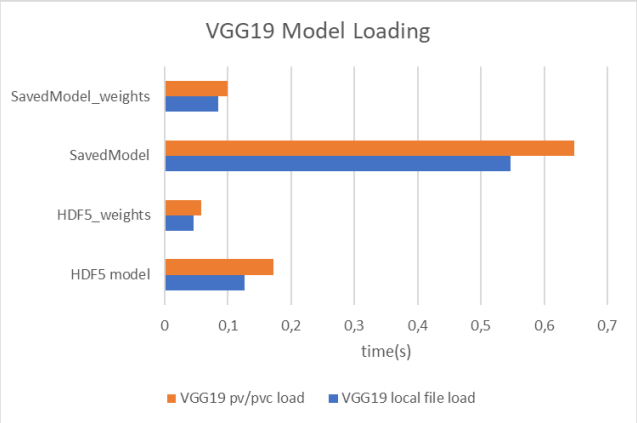

Figure 7: Comparison of read/write speeds of GPU-using KNIX sandboxes

advantage of 14-26%, depending on the data loading scheme. We therefore claim that KNIX offers additional benefit through the grouping of functions inside the same KNIX sandbox, because of the model sharing speed-up through the direct data exchange via the sandbox file system. However, in a distributed FaaS environment, if a model first needs to be unloaded from the GPU memory after a microfunction finishes, and before the follow-up function can execute, this unloading time would represent a challenge for the function start-up times because this would add additional overhead. More detailed evaluations of interactions between workflows with sequential or concurrent microfunction executions and the KNIX serverless platform are left for future work.

8 CONCLUSION

In this paper, we apply an approach for sharing GPU memory and computing resources to our high-performance serverless platform KNIX. Owing to its open design, only selected parts the framework required modifications. Employing installations of nvidia-docker, Kubernetes, KNative and the GPU-manager framework, we succeeded to partition a physical GPU into multiple vGPUs, and assigned these vGPUs to serverless KNIX microfunctions and workflows. The selected approach enables allocation elasticity by temporarily modifying container resources, therefore further improving GPU resource utilisation.

Experimental studies have been conducted to evaluate the performance impacts and overheads of GPU sharing using to different DL frameworks. The results reveal that the measured overhead introduced by the GPU sharing framework to DL frameworks is <15%, so that in general the GPU resources can still be considered as effectively managed. When measured over long execution times the fairness variations for concurrently executing functions remains low (<0.13%). However, our measurements still show potential for further optimisation as dynamic GPU resource allocation and deallocation results in strong fluctuations of assigned function's GPU resources. Finally, measurements on application performance and fast data sharing between different shared GPU-using KNIX microfunctions show potential for serverless application speed-up.

REFERENCES

[1] 2016. OpenLambda Dev Meeting July 5. http://open-lambda.org/resources/slides/july-5-16.pdf.
[2] 2021. AWS Lambda. https://aws.amazon.com/lambda/.
[3] A MNIST-like fashion product database. Benchmark 2021. A MNIST-like fashion product database. Benchmark. https://github.com/zalandoresearch/fashion-mnist.
[4] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A system for large-scale machine learning. arXiv:1605.08695 [cs.DC]
[5] Istemi Ekin Akkus, Ruichuan Chen, Ivica Rimac, Manuel Stein, Klaus Satzke, Andre Beck, Paarijaat Aditya, and Volker Hilt. 2018. SAND: Towards High-Performance Serverless Computing. In *2018 USENIX Annual Technical Conference (USENIX ATC 18)*. USENIX Association, Boston, MA, 923–935. https://www.usenix.org/conference/atc18/presentation/akkus
[6] Amazon Elastic Inference 2021. Amazon Elastic Inference. https://aws.amazon.com/machine-learning/elastic-inference/.
[7] Autonomous Vehicle and ADAS development on AWS 2020. Autonomous Vehicle and ADAS development on AWS. https://aws.amazon.com/blogs/industries/autonomous-vehicle-and-adas-development-on-aws-part-1-achieving-scale/.
[8] Azure Functions—Serverless Architecture | Microsoft Azure 2021. Azure Functions—Serverless Architecture | Microsoft Azure. https://azure.microsoft.com/en-us/services/functions/.
[9] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. 2015. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274* (2015).
[10] Cloud Functions - Serverless Environment to Build and Connect Cloud Services | Google Cloud Platform 2021. Cloud Functions - Serverless Environment to Build and Connect Cloud Services | Google Cloud Platform. https://cloud.google.com/functions/.
[11] Container Service for Kubernetes [n.d.]. Container Service for Kubernetes. https://www.alibabacloud.com/product/kubernetes.
[12] J. Duato, A. J. Peña, F. Silla, R. Mayo, and E. S. Quintana-Ortí. 2010. rCUDA: Reducing the number of GPU-based accelerators in high performance clusters. In *2010 International Conference on High Performance Computing Simulation*. 224–231. https://doi.org/10.1109/HPCS.2010.5547126
[13] Alex Ellis. 2017. Functions as a Service (FaaS). https://blog.alexellis.io/functions-as-a-service/.
[14] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. 2015. A Neural Algorithm of Artistic Style. *CoRR* abs/1508.06576 (2015). arXiv:1508.06576 http://arxiv.org/abs/1508.06576
[15] GPU Manager is used for managing the nvidia GPU devices in Kubernetes cluster. 2021. GPU Manager is used for managing the nvidia GPU devices in Kubernetes cluster. https://github.com/tkestack/gpu-manager.
[16] GPU Sharing Scheduler for Kubernetes Cluster [n.d.]. GPU Sharing Scheduler for Kubernetes Cluster. https://github.com/AliyunContainerService/gpushare-scheduler-extender.
[17] IBM Cloud Functions [n.d.]. Cloud Functions - Overview | IBM Cloud. https://www.ibm.com/cloud/functions.
[18] J. Kim, T. J. Jun, D. Kang, D. Kim, and D. Kim. 2018. GPU Enabled Serverless Computing Framework. In *2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*. 533–540. https://doi.org/10.1109/PDP2018.2018.00090
[19] KNIX 2021. KNIX. https://github.com/knix-microfunctions/knix.
[20] Kubernetes Schedule GPUs 2021. Kubernetes Schedule GPUs. https://kubernetes.io/docs/tasks/manage-gpus/scheduling-gpus/.
[21] Nuclio Serverless Functions [n.d.]. Nuclio Serverless Functions. https://nuclio.io/.
[22] NVIDIA-Docker [n.d.]. Build and run Docker containers leveraging NVIDIA GPUs. https://github.com/NVIDIA/nvidia-docker.
[23] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. 2017. Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration. *PyTorch: Tensors and dynamic neural networks in Python with strong GPU acceleration* 6 (2017), 3.
[24] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556 [cs.CV]
[25] tf.linalg.matmul API 2021. tf.linalg.matmul API. https://www.tensorflow.org/api_docs/python/tf/linalg/matmul.
[26] The CIFAR-10 dataset 2021. The CIFAR-10 dataset. https://www.tensorflow.org/datasets/catalog/cifar10.