UNIVERSITA' DEGLI STUDI DI
NAPOLI FEDERICO II

Scuola Politecnica e delle Scienze di Base
Corso di Laurea in Ingegneria Informatica

Elaborato finale in **Elaborazione di segnali multimediali**

Segmentazione semantica per autonomous driving

Anno Accademico 2020/2021

Candidato:

Simeoni Ildebrando

matr. N46003995

Alla mia famiglia, ai miei amici e a me.
“J’avemo pagato le nocchie”

Indice

Introduzione

Nell'era dell'Intelligenza Artificiale, una delle applicazioni più attivamente studiate è la guida autonoma nell'industria automobilistica. In particolare, i recenti progressi tecnologici nei campi del deep learning, della computer vision e di sensori quali RADAR, LIDAR e GPS, hanno permesso ai produttori di aumentare il livello di autonomia dei propri veicoli.

Sono sempre più le multinazionali che stanno investendo risorse umane e materiali nella ricerca e nello sviluppo di sistemi hardware e software che possano supportare questa funzione; a dimostrazione di ciò, il mercato delle auto autonome è stato valutato a 20.97 miliardi di dollari nel 2020, e si prevede che possa raggiungere i 61.93 miliardi con un CAGR (tasso composto di crescita annuale) del 22. 75%, nel prossimo futuro [1].

Spesso si ritiene che progettare un'auto a guida autonoma non sia un compito estremamente arduo, sottovalutando la complessità delle azioni e dei processi in atto durante la quotidiana azione di guida di un veicolo. Il semplice rispetto delle norme stradali non è infatti sufficiente per raggiungere il livello di abilità di guida di un essere umano; vanno considerate le capacità di: gestire situazioni impreviste, reagire opportunamente alle diverse condizioni meteorologiche, prendere decisioni che sono contro le regole ove necessario, ad esempio, per evitare di mettere in pericolo una vita umana.

Il successo dei veicoli a guida autonoma è in gran parte basato su un riconoscimento delle immagini accurato, compito attualmente affrontabile mediante *deep neural networks*.

Il complesso sistema di un veicolo autonomo si basa su diversi componenti hardware e software, i quali coprono molti aspetti necessari al corretto riconoscimento degli oggetti; la segmentazione semantica è un elemento fondamentale di tale processo.

Essa è una particolare fase di elaborazione delle immagini in cui si cerca di separare due o più elementi di un'immagine e definire i confini delle singole entità. Tuttavia, la maggior parte della ricerca sulla segmentazione semantica si concentra sul miglioramento della precisione di tale tecnica con meno attenzione alle soluzioni computazionalmente efficienti.

Gli obiettivi di questo lavoro sono quelli di:

- Fornire una breve panoramica sulle tecniche di segmentazione semantica e sulle applicazioni per autonomous driving.
- Analizzare in maniera breve, ma completa, un’architettura di deep learning per la segmentazione semantica delle immagini proposta nel paper scientifico “Speeding up Semantic Segmentation for Autonomous Driving” [2], caratterizzata da alta precisione pur essendo abbastanza efficiente e quindi adattabile a dispositivi embedded, reimplementando inoltre il relativo codice, adattandolo ai più recenti aggiornamenti dei principali moduli e tecnologie utilizzate.
- Confrontare l’architettura proposta con l’attuale stato dell’arte nel settore, di cui è proposto un rapido excursus.

Capitolo 1: Segmentazione semantica per autonomous driving

I nostri "sensori" più importanti nella guida su strada sono i nostri occhi, con i quali possiamo identificare ed analizzare l'ambiente che ci circonda velocemente e accuratamente. Sulla base di questa osservazione, è possibile affermare che il successo dei veicoli a guida autonoma è in gran parte basato su un riconoscimento delle immagini ben funzionante.

La maggior parte delle prove disponibili indica che questo compito è più facilmente realizzabile utilizzando *deep neural networks*, le quali costituiscono attualmente lo stato dell'arte in numerosi campi dell'Intelligenza Artificiale.

Dal punto di vista tecnico, il processo di segmentazione semantica prevede di raggruppare l'immagine generalmente a livello di pixel e assegnare un'etichetta, indicante una delle classi predefinite, a ciascuno di essi.

Più in generale, la segmentazione è il processo che permette di partizionare un'immagine in regioni (o oggetti) con un livello di accuratezza richiesto che varia in base al problema da affrontare. Risulta quindi evidente quanto sia di fondamentale importanza ottenere un livello sufficientemente elevato di segmentazione dell'immagine, poiché questa costituirà la base per successive procedure di analisi computerizzata dell'immagine stessa.

E' inoltre possibile suddividere la segmentazione semantica in due tipologie:

- Segmentazione effettuata mediante tecniche edge-based; fornisce in uscita la mappa dei contorni dell'immagine, tale tecnica si basa sull'individuazione di oggetti rilevandone i bordi, basandosi prevalentemente sulle discontinuità presenti nell'immagine, associando quindi alla posizione degli oggetti di interesse rapidi cambiamenti nei valori di intensità luminosa.
- Segmentazione effettuata mediante tecniche class-based; fornisce invece in uscita la mappa delle etichette, tale tecnica si basa sull'individuazione di oggetti suddividendo l'immagine in regioni tra loro simili sulla base di un determinato criterio (e.g. luminosità, colore).

1.1 Segmentazione semantica

La segmentazione semantica delle immagini è definita come il compito di classificare ogni pixel di un'immagine a partire da un insieme predefinito di classi; nella figura 1 ne è riportato un esempio.

È necessario innanzitutto fornire una distinzione tra processi che spesso vengono erroneamente confusi tra loro.

Figura 1: esempio di segmentazione semantica (fonte: ACM Digital Library)

La classificazione di immagini consiste nel fare una previsione per un intero input; la localizzazione/rilevazione fornisce non solo le classi, ma anche informazioni aggiuntive riguardanti la posizione spaziale di queste ultime nell'immagine; infine, la segmentazione semantica, mediante la quale è possibile effettuare previsioni dense che deducono le etichette per ogni pixel nell'immagine di input iniziale, in modo tale che ogni pixel sia etichettato con la classe relativa al suo oggetto o alla regione circostante.

La segmentazione semantica, inoltre, si distingue dalla semplice segmentazione di oggetti [3], quest'ultima è definibile come una suddivisione di un'immagine in più parti "coerenti" tra loro ma senza alcun tentativo di capire cosa esse rappresentino, in questo caso la coerenza tra regioni dell'immagine viene valutata sulla base di caratteristiche di basso livello come il colore, le texture e la luminosità, senza quindi valutare un collegamento semantico tra i pixel nell'immagine.

Al contrario, la segmentazione semantica tenta di suddividere l'immagine in parti semanticamente significative e di classificare ciascuna di esse in una delle classi predeterminate.

È possibile, inoltre, suddividere ulteriormente la segmentazione in: *semantic*, *instance e panoptic segmentation*. La segmentazione semantica (i.e. *semantic segmentation*) associa ad ogni pixel di un'immagine un'etichetta relativa ad una classe (e.g. persona, fiore, auto, cane) e tratta più oggetti della stessa classe come una singola entità. Al contrario, l'*instance segmentation* tratta molteplici oggetti della stessa classe come istanze individuali distinte. Infine, per combinare i concetti di *instance* e *semantic segmentation*, la *panoptic segmentation* assegna due etichette a ciascuno dei pixel di un'immagine: un'etichetta semantica e un id relativo alla singola istanza. I pixel etichettati in modo identico sono considerati appartenenti alla stessa classe semantica e i loro id ne distinguono le differenti istanze.

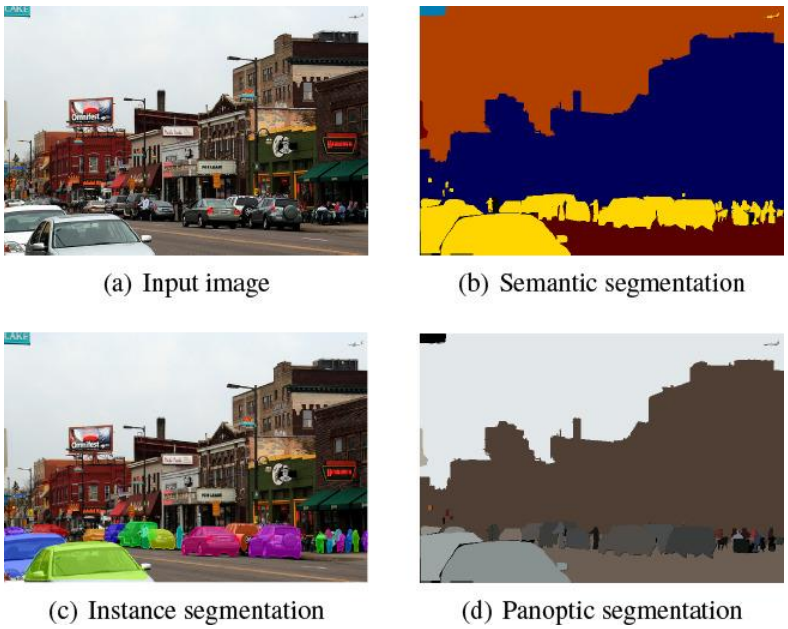

Figura 2: differenti tipi di segmentazione (fonte: ResearchGate)

Un esempio delle differenze tra questi tipi di segmentazione è riportato in figura 2, nella quale è possibile notare come le macchine dell'immagine in input siano identificate da una singola classe generica nella *semantic segmentation*, mentre le stesse sono classificate separatamente nelle applicazioni di *instance e panoptic segmentation*; in particolare in quest'ultimo caso, a differenza dell'*instance segmentation* vengono anche classificati altri oggetti quali gli edifici, il manto stradale ed il cielo.

1.1.1 Applicazioni

Numerosi e disparati sono i campi in cui la tecnologia della segmentazione semantica è applicata; non solo per autonomous driving essa spazia dall'imaging biomedico all'e-commerce, solo per citarne alcuni.

Di seguito sono riportati alcuni dei principali casi di utilizzo:

- Imaging biomedico; la segmentazione semantica trova ampio utilizzo nel campo dell'imaging biomedico, in particolare essa è applicata per localizzare tumori o altre patologie, misurare il volume di tessuti,

effettuare diagnosi, pianificare interventi chirurgici e realizzare simulazioni di interventi complessi in contesti di realtà aumentata.

- Face recognition; è sempre più richiesta un'accurata segmentazione semantica al fine di individuare all'interno di un'immagine raffigurante un volto umano categorie come: occhi, naso, bocca, pelle, capelli e sfondo. La segmentazione dettagliata delle caratteristiche facciali può infatti essere usata per addestrare le applicazioni di computer vision a distinguere l'etnia, l'età e l'espressione di un individuo.
- GeoSensing; di fondamentale importanza è la segmentazione semantica nella mappatura di immagini satellitari del suolo. Essa infatti è fondamentale per numerose applicazioni quali: il monitoraggio delle aree di deforestazione, il rilevamento di strade ed edifici per la gestione del traffico, il monitoraggio della condizione delle strade e la pianificazione delle città.
- Content base image retrieval (CBIR); ovvero l'applicazione di tecniche di computer vision, tra cui, la segmentazione semantica, al problema dell'image retrieval, ovvero la ricerca di immagini digitali in grandi database effettuando una ricerca per contenuto che si oppone alla tradizionale ricerca basata sui concetti.
- Fingerprint recognition e iris recognition; a causa della loro unicità e permanenza, le impronte digitali e l'iride sono emerse come due degli identificatori biometrici più utilizzati negli ultimi decenni. I sistemi automatizzati di verifica delle impronte digitali e di riconoscimento dell'iride sono stati sviluppati per soddisfare le esigenze delle forze dell'ordine, ma il loro uso è diventato sempre più diffuso anche nelle applicazioni civili negli ultimi anni. Nonostante la loro ampia diffusione tuttavia, la verifica automatizzata affidabile è rimasta una sfida ed è stata ampiamente studiata nel contesto del riconoscimento dei modelli e dell'elaborazione delle immagini.

- Agricoltura di precisione; i robot per l'agricoltura di precisione possono ridurre la quantità di erbicidi che devono essere spruzzati nei campi e la segmentazione semantica delle colture e delle erbacce li assiste in tempo reale per attivare tali azioni di diserbo.
- E-commerce; la categorizzazione accurata dell'abbigliamento può essere un compito complesso a causa della grande varietà di articoli presenti. La segmentazione semantica per gli articoli di abbigliamento ha reso possibile lo sviluppo di applicazioni quali ‘Amazon virtual mirror’, che consente di provare in un contesto di realtà aumentata capi di abbigliamento direttamente da casa.

1.2 Autonomous driving

Un'auto a guida autonoma, nota anche come veicolo autonomo, auto senza conducente o robo-auto, è un veicolo che è in grado di rilevare il suo ambiente e muoversi in sicurezza con poco o nessun input umano.

È doveroso innanzitutto fare un'importante premessa in merito al termine "autonomous driving", infatti, la Society of Automotive Engineers (SAE) utilizza il termine automated (automatizzato) invece di autonomous (autonomo). Questo perché la parola "autonomia" ha implicazioni che vanno oltre il mero ambito elettromeccanico; un'auto completamente autonoma infatti, sarebbe autoconsapevole e capace di compiere le proprie scelte, un'auto completamente automatizzata, invece, sarebbe solamente in grado di rilevare gli ordini e poi eseguirli da sola.

Il termine self-driving è spesso usato in modo intercambiabile con autonomous, tuttavia anche in questo caso si indica un concetto leggermente differente. Le auto a guida autonoma rientrano, facendo riferimento ai livelli definiti dalla SAE successivamente analizzati più nel dettaglio, nel livello 3 (automazione di guida condizionata) o nel livello 4 (alta automazione di guida). Esse sono inoltre soggette a geofencing, ovvero all'utilizzo di un perimetro virtuale associato a un'area geografica del mondo reale utilizzato per determinare la posizione del dispositivo. Al contrario, un'auto di livello 5 completamente autonoma potrebbe tecnicamente andare ovunque, senza essere soggetta alle precedenti limitazioni, in questo caso si parlerebbe di self-driving.

Le auto a guida autonoma combinano una varietà di sensori per percepire l'ambiente circostante, come radar, lidar, sonar, GPS e unità di misura inerziale. In particolare, sono le tecnologie Lidar che costituiscono la principale spinta alla crescita di tale mercato [1], esse agiscono come un occhio per i veicoli a guida autonoma in quanto forniscono una visione a 360

gradi di ciò che circonda il veicolo, il che li aiuta a guidare autonomamente e in modo sicuro. Sono usate da molti veicoli autonomi per navigare negli ambienti in tempo reale e tra i loro vantaggi è necessario menzionare in particolare un'accurata percezione della profondità, che permette al LiDAR di conoscere la distanza di un oggetto a partire da pochi centimetri fino a 60 metri di distanza.

I sistemi di controllo avanzati in tali veicoli interpretano le informazioni sensoriali precedentemente menzionate al fine di identificare i percorsi di navigazione appropriati, così come gli ostacoli e la segnaletica rilevante. Possibili implementazioni di tale tecnologia includono veicoli personali a guida autonoma, robotaxi condivisi, plotoni di veicoli connessi e trasporti su lunghe distanze.

Diversi progetti per sviluppare veicoli commerciali completamente a guida autonoma sono in varie fasi di sviluppo in tutto il mondo. Waymo è diventato il primo fornitore di servizi ad offrire corse robotaxi al pubblico in Phoenix, Arizona, nel 2020; Tesla ha affermato la possibilità di offrire una "guida completamente autonoma" in abbonamento ai proprietari di veicoli privati nel 2021; Nuro è stata autorizzata ad avviare operazioni di consegna commerciale autonoma in California nel 2021. In Cina sono state lanciate due prove di robo-taxi accessibili al pubblico, nel 2020 nel distretto Pingshan di Shenzhen dalla società cinese AutoX e nel 2021 al parco Shougang di Pechino.

La SAE definisce attualmente 6 livelli di automazione della guida che vanno dal livello 0 (completamente manuale) al livello 5 (completamente autonomo).

Questi livelli sono stati adottati dal Dipartimento dei Trasporti degli Stati Uniti, e sono descritti come segue:

- Livello 0: il sistema automatizzato non ha il controllo del veicolo, ma può avvisare il conducente dei pericoli.

- Livello 1: il conducente e il sistema automatizzato condividono il controllo del veicolo. Esempi di questo possono essere trovati nella maggior parte delle auto dotate di ADAS (Advanced driver-assistance systems).
- Livello 2: il sistema automatizzato è in grado di prendere il pieno controllo del veicolo, tuttavia, il conducente deve essere pronto a intervenire se il sistema non riesce a riconoscere un potenziale pericolo.
- Livello 3: Il sistema automatizzato prende il pieno controllo del veicolo e il passeggero può tranquillamente distogliere la sua attenzione dai compiti di guida, tuttavia, deve comunque essere in grado di intervenire.
- Livello 4: Il guidatore può distogliere con sicurezza la sua attenzione dai compiti di guida e lasciare che il sistema automatizzato prenda il pieno controllo. Questa funzionalità è attualmente limitata a specifiche aree "geofenced" e ad altri ambienti relativamente controllati.
- Livello 5: non è richiesto alcun intervento umano.

In un mondo sempre più dominato da dispositivi IoT (Interet of Things), dove ogni cosa, che sia un veicolo motorizzato o un semaforo, è collegata a una rete ad alta velocità di qualche tipo, evidenti sono le possibili svolte nel campo dell’autonomous driving in merito alla comunicazione tra veicoli (V2V), tra veicoli e infrastrutture (V2I) e soprattutto tra veicoli e pedoni (V2P), ad esempio attraverso gli smartphone.

1.2.1 Vantaggi

Numerosi sono i vantaggi dell'applicazione di veicoli a guida autonoma su vasta scala, tra i principali è doveroso menzionare:

- Riduzione degli incidenti mortali e delle collisioni stradali.
- Riduzione della congestione del traffico e conseguente riduzione dei costi nella creazione di infrastrutture ad esso collegate.
- Risparmio di carburante con conseguente riduzione delle emissioni di CO2 relative ai veicoli.

Altrettanto numerosi sono i possibili campi di applicazione di tale tecnologia, tra cui: spedizioni e consegne, trasporto pubblico, trasporto di emergenza, fino ad arrivare al trasporto privato.

1.2.2 Sfide

Auto completamente autonome (livello 5) sono in fase di test in diverse zone del mondo, ma nessuna è ancora disponibile al pubblico. Siamo ancora lontani da questo, anche a causa di sfide che spaziano da quelle tecnologiche e legislative a quelle ambientali e filosofiche.

Alcune di esse sono:

- Le tecnologie LiDar sono ancora costose e non accessibili alla maggior parte della popolazione; inoltre non essendo ancora stato fatto un testing su larga scala molti sono i dubbi ancora irrisolti, la possibilità che i segnali lidar di più veicoli possano interferire tra loro, o ancora se le frequenze radio disponibili saranno in grado di sostenere la produzione in massa di auto autonome, sono solo alcuni di essi.
- La guida in condizioni metereologiche avverse; ad esempio in casi di scarsa visibilità a causa di precipitazioni o di parziale/totale occlusione di segnaletica stradale a causa di neve, detriti o ghiaccio. Condizioni che rendono ancora più complesso il processo decisionale di un'auto a guida autonoma.
- Responsabilità per gli incidenti; sarà sempre più difficile attribuire la

responsabilità di un incidente causato da un'auto all'aumentare del livello di autonomia di quest'ultima; tanto che auto completamente autonome (livello 5) potrebbero non avere un cruscotto o un volante, renendo quindi il passeggero umano impossibilitato a prendere il controllo del veicolo anche in caso di emergenza.

- Capacità di astrazione e intelligenza emotiva umana; spesso i conducenti umani si affidano a sottili spunti e alla comunicazione non verbale, come il contatto visivo con i pedoni o la lettura delle espressioni facciali e del linguaggio del corpo degli altri conducenti, per prendere decisioni in frazioni di secondo e prevederne i comportamenti. Sarà un arduo compito progettare auto in grado di replicare questa abilità.
- Capacità di funzionamento in ambienti cittadini caotici; lo stato dell'arte nel campo dell'AI è ancora lontano dal fornire alle auto la capacità di destreggiarsi in ambienti caotici come quelli cittadini densamente popolati.
- Sicurezza informatica dei dispositivi di controllo; veicoli a guida autonoma sono potenzialmente soggetti ad attacchi informatici, una sfida non semplice è quindi anche quella di garantire la sicurezza e robustezza del software di tali veicoli.

Capitolo 2: Metodologia

In questo capitolo verrà presentato l'approccio implementato a partire dal paper scientifico di riferimento [2], con particolare attenzione agli accorgimenti adottati al fine di rendere l'implementazione più rapida ed efficiente possibile, pur mantenendo un buon livello di precisione.
Verrà infine proposta una breve panoramica sullo stato dell'arte attuale nel campo della segmentazione semantica, riferendosi in particolare alle applicazioni relative all'autonomous driving, al fine di confrontare l'approccio qui utilizzato con le più recenti implementazioni nel settore.

2.1 Approccio implementato

Con la popolarità del deep learning negli ultimi anni, molti problemi di segmentazione semantica sono stati affrontati utilizzando *deep neural networks*, ed in particolare *convolutional neural networks* CNN (i.e. reti neurali convoluzionali), dimostratesi superiori rispetto agli altri approcci con un ampio margine in termini di precisione ed efficienza.
Di seguito sarà analizzato brevemente il codice di riferimento nei suoi punti fondamentali e gli accorgimenti applicati alla rete presa in considerazione.

```
label_defs = [
    Label('unlabeled',     (0,     0,   0)), Label('dynamic',       (111,  74,   0)),
    Label('ground',        ( 81,   0,  81)), Label('road',          (128,  64, 128)),
    Label('sidewalk',      (244,  35, 232)), Label('parking',       (250, 170, 160)),
    Label('rail track',    (230, 150, 140)), Label('building',      ( 70,  70,  70)),
    Label('wall',          (102, 102, 156)), Label('fence',         (190, 153, 153)),
    Label('guard rail',    (180, 165, 180)), Label('bridge',        (150, 100, 100)),
    Label('tunnel',        (150, 120,  90)), Label('pole',          (153, 153, 153)),
    Label('traffic light', (250, 170,  30)), Label('traffic sign',  (220, 220,   0)),
    Label('vegetation',    (107, 142,  35)), Label('terrain',       (152, 251, 152)),
    Label('sky',           ( 70, 130, 180)), Label('person',        (220,  20,  60)),
    Label('rider',         (255,   0,   0)), Label('car',           (  0,   0, 142)),
    Label('truck',         (  0,   0,  70)), Label('bus',           (  0,  60, 100)),
    Label('caravan',       (  0,   0,  90)), Label('trailer',       (  0,   0, 110)),
    Label('train',         (  0,  80, 100)), Label('motorcycle',    (  0,   0, 230)),
    Label('bicycle',       (119,  11,  32)), Label('pole group',    (153, 153, 153)),]
```

Figura 3: classi (fonte: Script)

2.1.1 Data processing

Il dataset di riferimento è costituito da 30 classi per l'annotazione, mostrate in figura 3, raggruppate in 8 categorie: flat, construction, nature, vehicle, sky, object, human, and void. Secondo l'approccio adottato nel paper "The Cityscapes Dataset for Semantic Urban Scene Understanding" [5], le classi sono state selezionate in base alla loro frequenza, alla rilevanza da un punto di vista applicativo, a considerazioni pratiche riguardanti lo sforzo di annotazione, così come per facilitare la compatibilità con dataset esistenti (e.g. Kitti dataset).

Le classi troppo rare sono state quindi escluse dal benchmark e sono state considerate appartenenti alla classe unlabeled, valutando così 19 classi rimanenti, più il background.

Le immagini, automaticamente suddivise in cartelle di training, validation e test, sono state poi organizzate in liste di file mediante la funzione "build_file_list".

Successivamente sono stati caricati i dati presenti in un'apposita cartella (nel caso dell'implementazione qui realizzata, una cartella su Google drive) alla quale accedere mediante Google Colab, in modo tale da prelevare le

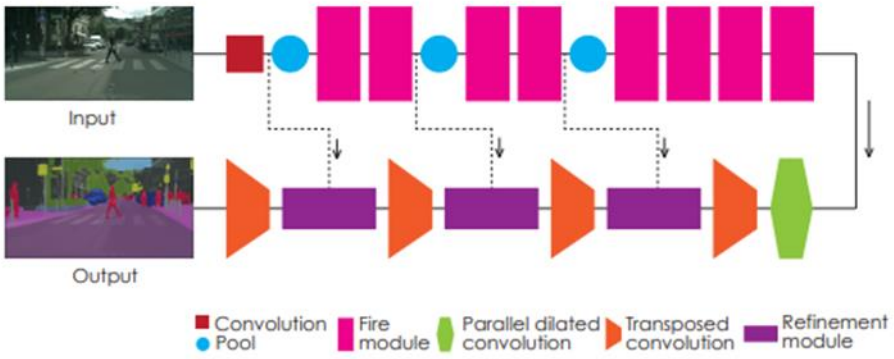

Figura 4: struttura architettura (fonte: [2])

immagini e le rispettive label precedentemente scaricate, creando così, tramite la funzione precedentemente menzionata "build_file_list", la lista delle immagini di training, validation e test, oltre che a definire il numero di classi e il relativo colore al fine di effettuare la predizione nella successiva fase di test.

2.1.2 Rete

Al fine di ottenere la segmentazione semantica in tempo reale, è necessario effettuare un trade-off tra la velocità di esecuzione della rete e la sua precisione, ed è a questo particolare trade off che l'architettura mostrata in seguito, ed illustrata in figura 4, fa attenzione.

Come la maggior parte delle reti di segmentazione di successo (e.g. LedNet [6], RedNet [7], U-Net, SegNet [8]), la rete qui analizzata è formata da una coppia encoder-decoder.

Un'architettura generale di segmentazione semantica può essere pensata come una rete di codifica (i.e. encoder) seguita da una rete di decodifica (i.e. decoder). In tale architettura l'encoder è di solito una rete di classificazione pre-addestrata il cui scopo è quello di estrarre le informazioni essenziali dall'immagine effettuando ciò che viene denominato *downsampling*. Quest'ultima è un'operazione che mira a ridurre le dimensioni spaziali

dell'immagine mediante operazioni matematiche (i.e. *max/average pooling*, *strided convolution*), i cui principali scopi sono: sopprimere il rumore, rendere la rete invariante alla traslazione di oggetti nell'immagine in input durante processi di classificazione e ridurre le risorse necessarie per la memorizzazione delle immagini.

Gli strati iniziali dell'encoder posseggono le informazioni spaziali ad alta risoluzione riguardanti l'immagine, essi infatti sono in grado di individuare la posizione degli oggetti presenti, ma non sono in grado di classificarli; al contrario, in seguito al processo di downsampling, gli strati dell'encoder hanno una miglior comprensione di ciò che è presente nell'immagine, grazie alle convoluzioni che hanno determinato l'aggiornamento dei pesi della rete, ma solo un'approssimativa consapevolezza di dove gli oggetti siano collocati spazialmente.

L'encoder genera, a partire da un'immagine, un vettore di feature multi-dimensionale nel quale sono codificate (da qui il termine encoder) le informazioni spaziali relative agli oggetti presenti.

Tale vettore costituisce a sua volta l'input per la seconda parte dell'architettura, costituita dal decoder; esso è una rete di decodifica che effettua un processo inverso a quello del *downsampling* denominato *upsampling*.

Durante la fase di *upsampling* vengono utilizzate informazioni dagli strati dell'encoder per effettuare la ricostruzione dell'output, che sarà, al termine del processo di decodifica, una mappa di segmentazione di dimensioni pari a quelle dell'immagine in input.

In particolar modo l'encoder è in questo caso un'architettura SqueezeNet 1.1 modificata [9] che è stata progettata come una rete a bassa latenza (i.e. il ritardo di comunicazione, ovvero il tempo che intercorre tra l'invio di un'informazione e la sua risposta) per il riconoscimento delle immagini, pur mantenendo una precisione prossima a quella di AlexNet. I principali moduli

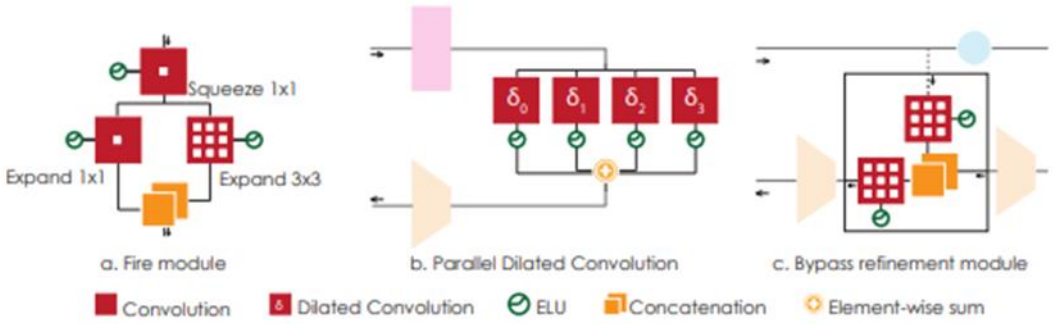

Figura 5: moduli rete (fonte:[2])

computazionali di SqueezeNet sono i cosiddetti moduli *fire*, mostrati in figura 5.a, che consistono in tre operazioni convoluzionali. L'encoder è composto da otto moduli *fire*, intervallati da un totale di tre strati di max-pooling per il downsampling.

Il decoder è costituito in particolare da un modulo di *parallel dilated convolution,* figura 5.b, in cui i risultati di quattro *dilated convolution* sono sommati tra loro, ciò permette di evitare di utilizzare strati fully connected che sarebbero altrimenti computazionalmente più onerosi.

L'upsampling è qui efettuato mediante i blocchi di *transposed convolution* i cui output sono combinati con informazioni provenienti dai livelli inferiori dell'encoder mediante i blochi di *bypass refinement*, figura 5.c.

In questo lavoro le funzioni di attivazione non lineari dell'architettura originale, denominate Rectified Linear Units (RELU), sono state sostituite con Exponential Linear Units (ELU), entrambe le funzioni sono mostrate in figura 6. ELU è una funzione che tende a far convergere la loss function a zero velocemente e a produrre risultati accurati, è molto simile a RELU, entrambe le funzioni sono infatti in forma di funzione identica per gli input non negativi, caratteristica che permette l'addestramento di *deep neural networks* evitando il problema dei vanishing gradients, mentre differiscono per gli input negativi; la funzione ELU infatti fa un uso più efficiente dei

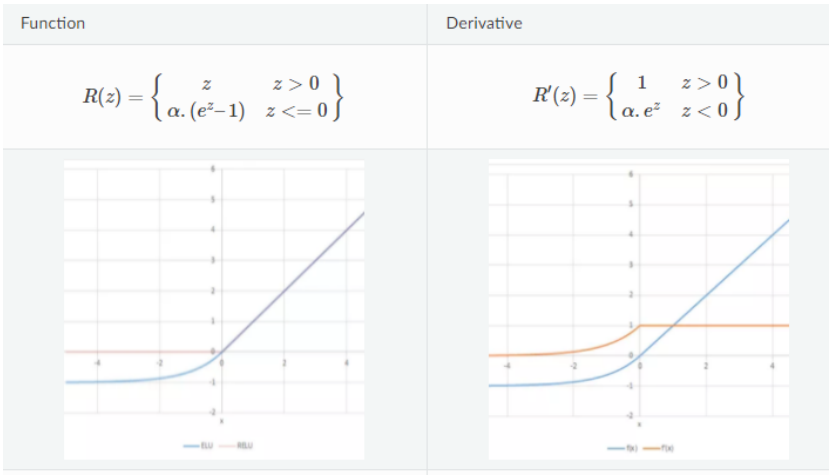

Figura 6: ELU vs RELU (fonte: [10])

parametri trasmettendo anche informazioni nella parte negativa dell'attivazione.

Questa caratteristica consente di risolvere il problema della dying RELU, evento che si verifica nel caso di valori di attivazione in input minori di 0, per i quali il valore del gradiente sarà pari a 0, causando una non opportuna regolazione dei pesi della rete durante il processo di training e la conseguente perdita dei neuroni che entrano in tale stato, poiché essi smetteranno di rispondere alle variazioni di errore/ingresso [10].

Un ulteriore accorgimento per aumentare l'efficienza della rete è l'utilizzo della normalizzazione L2.

Come la norma L1, anche la norma L2 è spesso utilizzata per l'addestramento di algoritmi di machine/deep learning come metodo di regolarizzazione, poiché, matematicamente, entrambe sono misure della grandezza dei pesi: la somma dei valori assoluti nel caso della norma L1, e la somma dei valori al quadrato per la norma L2. Quindi, pesi più grandi risultano in una norma più grande. Questo significa che, viceversa, minimizzare la norma è un metodo per mantenere i coefficienti del modello piccoli e, a sua volta, il modello meno complesso.

Inoltre, la norma L2 è computazionalmente meno onerosa della norma L1.

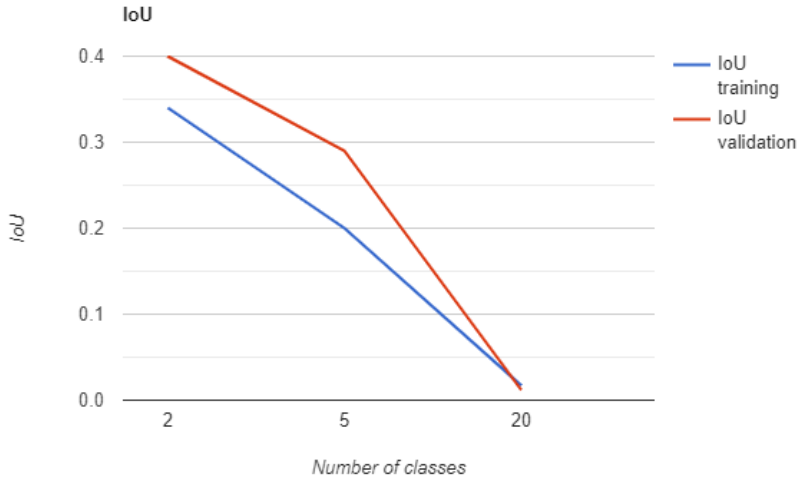

Figura 7: risultati training con 2 epoche (fonte: Script)

2.1.3 Training

La fase di training prevede l'addestramento del codificatore SqueezeNet mediante il metodo fit di Keras, il quale ha come parametri di input:

- Il numero di etichette, ovvero il numero di classi considerate in questo lavoro (pari a 20, considerando anche il background).
- Il batch size, che indica il numero di campioni (immagini) per gruppo che verranno utilizzati per addestrare la rete; alcuni dei vantaggi di scegliere un batch size minore dell'intero numero di campioni presenti sono la richiesta inferiore di memoria per la fase di training e una velocizzazione di tale fase, in quanto i pesi della rete vengono aggiornati al termine di ogni batch. In questo lavoro è stato imposto pari a 5 tale valore.
- Il numero di epoche è un iperparametro che definisce il numero di volte che l'algoritmo di apprendimento scorrerà l'intero dataset di training. Il termine di un'epoca indica che ogni campione nel dataset di training ha avuto l'opportunità di aggiornare i parametri interni del modello. Un'epoca è composta da uno o più batch. In questo lavoro era

inizialmente stato imposto pari a 1 tale valore, ma nella reimplementazione del codice su macchine meno performanti è stato necessario aumentare a 2 il suddetto parametro, in modo da aumentare il livello di IoU di training e validation della rete, mostrato in figura 7, pur non raggiungendo i valori del paper [2]; in particolare è evidente come all'aumentare del numero di classi corrisponde un notevole decremento della precisione da parte della rete nell'etichettare correttamente i pixel dell'immagine in input, tale aspetto verrà più approfonditamente trattato nel capitolo dei risultati sperimentali.

- I passi di training e validation, che rappresentano rispettivamente la divisione, arrotondata per intero, tra il numero di immagini di training e validation ed il batch size.
- Pesi salvati, parametro che indica il path relativo al file .h5 contenente i pesi aggiornati e salvati dell'architettura dopo la fase di training grazie al metodo callbacks.ModelCheckpoint di Keras.
- Dimensione delle immagini in input al modello, in questo caso pari a (128, 256, 3) secondo un formato channel-last.
- Data generators (training e validation), necessari in un mondo in cui vasti dataset sono sempre più alla portata di tutti, essi sono utilizzati per operare su tali dataset in tempo reale, sfruttando tecniche di multiprocessing e multithreading in modo da evitare di sfruttare eccessiva memoria.

La rete è stata addestrata con immagini a piena risoluzione, senza l'ausilio di dataset aggiuntivi o di operazioni di data augmentation.

La loss function utilizzata è la categorical_crossentropy ottimizzata con Stochastic Gradient Descent, utilizzando un ottimizzatore RMSprop con learning rate pari a 1e-4, valore di rho di default pari a 0.9 e costante epsilon per la stabilità pari a 1e-08.

2.2 Stato dell’arte

Numerose sono le architetture implementate in seguito a quella di riferimento [2] per la segmentazione semantica di scene stradali, in particolar modo seguono in tale paragrafo alcuni dei metodi più innovativi che hanno raggiunto i migliori risultati su dataset di immagini stradali (e.g. Cityscapes, Mapillary, PASCAL) in termini di IoU (Intersection over Union) media. L’IoU è una delle principali metriche utilizzate in ambito di segmentazione semantica, essa valuta l'area di sovrapposizione tra la previsione di segmentazione e la ground truth divisa per l'area di unione tra di esse, questa metrica varia tra 0 e 1 (0-100%) con il valore 0 che indica nessuna sovrapposizione e 1 che indica una segmentazione perfettamente sovrapposta. La maggior parte delle tecniche di segmentazione semantica adotta una rete fully-convolutional (FCN) con un'architettura encoder-decoder, come visto in precedenza, in cui il codificatore riduce progressivamente la risoluzione spaziale e apprende concetti visivi più astratti/semantici con campi recettivi più grandi. Poiché la modellazione del contesto è fondamentale per la segmentazione, gli sforzi più recenti si sono concentrati sull'aumento del campo recettivo, attraverso *dilated/atrous convolution* o l'inserimento di moduli di attenzione. Tuttavia, l'architettura FCN basata sul codificatore-decodificatore rimane invariata. Nel lavoro "Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers" [11] invece, si fornisce una prospettiva alternativa trattando la segmentazione semantica come un compito di predizione sequenza-sequenza. In particolare, viene impiegato un trasformatore puro (cioè, senza convoluzione e riduzione della risoluzione) per codificare un'immagine come una sequenza di blocchi. Con il contesto globale modellato in ogni strato del trasformatore, questo codificatore può essere combinato con un semplice decodificatore per fornire un potente modello di segmentazione, chiamato SEgmentation TRansformer

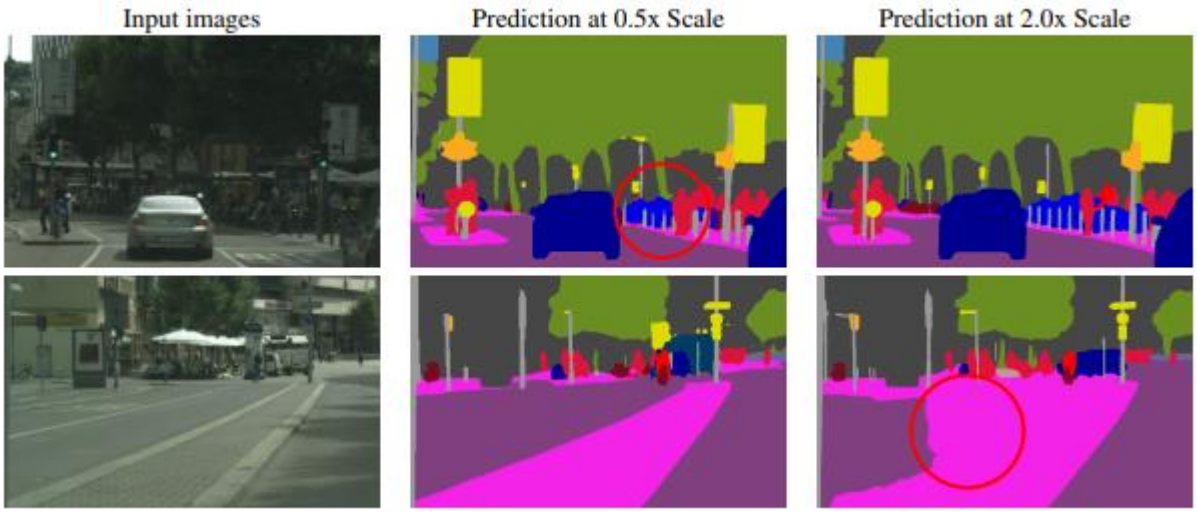

Figura 8: predizioni a differenti scale (fonte: [12])

(SETR). Esperimenti mostrano che SETR raggiunge un nuovo stato dell'arte su ADE20K (50,28% mIoU), Pascal Context (55,83% mIoU) e risultati competitivi su Cityscapes.

O ancora, l'approccio descritto nell'articolo "Hierarchical Multi-Scale Attention for Semantic Segmentation" [12], che attualmente detiene il più alto livello di IoU media (pari a 85.1%) sul dataset Cityscapes, basato sulla multi-scale inference, comunemente usata per migliorare i risultati della segmentazione semantica; più scale di immagini sono passate attraverso una rete e poi i risultati sono combinati attraverso *mean o max pooling*. In particolare nel paper si mostra come le predizioni a determinate scale siano migliori, come mostrato in figura 8, e come la rete impari a favorire queste ultime in tali casi al fine di generare previsioni più accurate. I risultati di tale rete raggiungono lo stato dell'arte sia per il dataset Mapillary che per Cityscapes.

Tuttavia, come evidenziato dal lavoro "ContextNet: Exploring Context and Detail for Semantic Segmentation in Real-time" [13], le moderne implementazioni di *deep neural networks* producono risultati molto accurati

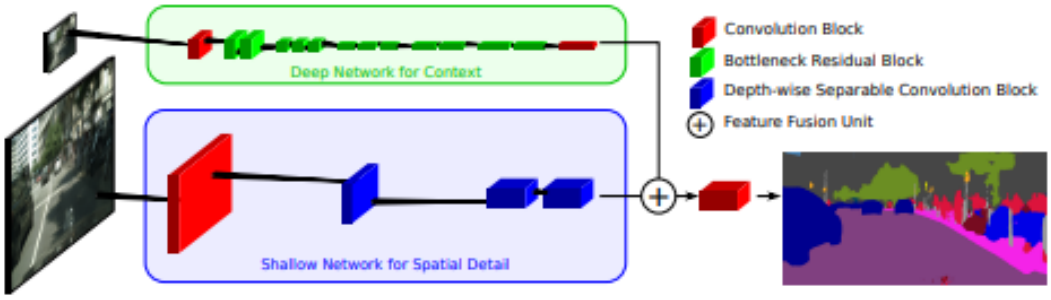

Figura 9: architettura ContextNet (fonte: [13])

su una varietà di dataset impegnativi per la segmentazione semantica, ma tali soluzioni non sono direttamente utilizzabili da applicazioni che operano in tempo reale o da dispositivi embedded, poiché il semplice riadattamento di tali sistemi per ridurre il costo computazionale (i.e. velocità, memoria ed energia) ne provoca un calo significativo della precisione. Per questo motivo nell'articolo viene proposta una soluzione il cui scopo è molto simile a quello di [2]; ContextNet, è un'architettura, mostrata in figura 9, che si basa sulla convoluzione fattorizzata, la compressione della rete e la rappresentazione piramidale per produrre una segmentazione semantica competitiva in tempo reale con ridotte richieste computazionali. Essa combina un ramo di *deep neural network* a bassa risoluzione che cattura informazioni sul contesto in modo efficiente con un ramo poco profondo che si concentra sulla segmentazione dei dettagli ad alta risoluzione spaziale, raggiungendo un livello di precisione del 66,1% sul dataset Cityscapes a 18,3 fotogrammi al secondo a piena risoluzione (1024 ×2048).

Capitolo 3: Risultati

In questo ultimo capitolo verrà brevemente descritto il dataset di interesse per l'implementazione del lavoro; saranno inoltre presentati i risultati sperimentali ottenuti dalla rete mostrata in precedenza.

3.1 Dataset

Cityscapes è un dataset su larga scala che si concentra sulla comprensione semantica delle scene stradali urbane, un esempio di immagine del dataset è mostrato in figura 10. Esso fornisce annotazioni semantiche per 30 classi raggruppate in 8 categorie (i.e. superfici piane, persone, veicoli, costruzioni, oggetti, natura, cielo e void) e consta di circa 5.000 immagini con annotazioni fini e 20.000 con annotazioni grossolane.

I dati sono stati catturati in 50 città durante diversi mesi, di giorno e in buone condizioni atmosferiche.

È stato originariamente registrato come video, quindi i fotogrammi sono stati selezionati manualmente per avere le seguenti caratteristiche: un gran numero di oggetti dinamici, una disposizione variabile della scena e uno sfondo variabile.

Figura 10: esempio dataset Cityscapes (fonte: ResearchGate)

Nello svolgimento del lavoro sono state considerate le ground truth fine annotations (gtFine_trainvaltest), questo tipo di annotazioni è usato per la validation ed il training (con 3475 immagini annotate) e per il test (1525 dummy annotations). Il dataset è stato suddiviso in 2975, 500 e 1525 immagini, di risoluzione 2048x1024 pixel, rispettivamente per training, validation e test.

Le annotazioni sono codificate utilizzando file json contenenti i singoli poligoni. Le immagini annotate standard in formato LDR a 8 bit (i.e. leftImg8bit) sono già suddivise nelle relative sezioni di training, validation e test.

Sono presenti inoltre nel dataset: annotazioni relative ad altre 20.000 immagini da fornire per un eventuale addestramento aumentato, immagini con volti e targhe oscurate, numerosi pacchetti da terze parti derivati da ulteriori pubblicazioni che includono tra le altre: annotazioni per *panoptic segmentation*, immagini in condizioni meteo non favorevoli (i.e. nebbia e pioggia) e annotazioni 3D sui bounding box per i veicoli.

3.2 Risultati sperimentali

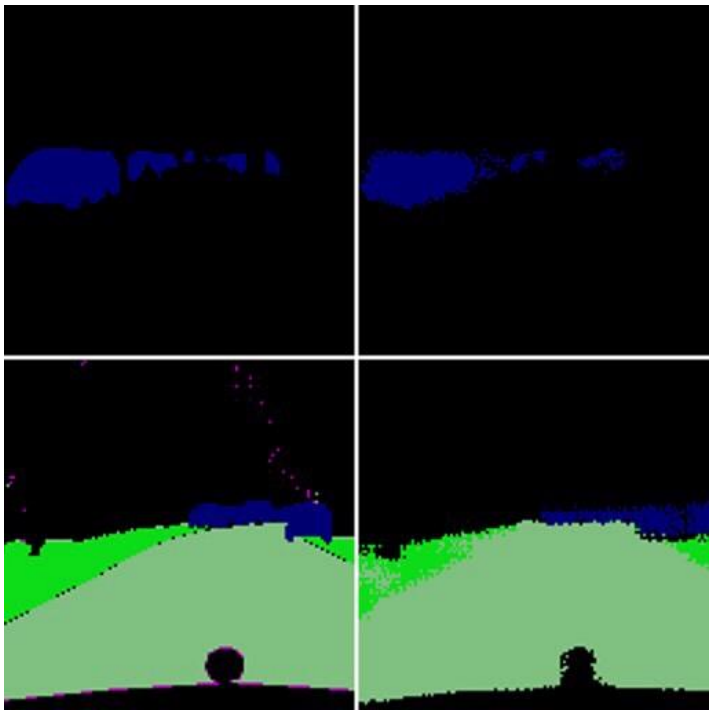

Figura 11: risultati con 2 e 5 classi (fonte: Script)

Come precedentemente anticipato nei risultati del training della rete, l'addestramento mediante la GPU Tesla 4 fornita da Colab e della durata complessiva di 3 ore non raggiunge il livello mostrato nel lavoro [2], il cui addestramento, della totale durata di 22 ore, è stato realizzato mediante 2 GPU Titan X.

In particolare, come è possibile osservare nella figura 11, si ottengono risultati soddisfacenti per un limitato numero di classi da riconoscere nell'immagine in input; sono stati riportati i risultati considerando rispettivamente 2 (i.e. car, unlabeled) e 5 (i.e. car, road, terrain, sidewalk, unlabeled) classi, riportando la ground truth (a sinistra) e la relativa predizione (a destra).

E' evidente come la rete, seppur individui con sufficiente precisione la posizione degli oggetti presenti nell'immagine, difficilmente riesca a delineare con accuratezza i contorni di questi ultimi, spesso erroneamente

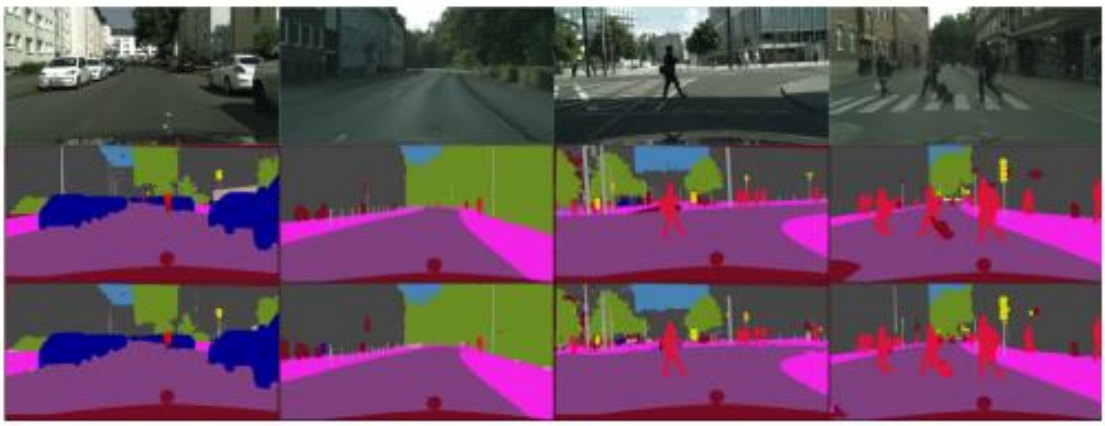

Figura 12: immagini input (in alto), ground truth (centro),
output della rete (in basso) (fonte: [2])

etichettati come unlabeled poiché ritenuti appartenenti a classi non considerate in fase di test.

La rete è in grado di identificare in maniera accurata classi che si ripresentano più frequentemente nel dataset (e.g. car, road, terrain), mentre difficilmente individua oggetti appartenenti a classi meno predominanti, in particolar modo nei casi di occlusione visiva di questi ultimi a causa di oggetti appartenenti a classi più facilmente riconoscibili dalla rete.

Aumentando il numero di classi prese in considerazione, l'*accuracy* (i.e. accuratezza) della rete, valutata in termini di IoU, decresce in maniera esponenziale; in particolar modo, dopo numerosi test, sono stati ritenuti soddisfacenti i risultati ottenuti considerando contemporaneamente al più 8 delle 19 classi iniziali.

Poiché le limitazioni legate ai risultati sperimentali sono dovute ai limiti delle risorse utilizzate nel training della rete, al fine di avvalorare maggiormente la tesi del lavoro svolto, sono di seguito riportati e commentati i risultati sperimentali ottenuti nel paper di riferimento [2].

La rete ottiene risultati confrontabili con la rete di segmentazione ENet [14], che rappresenta una delle principali soluzioni progettate per la segmentazione semantica per dispositivi a bassa latenza.

I risultati, utilizzando il server di valutazione ufficiale Cityscapes, sono di IoU media per classe di 59.8% e di IoU media per categoria di 84.3%, indicando in questi termini la capacità della rete di etichettare correttamente i pixel relativamente alla classe di appartenenza (una tra le 19 considerate in tale lavoro) e relativamente alla più generica categoria di appartenenza (una tra le 8 appartenenti al dataset).

L'architettura è quindi in grado di ottenere migliori prestazioni sia di ENet [14] che di SegNet, migliorando la IoU per classe per tutte le classi tranne 5 (i.e. wall, camion, autobus, train, motorcycle).

L'ispezione visiva delle previsioni della rete, di cui ne sono mostrati esempi in figura 12, mostra risultati soddisfacenti su immagini tipiche della scena urbana, mostrando contorni degli oggetti segmentati in modo netto.

Tali risultati sono confrontabili con un'altra moderna soluzione al problema della segmentazione semantica in tempo reale che è Context Aggregation Network [15], una rete neurale convoluzionale a doppio ramo, anch'essa con costi computazionali significativamente inferiori rispetto allo stato dell'arte, pur mantenendo una precisione di previsione competitiva. Ricalcando le architetture a doppio ramo esistenti per la segmentazione semantica ad alta velocità, essa è costituita da un ramo ad alta risoluzione per valutare i dettagli spaziali e un ramo di contesto con versioni leggere dei blocchi di aggregazione globale e di distribuzione locale, utili per catturare sia le dipendenze contestuali a lungo raggio che quelle locali necessarie per un'accurata segmentazione semantica, con bassi costi di calcolo.

Conclusioni

La segmentazione semantica di immagini stradali rappresenta un elemento cruciale nello sviluppo dei veicoli a guida autonoma, che è stato dimostrato essere un mercato in esponenziale crescita in tutto il mondo, e uno dei punti fondamentali nella rivoluzione dell'intelligenza artificiale di cui siamo testimoni.

Come visto in questo elaborato le *deep neural networks* costituiscono attualmente lo stato dell'arte dei sistemi di segmentazione semantica delle immagini; esse non sono tuttavia esenti da problemi, hanno infatti un elevato costo computazionale che non sempre le rende adatte, se non previ opportuni adattamenti, ad applicazioni real time in sistemi embedded quali quelli relativi ai veicoli a guida autonoma.

In questo lavoro è stata analizzata, descritta e reimplementata una delle possibili soluzioni a tale problema, confrontandola successivamente con altre architetture che costituiscono l'attuale stato dell'arte nella segmentazione semantica, con particolare attenzione alla segmentazione di immagini stradali. Tale struttura costituisce un punto di inizio verso possibili implementazioni in grado di ridurre ancor più gli oneri computazionali (e.g. riducendo ancor più la profondità della rete mediante Knowledge Distillation) pur raggiungendo livelli di accuratezza pari all'attuale stato dell'arte.

Bibliografia

[1] ReportLinker, https://www.reportlinker.com/p06101202/Autonomous-Driverless-Car-Market-Growth-Trends-COVID-19-Impact-and-Forecast.html?utm_source=GNW, 1 Aug 2021.

[2] Michael Treml, Jose A. Arjona-Medina, Thomas Unterthiner, Rupesh Durgesh, Felix Friedmann, Peter Schuberth, Andreas Mayr, Martin Heusel, Markus Hofmarcher, Michael Widrich, Bernhard Nessler, Sepp Hochreiter, "Speeding up Semantic Segmentation for Autonomous Driving", NIPS 2016 workshop MLITS submission, 2016, pagine 1:6.

[3] Jianbo Shi, J. Malik, "Normalized Cuts and Image Segmentation", IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 22, pagine 888:905, Aug 2000.

[4] Nanonets, https://nanonets.com/blog/how-to-do-semantic-segmentation-using-deep-learning/, 2 Aug 2021.

[5] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, Bernt Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding", 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2016, pagine 2-3.

[6] Yu Wang, Quan Zhou, Jia Liu, Jian Xiong, Guangwei Gao, Xiaofu Wu, L. Latecki, "A Lightweight Encoder-Decoder Network for Real-Time Semantic Segmentation", 2019 IEEE International Conference on Image Processing (ICIP), IEEE, 2019, pagine 1:5.

[7] Jindong Jiang, Lunan Zheng, Fei Luo, Zhijun Zhang, “Residual Encoder-Decoder Network for indoor RGB-D Semantic Segmentation”, 2018, pagine 1:12.

[8] Vijay Badrinarayanan, Alex Kendall, R. Cipolla, “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation “, IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 39, pagine 2481:2495, 2017.

[9] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, and K. Keutzer, “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size”, ICLR 2017, 2017, pagine 1:13.

[10] ML Glossary, https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html, 2 Aug 2021.

[11] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, Li Zhang, “Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers”, 2020, pagine 1:12.

[12] Andrew Tao, Karan Sapra, Bryan Catanzaro, “Hierarchical Multi-Scale Attention for Semantic Segmentation”, 2020, pagine 1:11.

[13] Rudra P K Poudel, Ujwal Bonde, Stephan Liwicki, Christopher Zach, “ContextNet: Exploring Context and Detail for Semantic Segmentation in Real-time”, BMVC 2018, 2018, pagine 1:12.

[14] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, Eugenio Culurciello, “ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation”, 2020, pagine 1:10.

[15] Michael YingYang, Saumya Kumaar,Ye Lyu, Francesco Nex, “Real-time Semantic Segmentation with Context Aggregation Network”, 2020, pagine 1:31.