Extending Path Development
Manifest

October 20, 2024

1 Introduction

There are already numerous supervised learning methods adapted to sequences : combining CNN, RNN / LSTM, not to mention Transformers which overcame the cost of representing arbitrary long-term dependencies.

Path Development arises in Rough-Path literature as an alternative, finite-dimensional, parameterized exponential (in a Controlled Differential Equation sense) to the signature transform.

The idea is to represent a path incrementally into a matrix Lie group. Choosing a Lie group boils down to enforcing a structure on the parameterization.

Contrasting with many ML representations, path development is mathematically well principled. In theory, it is an algebra homomorphism away from the signature transform of a sequence [3], and the signature is characteristic and universal on compact sets of paths (in the product topology). Path development is even universal and characteristic beyond the compact space [3] - providing one finds the right algebra homomorphism (relating to the parameterisation of the exponential).

It has already been succesfully applied to tackle gradient issues in LSTM.

I believe there is room for combining further path developments with celebrated machine-learning models, including :

- As a learnable feature extraction layer of different-size input sequences,
- As a standalone model, providing the output matrix can be seen as an element of a Hilbert space, enabling the use for instance of Hilbert-Schmidt Independance Criterion [4], defining distances between representations, ...
- As a layer in deep neural network, whether to contract the sequentiality of the data, or as an action on another branch of the neural network.

Indeed the study of Lie groups is originally motivated partly by the potential of group-action and theory of representation.

Learning on manifolds Restricting the state space to be included on a manifold reduces the effective dimensionality of the representations, which could reduce the risk to overfit. To bridge the gap with the manifold hypothesis, one needs to make sure the selected manifold has the right topology and is adapted to the problem.

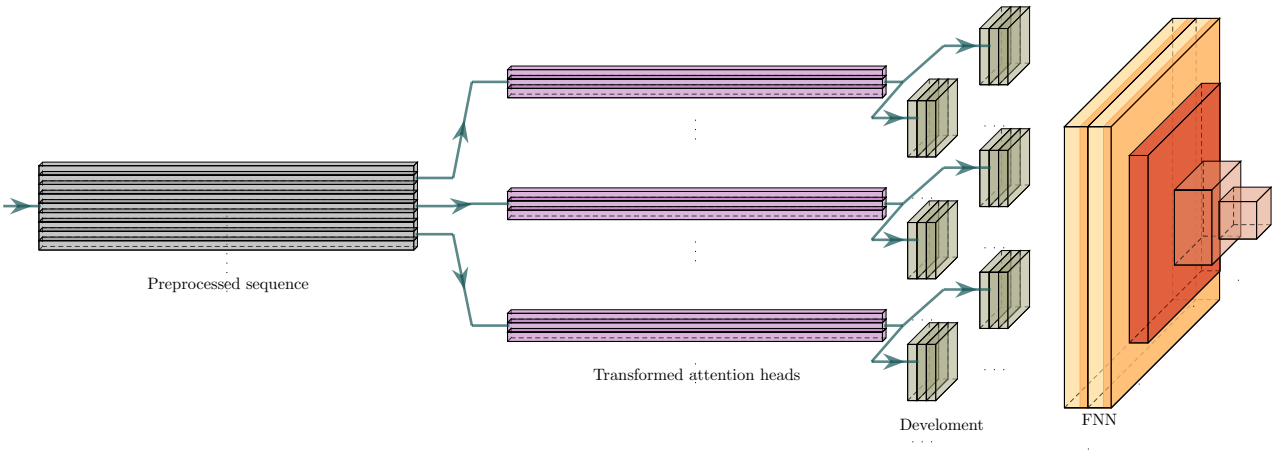
To do so, an exploratory data analysis, in spite of a good intuition, could help make the right choice ; albeit there is no such method to my knowledge.

2 Objectives

Extending available Lie groups An implementation of path development already exists from [2]. Lie groups that could be added include :

- *General Linear group* : the Lie algebra has no constraints,
- *General Orthogonal group* : a distorted version of the orthogonal group,
- *Nilpotent Lie groups* including the Heisenberg group and higher-order Carnot groups : learn a universal lift of nilpotent diffusions.

Including path development as a layer in a DL architecture



The general idea is that having a large dimensional path, computing a first cross-channel representation, then split the path into attention heads, perform multi-head self-attention. Eventually each sub-sequence can be embedded in multiple Lie groups and multiple channels with path development.

Depending on the task, the last part of the network could be adapted : the Hilbert structure of matrices can be used as is to compare elements [4], otherwise a FFN can be plugged to interpret each channel of each group and aggregate them.

Revamp Path Development library Package and ship the path development tools with latest pytorch support : group structures, layers, architecture support and statistics. Implement a JAX equivalent.

References

[1] Baudoin, Fabrice. An introduction to the geometry of stochastic flows. World Scientific, 2004.

[2] Lou, Hang, Siran Li, and Hao Ni. "Path development network with finite-dimensional Lie group representation." arXiv preprint arXiv:2204.00740 (2022).

[3] Chevyrev, Ilya, and Terry Lyons. "Characteristic functions of measures on geometric rough paths." (2016): 4049-4082.

[4] Gretton, Arthur, et al. "A kernel statistical test of independence." Advances in neural information processing systems 20 (2007).