Réalisation et évaluation d'un prototype FaaS d'inférence sur des modèles d'IA

Encadrement

- Camélia Slimani

- Daniel Hagimont et/ou Boris Teabe

Description du Contexte Global

Le **paradigme Function-as-a-Service** est aujourd'hui largement adopté par les fournisseurs de cloud pour des services d'inférence sur des modèles de machine learning (ML). Dans ce modèle d'exécution, le client publie le modèle de ML ainsi que la fonction d'inférence, et délègue au fournisseur de cloud la responsabilité de gestion des ressources et de mise à l'échelle de la fonction d'inférence. Cela a pour avantages (1) de décharger le développeur de la gestion de l'infrastructure de déploiement, (2) la facturation à l'utilisation réelle des ressources, puisque le coût n'est engagé que lorsque le code s'exécute, (3) la mise à l'échelle automatique, le nombre d'instances déployées de la fonction est adapté aux besoins.

Bien que les GPU (Graphical Processing Unit) soient des accélérateurs matériels intensivement utilisés pour accélérer l'excéution de fonctions d'inférences sur des modèles d'apprentissage profond, il existe peu de solutions permettant de les utiliser dans le contexte l'inférence en FaaS [1][2][3], en raison de la complexité inhérente à la programmation et à la virtualisation des GPU : entre autres, la gestion des transferts de données entre la mémoire hôte et la mémoire du GPU, l'exécution concurrente de plusieurs fonctions.

Objectif du projet :

L'équipe SEPIA de l'IRIT travaille à mettre en place des mécanismes permettant un déploiement de moteurs d'inférence selon le paradigme FaaS. Afin d'experimenter les approches proposées, la première étape est de réaliser un prototype simple de FaaS exploitant un GPU. Il s'agit de l'objectif de ce projet.

Etapes de réalisation du projet :

Les étapes de réalisation du projet sont les suivantes :

1. Faire un état de l'art des solutions existantes de plateformes FaaS utilisant du GPU ;
2. Identifier les spécifications auxquelles devra répondre votre plateforme FaaS

3. Concevoir un prototype de plateforme FaaS utilisant le GPU pour l'inférence sur des modèles d'apprentissage profond, en utilisant Apache OpenWhisk par exemple.
4. Définir un protocole expérimental pour évaluer la plateforme :
    1. Choisir les modèles à utiliser ;
    2. Coder la/les fonctions d'inférence;
    3. Définir les scénarios d'exécution ;
    4. Identifier les métriques d'évaluation de la plateforme : taux d'occupation du GPU, utilisation des ressources, temps de démarrage des fonctions, temps de réponse, etc.)
5. Exécuter le protocole expérimental.

Ressources :

- Matériel : Nvidia Jetson Nano (fourni) + machines personnelles
- Logiciel :
    - Programmation en C/C++ et CUDA[4].
    - Utilisation d'OpenWhisk [5] pour la mise en œuvre de la plateforme.
    - Utilisation de la bibliothèque TensorRT[6] pour la programmation des fonctions d'inférence
    - Utilisation de Nvidia Nsight [7] pour l'évaluation.

Intérêt pour les étudiants :

Le projet porte sur un sujet d'actualité qui est le déploiement d'une platforme FaaS pour l'apprentissage profond. Par ailleurs, le projet permet de comprendre l'architecture et les contraintes des GPU, et d'utiliser et de maitriser des outils valorisables pour un ingénieur : OpenWhisk et Nvidia Nsight.

Bibliographie

1: Minchen Yu Ao Wang Dong Chen Haoxuan Yu Xiaonan Luo Zhuohao Li Wei Wang Ruichuan Chen Dapeng Nie Haoran Yang, FaaSwap: SLO-Aware, GPU-Efficient Serverless Inference via Model Swapping, 2024
2: Wu, H., Yu, Y., Deng, J., Ibrahim, S., Wu, S., Fan, H., ... & Jin, H., WU, Hao, YU, Yue, DENG, Junxiao, et al. {StreamBox}: A Lightweight {GPU}{SandBox} for Serverless Inference Workflow, 2024
3: Lukas Tobler, GPUless – Serverless GPU Functions, 2022
3: Cheng, J., Grossman, M., & McKercher, T. , Professional CUDA c programming,
4: , Open Source Serverless Cloud Platform, , https://openwhisk.apache.org/
5: , Nvidia TensorRT, , https://developer.nvidia.com/tensorrt
6: , NVIDIA Nsight Systems, , https://developer.nvidia.com/nsight-systems