Geometry, Graph and GPT

Driving Geometric Transformation through Prompts

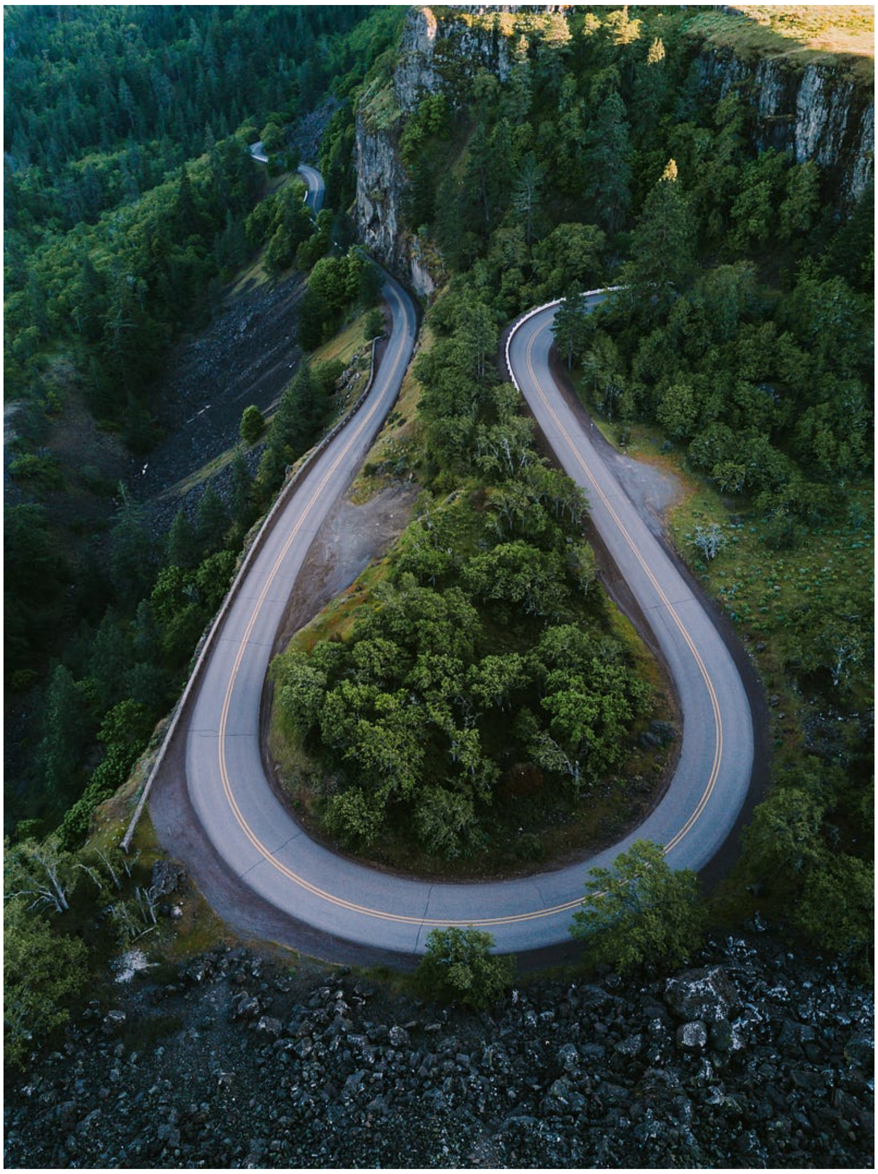
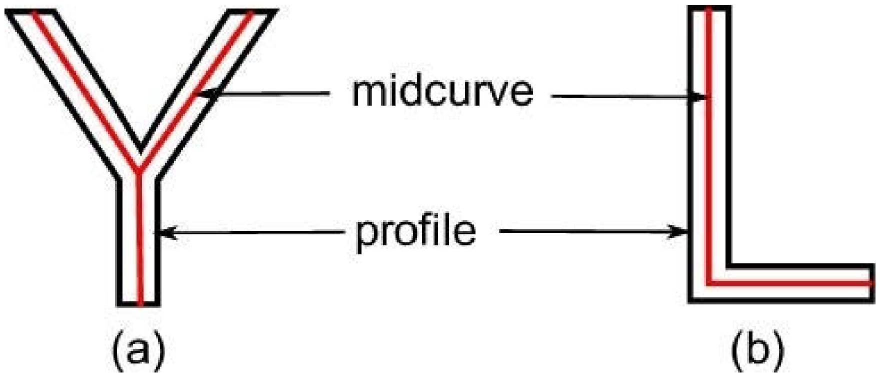


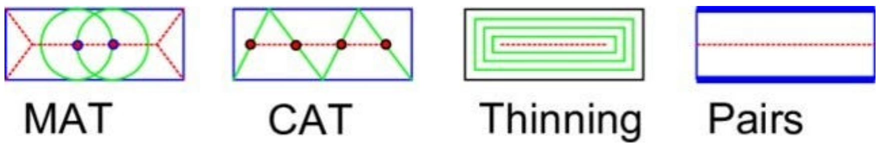
Photo by David Talley on Unsplash

A midcurve of a 2D geometric profile is a curve that lies equidistant from the bounding curves of the profile. It is a curve that represents the “middle” of the profile. The midcurve can be used to represent the shape of the profile in a simpler way than the original profile, while still retaining important geometric information about the profile.



Examples of Midcurves of 2D Geometric profiles (ref)

Although many approaches like Medial Axis Transform, Chordal Axis Transform, Thinning, Pairing, etc., have been tried for decades, the problem remains unsolved due to complexity of shapes and variety of connections.



Midcurve Approaches (ref)

Research project MidcurveNN attempts to evaluate if Neural Networks can be used to generate the midcurve of a 2D geometric profile.

Problem Statement

> **Goal**: Given a 2D closed shape (closed polygon) find its midcurve (polyline, closed or open)
>
> **Input**: set of points or set of connected lines, non-intersecting, simple, convex, closed polygon
>
> **Output**: another set of points or set of connected lines, open/branched polygons possible

Essentially, if we consider vertices as nodes and lines as arcs then the polygon/polyline profile is nothing but a Graph and thus the midcurve generation is a Graph Summarization/Dimension-Reduction/Compression issue, i.e. reducing a large graph to a smaller graph preserving its underlying structure, similar to text summarization, which attempts to keep the essence.

Geometry Representation

Summarization/Dimension-Reduction/Compression can be viewed as encoder-decoder problem where larger input is fed to the encoder side whereas reduced output is generated by the decoder side.

To be able to use Neural Network based encoder decoders, both, input and output needs to be in a fixed size vector form. So, can variable shape polygons/polylines be fed to the neural networks? Issues are:

> Shapes can not be modeled as sequences. Although polygon shape L may appear as sequence of points, it is not.

All shapes can not be drawn without lifting a pencil, which is possible for sequences. Say, Shapes like Y or concentric O, cannot be modeled as sequences. So, Midcurve transformation cannot be modeled as Sequence 2 Sequence network.

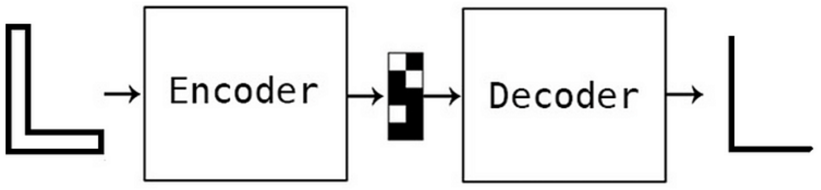
How to represent a geometric figure to feed to any Machine/Deep Learning problem, as they need numeric data in vector form?

How to convert geometric shapes (even after restricting the domain to 2D linear profile shapes) to vectors?

Closest data structure is graph, but that's predominantly topological and not geometrical. Meaning, graphs represent only connectivity and not spatial positions. Here, nodes would have coordinates and arcs would have curved-linear shape. So, even Graph Neural Networks, which convolute neighbors around a node and pool the output to generate vectors, are not suitable.

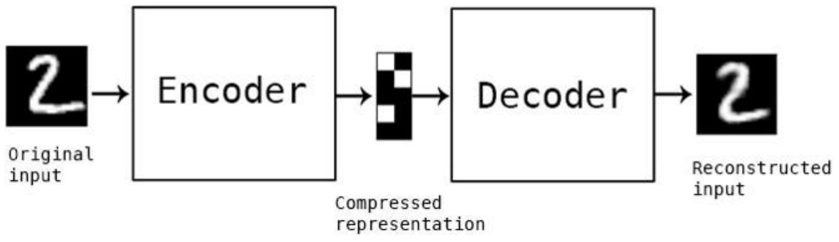
Need RnD to come up with a way to generate geometric-graph embedding that will convolute at nodes (having coordinates) around arcs (having geometry, say, a set of point coordinates), then pool them to form a meaningful representation. Real crux would be how to formulate pooling to aggregate all incident curves info plus node coordinates info into a single number!!

So, though graphs are not perfect but with insertion of geometric information they can represent the 2D profiles well. Can Encoder-Decoder architectures (Transformers with Attention) be applied to generate the midcurve?



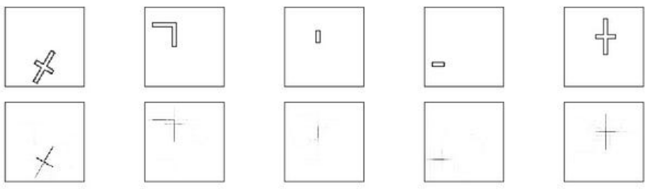
Midcurve by Encoder Decoder (ref)

Such Transformers taking variable length input and output graphs have not been established. So, would taking this problem to image domain help?



Auto Encoder on images (ref)

Various Image2Image networks have been tried by MidcurveNN project but with limited success. Here is a sample:

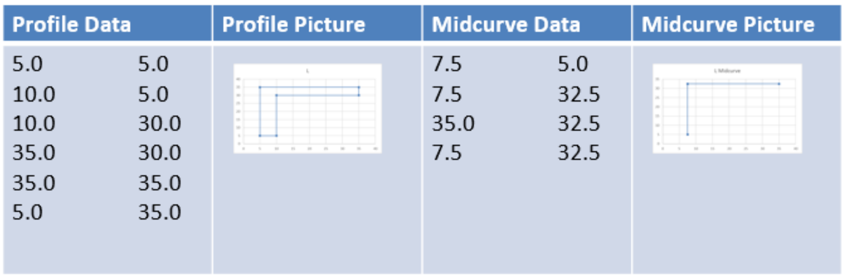

Simple Encoder Decoder network in Tensorflow/Keras (ref)

Graph, Geometry and GPT

Now evaluating if LLMs (Large Language Models), like can be employed to generate the midcurve. Idea is to give prompt telling what needs to be done along with some examples, and see if LLMs can generate shape for the test example.

```
You are a geometric transformation program that transforms input 2D polygonal pr
Input 2D polygonal profile is defined by set of connected lines with the format
input : [line_1, line_2, line_3,....] where lines are defined by two points, whe
line_1 is defined as ((x_1, y_1), (x_2,y_2)) and similarly the other lines.
Output is also defined similar to the input as a set of connected lines where li
output : [line_1, line_2, line_3,....]

Below are some example transformations, specified as pairs of 'input' and the co
Do not write code or explain the logic but just give the list of lines with poin

input:[((5.0,5.0), (10.0,5.0)), ((10.0,5.0), (10.0,30.0)), ((10.0,30.0), (35.0,3
output: [((7.5,5.0), (7.5, 32.5)), ((7.5, 32.5), (35.0, 32.5)), ((35.0, 32.5) (7

input: [((5,5), (10, 5)), ((10, 5), (10, 20)), ((10, 20), (5, 20)), ((5, 20),(5,
output: [((7.5, 5), (7.5, 20))]

input: [((0,25.0), (10.0,25.0)), ((10.0,25.0),(10.0, 45.0)), ((10.0, 45.0),(15.0
output: [((12.5,0), (12.5, 22.5)), ((12.5, 22.5),(12.5,45.0)), ((12.5, 22.5), ((

input:[((0, 25.0), (25.0,25.0)),((25.0,25.0),(25.0,20.0)), ((25.0,20.0),(15.0, 2
output:
```

The first input example above represents ‘L’ shape (shown below) and the second is an ‘I’, whereas the 3rd is a ‘Plus’ sign shape.



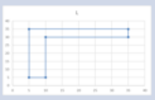
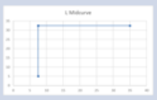
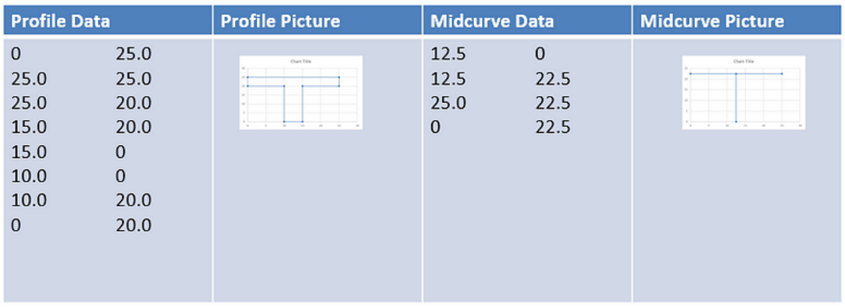
Profile Data		Profile Picture	Midcurve Data		Midcurve Picture
5.0	5.0		7.5	5.0	
10.0	5.0		7.5	32.5	
10.0	30.0		35.0	32.5	
35.0	30.0		7.5	32.5	
35.0	35.0				
5.0	35.0				

Image by Author

The last shape for which LLM has been asked for the answer is actually a 'T' shape. The picture below shows the correct/actual answer as well.



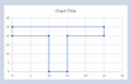
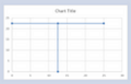
Profile Data		Profile Picture	Midcurve Data		Midcurve Picture
0	25.0		12.5	0	
25.0	25.0		12.5	22.5	
25.0	20.0		25.0	22.5	
15.0	20.0		0	22.5	
15.0	0				
10.0	0				
10.0	20.0				
0	20.0				

Image by Author

And the outputs computed by various LLMs (ChatGPT, Perplexity AI, Bard) , along with the real/actual answer:

```
Actual: [((12.5,0), (12.5,22.5)), ((12.5,22.5),(25.0,22.5)), ((12.5,22.5),(0,22
ChatGPT: [((2.5, 0), (2.5, 22.5)), ((2.5, 22.5), (2.5, 45.0)), ((2.5, 22.5), (25
Perplexity: [((12.5,0), (12.5, 22.5)), ((12.5, 22.5),(12.5,45.0)), ((12.5, 22.5)
Bard: [((12.5, 0), (12.5, 25.0)), ((12.5, 25.0), (25.0, 25.0)), ((25.0, 25.0),
```

Visually here is how results from different LLMs look:

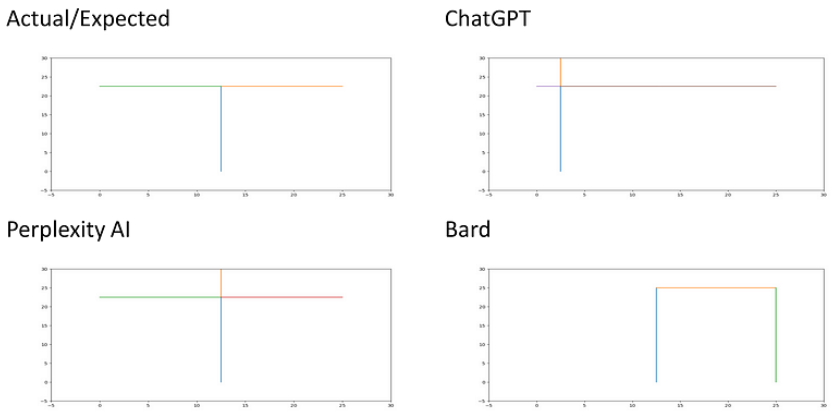


Image by Author

Obviously all the LLMs seems to have failed and there could be two prominent reasons:

Prompt design was not effective and could be improved upon.

The LLM model itself is not able to learn the pattern and predict well.

Presenting this problem here so that LLM community, if interested, can working on either/both of the above mentioned issues.

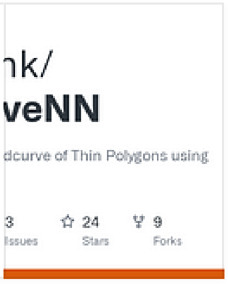
Exiting times ahead!!

References

GitHub - yogeshhk/MidcurveNN: Computation of Midcurve of Thin…

Midcurve by Neural Networks Goal: Given a 2D closed shape (closed polygon) find its…

github.com


nk/
veNN
dcurve of Thin Polygons using
3 Issues
24 Stars
9 Forks