Google Developer Groups

On Campus • Indian Institute of Information Technology Kalyani

Winter of Code 5.0

Proposal

for the Project

“High Performance Augmentation Benchmarking”

under

KORNIA

About Me

Name - Suryam Singh
Email-Id - suryam1802singh@gmail.com
GitHub Username - kaemi-kun
Country - India
Timezone - IST (Indian Standard Time) — UTC +05:30
Primary Language - English
LinkedIn-Profile-Link- - https://www.linkedin.com/in/suryam-singh-626061346?utm_source=share&utm_campaign=share_via&utm_content=profile&utm_medium=android_app
Other Links (If Applicable) - github.com/kaemi-kun

Synopsis

Kornia offers a rich collection of computer vision operators that integrate tightly with PyTorch, making it a powerful choice for research and applied machine learning. However, when these operators are used in real production workflows, users often face difficulties related to torch.compile compatibility and ONNX export reliability. Even small incompatibilities can prevent models from being compiled, optimized, or deployed smoothly, creating a gap between experimentation and real-world usage.

This project focuses on improving the practical usability of Kornia by identifying and reducing these compatibility gaps. The work will involve auditing selected modules, refactoring non-traceable code patterns into more symbolic and traceable forms, and building verification workflows to ensure that ONNX-exported outputs remain consistent with PyTorch results. Rather than redesigning Kornia's core algorithms, the goal is to make existing operators more reliable, predictable, and easier to deploy across different environments.

By improving compatibility, testing, and documentation, this project helps

Kornia become a more production-ready library, enabling users to trust its operators not only during training, but also during optimization and deployment. This directly supports Kornia's mission of providing high-quality, differentiable computer vision tools for modern machine learning pipelines.

Benefits to the Community

Kornia provides powerful computer vision operators for PyTorch, but using them smoothly in production workflows can still be difficult due to limitations in torch.compile compatibility and ONNX export reliability. These issues often create a gap between experimentation and deployment. This project focuses on improving the practical usability of Kornia by identifying compatibility gaps, refactoring non-traceable code patterns, and validating exported models through systematic testing. The goal is not to redesign Kornia's algorithms, but to make existing operators more reliable, predictable, and easier to deploy. By improving compatibility, testing, and documentation, this project helps Kornia users move more confidently from research to real-world applications.

Project Plan

The project will be carried out in small, incremental phases to ensure stability, correctness, and continuous feedback from the maintainers. I will begin by exploring Kornia's codebase and selecting a limited set of operators that show compatibility issues with torch.compile or ONNX export. This initial phase will focus on understanding why certain patterns fail during tracing or export.

In the next phase, I will improve parts of the code that currently cause problems during tracing or export by simplifying control flow and making the logic more consistent with PyTorch's tracing requirements. These changes will be applied carefully and in small steps so that the original behavior of the operators remains unchanged. Each improvement will be tested to make sure it does not introduce unexpected side effects.

To check correctness, I will build simple comparison tests that verify whether ONNX-exported outputs closely match the original PyTorch results. This helps ensure that compatibility improvements do not reduce numerical accuracy. Along with these updates, I will document common limitations, working patterns, and practical tips for using Kornia operators with torch.compile and ONNX export, so that users can avoid common pitfalls.

Throughout the project, I will regularly incorporate mentor feedback, refine implementations, and improve documentation so that the final outcome is both technically correct and easy for users to adopt.

Tech Stack

- **Python**
- **PyTorch**
- **Kornia**
- **torch.compile**
- **ONNX**
- **ONNX Runtime**
- **Pytest**

Milestones

Milestone	Tentative Date	KPI
Week 1	1/02/26 - 7/02/26	**Exploration**:Set up Kornia locally, run torch.compile and ONNX export on selected operators, document failures and warnings
Week 2	8/02/26 - 14/02/26	**Compatibility Analysis**:Identify specific code patterns causing tracing/export issues in selected modules
Week 3	15/02/26 - 21/02/26	Apply small, safe refactors to improve traceability while preserving original behavior
Week 4	22/02/26 - 28/02/26	**Output Verification:**Implement PyTorch vs ONNX output comparison tests with tolerance checks
Week 5	1/03/26 - 7/03/26	**Regression Testing**:Add automated regression tests to prevent future compatibility breaks

Week 6	8/03/26 - 14/03/26	**Documentation**:Document limitations, supported patterns, best practices, and refine code based on mentor feedback

Deliverables

By the end of the contribution period, the following deliverables will be completed:

- **A documented list of Kornia operators tested for torch.compile and ONNX export compatibility, along with observed issues and limitations.**
- **Refactored implementations for a selected set of operators to improve traceability and export reliability without changing numerical behavior.**
- **Simple automated tests that compare PyTorch and ONNX results to ensure that compatibility improvements do not change expected behavior.**
- **Additional regression tests to help prevent future compatibility problems.**
- **Practical documentation describing what works well, what does not, and how users can avoid common export and compilation issues.**
- **At least one small contribution improving error clarity, comments, or structure in the affected modules**

Acknowledgement

I sincerely thank the Kornia maintainers and the Winter of Code team for creating this opportunity to learn, contribute, and grow through open source development. I look forward to collaborating with the mentors and community and contributing meaningfully to the project.