BENCHMARK TRA FPGA E GPU EMBEDDED PER MODELLI DI DEEP LEARNING

C. Nuzzi[(1)], S. Pasinetti[(1)], F. Docchio[(1)], G. Sansoni[(1)]
[(1)]Dip. di Ingegneria Meccanica e Industriale, Università degli Studi di Brescia
mail autore di riferimento: c.nuzzi@unibs.it

1. INTRODUZIONE

Le soluzioni embedded sul mercato per applicazioni di intelligenza artificiale sono sempre più frequenti, accessibili a prezzi che vanno sempre più riducendosi. Se le FPGA sono una piattaforma già affermata e tipicamente prestante, le GPU di contro sono state introdotte solo recentemente nel mercato embedded, proprio grazie alla forte spinta che la ricerca ha dato nell'ambito delle applicazioni di Deep Learning e intelligenza artificiale. In questo lavoro abbiamo valutato le prestazioni di riconoscimento di un modello di rete neurale di convoluzione rispettivamente su una **FPGA Xilinx, ZCU104** e su una **GPU embedded Nvidia Jetson TX2**.

2. IL MODELLO DI VALUTAZIONE

Per questo benchmark si è scelto di utilizzare il modello di rete neurale VGG-16 [1] addestrato sul dataset CIFAR-10 [2]. L'addestramento è stato eseguito su un computer host, ottenendo una rete finale con rappresentazione a virgola mobile a 32 bit (**FP32**). Questa rappresentazione permette alla rete di ottenere un'accuratezza elevata, ma di contro implica una dimensione non trascurabile del modello, caratteristica fondamentale nel caso dei dispositivi embedded. Per questo motivo è necessario ridurre la dimensione del modello: grazie ai processi di quantizzazione e pruning del modello originale è stato possibile passare a una rappresentazione a interi a 8 bit (**INT8**) nel caso della FPGA (eseguita tramite software proprietario) e ad una rappresentazione a virgola mobile a 16 bit (**FP16**) nel caso della scheda Jetson TX2 (eseguita tramite TensorRT, software open source sviluppato da Nvidia).

3. RISULTATI SPERIMENTALI

Sono stati valutati due modelli diversi per la FPGA: "**quant8**", dove il modello originale è stato sottoposto solo alla procedura di quantizzazione per passare alla rappresentazione INT8, e "**pruned8**", dove il modello originale è stato inoltre sottoposto alla procedura di pruning (Tabella 1). Per quanto riguarda le prestazioni della scheda Jetson TX2, si è valutato solamente un modello chiamato "**quant16**", dove il modello originale è stato sottoposto a quantizzazione per passare da FP32 a FP16 (Tabella 2). Si noti che la scheda Jetson TX2 presenta una CPU Denver a 2 core e una CPU ARM a 4 core: il valore complessivo dell'utilizzo della CPU è dunque riportato come media tra le singole percentuali dei core attivi in quella particolare configurazione. La FPGA possiede invece una singola CPU a 4 core e una "Deep Processing Unit" (DPU) dedicata all'elaborazione di algoritmi di Deep Learning, il cui utilizzo e prestazioni si contrappongono a quelle delle normali GPU. Pertanto, per la scheda ZCU104 le modalità di funzionamento si differenziavano per il numero di processi in parallelo utilizzati (threads da 1 a 6) mentre la scheda Jetson TX2 presenta 5 modalità di funzionamento differenti a seconda di quali e quanti core delle due CPU utilizzare. Altra metrica fondamentale per il caso in esame sono i frame per secondo (FPS) che le schede sono in grado di gestire e il tempo di inferenza (inference time), che indica quanto tempo è richiesto alla scheda per produrre in uscita una predizione sull'immagine corrente.

4. CONCLUSIONE

Dai risultati ottenuti emerge che le configurazioni migliori per la FPGA risultano essere in entrambi i casi quelle a 4 threads, mentre per la scheda Jetson TX2 non sorprende che la configurazione migliore risulti essere quella in modalità Max-N, ovvero la modalità che consente le massime prestazioni della scheda. In generale, anche considerando il prezzo dei due dispositivi valutati, le performance della GPU embedded sono migliori e accessibili ad un prezzo molto ragionevole. Per quanto riguarda le accuratezze raggiunte dai tre modelli, si ha una Top1-Accuracy dell'86.60% nel caso "*quant8*", dell'84.20% nel caso "*pruned8*" e dell'85.80% nel caso "*quant16*". Il lieve scarto di accuratezza tra i casi "*quant8*" e "*quant16*" è dovuto al software di quantizzazione utilizzato che, nel primo caso, è ottimizzato per le architetture target della compagnia.

Tabella. 1 *Performance della configurazione "quant8" e della configurazione "pruned8" della FPGA.*

	Threads	FPS	Corrente (A)	Potenza (W)	Efficienza energetica (FPS/W)	Inference time (ms)	CPU (%)	DPU (%)
QUANT8	1	840.49	1.69	20.28	41.44	1.19	25.30	99.00
	2	1547.10	1.74	20.88	74.09	0.65		
	3	1781.74	1.76	21.12	84.36	0.56		
	4	**1886.96**	**1.77**	**21.24**	**88.84**	**0.53**		
	5	1723.23	1.79	21.48	80.22	0.58		
	6	1710.18	1.82	21.84	78.30	0.58		
PRUNED8	1	1423.19	1.66	19.92	71.45	0.70	25.30	99.00
	2	2639.07	1.70	20.40	129.37	0.38		
	3	2979.68	1.72	20.64	144.36	0.34		
	4	**3625.26**	**1.75**	**21.00**	**172.63**	**0.28**		
	5	3141.14	1.77	21.24	147.89	0.32		
	6	2922.84	1.80	21.60	135.32	0.34		

Tabella. 2 *Performance della configurazione "quant16" della scheda Jetson TX2..*

	Modalità	FPS	Corrente (A)	Potenza (W)	Efficienza energetica (FPS/W)	Inference time (ms)	CPU (%)	GPU (%)
QUANT16	**Max-N**	**1697.15**	**0.56**	**10.64**	**159.51**	**0.59**	**38.83**	**79.00**
	Max-Q	1175.13	0.32	6.06	193.88	0.85	64.50	75.00
	Max-P	1439.38	0.41	7.85	183.43	0.69	45.16	75.00
	Max-P ARM	1582.22	0.44	8.40	188.40	0.63	57.50	81.00
	Max-P Denver	690.45	0.33	6.19	111.47	1.45	63.33	64.00

RIFERIMENTI BIBLIOGRAFICI

[1] K. Simonyan e A. Zisserman, «Very Deep Convolutional Networks for Large-Scale Image Recognition,» in *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, 2014.

[2] A. Torralba, R. Fergus e W. T. Freeman, «80 Million Tiny Images: A Large Data Set for Nonparametric Object and Scene Recognition,» in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008.