Kartik Sachdev
Lerchenstrasse 123
80995, Munich, Germany
Email: kartik.sachdev@rwth-aachen.de
Tel: +49-176-45791369

Munich, 17 April 2022

Machine Learning for Science (ML4Sci)

Dear Mentors,

Proposal for Transformers for Dark Matter Morphology with Strong Gravitational Lensing

Thank you for allowing me to convey my proposal to you. Through this letter, I would like to express my ardent desire to contribute to the project Transformers for Dark Matter Morphology with Strong Gravitational Lensing for Google Summer of Code (GSoC) 2022. The proposal consists of my brief introduction, timeline with task breakdown and finally, a research abstract comparing the top two solution for the transformer's evaluation task.

I am currently studying M.Sc. Robotic Systems Engineering at RWTH Aachen University. Having completed my third semester, I have acquired strong fundamental knowledge in the fields of Artificial Intelligence- namely Reinforcement Learning, Deep Learning for Computer Vision, and Machine Learning. To strengthen my academic knowledge, I have undertaken projects related to Reinforcement Learning like Deep Q-Networks, Policy Gradients, and to Computer Vision like Image Classification using ResNets, transfer learning, hyperparameter tuning (Population-Based Training, ASHA) using PyTorch and Ray-tune libraries.

My recent experience has been with the BMW Group as a Robotics Software Engineering Intern in the decision-making team of the Logistic Robotics. I developed the behavior design of an early-stage logistic robot with State Machine using Python, SMACH and ROS combining computer vision, user interface, safety, etc. Our team successfully tested the state machine for simulation as well as the real robot for the first release. I have also simulated, tested various motion planners, and developed algorithms for the purpose of conceptualization, feasibility study, and digital twin using MoveIt! and Gazebo. I also handled the deployment of the robotic software suite on the real-world robotic system and of their respective simulations using Docker and Docker-Compose. Apart from technical responsibilities, I also worked with my supervisors and product owners to develop new features; to improvise layout around the robots; to collaborate with other departments consisting of Ph.D. students and interns; following agile methodologies and version control system for software development. I believe that my hands-on software development experience and awareness of standard industrial practices provide me with a strong foundation for my time at GSoC.

I am a self-motivated person and prefer to learn about the latest technologies by undertaking projects or participating in technical events. Last year, I participated in AutoML Fall School organized by the University of Freiburg. Our team stood third in the hackathon where we used and implemented Autosklearn, Autogluon and AutoPyTorch on a tabular data. In one of the hackathons, I learnt about explainable AI using SHAP and CXPlain libraries and our team ended up in the second position. I have also contributed to an open-source repository on GitHub namely Lab-ml where I worked on PyTorch implementations for Computer Vision such as ResNets, Cross Validation; hyperparameter tuning with explanations to make Deep Learning codes more interpretable. In of the online paper discussion event organized by "Machine Learning Tokyo", I explained the paper "SimCLR" on Self-Supervised Learning by Google Research and it was attended by over 20 researchers and ML enthusiasts across the globe.

Before the commencement of my graduation, I worked as a Mechanical Research and Design Engineer at TBK Co., Ltd, Tokyo, Japan where owing to my projects I was able to sharpen my work style and critical thinking ability. Moreover, having worked in India, Japan, and Germany for over three years, I have learned the professional skills of planning and cooperating with people from different cultures that would enable me to collaborate better with the mentors from different geographies.

What attracts me the most to GSoC is the structured program. Sessions like Community Bonding, evaluation, and guidance from the mentors, GSoC provides an innovative learning platform and gives an exposure to the tools and techniques followed in open-source community in short time duration. The biggest learning point would be the code review sessions and feedback from seasoned mentors as it would help me improve my code quality and learn the best practices. This steep learning curve is especially needed for gaining expertise in the fast-paced domain of software development. Moreover, I would like to gain hands-on experience with Vision Transformers, hence, this project would an ideal starting point.

Lastly, I believe that with a fundamental knowledge in Deep Learning along with the professional experience, I would be able to contribute to the best of my level at GSoC 2022. If given a chance, I am determined to make the most of it by delivering at my best level. Even after GSoC, I would continue to contribute to the open-source community with the lessons learnt during the program.

Thank you for spending your valuable time on my letter. Please find attached the proposed timeline and the research abstract.

Yours sincerely,
Kartik Sachdev

Timeline

Milestones	Dates	Tasks details
Pre GSoC period	Upto May 20	Understand the papers: "Deep Learning the Morphology of Dark Matter Substructure", "Decoding Dark Matter Substructure without Supervision" and "Domain Adaptation for Simulation-Based Dark Matter Searches Using Strong Gravitational Lensing"
		Literature review on Vision Transformers implementations for classification and regression
Community Bonding period	May 20 - June 12	Go through the Equivariant DeepLense codebase and documentation
		Discuss and finalize the implementation, tasks, and deliverables of the project with mentors
		Get feedback on the submitted solutions, areas of improvements etc.
		Finalize the timeline of the project
Coding period (first half)	June 13 - July 13	Testing the submitted solution (approach 1): E(2)CNN-CvT on a bigger dataset - more classes, more data for classification
		Testing to include more symmetries like C8, D4, and D8 to E(2)CNN-CvT architecture
		Benchmarking with other Vision Transformer implementations on critereas such as accuracy, ROC curve, AUC, training time, generalizabilty, adaptibility to new classes etc.
Evaluation period preparation	July 14 - July 25	Preparing documentation
		Adding changes to improve implementation, feedback from Mentors
Coding period (second half)	July 26 - Aug. 13	With results and lesson learnt from the first half coding period, testing the network for regression task
		Benchmarking with other Vision Transformer implementations on critereas such as MSE, training time, generalizabilty, adaptibility to new classes etc.
Evaluation period preparation	Aug. 14 - Aug. 27	Solving issues, bugs and thorough testing
		Preparing documentation for final submission
		Feedback from Mentors on the final codebase, documentation
Buffer period	Aug. 28 - Sept. 4	Accounting for last-minute changes in plan

Rotation Equivariant Convolutional Vision Transformer

Kartik Sachdev
RWTH Aachen University
Munich, Germany
kartik.sachdev@rwth-aachen.de

Abstract

To induce rotational equivariance to a cyclic subgroup C_4 in the transformer architecture, $\mathrm{E}(2)$-steerable convolutions are applied before the Convolutional Vision Transformers (CvT) for binary classification on a highly symmetrical dataset of simulated strong gravitational lensing images with and without substructure. The model was trained with only 65k parameters and achieves a test accuracy of over 97.1%. Notably, the proposed architecture has approximately one tenth of parameters compared to CvT, while achieving a comparable test accuracy.

1 Introduction

$\mathrm{E}(2)$ steerable convolutional layers [5] are strategically placed before [7] the Convolutional Vision Transformer (CvT) [6] to empirically verify a hypothesis of making Vision Transformers equivariant under rotation. The main idea is to exploit the known inherent rotational symmetries present in the dataset using equivariant CNNs and at the same time leverage the advantages of CvT architecture - shift, scale invariance, global context and ability to train on small dataset [6]. The rest of this research abstract is organized as follows: Related work is discussed in Section 2, followed by Implementation Details in Section 3, Experiments in Section 4 and lastly, Results And Future Work in Section 5.

2 Related Work

Convolutional Vision Transformers uses multi-stage hierarchical design with each stage consisting of the Convolutional Token Embedding and Convolutional Projection. The Convolutional Token Embedding layer takes the input image or 2D reshaped token maps from the previous layer to learn a function that maps them into new tokens which are then flattened and normalized by layer normalization [2] for input into the subsequent stack of Convolutional Transformer Blocks. In each block, Convolutional Projection is applied for query, key, and value embeddings which are passed to the Multi-Head Attention layer. In the last stage, fully connected head is used to predict the class [6].

$\mathrm{E}(2)$-equivariant networks are equivariant under all isometries of the image plane $\mathbb{R}^2$, that is, under translations, rotations and reflections. In other terms, if a model is equivariant to a subgroup $G \leq O(2)$ of the orthogonal group, then the output produced by the network transforms consistently when the input is transformed under the action of an element $g \in G$. For example, a cyclic subgroup C_6 models 6 rotations in the multiples of $\pi/3$. General $\mathrm{E}(2)$-equivariant Steerable CNNs framework [5] provides a set of tools and general strategy for variety of such groups. Of particular interest, $\mathrm{E}(2)$-steerable convolutions guarantee an equivariant mapping between Input and Output field type which defines a transformation law or how the signals sampled on the plane $\mathbb{R}^2$ transform under g.

Code and trained models: https://github.com/sachdevkartik/GSoC-2022.

	EqCvT		**CvT**	
	Layer Name	Details	Layer Name	Details
Stage 1	Conv. Point Avg. Pool Group Pool	$\begin{bmatrix} 7 \times 7, \text{Stride=1} \\ \text{Field Type=Regular} \\ \text{Fields=10} \end{bmatrix} \times 1$	Conv. Embed.	$7 \times 7, 64$
			Conv. Proj. MHSA MLP	$\begin{bmatrix} 3 \times 3, 64 \\ \text{H}_1\text{=2}, \text{D}_1\text{=64} \\ \text{R}_1\text{=4} \end{bmatrix} \times 2$
Stage 2	Conv. Embed.	$3 \times 3, 32, \text{Stride=2}$	Conv. Embed.	$3 \times 3, 128, \text{Stride=2}$
	Conv. Proj. MHSA MLP	$\begin{bmatrix} 3 \times 3, 32 \\ \text{H}_1\text{=3}, \text{D}_1\text{=32} \\ \text{R}_1\text{=2} \end{bmatrix} \times 2$	Conv. Proj. MHSA MLP	$\begin{bmatrix} 3 \times 3, 64 \\ \text{H}_1\text{=3}, \text{D}_1\text{=64} \\ \text{R}_1\text{=4} \end{bmatrix} \times 1$
Head	Linear	64	Linear	128
Params	65k		463k	
Accuracy	97.1%		98.1%	

Table 1: Architectures for EqCvT and smaller version of CvT [6] used for comparison

3 Implementation Details

A single convolution block consisting of modules - $\mathrm{E}(2)$-steerable convolution, batch-normalization and ReLU is added before the single stage of CvT. The input and output field types of convolution block are composed of 3 trivial and 10 regular representations with rotational action of C_4. Each module works on a geometric tensor which wraps a common tensor and augment tensor with a compatible field type [5]. Images with a size 129×129 are converted to geometric tensor before feeding to the network. The output of the convolution block is passed through 2 pooling layers namely, anti-aliased channel-wise average pooling based on [8] and group pooling similar to max pooling, which pool over the spatial dimensions and over the group respectively.

To make the initial layers compatible with the CvT layers, the geometric tensor is unwrapped by keeping the tensor and discarding the associated field type. The resulting tensor is then, fed to the single stage of CvT as discussed in the Section 2. The stage consists of Convolutional Token Embedding layer to generate tokens for the Convolutional Transformer Blocks which performs the Multi-Head Self-Attention. Finally, an MLP layer is used for binary classification.

4 Experiments

The proposed model is evaluated on the simulated strong lensing images dataset for binary classification. The dataset was generated using the package PyAutoLens [4] and consists of 5000 images each of simulated strong lensing images with and without dark matter substructure [1]. Adam [3] optimizer is used with the weight decay of 1e-7 with β_1, β_2 as 0.9 and 0.999 respectively. The models are trained with a constant learning rate of 1e-4 for 40 epochs and batch size of 8 and 64 for EqCvT and CvT respectively. Data augmentation follows a similar scheme to the original implementation of General $\mathrm{E}(2)$-Equivariant Steerable CNN for MNIST dataset with continuous rotations [5].

5 Results And Future Work

EqCvT obtains an accuracy of 97.1% which is 1% less than 2-stage CvT but with a significant reduction of 86% in parameter count. To check for the rotational equivariance, the standard deviation of the output logits were compared for an input with 8 rotations in multiples of $\pi/4$. EqCvT had a standard deviation of 0.48 while CvT had 0.96, empirically showing more robustness to rotation.

This work provides one of the basis for combining equivariant networks with transformers. One possible direction could be partially replacing the convolutional part of CvT with steerable convolutional layers to make the network more robust to known symmetries. It would be interesting to see how well the architecture performs on different and bigger datasets. As the results of this work are more empirical, proving the equivariance of this architecture theoretically could also an area to be explored.

Acknowledgments

I would like to thank Machine Learning for Science (ML4Sci) with the participating organizations University of Alabama, Brown University and BITS Pilani Hyderabad for providing the dataset.

References

[1] Stephon Alexander, Sergei Gleyzer, Evan McDonough, Michael W Toomey, and Emanuele Usai. Deep learning the morphology of dark matter substructure. *The Astrophysical Journal*, 893(1):15, 2020.

[2] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

[3] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[4] J. W. Nightingale, R. G. Hayes, Ashley Kelly, Aristeidis Amvrosiadis, Amy Etherington, Qiuhan He, Nan Li, XiaoYue Cao, Jonathan Frawley, Shaun Cole, Andrea Enia, Carlos S. Frenk, David R. Harvey, Ran Li, Richard J. Massey, Mattia Negrello, and Andrew Robertson. 'pyautolens': Open-source strong gravitational lensing. *J. Open Source Softw.*, 6(58):2825, 2021.

[5] Maurice Weiler and Gabriele Cesa. General E(2)-Equivariant Steerable CNNs. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[6] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021.

[7] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollar, and Ross Girshick. Early convolutions help transformers see better. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 30392–30400. Curran Associates, Inc., 2021.

[8] Richard Zhang. Making convolutional networks shift-invariant again. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7324–7334. PMLR, 09–15 Jun 2019.