`DeltaNN`: Differential Testing to Evaluate Robustness of Image Recognition Models

I. Appendix

A. Output Label Differences Between Native DL Framework Models

In Figure 1, we present the percentage of discrepancies of our models under test, built with the respective DL framework, in comparison to all other native DL frameworks. We observe up to 57% discrepancies in inference output on `MobileNetV2`, up to 40% on `ResNet101` and up to 33% on `InceptionV3`.

B. Fault Localization Analysis

As described in our main contribution, we performed a fault localization analysis for DL framework conversions in `InceptionV3`, utilizing `TF` as source DL framework, and `TFLite` as target DL framework. Figure 2 demonstrates the maximum and the standard deviation discrepancies across layer activations and respective parameters, for convolution and bias addition layers.

C. Execution Times across DNN Frameworks

As about our observations in Figures 6 - 14:

- *InceptionV3* appears the most robust across all devices, for all the DNN frameworks (except *Keras* which did not run), posing a maximum (small, but non-negligible) relative change of 8% (*TF/TFLite* to *Keras* configuration, `Hikey` device).
- *MobileNetV2* appears to be the less robust configuration, with a wide range of deviations across devices. apart from the *PyTorch* configuration, `Xavier` failed to execute, therefore we do not present any comparisons for that device.
- Considerable deviations were observed in the devices that run the experiments, especially between *Keras* and *PyTorch* (maximum relative change of 16-18% on (`Server`) across optimizations).
- *ResNet101V2* was proven robust for the majority of its DNN Framework configurations, however large discrepancies were observed for its PyTorch configuration. We consider that the *V1* DNN Framework backend used can pose significant differences between the other configurations - going out of range in Figures 12- 14 in the last column. Overall, the relative change goes up to 100% for *PyTorch* in comparison to other DNN Framework configurations.

D. Execution Time Measurements of Deep Learning Frameworks Per-Device

Figures 16-15 show the execution times under different optimization setting for each one of the DL Frameworks under test. By observing for instance Figures 16a-16c, we can observe that the difference in execution times between `Server` and `Hikey` widens significantly. We also consider differences across our converted models, so, for instance PyTorch-to-TFLite refers to the respective TFLite model converted from PyTorch.

E. Execution Time Measurements of Deep Learning Frameworks Per-Optimization

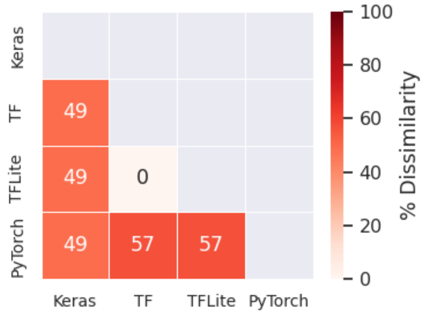
As described in the main contribution, we conducted a small-size experiment, utilizing part of our dataset (100 images), generating model configurations for each optimization setting, for ResNet101 using TensorFlow and for InceptionV3 using PyTorch, both on `Local` device. The total execution time for all images, along with the percentage of effect in comparison with the `Basic` optimization, is depicted on Tables III and IV. Note that a positive number resulted into a better performance in comparison with the `Basic` optimization, while a negative number resulted into a degraded performance.

F. Execution Time Measurements of Optimization Comparisons

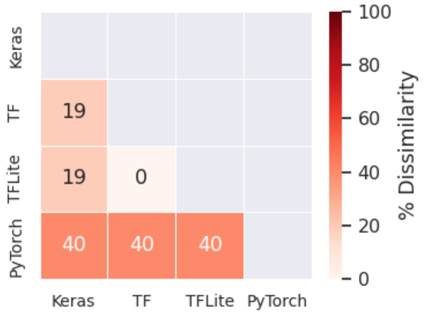
We present the complete set of data related to execution time comparisons between *Basic*, *Default* and *Extended* optimizations on Tables I-XI. Between the comparison pair *(A, B)* per-case positive number indicates that B is faster than A, while a negative number that B is slower than A. For instance, in Table I, where `A=Basic` and `B=Default` optimizations, on `Server`, the `Keras` version with *Default* optimizations was 6.5% faster than the one with *Basic* optimizations.

	Server	Xavier	Hikey	Intel
Keras	6,5	-	-2,3	30,5
TensorFlow	1,9	-	-5,8	27,7
TFLite	2,9	-	4,3	18,4
PyTorch	2,8	5,4	-10,5	16,2

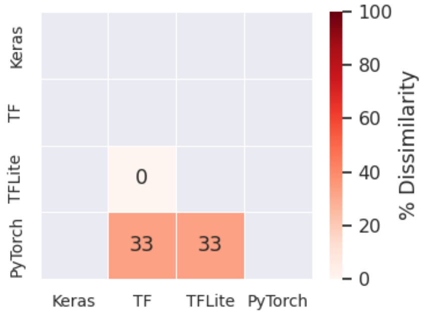
TABLE I: MobileNetV2, Basic / Default optimization execution time comparisons (%).



(a) MobileNetV2

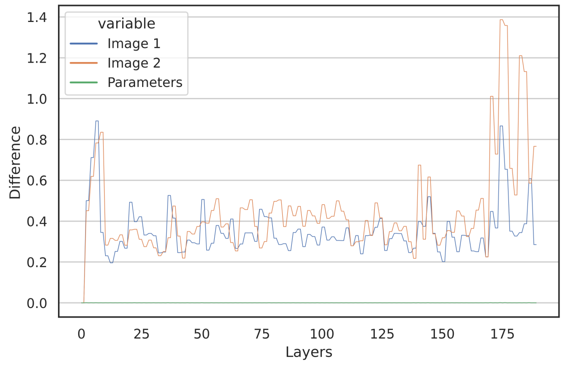


(b) ResNet101

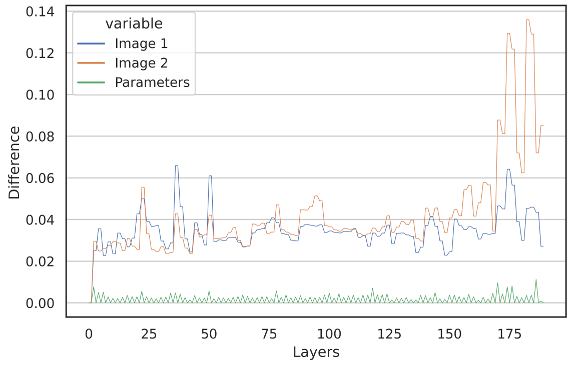


(c) InceptionV3

Fig. 1: Pairwise comparison of output label dissimilarities (%) between DL frameworks for our 3 models, running on `Server`, with *Default* optimization level.



(a) Standard deviation of differences per-layer



(b) Standard deviation of differences per-layer

Fig. 2: Maximum and standard deviation differences across model conv2D and bias add layers and their respective parameters, InceptionV3 model, with TF as source DL framework and TFLite as target.

	Server	Xavier	Hikey	Local
Keras	2,2	-	-81,4	20,1
TensorFlow	2,0	-	-76,9	18,2
TFLite	1,0	-	-79,3	10,6
PyTorch	0,9	-0,4	-78,7	32,4

TABLE II: MobileNetV2, Default / Extended optimization execution time comparisons (%).

	Total exec. time (ms)	% to Basic
All Opts Disabled	467,42	0,00
OpFusion	456,25	2,39
FoldConstant	463,14	0,92
FoldScaleAxis	480,1	-2,71
AlterOpLayout	454,67	2,73
CanonicalizeOps	454,49	2,77
CanonicalizeCast	454,57	2,75
EliminateCommonSubexpr	471,2	-0,81
CombineParallelConv2D	454,66	2,73
CombineParallelDense	454,26	2,82
CombineParallelBatchMatmul	455,25	2,60
FastMath	462,62	1,03
All Opts Enabled	475,45	-1,72

TABLE III: Execution time comparisons (%) across all optimization settings, ResNet101, Local device.

	Total exec. time	% to Basic opt.
All Disabled	63,43	0,00
OpFusion	64,11	-1,07
FoldConstant	60,32	4,90
FoldScaleAxis	64,28	-1,34
AlterOpLayout	FAILED	FAILED
CanonicalizeOps	62,68	1,18
CanonicalizeCast	62,78	1,02
EliminateCommonSubexpr	62,96	0,74
CombineParallelConv2D	63,89	-0,73
CombineParallelDense	66,9	-5,47
CombineParallelBatchMatmul	63,15	0,44
FastMath	66,8	-5,31
All Enabled	57,11	9,96

TABLE IV: Execution time comparisons (%) across all optimization settings, InceptionV3, Local device.

	Server	Xavier	Hikey	Local
Keras	8,9	-	-81,8	56,6
TensorFlow	4,0	-	-78,2	50,8
TFLite	3,9	-	-78,4	30,9
PyTorch	3,8	4,9	-80,9	53,9

TABLE V: MobileNetV2, Basic / Extended optimization execution time comparisons (%).

	Server	Xavier	Hikey	Local
Keras	5,2	0,7	-1,4	-0,1
TensorFlow	4,9	0,4	-1,0	2,4
TFLite	4,8	0,4	-3,3	1,5
PyTorch	-30,9	5,3	11,5	17,6

TABLE VI: ResNet101, Basic / Default optimization execution time comparisons (%).

	Server	Xavier	Hikey	Local
Keras	5,2	0,5	-9,4	1,6
TensorFlow	5,0	0,4	-45,2	-0,7
TFLite	4,8	0,4	114,1	0,3
PyTorch	4,3	7,8	-50,4	17,0

TABLE VII: ResNet101, Basic / Extended optimization execution time comparisons (%).

	Server	Xavier	Hikey	Local
Keras	0,0	-0,3	-8,1	1,7
TensorFlow	0,1	-0,1	-44,7	-3,0
TFLite	-0,1	0,0	121,4	-1,1
PyTorch	50,9	2,4	-55,6	-0,5

TABLE VIII: ResNet101, Default / Extended optimization execution time comparisons (%).

	Server	Xavier	Hikey	Local
TensorFlow	3,3	1,4	-6,8	6,0
TFLite	6,0	1,5	-3,9	7,5
PyTorch	3,1	1,7	-5,6	5,2

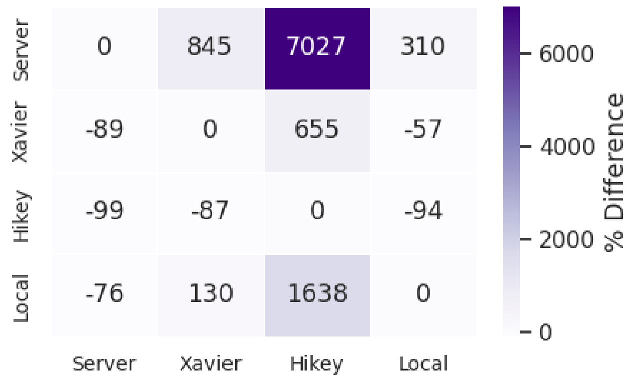
TABLE IX: InceptionV3, Basic / Default optimization execution time comparisons (%).

	Server	Xavier	Hikey	Local
TensorFlow	6,1	-35,4	-4,9	17,8
TFLite	8,1	-35,5	-2,1	16,6
PyTorch	6,3	-35,7	-4,8	19,7

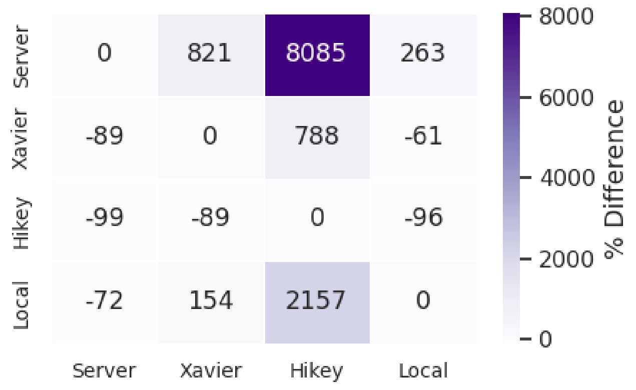
TABLE X: InceptionV3, Basic / Extended optimization execution time comparisons (%).

	Server	Xavier	Hikey	Local
TensorFlow	2,7	-36,3	2,0	11,2
TFLite	2,0	-36,4	1,9	8,5
PyTorch	3,2	-36,8	0,9	13,8

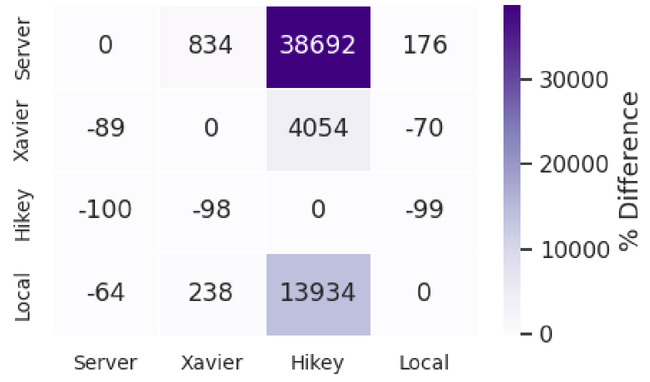
TABLE XI: InceptionV3, Default / Extended optimization execution time comparisons (%).



(a) *Basic* optimization

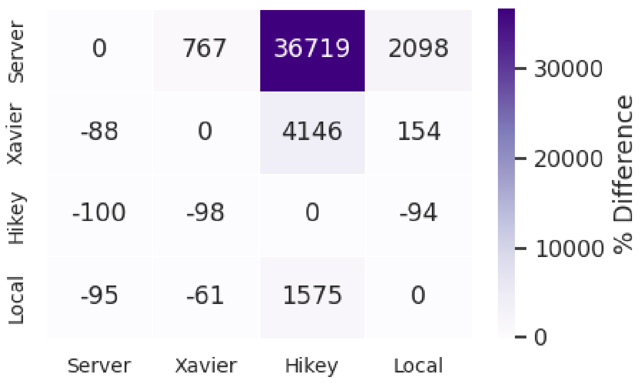


(b) *Default* optimization

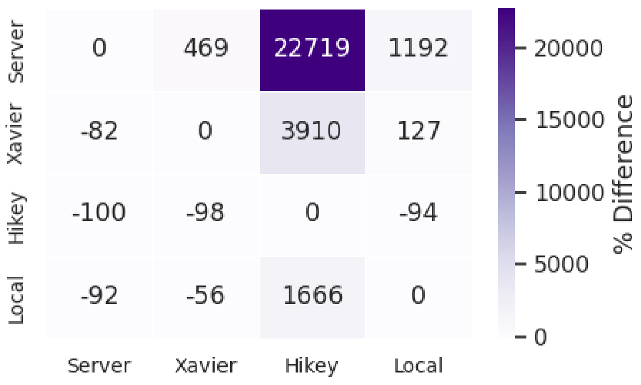


(c) *Extended* optimization

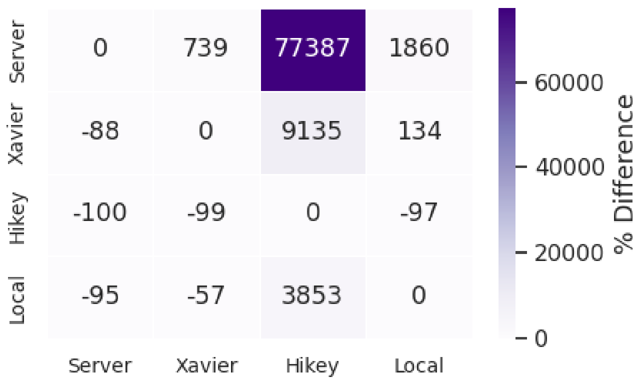
Fig. 3: Pairwise comparison of execution times across hardware acceleration devices per-optimization setting for `MobileNetV2`, `PyTorch` DNN Framework.



(a) *Basic* optimization

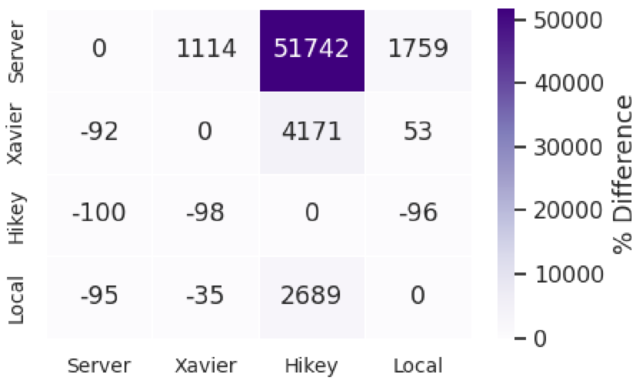


(b) *Default* optimization

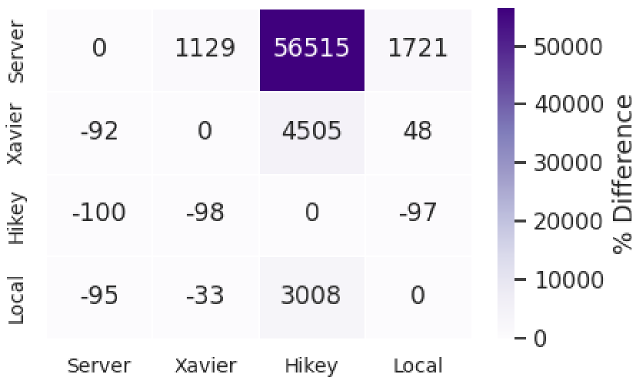


(c) *Extended* optimization

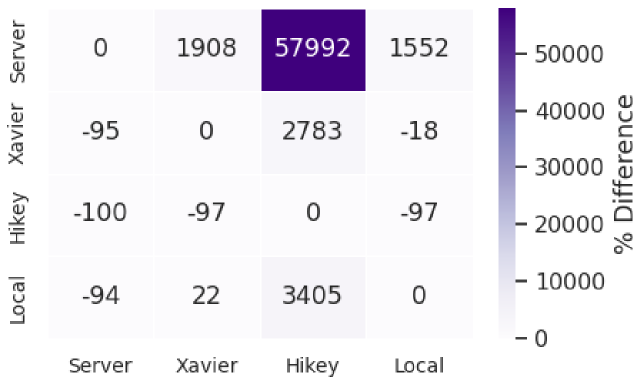
Fig. 4: Pairwise comparison of execution times across hardware acceleration devices per-optimization setting for `ResNet101V2`, `PyTorch` DNN Framework.



(a) *Basic* optimization

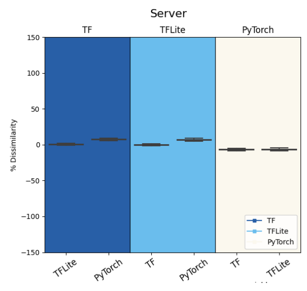


(b) *Default* optimization

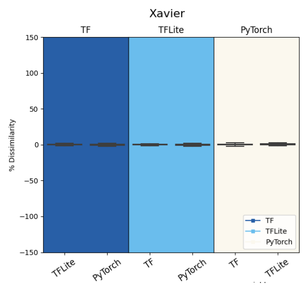


(c) *Extended* optimization

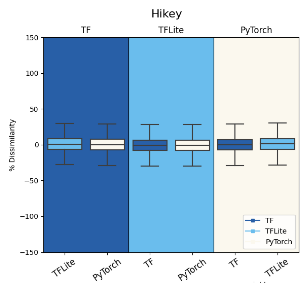
Fig. 5: Pairwise comparison of execution times across hardware acceleration devices per-optimization setting for `InceptionV3`, in `PyTorch` DNN Framework.



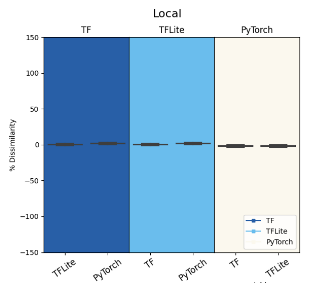
(a) `Server`



(b) `Xavier`

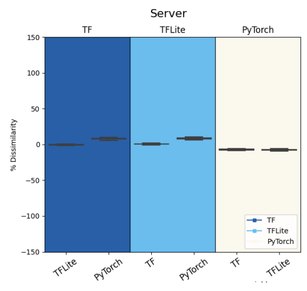


(c) `Hikey`

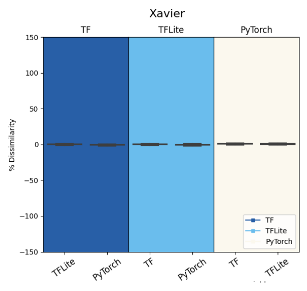


(d) `Local`

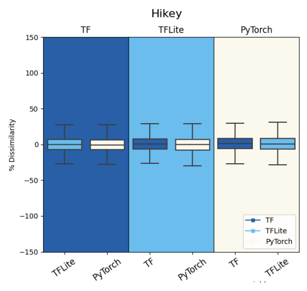
Fig. 6: Pairwise comparison of execution times across DNN Frameworks for each hardware acceleration device, *InceptionV3*, *Basic* optimization.



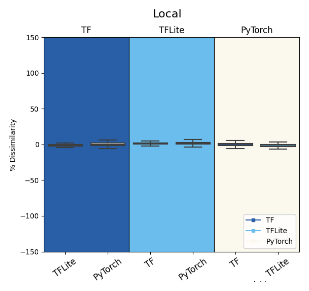
(a) `Server`



(b) `Xavier`

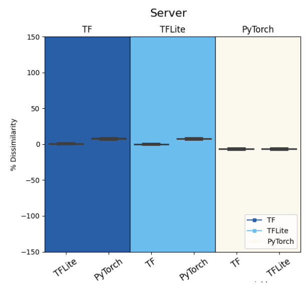


(c) `Hikey`

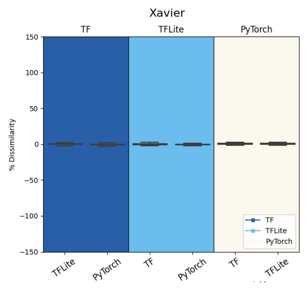


(d) `Local`

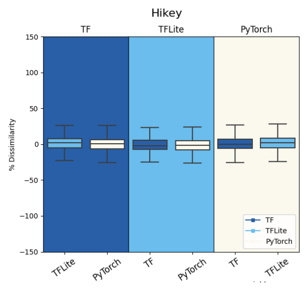
Fig. 7: Pairwise comparison of execution times across DNN Frameworks for each hardware acceleration device, *InceptionV3*, *Default* optimization.



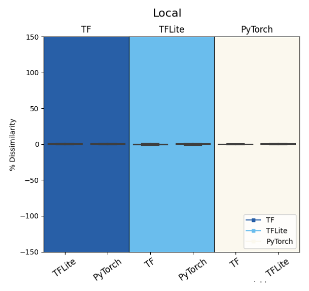
(a) `Sovu`



(b) `Xavier`

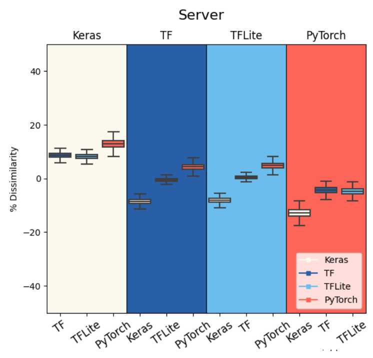


(c) `Hikey`

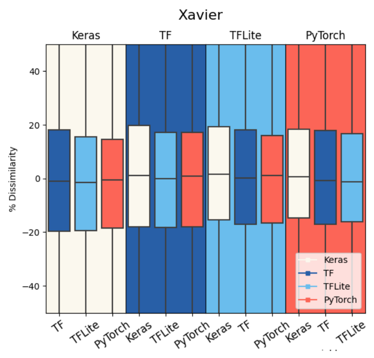


(d) `Local`

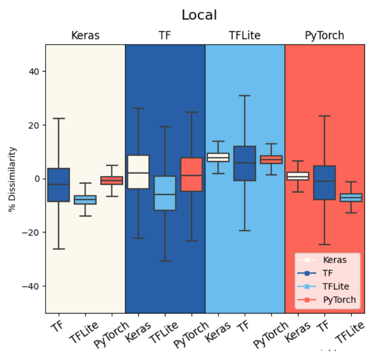
Fig. 8: Pairwise comparison of execution times across DNN Frameworks for each hardware acceleration device, *InceptionV3*, *Extended* optimization.



(a) `Server`

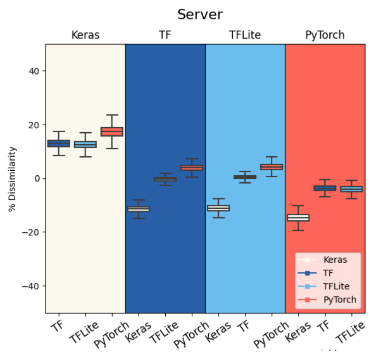


(b) `Hikey`

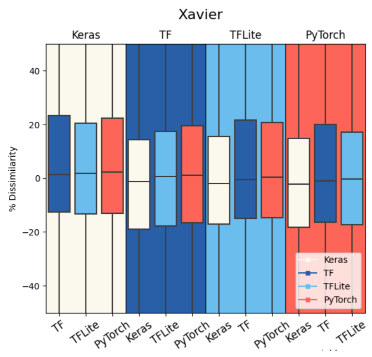


(c) `Local`

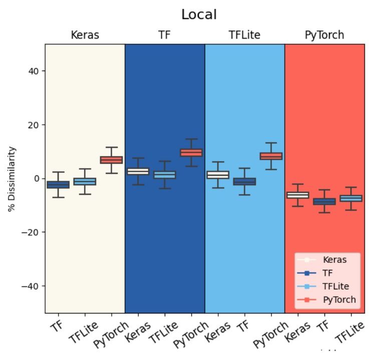
Fig. 9: Pairwise comparison of execution times across DNN Frameworks for each hardware acceleration device, `MobileNetV2`, *Basic* optimization.



(a) `Server`

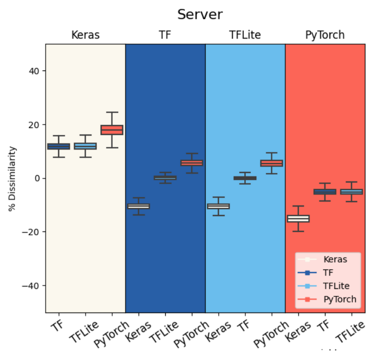


(b) `Hikey`

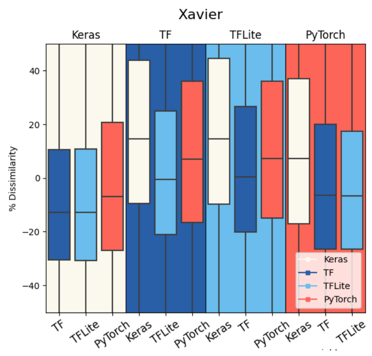


(c) `Local`

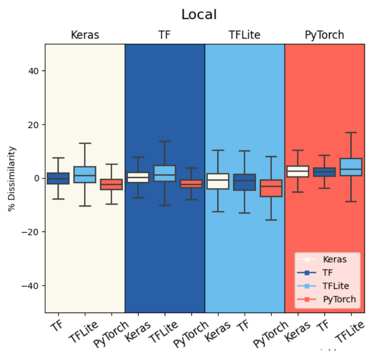
Fig. 10: Pairwise comparison of execution times across DNN Frameworks for each hardware acceleration device, `MobileNetV2`, *Default* optimization.



(a) `Server`

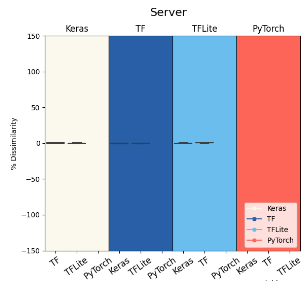


(b) `Hikey`

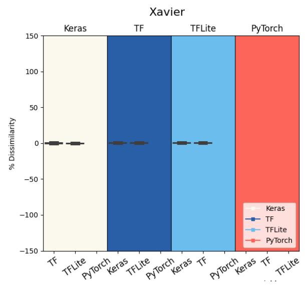


(c) `Local`

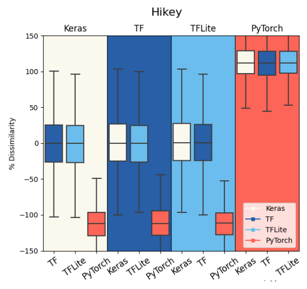
Fig. 11: Pairwise comparison of execution times across DNN Frameworks for each hardware acceleration device, `MobileNetV2`, *Extended* optimization.



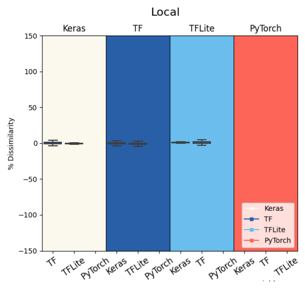
(a) `Server`



(b) `Xavier`

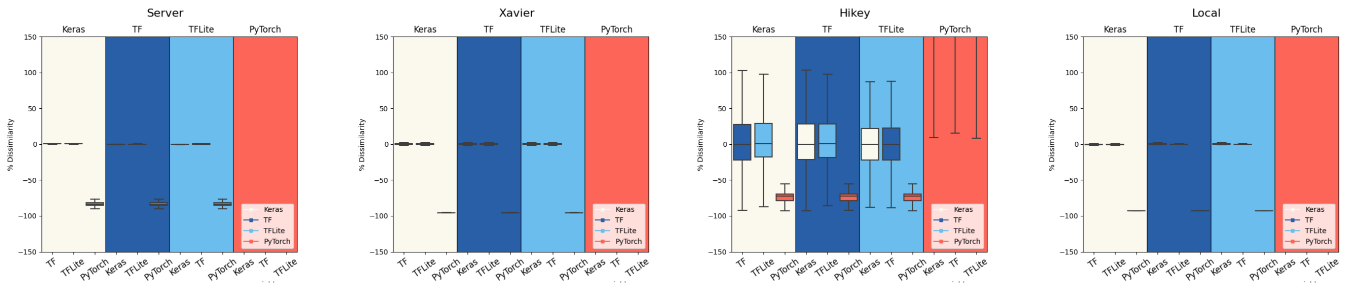


(c) `Hikey`



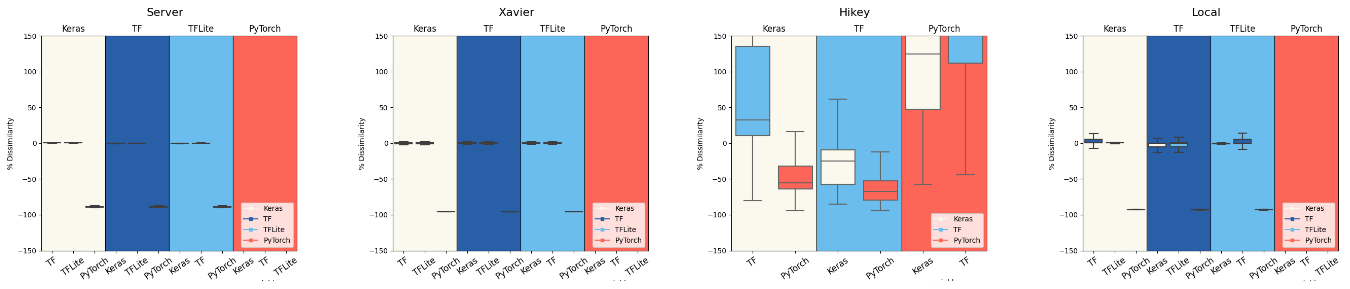
(d) `Local`

Fig. 12: Pairwise comparison of execution times across DNN Frameworks for each hardware acceleration device, `ResNet101V2`, *Basic* optimization.



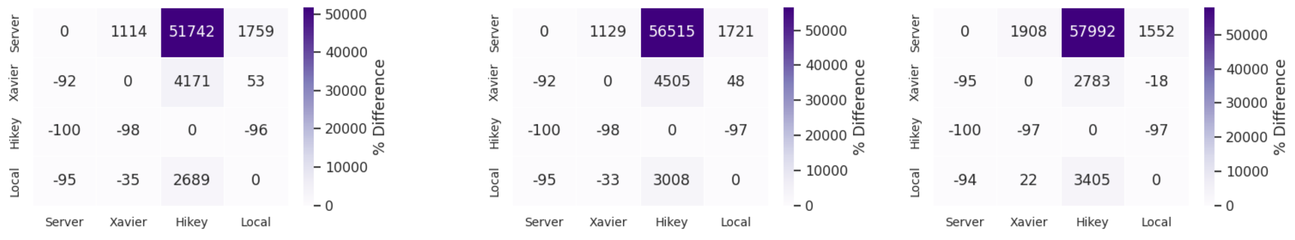
(a) `Server` (b) `Xavier` (c) `Hikey` (d) `Local`

Fig. 13: Pairwise comparison of execution times across DNN Frameworks for each hardware acceleration device, `ResNet101V2`, *Default* optimization.



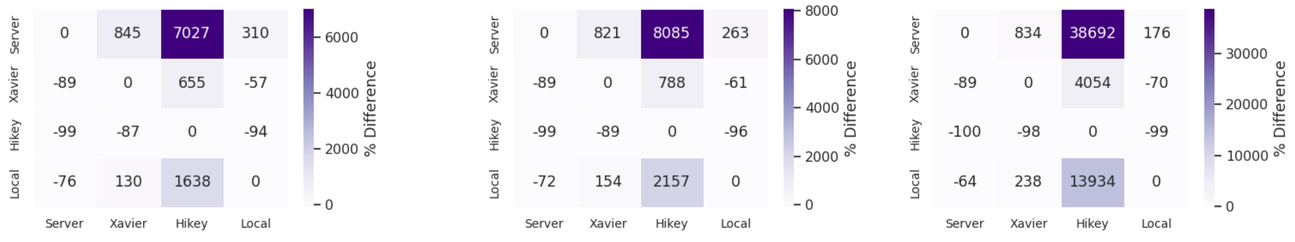
(a) `Server` (b) `Xavier` (c) `Hikey` (d) `Local`

Fig. 14: Pairwise comparison of execution times across DNN Frameworks for each hardware acceleration device, `ResNet101V2`, *Extended* optimization.



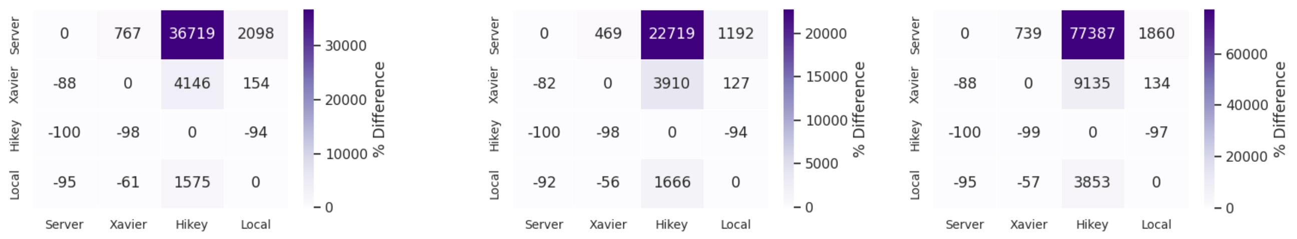
(a) *Basic* optimization (b) *Default* optimization (c) *Extended* optimization

Fig. 15: Pairwise comparison of execution times across hardware acceleration devices per-optimization setting for *InceptionV3*, `PyTorch` DNN Framework.



(a) *Basic* optimization (b) *Default* optimization (c) *Extended* optimization

Fig. 16: Pairwise comparison of execution times across hardware acceleration devices per-optimization setting for `MobileNetV2`, `PyTorch` DNN Framework.



(a) *Basic* optimization (b) *Default* optimization (c) *Extended* optimization

Fig. 17: Pairwise comparison of execution times across hardware acceleration devices per-optimization setting for `ResNet101V2`, `PyTorch` DNN Framework.