Guida per scelta della GPU

In questo documento vengono presentati dei test fatti con la libreria SB-SDK al fine di dare dei criteri oggettivi per la scelta di una GPU NVidia che possa essere il miglior compromesso fra costo e prestazioni per la vostra applicazione.

Si premette che in generale per il Deep Learning si consigliano le GPU NVidia della serie GeForce rispetto alla serie Quadro per motivi di velocità di calcolo e di costo.

I parametri da valutare per la scelta di una GPU NVidia si riducono sostanzialmente ai seguenti:

- quantità di RAM
- numero di core
- costo

Di seguito i link al sito web della NVidia con le specifiche delle GPU GeForce serie 30 e 40 da cui sono estratte anche le tabelle, riportate qui solo per completezza:

https://www.nvidia.com/it-it/geforce/graphics-cards/30-series/

https://www.nvidia.com/it-it/geforce/graphics-cards/40-series/

	GeForce RTX 3090 Ti	GeForce RTX 3090	GeForce RTX 3080 Ti	GeForce RTX 3080	GeForce RTX 3070 Ti	GeForce RTX 3070	GeForce RTX 3060 Ti	GeForce RTX 3060	GeForce RTX 3050
Core NVIDIA CUDA	10752	10496	10240	8960 / 8704	6144	5888	4864	3584	2560 / 2304
Boost Clock (GHz)	1.86	1.70	1.67	1.71	1.77	1.73	1.67	1.78	1.78 / 1.76
Dimensioni della memoria	24 GB	24 GB	12 GB	12 GB / 10 GB	8 GB	8 GB	8 GB	12 GB	8 GB
Tipo di memoria	GDDR6X	GDDR6X	GDDR6X	GDDR6X	GDDR6X	GDDR6	GDDR6	GDDR6	GDDR6

	GeForce RTX 4090	GeForce RTX 4080	GeForce RTX 4070 Ti	GeForce RTX 4070	GeForce RTX 4060 Ti	GeForce RTX 4060
Core NVIDIA CUDA	16384	9728	7680	5888	4352	3072
Boost Clock (GHz)	2.52	2.51	2.61	2.48	2.54	2.46
Dimensioni della memoria	24 GB	16 GB	12 GB	12 GB	16 GB oppure 8 GB	8 GB
Tipo di memoria	GDDR6X	GDDR6X	GDDR6X	GDDR6X	GDDR6	GDDR6

Per i test sono state utilizzate 3 GPU differenti, due recenti ed una molto datata, con differenti dimensioni della memoria e del numero di NVIDIA core:

	RAM [GB]	Core NVIDIA CUDA	Anno
GeForce RTX 3090	24	10496	2020
GeForce RTX 3060	12	3584	2020
GeForce RTX 850M	4	640	2013

Si è scelta la RTX 3090 e la RTX 3060 come esempi di una GPU in fascia alta ed una in fascia bassa come può risultare dalla tabella delle caratteristiche delle GPU della serie GeForce 30, in modo che si possano valutare le differenze di prestazione.

FABERVISION Srl
Via Tonale 9
24061 Albano S. Alessandro (BG)
ITALY

Phone +39 035 19752207
Vat IT03929430985
E-Mail info@fabervision.com
Website www.fabervision.com

Le GPU RTX 3090 e la RTX 3060 sono state montate su una workstation HP Z4G4 con un processore Intel i9-10900X con 10 core, mentre la GPU RTX 850M su un notebook ASUS con un processore i7 4710HQ con 4 core.

I test sono stati fatti con la versione 1.13.0.9 dell'SB-SDK. In questa versione sono implementate 8 reti differenti di Deep Learning, 3 per **Deep Surface** e 5 per **Deep Cortex**.

1 Da che parametri dipendono l'uso della memoria della GPU

Come criterio generale si consideri che i fattori che influiscono sia sull'utilizzo della memoria della GPU che sul tempo di inferenza sono:

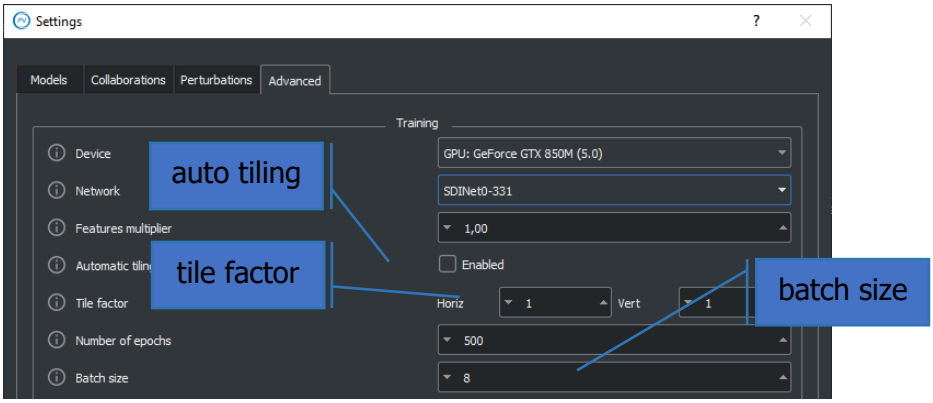
1. Il numero di immagini che devono essere elaborate contemporaneamente
2. la dimensione della immagine di ingresso della rete
3. il numero di parametri della rete

Questi fattori dipendono da alcuni parametri di progetto che vengono impostati nel menù settings della **SB-GUI** oppure da software con la funzione sb_project_set_par e che verranno spiegati nei prossimi capitoli.

1.1 Numero di immagini elaborate contemporaneamente

Il primo fattore, ossia il numero di immagini che devono essere elaborate contemporaneamente, è legato a due gruppi di parametri della libreria che sono:

- batch_size che influenza solo l'**SVL**
- auto_tiling, scale e tile_factor con i quali si regola il numero di tile con cui deve essere piastrellata l'immagine e che influenzano solo l'**inferenza**.



1.1.1 batch_size

Il batch_size è un parametro esclusivo dell'SVL per cui non influenza l'inferenza. Per quanto riguarda l'utilizzo delle risorse della GPU questo parametro è importante perché tutte le immagini del batch vengono elaborate contemporaneamente e quindi richiedono una certa quantità di memoria della GPU.

Il valore di batch_size va impostato in base alla dimensione del dataset di SVL e più immagini ci sono più il valore deve essere aumentato.

- **Deep Surface**: il numero delle immagini su cui calcolare batch_size corrisponde al numero di tile totali di tutte le immagini effettivamente utilizzabili dalla SVL, ossia tutti i tile in cui ci sia della ROI di analisi. Solitamente Deep Surface avendo necessità di poche immagini, richiede valori di batch_size fra 4 e 16.
- **Deep Cortex**: il numero della immagine corrisponde al numero delle immagini del dataset di SVL. Avendo bisogno di molte immagini (centinaia) servono valori di batch_size fra 8 e 64.

Di seguito una tabella che mostra il valore massimo del parametro batch_size a seconda della rete, con il parametro feature_multiplier impostato a 1, e di alcune GPU con differente dimensione della memoria. Superando questo valore la funzione sb_svl_run ritornerà l'errore SB_ERR_DL_CUDA_OUT_OF_MEMORY.

FABERVISION Srl
Via Tonale 9
24061 Albano S. Alessandro (BG)
ITALY

Phone +39 035 19752207
Vat IT03929430985
E-Mail info@fabervision.com
Website www.fabervision.com

	Tipo di rete sb_t_network_type	batch_size massimo		
		RTX 3090	RTX 3060	RTX 850M
DEEP SURFACE	*SB_NETWORK_TYPE_SDINET0_331x128*	256	128	64
	SB_NETWORK_TYPE_SDINET0_400x160	256	64	32
	SB_NETWORK_TYPE_SDINET0_331	128	64	16
DEEP CORTEX	*SB_NETWORK_TYPE_ICNET0_64*	512	512	512
	SB_NETWORK_TYPE_ICNET0_128	512	512	512
	SB_NETWORK_TYPE_EFFICIENTNET_B0	128	64	16
	SB_NETWORK_TYPE_EFFICIENTNET_B1	128	32	8
	SB_NETWORK_TYPE_EFFICIENTNET_B2	64	32	8

Si consideri anche che al diminuire del parametro feature_multiplier il numero massimo di batch aumenta e viceversa.

Si veda anche la descrizione del parametro batch_size nella documentazione dell'SB-SDK.

1.1.2 auto_tiling, scale e tile_factor

Con questi parametri, utilizzati solo da **Deep Surface,** si imposta il numero di tile con cui deve essere piastrellata l'immagine da elaborare e che, durante l'inferenza, verranno elaborati contemporaneamente richiedendo quindi una certa quantità di memoria.

- **La dimensione della memoria della GPU** limita il numero massimo di tile impostabile il quale dipende, ovviamente, anche dal tipo di rete. Superando questo valore la funzione sb_project_detection ritornerà l'errore SB_ERR_DL_CUDA_OUT_OF_MEMORY.
- **Il numero di core della GPU** limita il numero di tile elaborabili in un certo tempo, questo punto verrà affrontato nel capitolo 2 *Tempi di inferenza in Runtime*.

Di seguito una tabella con il numero massimo di tile a seconda del tipo di rete, con il parametro feature_multiplier impostato a 1, e di alcune GPU con differenti dimensione della memoria.

	Tipo di rete sb_t_network_type	Numero massimo di tile		
		RTX 3090	RTX 3060	RTX 850M
DEEP SURFACE	*SB_NETWORK_TYPE_SDINET0_331x128*	1450	730	170
	SB_NETWORK_TYPE_SDINET0_400x160	790	440	100
	SB_NETWORK_TYPE_SDINET0_331	580	225	64

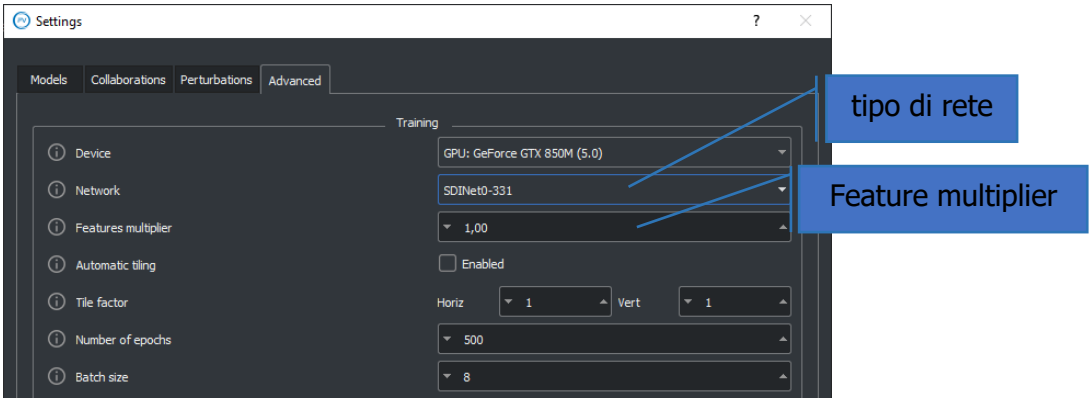
Si consideri anche che al diminuire del parametro feature_multiplier il numero massimo di tile aumenta e viceversa.

Si veda anche la descrizione di parametri auto_tiling, scale e tile_factor nella documentazione dell'SB-SDK.

1.2 Dimensione della immagine e numero di parametri della rete

Gli ultimi due fattori, ossia la dimensione della immagine di ingresso della rete e il numero di parametri della rete, dipendono da due parametri:

- type che specifica il tipo di rete e di conseguenza anche la dimensione della immagine in ingresso alla rete e il numero di parametri della rete;
- feature_multiplier che modula la complessità della rete e di conseguenza anche il numero dei parametri della rete. In particolare il valore può essere minore di 1 se si vuole ridurre la complessità della rete oppure maggiore di 1 se la si vuole aumentare.



La tabella che segue mostra:

- per ogni tipo di rete la dimensione della immagine in ingresso alla rete stessa;
- come varia il numero di parametri della rete al variare del parametro feature_multiplier. Come esempio si sono scelti 3 valori 0.7, 1.0 (default) e 1.3, ma il parametro può variare a passi di 0.1 tra 0.5 e 1.5.

	Tipo di rete type	**Dimensione di ingresso della rete [pixel]**	**Numero di parametri [milioni]** feature_multiplier		
			0.7	**1.0**	**1.3**
DEEP SURFACE	*SB_NETWORK_TYPE_SDINET0_331x128*	331 x 128	0.35	0.70	1.18
	SB_NETWORK_TYPE_SDINET0_400x160	400 x 160	0.35	0.70	1.18
	SB_NETWORK_TYPE_SDINET0_331	331 x 331	0.35	0.70	1.18
DEEP CORTEX	*SB_NETWORK_TYPE_ICNET0_64*	64 x 64	0.06	0.13	0.19
	SB_NETWORK_TYPE_ICNET0_128	128 x 128	0.09	0.24	0.25
	SB_NETWORK_TYPE_EFFICIENTNET_B0	224 x 224	2.06	4.02	6.63
	SB_NETWORK_TYPE_EFFICIENTNET_B1	240 x 240	3.33	6.52	10.82
	SB_NETWORK_TYPE_EFFICIENTNET_B2	260 x 260	3.89	7.71	12.94

Si veda anche la descrizione di parametri type e feature_multiplier nella documentazione dell'SB-SDK.

2 Tempi di inferenza in runtime

Nei paragrafi precedenti sono stati spiegati i parametri della libreria che influenzano l'uso delle risorse della GPU. In particolare, per quanto riguarda il tempo di inferenza in runtime, di seguito verrà spiegato per ognuno dei moduli **Deep Cortex** e **Deep Surface** da quali parametri dipende:

- **Deep Cortex** da type e feature_multiplier
- **Deep Surface** da type, feature_multiplier, auto_tiling, scale e tile_factor

Per tempo di inferenza si intende il tempo di esecuzione delle funzioni sb_project_detection e sb_project_get_res che vengono chiamate in sequenza, la prima funzione esegue l'inferenza vera e propria la seconda serve per avere i risultati della inferenza.

L'inferenza di una immagine coinvolge oltre alla GPU anche la CPU la quale si occupa del pre-processing (ad esempio il ridimensionamento della immagine e la suddivisione in tile) e del post-processing (ad esempio creazione del piano di voto, del piano modello, del piano verità e della blob analisi). Nella tabella quindi viene dato sia il tempo della GPU che quello della CPU, dove, ovviamente, il tempo totale è dato dalla somma dei due tempi.

2.1 Deep Surface

Per questo test è stata utilizzata la soluzione di esempio *deep_surface_coins* rilasciata insieme all'**SB-SDK**. A partire da questa soluzione sono state create delle nuove soluzione in ognuna delle quali le immagini e le ROI sono state ridimensionate al fine di ottenere il numero di tile desiderato.

Vengono presentati i tempi di calcolo di **Deep Surface** con la seguente configurazione:

Parametro	Valore
type	SB_NETWORK_TYPE_SDINET0_331
feature_multiplier	1.0
auto_tiling	Enable
scale	1.0
tile_factor	con auto_tiling abilitato non viene utilizzato

Per quanto riguarda il tempo di pre e post processing si fa notare che nel test è stato abilitato l'auto_tiling con il parametro scale impostato a 1.0 in modo da non fare nessun ridimensionamento della immagine il quale comporta un aumento del tempo di calcolo solo a carico della CPU.

Di seguito una tabella con i tempi di inferenza in ms al variare del numero di tile e con differenti GPU.

		SB_NETWORK_TYPE_SDINET0_331									
Image resolution	Numero di tile	LINUX UBUNTU 20,04				WINDOWS 10					
		RTX 3090		RTX 3060		RTX 3090		RTX 3060		RTX 850M	
		CPU	GPU	CPU	GPU	CPU	GPU	CPU	GPU	CPU	GPU
300x300	**1 (1x1)**	1	6	1	6	2	7	2	7	2	22
400x400	**4 (2x2)**	2	7	2	10	4	9	4	11	5	79
800x800	**9 (3x3)**	5	10	6	20	14	13	14	22	19	178
1150x1150	**16 (4x4)**	10	17	11	35	23	21	23	38	33	313
1450x1450	**25 (5x5)**	17	26	19	53	45	33	43	57	68	492
1750x1750	**36 (6x6)**	37	42	32	76	53	43	53	82	77	711
2350x2350	**64 (8x8)**	68	65	68	134	93	76	94	144	128	1260
3500x3500	**144 (12x12)**	167	166	168	322	208	167	203	319	n.d.	n.d.
4700x4700	**256 (16x16)**	286	296	285	574	373	298	382	580	n.d.	n.d.

Segue una tabella con i tempi di calcolo per singolo tile ottenuti dai tempi della tabella precedente divisi per il numero dei tile. È interessante notare che oltre a 4 tile il tempo di elaborazione per tile sia della CPU che della GPU rimane costante. Ad esempio, su windows, a partire da almeno 4 tile, per elaborare un tile della rete SB_NETWORK_TYPE_SDINET0_331 con la GPU RTX 3090 occorre mediamente 1.2 ms con la RTX 3060 2.3 ms e con la RTX 850M 19.7 ms. Diversamente accade per un numero di tile inferiore a 4 ove il tempo per tile aumenta perché l'hardware viene utilizzato in modo meno efficiente rispetto a fargli elaborare in una volta sola molti tile contemporaneamente.

Numero di tile	SB_NETWORK_TYPE_SDINET0_331									
	LINUX UBUNTU 20,04				WINDOWS 10					
	RTX 3090		RTX 3060		RTX 3090		RTX 3060		RTX 850M	
	CPU	GPU	CPU	GPU	CPU	GPU	CPU	GPU	CPU	GPU
1	0,7	5,7	0,7	6,0	1,5	7,1	1,5	7,1	2,2	22,4
4	0,4	1,7	0,5	2,5	0,9	2,2	0,9	2,8	1,3	19,8
9	0,5	1,1	0,7	2,2	1,5	1,5	1,5	2,5	2,1	19,8
16	0,6	1,1	0,7	2,2	1,4	1,3	1,4	2,4	2,0	19,6
25	0,7	1,0	0,8	2,1	1,8	1,3	1,7	2,3	2,7	19,7
36	1,0	1,2	0,9	2,1	1,5	1,2	1,5	2,3	2,1	19,7
64	1,1	1,0	1,1	2,1	1,4	1,2	1,5	2,2	2,0	19,7
144	1,2	1,2	1,2	2,2	1,4	1,2	1,4	2,2		
256	1,1	1,2	1,1	2,2	1,5	1,2	1,5	2,3		

2.2 Deep Cortex

Per questo test è stata utilizzata la soluzione di esempio *deep_cortex_nuts* rilasciata insieme all'**SB-SDK**. Vengono presentati i tempi di calcolo di **Deep Cortex** con la seguente configurazione:

Parametro	Valore
type	Tutte le reti
feature_multiplier	1.0

Di seguito una tabella con i tempi di inferenza in ms per tutte le reti di **Deep Cortex** con differenti GPU.

	LINUX UBUNTU 20.04				WINDOWS 10					
	RTX 3090		RTX 3060		RTX 3090		RTX 3060		RTX 850M	
TIPO DI RETE	**CPU**	**GPU**	**CPU**	**GPU**	**CPU**	**GPU**	**CPU**	**GPU**	**CPU**	**GPU**
SB_NETWORK_TYPE_ICNET0_64	0,3	0,8	0,4	0,8	0,5	1,6	0,5	1,6	0,5	1,6
SB_NETWORK_TYPE_ICNET0_128	0,5	0,9	0,5	0,9	1,0	1,7	1,0	1,8	1,1	1,9
SB_NETWORK_TYPE_EFFICIENTNET_B0	1,2	9,7	1,2	9,9	1,3	10,7	1,4	15,4	1,8	16,3
SB_NETWORK_TYPE_EFFICIENTNET_B1	1,3	13,1	1,3	13,4	1,5	14,7	1,5	20,0	2,0	25,2
SB_NETWORK_TYPE_EFFICIENTNET_B2	1,4	13,5	1,5	13,8	1,6	14,9	1,6	20,2	2,2	32,9

2.3 Analisi dei risultati

I risultati mettono in evidenza che:

- il sistema operativo Linux è sempre più veloce di windows;
- con un basso numero di tile le GPU RTX 3090 e RTX 3060 hanno tempi simili. Poi, all'aumentare del numero di tile aumenta il divario fra le due GPU;
- per **Deep Surface** nel tempo totale dell'inferenza la componente relativa alla CPU è importante tanto quanto quella relativa alla GPU per cui CPU e GPU devono essere bilanciate;
- per **Deep Cortex** il tempo relativo alla CPU è trascurabile rispetto a quello della GPU;

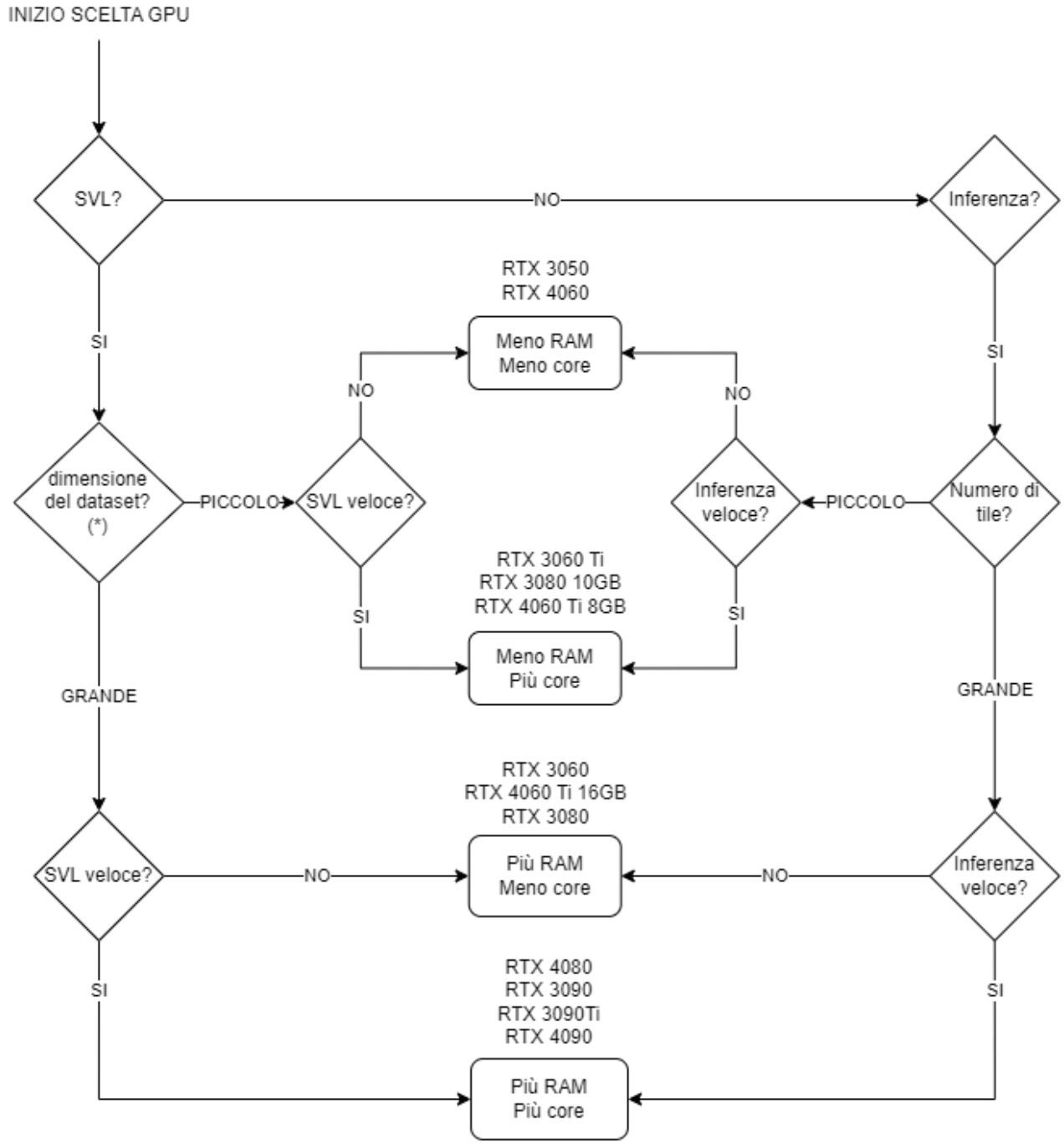
- Per **Deep Surface** il tempo di elaborazione per tile, oltre i 4 tile, rimane abbastanza costante e ciò permette di fare una previsione abbastanza precisa del tempo necessario per elaborare una immagine nel caso multi tile;
- per **Deep Surface** il numero massimo di tile utilizzabile in inferenza è limitato dalla dimensione della RAM della GPU;
- il valore massimo del batch_size utilizzabile in SVL è limitato la dimensione della RAM della GPU.

3 Conclusioni

Nelle conclusioni di quanto si è visto nei capitoli precedenti, e considerando che l'occupazione delle risorse della GPU in SVL e inferenza seguono due logiche differenti, si offrono queste riflessioni che possono essere utili per la scelta della GPU NVidia:

SVL	Solitamente il limite è dato dalla dimensione della memoria della GPU che va a limitare il batch_size massimo il quale va configurato in base alla dimensione del dataset. È più raro che sia richiesto che la SVL sia anche veloce a causa di vincoli applicativi, ma se così fosse sarà necessario scegliere la GPU anche in base al numero di core e non solo alla dimensione della RAM.
Inferenza	Solitamente il limite è dato dal numero di core della GPU che va a limitare il tempo di calcolo. Se, solo per **Deep Surface**, fosse necessario anche lavorare con un numero di tile alto allora si dovrà scegliere la GPU anche in base alla dimensione della RAM.
Deep Surface	Nel tempo totale dell'inferenza la componente relativa alla CPU è importante tanto quanto quella relativa alla GPU per cui CPU e GPU devono essere bilanciate, soprattutto per quanto riguarda i costi, ossia non è efficiente spendere molto per una GPU e abbinarla ad un processore di basse prestazioni e viceversa.
Deep Cortex	Il tempo di inferenza è quasi totalmente occupato dalla GPU per cui è meglio sbilanciare il sistema verso la GPU a scapito della CPU.
Multi threading	Un altro fattore da considerare è se l'applicativo avrà più **thread** che dovranno elaborare le immagini e quindi più thread che chiameranno contemporaneamente la funzione sb_project_detection. In questo caso le risorse della GPU, essendo condivise dai thread, dovranno essere dimensionate di conseguenza.
Sistema operativo	Linux permettere di avere dei tempi di inferenza sulla GPU inferiori a windows per cui in applicazioni che, per via di tempi ciclo molto ridotti, richiedono anche tempi in inferenza in runtime particolarmente bassi, si prenda in considerazione la possibilità di utilizzare Linux.
RAM della GPU	Per valutare approssimativamente le dimensioni della RAM della GPU necessarie alla vostra applicazione dovete tenere in considerazioni oltre che il tipo di rete utilizzata soprattutto quante immagini simultaneamente dovranno essere elaborate seguendo le indicazioni date nel capitolo relativo in cui ci sono della tabelle che possono fare da guida.
Numero di core della GPU	Per valutare il numero di core della GPU necessario alla vostra applicazione dovete tenere in considerazioni il tempo di inferenza in runtime richiesto per elaborare le immagini seguendo le indicazioni date nel capitolo relativo in cui ci sono della tabelle che possono fare da guida. Si consideri anche che con un numero basso di immagini o tile da elaborare contemporaneamente le GPU RTX 3090 e RTX 3060 hanno tempi simili pur avendo un numero di core molto differente, poi, all'aumentare del numero aumenta il divario fra le due GPU.
Un buon compromesso	per progetti non troppo esigenti, è la NVidia GeForce RTX 3060 che ha un costo basso, ben 12GB di RAM, 3584 core e in aggiunta ha anche delle dimensioni piuttosto ridotte, infatti occupa solo 2 slot PCI e, a seconda dei modelli, può essere lunga solo 18 cm.

Viene proposto un diagramma di flusso molto semplificato che può guidare nella scelta della GPU.



(*) DEEP SURFACE: la dimensione del dataset va calcolato in base al numero di tile che vengono utilizzati dalla SVL, ossia i tile che hanno ROI di analisi.
DEEP CORTEX: corrisponde al numero delle immagini

Altre informazioni si possono trovare nella documentazione dell'**SB-SDK** al link https://library.fabervision.com

FABERVISION Srl
Via Tonale 9
24061 Albano S. Alessandro (BG)
ITALY

Phone +39 035 19752207
Vat IT03929430985
E-Mail info@fabervision.com
Website www.fabervision.com