Poster n°40

Comparaison des frameworks d'apprentissage profond sur un seul nœud de calcul

Jean-Sébastien Lerat[1,2] Sidi Ahmed Mahmoudi[1] Saïd Mahmoudi[1]

[1]Service informatique logiciel et intelligence artificielle, département informatique et gestion, faculté polytechnique, UMONS

[2]Département des sciences et technologies, haute école en Hainaut

Introduction

Le *deep learning* est un domaine en plein expansion, expérimenté et mis en production à travers des *frameworks*. Ceux-ci sont exploités aussi bien sur des clusters de calcul que sur des cartes embarquées comme la Jeston Nano de Nvidia. De plus en plus souvent, la charge de travail est distribuée sur différents nœuds de calcul sans considérer le choix du framework. Dans ce travail, nous analysons et comparons ces frameworks.

Méthode

Le choix des frameworks a été fait en fonction de quatre critères : une **implémentation native** afin de profiter au mieux des ressources physiques, le support de **OpenCL & CUDA** afin de tirer parti des **GPUs**, un interfaçage vers **Python** afin de garantir un prototypage rapide et une **communauté** activement impliquée afin de garantir la longévité du framework. L'entraînement est mesuré sur quatre architectures neuronales différentes et bien connus, à savoir : **AlexNet**, **MobileNet** v2, **ResNet50** et **VGG16**. Celles-ci diffèrent en complexité. Deux jeux de données ont été utilisé pour entraîner un classifier à trois étiquettes (feu, fumée, pas de feu) : 791 et 6003 photos.

Résultats

En moyenne et dans le pire des cas, pyTorch est le plus rapide sur CPU et il est assez performant sur le GPU. D'autres tests ont également montré que pyTorch et Tensorflow sont les frameworks les plus stables car ils s'adaptent aux ressources qu'ils ont à disposition.

Frameworks	AlexNet		MobileNet		ResNet		VGG	
	CPU	*GPU*	*CPU*	*GPU*	*CPU*	*GPU*	*CPU*	*GPU*
MxNet	401 2993	19 114	181 1391	28 182	547 4181	33 232	2178 16480	28 185
Paddle	430 1107	328 2380	431 1045	338 2533	2273 5790	332 2500	7273 16493	337 2462
pyTorch	139 879	23 122	221 1459	40 254	480 3122	43 271	630 9010	34 190
Singa	273 /	6 /	430 /	6 /	977 /	6 /	1227 /	6 /
Tensorflow	528 3957	343 2542	420 3223	341 2564	835 6289	351 2605	1979 16397	350 2648

Conclusion

Lorsqu'il s'agit d'avoir un framework stable et efficace pyTorch semble être la meilleure option, en particulier lorsque les ressources sont limitées comme sur les cartes embarquées. Nous envisageons d'automatiser la compression de réseaux de neurones à l'aide de pyTorch et de distribuer le calcul entre les ressources.