GEOXPERIENCE ACADEMY

Tutorial

Deep Learning per il riconoscimento automatico di oggetti sulle immagini

planetek italia

Indice

Introduzione

Gli algoritmi (Spatial Models) costruiti in questo tutorial permettono di individuare automaticamente le chiome delle palme in un'immagine utilizzando un algoritmo di Deep Learning basato sulla CNN (Convolution Neural Network) disponibile in ERDAS IMAGINE.
I principi di base e la metodologia presentata nel tutorial possono essere applicati a qualsiasi genere di oggetto, per cui lo stesso algoritmo costruito potrà essere utilizzato per altre tipologie di oggetti.

L'output di un algoritmo di object detection di ERDAS IMAGINE è uno shapefile con una serie di poligoni (bounding box) che delimitano gli oggetti individuati nell'immagine; la procedura prevede tre passaggi fondamentali:

- Raccolta dei dati di training;
- Addestramento dell'algoritmo di machine learning (Initialization);
- Classificazione di un'immagine per il riconoscimento di oggetti (Detection).

Dati utilizzati per il tutorial

I dati utilizzati per il tutorial sono stati forniti ad Hexagon da Open Aerial Map (https://openaerialmap.org/)

Nel file zip allegato al tutorial, sono disponibili i dati specificati di seguito. Per comodità sono resi disponibili, oltre ai dati di input, anche i risultati; chi esegue l'esercizio può quindi scegliere se eseguire tutti i passaggi in completa autonomia o saltare qualche passaggio utilizzando i risultati già presenti.

- Nella cartella **"01_Training"** sono contenute le immagini utilizzate per la raccolta dei dati di training;
- Nella cartella **"02_Training_ready"** sono contenute le stesse immagini presenti nella cartella 01; in questo caso le immagini sono già processate e pronte per poter addestrare l'algoritmo (dopo aver letto qualche altra pagina del tutorial sarà più chiara la differenza tra la cartella 01 e la cartella 02);
- Nella cartella **"03_Addestramento"** è presente l'algoritmo, in formato spatial modeler (*.gmdx) che consente di addestrare la macchina; la cartella contiene anche una macchina già addestrata con i dati di training presenti nella cartella 02 (palmtrees_mi_25000.miz);
- Nella cartella **"04_Classificazione"** sono presenti: l'algoritmo per eseguire l'object detection "Detect Palm trees from images using DL.gmdx"; l'immagine sulla quale ricercare gli oggetti (presente nella cartella "Input"); un esempio di risultato (shapefile presente nella cartella "Output") ottenuto con i dati di training presenti nella cartella 02 e l'algoritmo presente nella cartella 03.

Fase 1- Raccolta di dati di Training

Per addestrare un classificatore machine learning, è necessario disporre di immagini di esempio che raffigurano il tipo di oggetto che si desidera trovare in altre immagini. La risoluzione spettrale e spaziale delle immagini di training deve essere coerente con il tipo di immagini su cui si desidera eseguire il rilevamento.

Il metodo più semplice per raccogliere *chip* di training adeguati utilizzando ERDAS IMAGINE prevede l'utilizzo degli strumenti disponibili nel layout di ERDAS dedicato al Machine Learning.

- In ERDAS IMAGINE Selezionare **File> Layout> Machine Learning Layout**

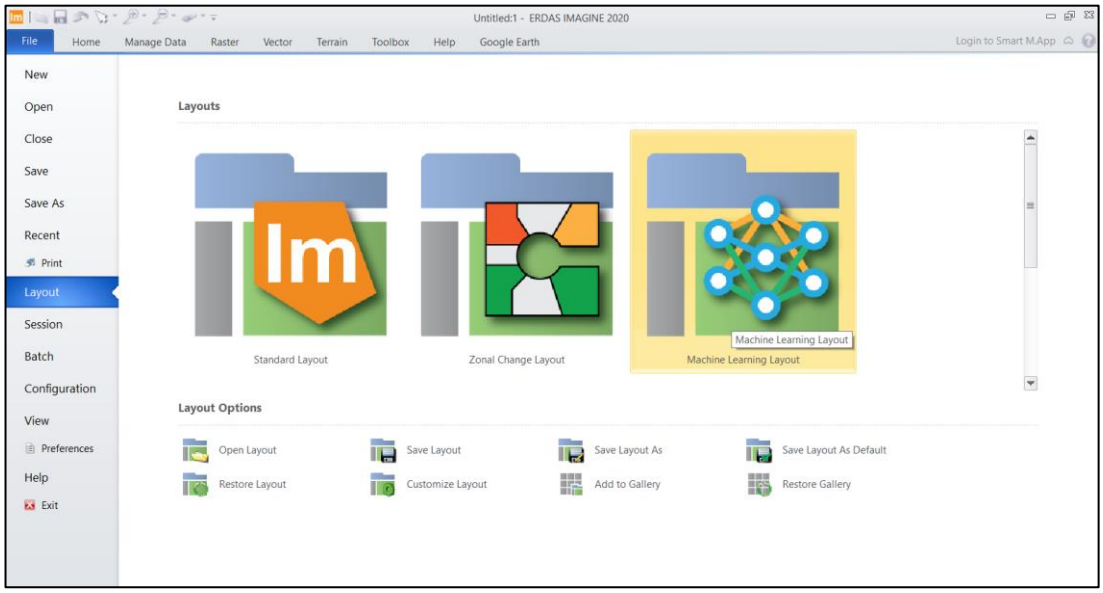


FIG.1 SELEZIONE DEL MACHINE LEARNING LAYOUT

- Nel layout di Machine Learning, attivare la scheda **Train**;
- Aprire l'immagine "palm_tile0_0.img" presente nella cartella "01_Training" e visualizzarla nella finestra 2D (2D view > Open Raster Layer oppure Drag & Drop);
- Nel menu a discesa "Type" (in altro a sinistra) selezionare "**Objects footprints only**"

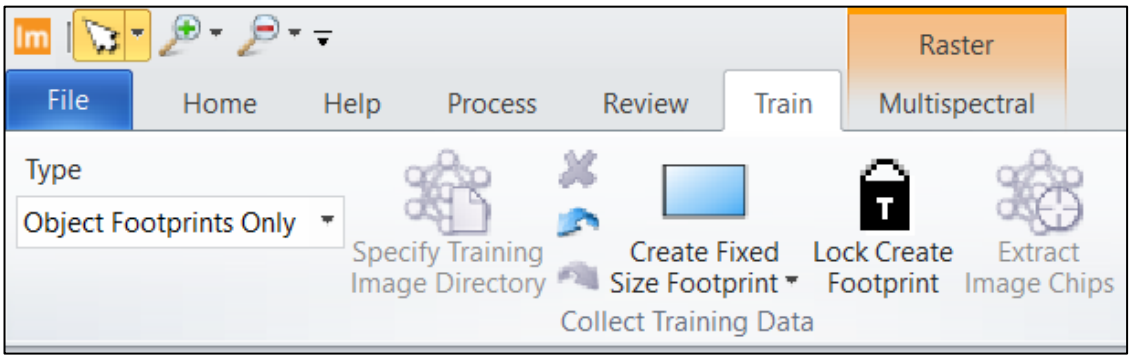


FIG.2 UTILIZZO DEL TAB "TRAIN" – PARTE 1

- Nella tabella "**Training Classes**" scrivere "Palm" nella colonna "Class" e scegliere il colore rosso nella colonna "Color";

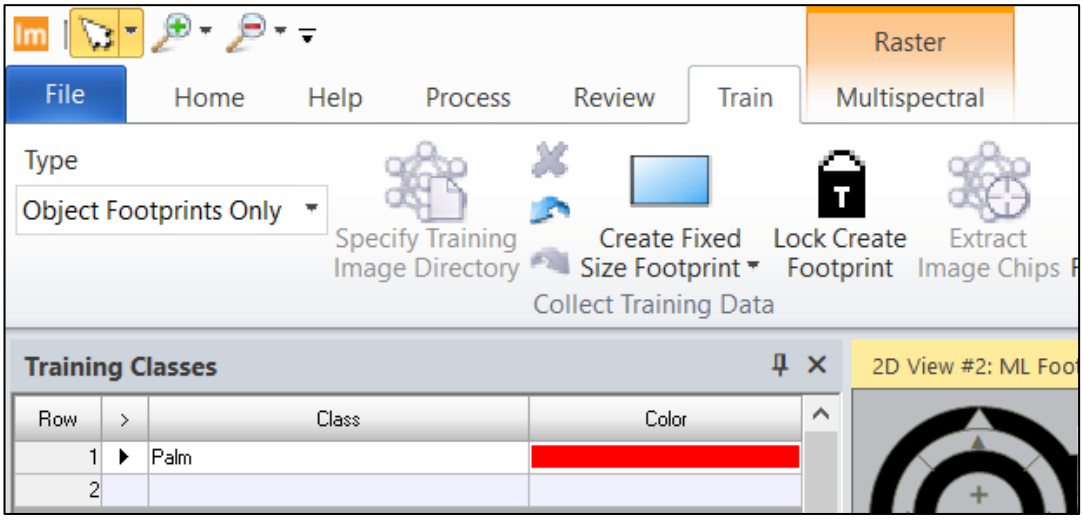


FIG.3 UTILIZZO DEL TAB "TRAIN" – PARTE 2

- Selezionare l'opzione **"Create Variable Size Footprint"** e attivare l'opzione "Lock Create Footprint";

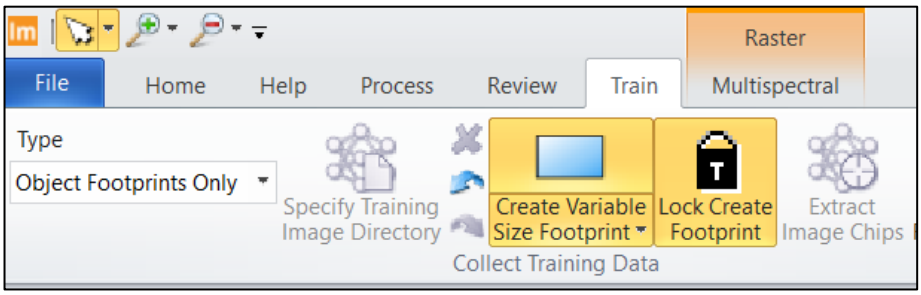


FIG.4 UTILIZZO DEL TAB "TRAIN" – PARTE 3

- A questo punto è possibile digitalizzare nell'immagine dei bounding box attorno alle chiome delle palme (vedi Fig.5). Quando sono state mappate tutte le palme chiaramente e interamente visibili nell'immagine (non porzioni di palme), cliccare sul tasto "**Save Footprints**".

FIG.5 ESEMPIO DI DIGITALIZZAZIONE MANUALE DEI DATI DI TRAINING

- Cliccare con il tasto destro su **2D view > Clear View** per eliminare l'immagine lavorata.
- Aprire l'immagine successiva ("palm_tile_0_1.img"), sempre presente nella cartella 01, e ripetere la procedura precedente;
- Dopo aver lavorato tutte le immagini presenti nella cartella 01 (per un buon addestramento dell'algoritmo possono essere necessarie centinaia, o anche migliaia, immagini di esempio in

funzione delle caratteristiche dell'oggetto da riconoscere) è possibile passare al layout standard di ERDAS, cliccando su **File > Layout > Standard Layout**.

Nota1. Chi vuole esercitarsi nella digitalizzazione può proseguire nella digitalizzazione dei footprints fino all'ultima immagine presente nella cartella 01, immagine "palm_tile_9_9.img"); chi volesse saltare questo passaggio ripetitivo può utilizzare per il proseguo del tutorial la cartella **"02_Training_ready"** in cui il lavoro di digitalizzazione è stato già completato.

Nota2. Quando si clicca su **"Save Footprints"** il software produce, nella stessa cartella in cui sono contenute le immagini (in questo caso "**01_Training**") dei file in formato xml, aventi lo stesso nome dell'immagine lavorata (ad esempio "palm_tile0_0.xml"; in questi file xml sono salvate le informazioni relative ai *bounding box* individuati sull'immagine dall'operatore. È importante, per il proseguo del tutorial, non spostare questi file xml dalla cartella, e non cancellarli.

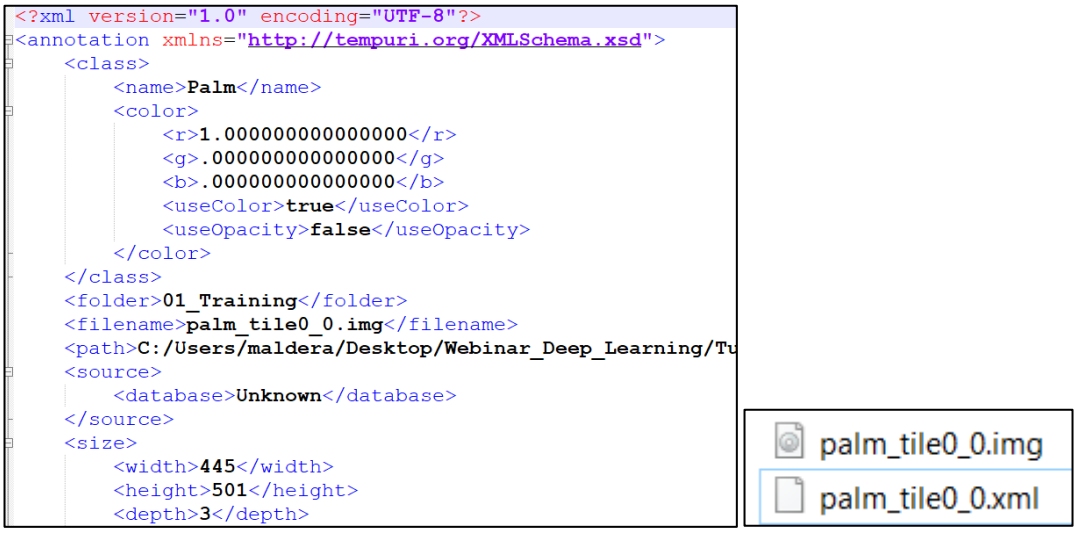


FIG.6 VISUALIZZAZIONE DEL CONTENUTO DELL'XML CHE CONTIENE LE INFORMAZIONI SUI FOOTPRINT DEGLI OGGETTI

Fase 2 - Addestramento dell'algoritmo di machine learning (Initializing)

Come anticipato, un algoritmo di machine learning (*machine intellect*) viene addestrato utilizzando esempi di oggetti (in questo caso palme) raccolti su altre immagini.

Per eseguire questo task è necessario utilizzare l'algoritmo **"Train a Machine Intellect to Detect Palm Trees.gmdx"** presente nella cartella **"03_Addestramento"**

- In ERDAS IMAGINE cliccare su **File > New > Spatial Modeler Editor** per aprire l'ambiente di lavoro;
- Posizionarsi con il mouse sulla parte bianca a quadretti e cliccare con il tasto destro e poi su **Open.** Selezionare il file "Train a Machine Intellect to Detect Palm Trees.gmdx".
- Verrà visualizzato l'algoritmo Spatial Model per l'addestramento (vedi Fig.7);

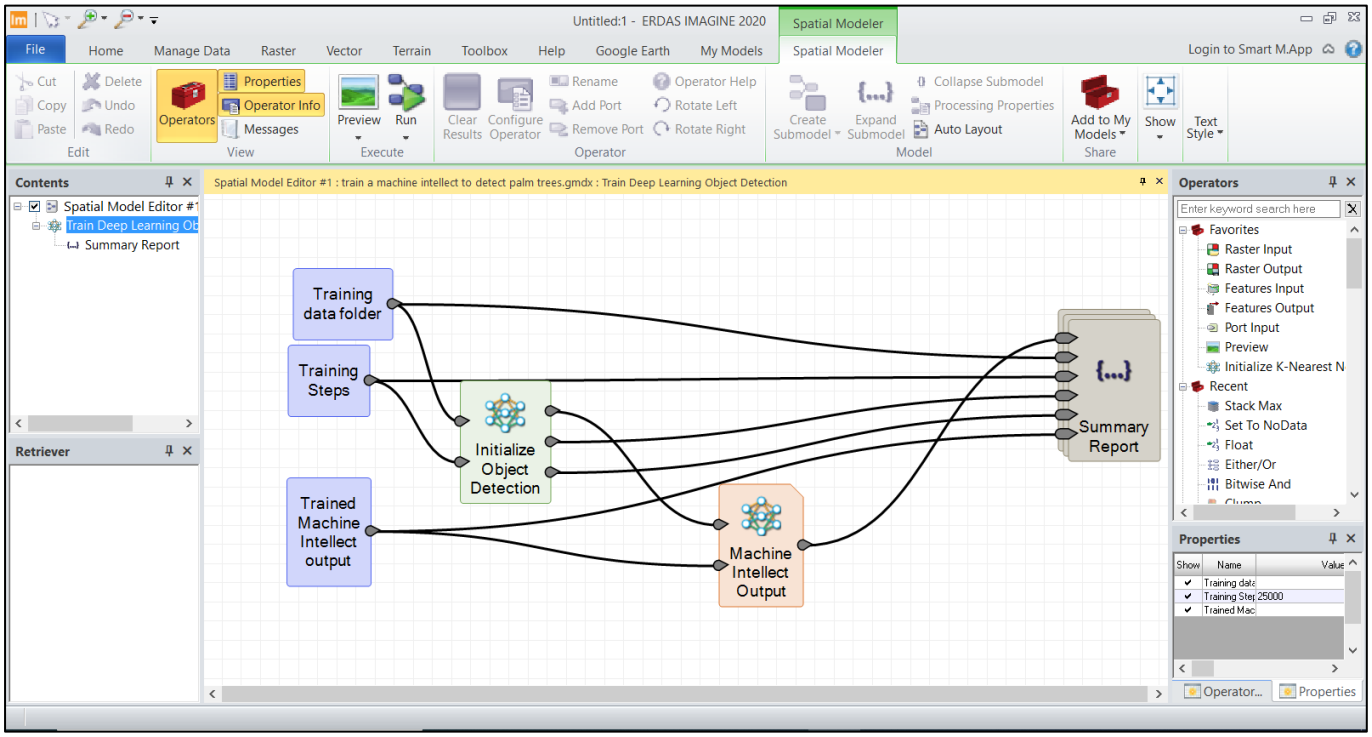


FIG.7 VISUALIZZAZIONE DELL'ALGORITMO PER L'ADDESTRAMENTO DEL MACHINE INTELLECT

- Per eseguire l'algoritmo è necessario cliccare sul tasto Run;
- Si aprirà la seguente finestra (Fig.8):

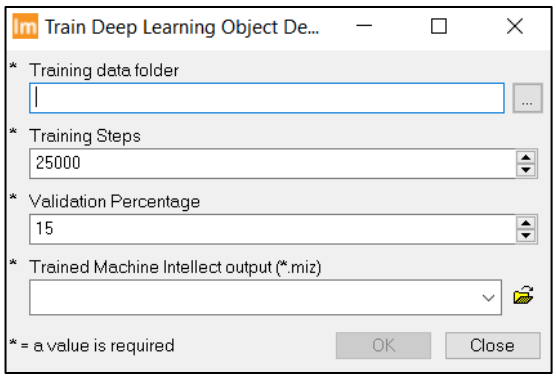


FIG.8 FINESTRA PER L'INSERIMENTO DEGLI INPUT E DEI PARAMETRI PER ESEGUIRE IL TRAINING

Per eseguire il training è necessario definire (tramite la finestra):

- **Training Data Folder**: la cartella che contiene i dati di training (quindi le immagini di esempio con xml associati). Per il tutorial va selezionata la cartella **"02_Training_ready"** (o la cartella 01 se avete completato la fase di raccolta di dati di training in autonomia);

- **Training Steps**: Il numero di volte in cui la rete neurale (l'algoritmo) apprende dai dati di addestramento. La rete deve elaborare i dati di training almeno una volta. Affinché la rete funzioni correttamente, deve elaborare più volte i dati di training. Con l'aumentare dei Training Steps, aumenterà anche il tempo di elaborazione. Il valore minimo accettabile è 1. Per il tutorial inseriamo un valore pari a 5.

- **Validation Percentage:** Una piccola percentuale dei dati di training (solitamente tra il 10 e il 20%) viene messa da parte per eseguire una validazione dell'algoritmo. Se ad esempio il dato di training è costituito da 1000 immagini e il valore di *validation percentage* è pari a 10, 100 immagini non verranno utilizzate per il training ma verranno utilizzate per la validazione. Per il tutorial è possibile utilizzare il valore di default, 15.

- **Trained Machine Intellect Output (*miz):** qui va specificato il nome e la posizione in cui salvare l'algoritmo addestrato. Per il tutorial ci posizioniamo nella cartella "03_Addestramento" e denominiamo il file **"palmtrees_mi_5_tutorial.miz"**

Una volta inseriti tutti gli input, cliccare su **OK.** L'algoritmo verrà eseguito, l'operatore "Initialize Object Detection" diventerà verde.

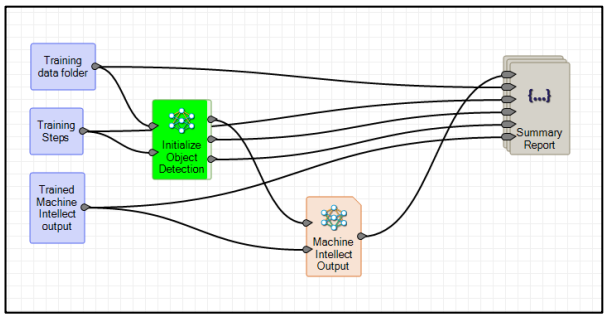


FIG.9 ALGORITMO DI TRAINING IN ESECUZIONE

Il processo può essere monitorato dalla barra di avanzamento presente in basso a destra. Inoltre quando il processing termina compaiono delle spunte verdi su tutti gli operatori (elementi) dell'algoritmo.

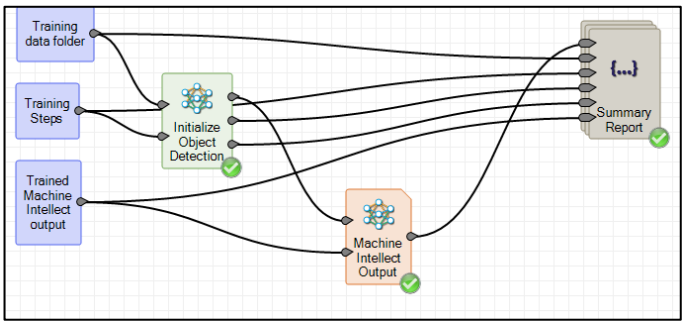


FIG.10 ALGORITMO DI TRAINING, PROCESSING ESEGUITO

Quando il processo sarà terminato nella cartella "03_Addestramento" troverete anche un file di testo (formato *.txt) denominato "palmtrees_mi_5_tutorial.txt" (lo stesso nome del file .miz), che denomineremo d'ora in poi report.

Ecco un esempio del contenuto del report:

c:\users\maldera\desktop\webinar_deep_learning\tutorial\03_addestramento\palmtrees_mi_5_tutorial.miz

Training Data:

c:/users/maldera/desktop/webinar_deep_learning/tutorial/02_training_ready/

Training Steps

5

Learning Accuracy:

0.0081797324156597587

Validation Accuracy:

0.0092120559875441856

Contiene quindi nell'ordine:

- **Path dell'outoput** e cioè l'algoritmo addestrato (file formato .miz);
- **Cartella** contenente i dati di training;
- **Numero di Training Steps**;
- **Validation Accuracy**: Una metrica che fornisce una misura dell'accuratezza dell'algoritmo relativamente al sottoinsieme dei dati di training utilizzato per la validazione. Se, ad esempio, vengono utilizzate 1000 immagini per addestrare l'algoritmo, da queste 1000 immagini viene estratto un sottoinsieme di 100 immagini per valutare la *validation accuracy* (quindi abbiamo usato un *validation percentage* di 10); queste 100 immagini vengono riclassificate:la classe degli oggetti è nota, l'algoritmo li classifica e confronta la classe ottenuta dalla classificazione con la classe effettiva. In questo modo si calcola la metrica validation accuracy. Il parametro può assumere valori compresi tra 0 e 1, con 1 che indica l'assenza di errori nella classificazione.
- **Learning Accuracy**: Una metrica che fornisce una misura dell'accuratezza dell'algoritmo relativamente ai dati effettivamente utilizzati per l'addestramento. Se, ad esempio, la cartella di training contiene 1000 immagini, da queste 1000 immagini viene estratto un sottoinsieme di 100 immagini per valutare la *validation accuracy* (quindi abbiamo usato un validation percentage di 10). Le altre 900 immagini vengono effettivamente utilizzate per addestrare l'algoritmo. Una volta addestrato l'algoritmo queste 900 immagini vengono riclassificate: la classe degli oggetti è nota, l'algoritmo li classifica e confronta la classe ottenuta dalla classificazione con la classe effettiva. In questo modo si calcola la metrica *learning accuracy*. Il parametro può assumere valori compresi tra 0 e 1, con 1 che indica l'assenza di errori nella classificazione.

Questo report permette quindi di valutare l'efficacia del training effettuato e consente di capire se occorre agire nuovamente sui dati di training per migliorare *learning accuracy* e *validation accuracy*. Nel caso del report generato, i valori delle due metriche sono:

Learning Accuracy: 0.0081797324156597587

Validation Accuracy: 0.0092120559875441856

Sono molto vicini allo zero, occorre aumentare il valore di *training steps* per permettere all'algoritmo di "imparare" meglio a distinguere le varie classi. Valgono le seguenti considerazioni:

- Il valore dei *training steps* può essere aumentato per tentativi, ad esempio di un ordine di grandezza alla volta (10, 100, 1000, ecc.), valutando al termine dell'addestramento le metriche learning e *validation accuracy*.
- Un buon addestramento si ha quando i due valori *learning accuracy* e *validation accuracy* sono superiori a 0.75 e sono abbastanza vicini come valore;
- Un *learning accuracy* basso evidenzia uno stato di *underfitting* (cioè il modello non riesce a classificare correttamente neanche i dati con cui è stato addestrato) potrebbe significare che: che le immagini a disposizione nel dato di training non sono sufficienti, sono stati commessi

degli errori nella raccolta dei dati di training o l'algoritmo utilizzato non è opportuno per quel tipo di immagini;
- Se passando da 100 a 1000 steps le due metriche di accuracy non migliorano di molto e il valore di *validation accuracy* si allontana dal *learning accuracy* allora è preferibile rimanere a 100 training steps;
- Se si ottiene un valore di *learning accuracy* molto alto e un valore di *validation accuracy* molto basso il modello è andato in *overfitting* (cioè funziona bene con i dati di addestramento, ma non con i dati di validazione, perché non è in grado di generalizzare il modello con dati non osservati); in questo caso potrebbe essere utile aumentare i dati di training e/o ridurre il numero di training steps;

Per lo svolgimento dell'esercizio è sufficiente eseguire nuovamente il modello di training l'algoritmo "Train a Machine Intellect to Detect Palm Trees.gmdx", cliccare sul tasto **Run** e aumentando il numero di training steps a 25000. Nella finestra che si apre bisogna inserire quindi i seguenti input:

- **Training Data Folder**: la cartella che contiene i dati di training (quindi le immagini di esempio con xml associati). Per il tutorial va selezionata la cartella "02_Training_ready" (o la cartella 01 se avete completato la fase di raccolta di dati di training in autonomia);

- **Training Steps**: 25000

- **Validation Percentage**: 15.

- **Trained Machine Intellect Output (*miz):** qui va specificato il nome e la posizione in cui salvare l'algoritmo addestrato. Per il tutorial ci posizioniamo nella cartella "03_Addestramento" e denominiamo il file "palmtrees_mi_25000_tutorial.miz"

Avendo impostato un valore pari a 25000 per i training steps il processing questa volta potrebbe durare anche molto (numerose ore), in base alle caratteristiche dell'hardware (CPU) che si utilizza. Per poter accelerare i tempi è molto utile processare i dati mediante la GPU, per questo tipo di processi in ERDAS è necessario disporre di schede grafica NVIDIA, con memoria interna almeno pari a 6GB. Per configurare ERDAS per poter utilizzare la GPU, consultare la guida in appendice. Chi volesse saltare questo passaggio e continuare il file può utilizzare il file disponibile "palmtrees_mi_25000.miz" già disponibile e presente nella cartella "03_Addestramento".

Una volta che è stato ottenuto un *machine intellect* (algoritmo, file.miz) addestrato che ha un buon valore di *learning accuracy* e un buon valore di *validation accuracy*, è possibile utilizzarlo per la fase di classificazione, in cui verrà chiesto al modello di classificare nuovi dati.

Fase 3 - Classificazione di nuove immagini

- Chiudere lo Spatial Modeler Editor, se è ancora aperto con l'algoritmo precedente;
- Cliccare su **File > New > Spatial Modeler Editor** per aprire un nuovo ambiente di editor vuoto:

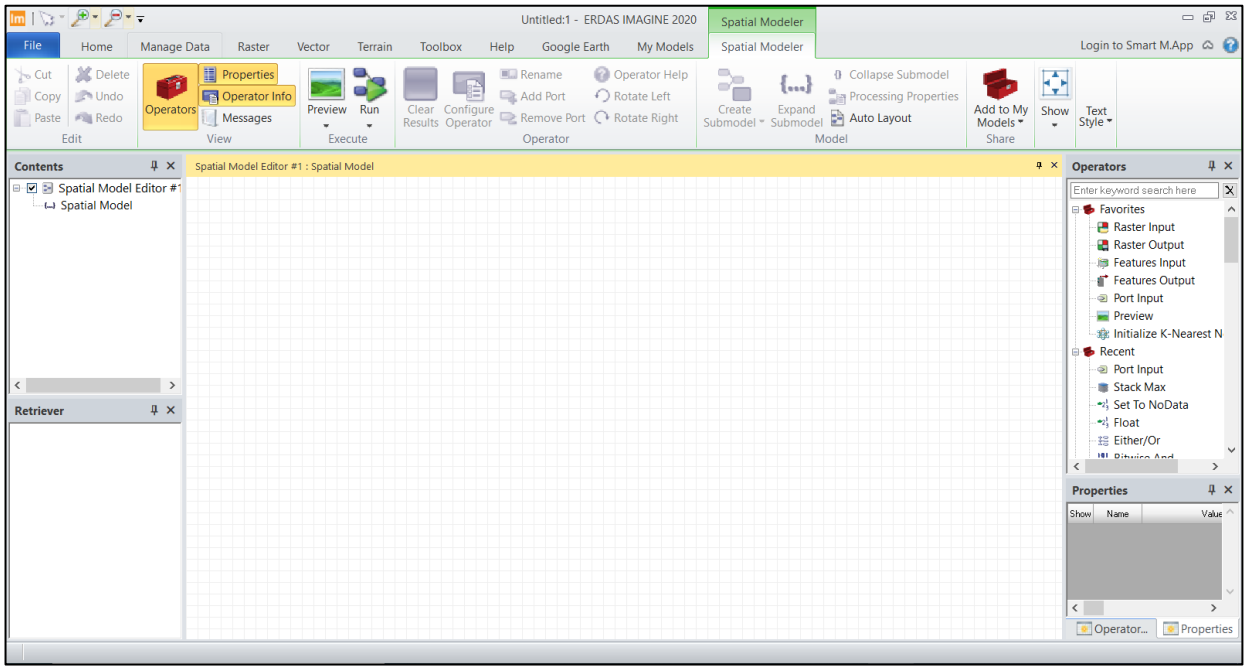


FIG.11 SPATIAL MODEL EDITOR

- Cliccare con il tasto destro nello spazio bianco a quadretti e quindi su Open;
- Selezionare il modello **"Detect Palm trees from images using DL.gmdx"** presente nella cartella "04_Classificazione";
- Si aprirà il modello che esegue la classificazione:

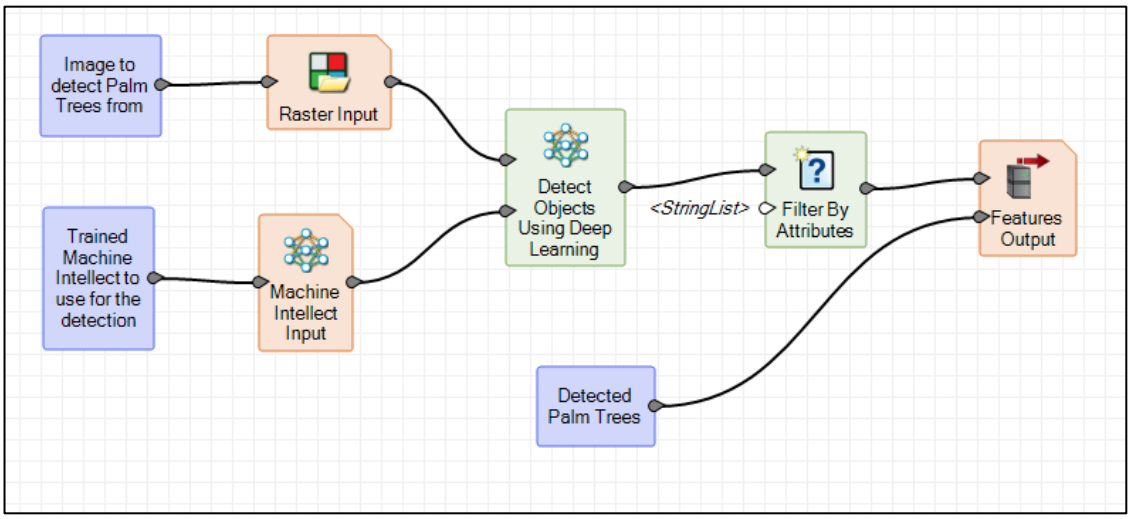


FIG.12 ALGORITMO DI OBJECT DETECTION

- Per eseguire l'algoritmo è necessario cliccare sul tasto Run;
- Si aprirà la seguente finestra:

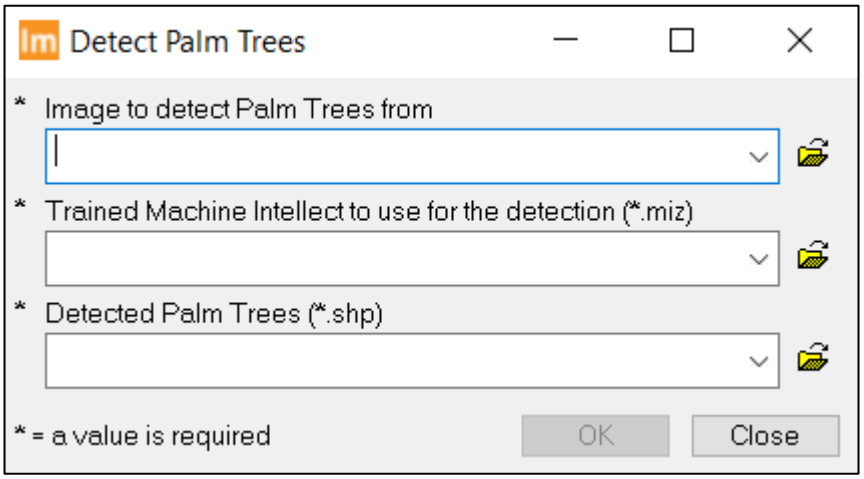


FIG.13 ALGORITMO DI OBJECT DETECTION – FINESTRA DEGLI INPUT

Per eseguire la classificazione occorre definire, quindi (tramite la finestra):

- **Images to detect Palm Trees From**: immagine da classificare sulla quale verranno individuati gli oggetti (palme in questo caso), per il tutorial utilizzare l'immagine "palm_trees_tile2_2.img" presente nella cartella "04_Classificazione\Input";

- **Trained Machine Intellect to use for the detection**: si tratta dell'algoritmo addestrato, per l'esecuzione del tutorial utilizzare il file "palmtrees_mi_25000_tutorial.miz" che è stato generato nel passaggio precedente e che dovrebbe essere presente nella cartella "03_Addestramento" (per chi ha seguito le indicazioni del tutorial). In alternativa, chi avesse saltato il passaggio relativo al training, può utilizzare il file "palmtrees_mi_25000.miz" presente sempre nella stessa cartella;

- **Detect Palm Trees (*.shp):** qui va definito il nome e la posizione del file di output, per il tutorial chiamare il file "detected_palm_trees_tile_25000_tutorial" e salvarlo nella cartella "04_Classificazione\Output".

Anche questo passaggio potrebbe durare diverse ore, in funzione delle capacità di calcolo del pc utilizzato. Chi volesse direttamente guardare il risultato trova nella cartella "04_Classificazione\Output" il file "detected_palm_trees_tile_25000.shp" e può utilizzare questo per il passaggio successivo.

Fase 4 - Analisi dei Risultati

- Chiudere lo **Spatial Modeler Editor**, se è ancora aperto con l'algoritmo precedente;
- Cliccare su **File > New > 2D View**;
- Aprire l'immagine "**palm_trees_tile2_2**" presente nella cartella"04_Classificazione\Input" e lo shapefile "detected_palm_trees_tile_25000_tutorial.shp" (o in alternativa "detected_palm_trees_tile_25000.shp") in sovrapposizione.
- Navigare nella finestra 2D per analizzare i risultati;

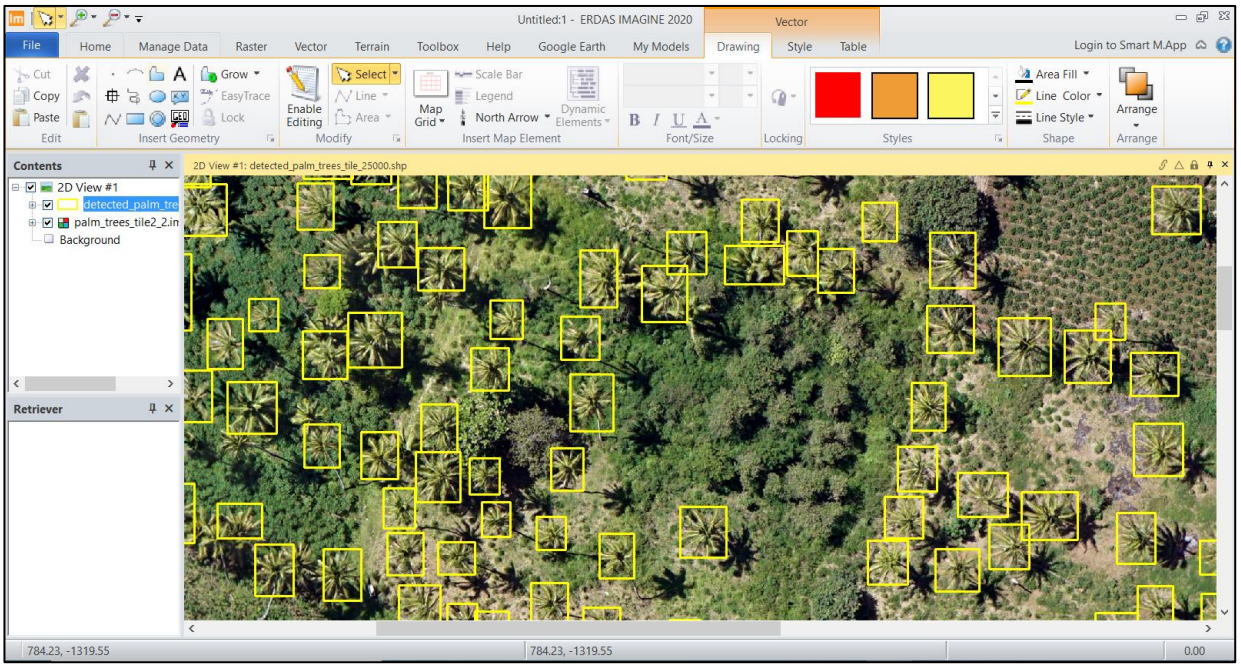


FIG.14 ANALISI DEI RISULTATI IN ERDAS IMAGINE

- Nella finestra "Contents" cliccare con il tasto destro sullo shapefile e quindi su "Display Attribute Table".

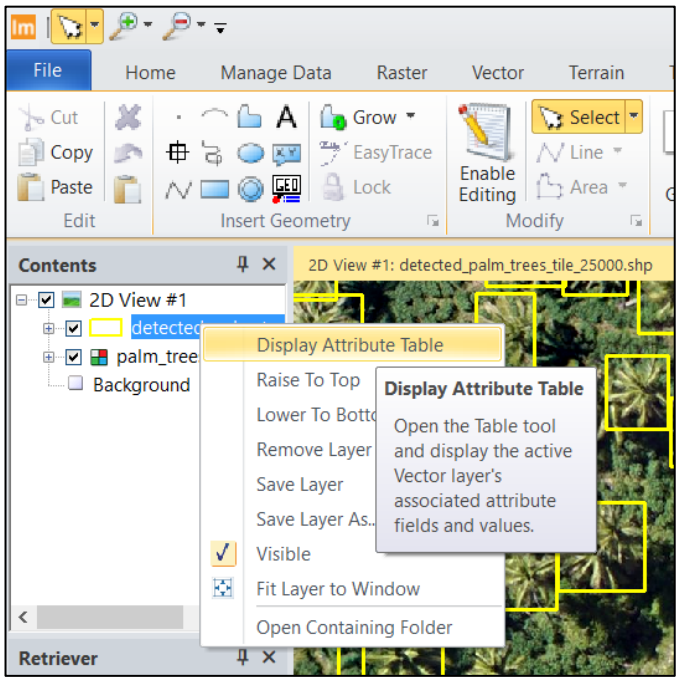


FIG.15 COMANDO DISPLAY ATTRIBUTE TABLE

- Verranno visualizzati gli attributi dello shapefile, in una tabella in basso. La tabella conterrà la colonna “CL_Class” con il nome della classe e la colonna “CL_Prob” con il valore della probabilità che ha quell’oggetto di appartenere a quella classe.

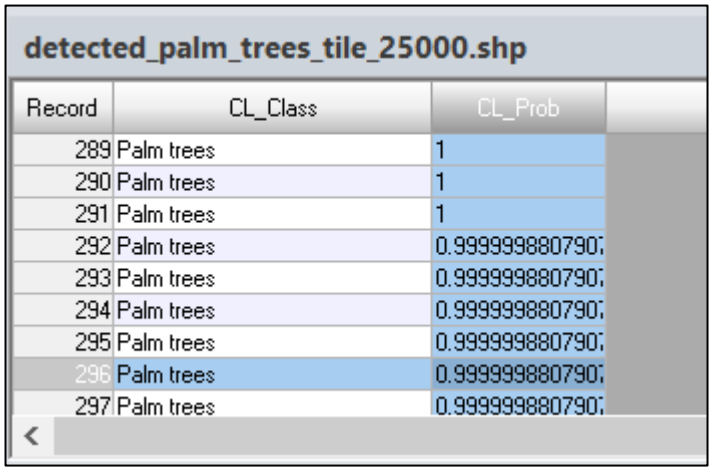
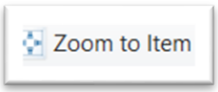


FIG.16 TABELLA DEGLI ATTRIBUTI

- Selezionando un'oggetto in mappa verrà selezionata la riga corrispondente in tabella. Per fare il contrario, cioè selezionare una riga in tabella e zoomare sull'oggetto in mappa bisogna prima selezionare la riga corrispondente e poi cliccare sul tasto "zoom to item", in alto nel tab "Table";



Analizzando i risultati ci si può rendere conto che ci sono degli oggetti che sono stati mappate erroneamente come palme (errore di commissione), ad esempio:

FIG.17 ESEMPIO DI ERRORE DI COMMISSIONE

Altre volte può capitare che una delle palme (o comunque degli oggetti che si vuole mappare) non venga mappata (errore di omissione)

FIG.18 ESEMPIO DI ERRORE DI OMISSIONE

Alcuni errori di omissione o commissione è normale che ci siano quando si processano immagini geospaziali tramite algoritmi di deep learning. Le cause sono da ricercare nella presenza di ombre o di altri oggetti vicini a quelli che si vuole cercare, nella eccessiva vicinanza di più oggetti simili, nelle caratteristiche spettrali particolari della scena, ecc. Bisogna inoltre considerare che gli output di questi algoritmi rappresentano un "supporto alle decisioni" e non possono sostituire in toto il lavoro di un fotointerprete. Questi errori possono essere quantificati e devono essere necessariamente inferiori a determinate soglie se vogliamo che l'algoritmo di classificazione automatica che stiamo utilizzando sia effettivamente utile e ci permetta di lavorare in tempi più rapidi rispetto ad un processo tradizionale di digitalizzazione mediante fotointerpretazione. Costruendo un buon dataset e addestrando correttamente l'algoritmo, si avrà a disposizione uno strumento che permette di processare le immagini in modo massivo, risparmiando tante ore o giornate di lavoro.

FIG.19 VISUALIZZAZIONE DEI RISULTATI

Appendice - Configurazione della scheda grafica

Per poter processare i dati mediante la GPU è necessario disporre di schede grafica NVIDIA, con memoria interna almeno pari a 6GB (ad esempio schede della serie Quadro o GeForce, per maggiori informazioni consultare i requisiti di sistema di ERDAS IMAGINE).

- In **ERDAS IMAGINE** cliccare su **File > Configuration > Configure OpenCL;**
- Si aprirà la finestra **Configure OpenCL Device Priority,** in questa finestra occorre assicurarsi che la scheda grafica che si vuole utilizzare sia selezionata e in cima alla lista (se ci sono più schede grafiche nel pc);
- Cliccare su **OK** e chiudere la finestra;

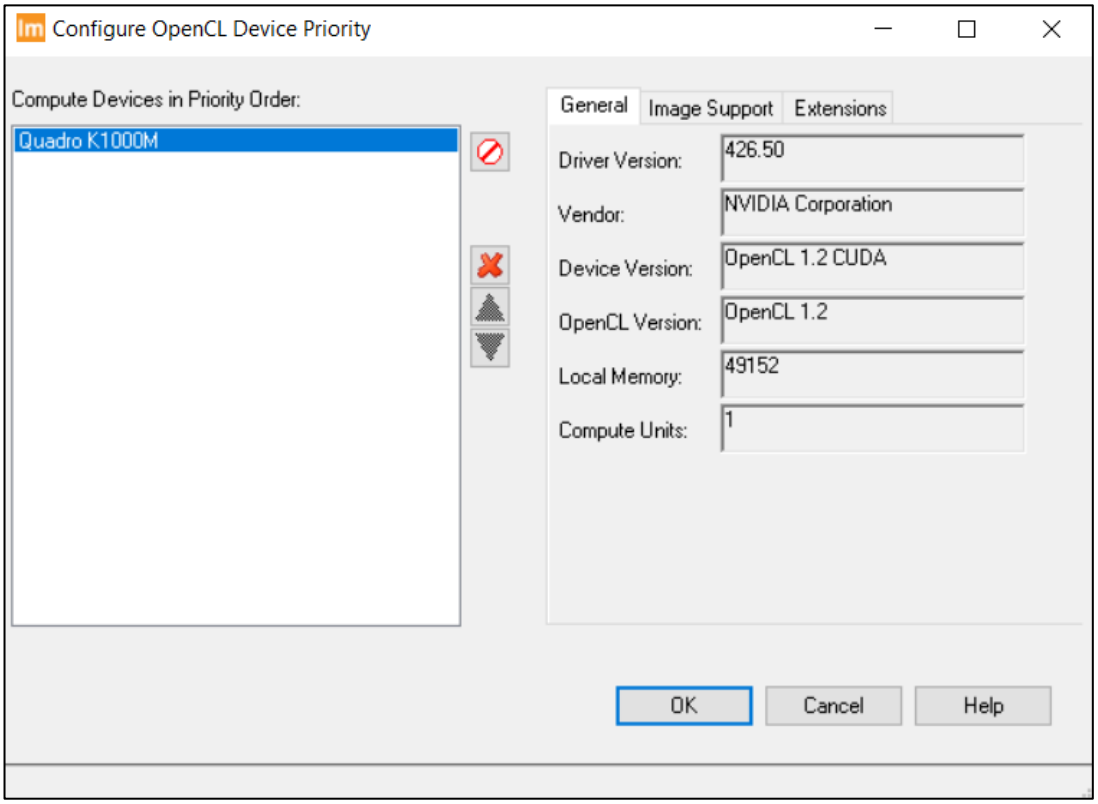


FIG.20 SELEZIONE DELLA SCHEDA GRAFICA DA UTILIZZARE

- Cliccare su **File > Preferences**
- Si aprirà la finestra "**Preference Editor**", nel box in alto a destra scrivere "GPU"

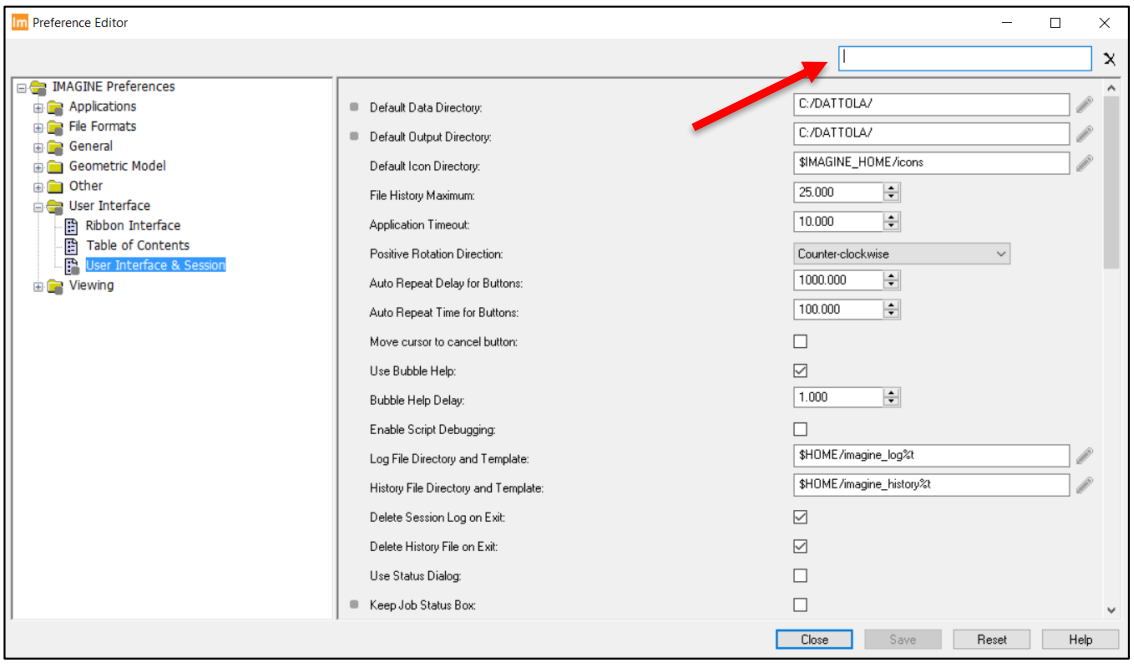


FIG.21 ABILITAZIONE DEL PROCESSING MEDIANTE GPU

- Dopo aver scritto “GPU”, la finestra si modificherà e compariranno solo le seguenti opzioni:

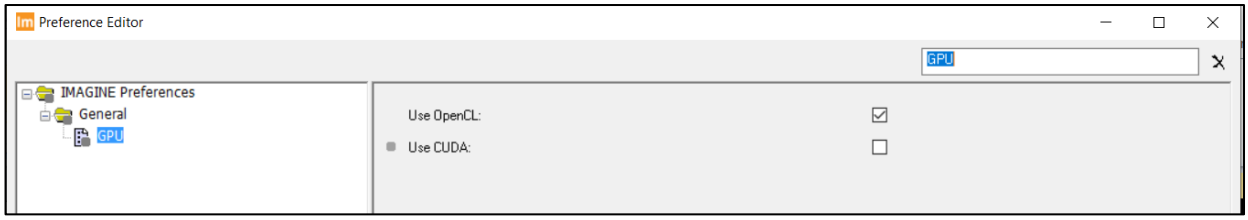


FIG.22 ABILITAZIONE DEL PROCESSING MEDIANTE GPU

- Bisogna selezionare l’opzione **“Use CUDA”**, in seguito cliccare su **Save** e **Close** (in basso a destra);
- Una volta chiusa la finestra **Preference Editor** occorre riavviare ERDAS IMAGINE.

Chi è Hexagon Geospatial

Hexagon è un'azienda attiva nel settore dell'alta tecnologia e in particolare nelle soluzioni per la raccolta e l'analisi dei dati geografici. La sede centrale è in Svezia, ma la multinazionale è attiva in oltre 50 Paesi con circa 20mila dipendenti, generando un fatturato di 3,9 miliardi di euro.

In Italia Hexagon è presente con cinque delle sue otto divisioni; sempre nel nostro Paese ha sede una delle società del Gruppo, IDS Georadar, che produce radar in grado di penetrare la crosta terrestre, utilizzati in diversi settori, tra cui quello delle utilities, della protezione civile e della difesa.

In estrema sintesi, l'attività di Hexagon si può riassumere in due filoni principali: le soluzioni e le tecnologie che servono a raccogliere dai grezzi e quelle che permettono di trasformare questi dati in informazioni utili per prendere decisioni. Molte delle soluzioni impiegano algoritmi di Intelligenza Artificiale e consentono di creare sistemi che vengono chiamati Autonomous Connected Ecosystem (sistemi autonomi connessi). Tra i segmenti più importanti di attività di Hexagon ci sono, ad esempio, telecamere intelligenti in grado di distinguere oggetti e persone e poi sistemi di comando e controllo utilizzati ad esempio dalla pubblica sicurezza.

La divisione Hexagon Geospatial, in particolare, è leader mondiale nel remote sensing, nel Gis e nel situation awareness in tempo reale.

Per saperne di più www.hexagongeospatial.com.

Chi è Planetek Italia

Planetek Italia è una PMI fondata nel 1994 che impiega oltre 60 uomini e donne, appassionate e competenti in Geomatica, scienze della Terra e software per le missioni spaziali. Forniamo soluzioni in grado di usare al meglio il valore dei dati geospaziali attraverso tutte le fasi del ciclo di vita dei dati: acquisizione, archiviazione, gestione, analisi e condivisione. Operiamo in molti campi di applicazione, che spaziano dal monitoraggio ambientale e del territorio, all'open-government e alle smart cities, alle soluzioni per la difesa e la sicurezza, l'ingegneria e le costruzioni, i trasporti, le utilities e l'energia, le risorse alimentari, fino alle missioni satellitari scientifiche e di esplorazione dello Spazio.
Planetek Italia è distributore per l'Italia dei software Hexagon Geospatial, fornitore di dati telerilevati da satellite, e centro di formazione ed addestramento all'utilizzo degli strumenti software.

Per saperne di più www.planetek.it.

Informazioni sul presente documento

Autori: Giuseppe Maldera

Editore: Planetek Italia S.r.l., Via Massaua, 12 - I-70132 Bari, Italia. Ref: PKID-2102554853-448111 v.5.0