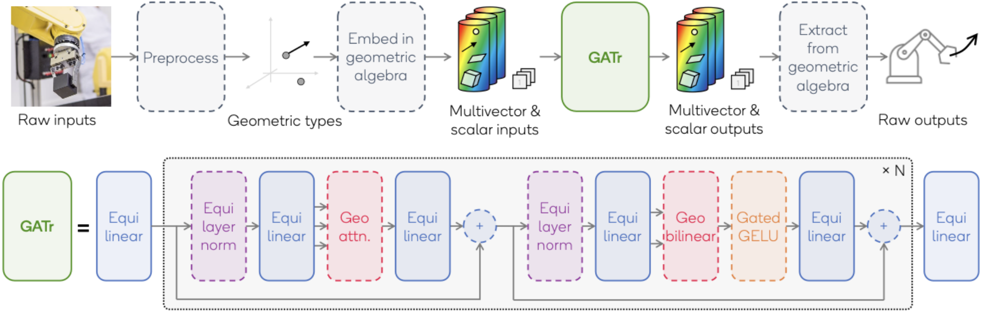
$$\rho_u(x) = \begin{cases} uxu^{-1} & \text{if } u \text{ is even} \\ u\hat{x}u^{-1} & \text{if } u \text{ is odd} \end{cases}$$

where $\hat{x}$ it's the *grade involution* of multivector x, meaning that it has flipped odd-grade components.

Table of Contents

Architecture

3 GATr



Linear layer

We want to construct a linear layer that is Pin-equivariant according to the previous definition. There are two types of equivariant function: one is the so-called *blade* $f(x) = \langle x \rangle_k$, that takes a multivector as input and put all its components of grade $\neq k$ to 0; the other one is $f(x) = e_0 \langle x \rangle_k$. In more detail, it is proved that every linear function f of the algebra $\mathbb{G}_{3,0,1}$ is of the form

$$f(x) = \sum_{k=0}^{4} w_k \langle x \rangle_k + \sum_{k=0}^{3} v_k e_0 \langle x \rangle_k \tag{4}$$

for parameters $w \in \mathbb{R}^5, v \in \mathbb{R}^4$.

Norm layer

We define a Pin-equivariant norm layer on multivectors given by the operation

$$\text{layerNorm}(x) = \frac{x}{\sqrt{\mathbb{E}_c \langle x, x \rangle}} \tag{5}$$

where $\mathbb{E}_c$ is the expectation over channels and we use the inner product of $\mathbb{G}_{3,0,1}$ algebra.

This implies $\mathbb{E}_c||\text{inputs}||^2 = 1$.

Attention layer

Regarding the attention mechanism we built a multi query approach as suggested by the paper since we have high dimensional data and we want to reduce computational costs. Just as in the original transformer, we compute scalar attention weights with a scaled dot product. The difference is that we use the inner product of $\mathbb{G}_{3,0,1}$ algebra, which is the standard dot product ignoring the dimensions containing e_0.

Geometric Bilinear layer

Because of the very limited grade mixing, equivariant linear maps are not sufficient to build expressive networks capable of build geometric features from existing ones. For this reason we introduce this layer given by

$$\text{Geometric}(x, y; z) = \text{Concatenate}_c(xy, \text{EquiJoin}(x, y; z)) \tag{6}$$

where $\text{EquiJoin}(x, y; z) = z_{0123}(x \vee y)$ and z it the average of all the input to the network.

Gated GELU layer

We use scalar-gated GELU nonlinearities given by

$$\mathrm{GatedGELU}(x) = \mathrm{GELU}(x_1)x$$

where x_1 is the scalar component of the multivector x.

Equivariance check

After implementing the model in Pytorch, we verified the numerical distance of the two members of the equation 3. Thus we generated a random versor u and check the difference for each layer. We have also considered separately the case where u is associated with a $Spin$ transformation.

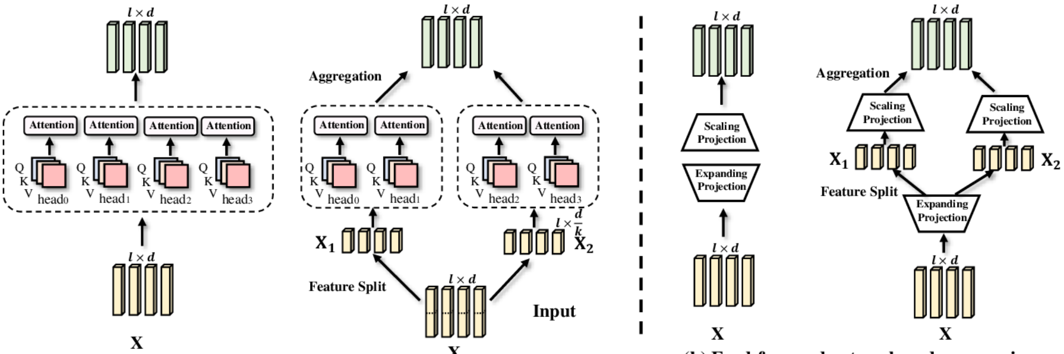
Layer	pin transformation	spin transformation
Linear layer	0.0000489	0.0000441
Norm layer	0.0000472	0.0000247
Geometric MLP	384.87	0.0000872
Attention layer	0.0000567	0.0000424

For implementing the attention mechanism for GATr architecture, we follow three mechanism: first is *multi-head* attention that is proposed in the reference paper, second one is *multi-query* attention, and third is *light-weight* attention. Multi-head attention provides the model with the flexibility to focus on different parts of the input sequence and leads to better model performance, but the model with multi-head attention would have more parameters and computational cost. For this reason authors test the GATr also with multi-query attention. Whenever multi-query attention is enough for our purpose based on the complexity of the task, it would result in lower number of parameters. The third approach that we use and is group-wise multi-head attention. The proposed approach is consistent with MHA, which can ensure LW-Transformer to learn similar attention patterns as original Transformer, while reducing parameters and computational costs a lot.

Reference: Light Weight Transformer

Light Weight Transformer

4 Light Weight GATr



(a) Multi-head attention and group-wise multi-head attention.

(b) Feed-forward network and group-wise feed-forward network.

Those features of different groups are transformed and the multi head attention will be compute within each group, and at the last the results will be aggregated. Below is its formula, where each X is feature of i-th group and $[\cdot]$ represent the concatenation:

$$G\text{-}MHA(X) = \left[\tau(X_1), \ldots, \tau(X_k)\right], \tag{7}$$

where $\forall i \in \{1 \ldots k\}$ we have

$$\tau(X_i) = MHA(X_i) \tag{8}$$

Table of Contents

5 Evaluations

Comparisons between models

We analysed the results of the previously discussed models, adding the *baseline Transformer*. In the last model, there isn't geometric representation of the data, but a flattened vector is given as input to the model in which all the concatenated sample variables are present.

We show the test accuracy, and F1 score for our task. F1 score is useful when we need a single metric to reflect both precision and recall: a high F1 score indicates that the model has a low rate of false positives and false negatives, suggesting it is robust and performs well across the aspects of precision and recall.

Comparisons between models

Another aspect that is important for evaluate and report, is the model parameters based on using different approaches (Multi-Head GATr and Light-Weight GATr). As it is shown below, there is a significant reduce in the number of parameters when using the light weight version of the GATr, while both give us the same accuracy.

Model	Accuracy	F1-score	Num Parameters
Transformer	0.99	0.95	4 K
MH GATr	1	1	258 K
LW GATr	1	1	41 K

Evaluations on rotated dataset

Finally, we tested the model by rotating the input data (such as *pos* and *wss*) to check equivariance. To do this, we used a 3D rotation matrix .

Reference: Rotation matrix

Below we report the results applying a $Spin$ trasformation composed of a line reflection and rotation around an axis.

Evaluations on rotated dataset

Model	Accuracy	F1-score
Transformer	0.98	0.95
LW GATr	0.99	1

This shows how the model is stable with respect to this transformation. This is not the case for any type of rotation. In fact, by making a rotation of 60 with respect to all axes (Pin transformation), the GATr model is affected by this transformation and the accuracy drops around 0.75. The problem is related to the GeoMLP layer and in particular to the EquiJoin operation, which from our study does not appear to be perfectly Pin-equivariant.

Geometric Algebra Transformer *Thank you*

for listening!

Any questions?