Alma Mater Studiorum · Università di Bologna

SCUOLA DI INGEGNERIA E ARCHITETTURA

Corso di Laurea Magistrale in Ingegneria Informatica

Domain adaptation per classificazione di point cloud mediante pseudo-annotazioni

Candidato:
Daniele Menchetti

Relatore:
Chiar.mo Prof. Luigi Di Stefano

Correlatori:
Dott. Adriano Cardace
Ph.D Pierluigi Zama Ramirez
Ph.D Riccardo Spezialetti

Sessione III
Anno Accademico 2021/2022

Indice

Capitolo 1

Introduzione

Negli ultimi decenni la quantità smisurata di dati accumulati attraverso l'utilizzo di dispositivi elettronici ha aperto le porte a nuovi campi di ricerca atti all'estrazione di informazioni e schemi interessanti da quest'ultimi. Nasce dunque una branca dell'intelligenza artificiale denominata **Apprendimento Automatico** o **Machine Learning** nella quale i sistemi coinvolti sono predisposti dell'abilità di apprendimento di qualsiasi cosa in maniera autonoma, senza che vi siano istruzioni esplicite. Tali algoritmi trovano ampio spazio in settori come la medicina, data science etc.

In un secondo momento, si sviluppa una sottobranca dell'apprendimento automatico definita **Apprendimento Profondo** o **Deep Learning** a partire dalla nascita del primo percettrone lineare. Qui, i modelli utilizzati vengono definiti reti neurali artificiali in quanto riflettono l'architettura e il comportamento del cervello umano imitando i processi di attivazione in cui questo è coinvolto. L'informazione passa dunque attraverso una sequenza di strati composti da neuroni ricevendo molteplici elaborazioni al fine di avere una rappresentazione di uscita dettata dall'obiettivo da perseguire. Deep Learning trova applicazioni in: visione artificiale, elaborazione del linguaggio naturale, riconoscimento audio, bioinformatica etc.

Uno dei problemi di ampio interesse all'interno dell'apprendimento profondo consiste nella classificazione di dati, ovvero l'attribuzione di un'etichetta a immagini, testo, audio etc. Tale problema diventa fortemente stimolante se applicato ad una recente tipologia di dato chiamata **nuvola di punti** o **point cloud**. Una nuvola di punti è un insieme di punti definiti in uno

spazio 3D e dunque rappresentati da 3 coordinate. Nel sistema di coordinate cartesiano ogni punto è raffigurato lungo le dimensioni X, Y e Z.
Solitamente le nuvole di punti rappresentano una forma oppure un oggetto nello spazio. In relazione ad oggetti reali essa può esser ottenuta attraverso opportuni scanner 3D oppure software di fotogrammetria, mentre è possibile generarla facilmente tramite software di progettazione e disegno 3D se si fa riferimento ad oggetti sintetici.

Una metodologia interessante di classificazione di nuvole di punti è adottata dal paper **RefRec: Pseudo-labels Refinement via Shape Reconstruction for Unsupervised 3D Domain Adaptation** [1]. In questo lavoro gli autori implementano idee brillanti per far fronte ad una situazione di **Adattamento al Dominio** o **Domain Adaptation**.

Con il termine **Adattamento al Dominio** si fa riferimento ad una particolare area di studio posta solitamente tra il campo di Machine Learning e quello di Transfer Learning. In particolare, la Domain Adaptation è la capacità di un modello allenato in un dominio definito sorgente o source di riuscire ad adattarsi ad un secondo dominio definito dominio di destinazione o target (legato al primo). Tale abilità trova facilmente applicazione in scenari in cui si necessita di un modello sufficientemente accurato avendo a disposizione solamente pochi dati reali etichettati per il suo addestramento. Per sopperire a tale mancanza si possono aggiungere dati provenienti da un secondo dominio (e.g., dati sintetici) così da incrementare le potenzialità del modello.

Questo progetto si pone l'obiettivo di condurre ulteriori analisi sul modello e i processi adottati dall'articolo di giornale RefRec al fine di migliorarne i risultati.

Capitolo 2

Lavori correlati

2.1 Deep Learning in visione artificiale

Il settore dell'intelligenza artificiale, in particolar modo quello dell'apprendimento profondo trova forte applicazione nella visione artificiale. Proprio in questa branchia dell'informatica le reti neurali di tipo convoluzionale hanno riscosso enorme successo in quanto evidenziano adeguatamente caratteristiche salienti locali da immagini e nuvole di punti permettendo ai modelli di apprendere schemi ricorrenti ben definiti. Quest'abilità rende possibile, attraverso un elaborazione in coda, portare a termine compiti di natura diversa (e.g., classificazione, segmentazione, tracciamento etc) con successo.

2.2 Supervised Learning

L'apprendimento supervisionato (SL) è una tecnica di addestramento basata su dati (input) ed annotazioni (output). Durante la fase iniziale di apprendimento, ciascuna coppia input-output viene fornita al modello di Machine Learning affinché esso possa apprendere una funzione la quale mappi i dati in ingresso nei corretti valori in uscita. Successivamente, il modello viene testato su coppie input-output mai viste fino a questo momento al fine di riuscire a capire quanto esso riesca a generalizzare su nuovi dati. Maggiore è la capacità di generalizzazione, migliore risulterà il modello sotto analisi.

Esso è uno dei più popolari metodi utilizzati in contesti come la classificazione e la regressione in quanto ha permesso di ottenere modelli di Machine

Learning molto buoni in scenari di diversa natura [2, 3]. A discapito delle eccezionali performance, tale tecnica richiede un enorme quantità di dati correlati di annotazioni poste manualmente da esperti di ogni settore generando dunque un collo di bottiglia non trascurabile.

2.3 Self-supervised Learning

L'apprendimento auto-supervisionato (SSL) è definito nel campo del Machine Learning come: *la macchina prevede qualsiasi parte del dato in ingresso per ciascuna parte osservata.*
Le idee principali dietro il concetto di SSL possono essere riassunte come:

1. Le etichette di supervisione sono generate dai dati stessi invece che tramite annotazione umana.

2. Il modello prevede parti dei dati da altre parti [4].

Questa tecnica di apprendimento permette quindi di alleviare la forte richiesta di dati etichettati manualmente, facendo così fronte al rallentamento posto dall'apprendimento supervisionato. Tuttavia, le performance di modelli addestrati in questo modo sono minori rispetto ad SL.

2.3.1 SSL in NLP

SSL è fortemente adottato nel settore dell'elaborazione del linguaggio naturale (NLP). Metodi SSL generativi come BERT [5] hanno riscosso un enorme successo attraverso l'utilizzo di un'attività preliminare atta al mascheramento di token in ingresso e al successivo pre-addestramento per prevedere i vocaboli originali.

2.3.2 SSL in Computer Vision

Inoltre, SSL trova importanti applicazioni nella visione artificiale tramite l'uso di metodi di SSL contrastanti [6, 7, 8, 9, 10] per discriminare il grado di somiglianza tra diverse immagini aumentate. Tali modelli hanno dominato la scena fino a quando tecniche di SSL generative [11, 12, 13] hanno portato prestazioni più competitive. Ad esempio, MAE [11] maschera casualmente le patch in ingresso e si occupa di recuperare tali parti nello spazio dei pixel

durante il pre-addestramento.
SSL è stato anche ampiamente studiato per l'apprendimento della rappresentazione delle nuvole di punti [14, 15, 16, 17, 18, 19, 20]. Tra questi, DepthContrast [15] imposta un'attività di discriminazione delle istanze per due versioni aumentate di una nuvola di punti in ingresso. OcCo [14] tenta di recuperare la nuvola di punti originale dalla point cloud occlusa nelle diverse viste della telecamera. IAE [19] adotta un autoencoder per ricostruire le caratteristiche implicite dalle point cloud aumentate. Point-BERT [20] propone una strategia di pre-addestramento in stile BERT nella quale si mascherano i token in input, mirando poi a prevederli tramite l'assistenza di dVAE [21].

2.4 Unsupervised Learning

L'apprendimento non supervisionato è un paradigma di apprendimento automatico per problemi in cui i dati disponibili sono costituiti da esempi senza etichetta associata. L'obiettivo di questi algoritmi è l'estrazione di proprietà strutturali interessanti dai dati. Esempi di apprendimento senza supervisione sono il raggruppamento, la riduzione delle dimensioni e la stima della densità.

Questo meccanismo di apprendimento trova applicazione in contesti di visione artificiale [22, 23], recupero di informazioni [24], neuroscienze [25] ecc.

2.5 Unsupervised Domain Adaptation

L'adattamento ad un nuovo dominio non supervisionato mira a ridurre la necessità di grandi quantità di dati annotati. L'idea chiave consiste nell'apprendere caratteristiche distintive nel dominio source e sfruttare tale rappresentazione nel dominio target. Molteplici lavori sono stati condotti per la classificazione di immagini [26, 27, 28, 29, 30], segmentazione [31, 32, 33, 34] e rilevamento di oggetti [35, 36, 37, 38]. L'approccio più comunemente adottato quando si fa riferimento all'adattamento ad un nuovo dominio non supervisionato consiste nel ridurre al minimo la discrepanza tra i domini così da evidenziare caratteristiche invarianti ad essi. Questo permette ad un classificatore di ottenere una buona performance in entrambi i contesti.

2.6 Unsupervised 3D Domain Adaptation

L'adattamento ad un nuovo dominio non superivsionato per la classificazione di nuvole di punti è un tema non ampliamente discusso all'interno della comunità scientifica. Tra i lavori condotti troviamo PointDAN [39], in cui viene proposto un classico approccio per adattamento ad un nuovo dominio 2D alterato al contesto 3D. Diversamente [40, 41], sfruttano un meccanismo di apprendimento auto-supervisionato per imparare simultaneamente caratteristiche distintive di entrambi i domini. Nel dettaglio [40] utilizza l'apprendimento auto-supervisionato introducendo un nuovo procedimento per estrarre forti caratteristiche dal dominio target.

2.7 Classificazione di nuvole di punti

Numerosi lavori di ricerca sono stati condotti per trovare architetture o metodologie in grado di riconoscere e classificare correttamente nuvole di punti. Particolare interesse destano i modelli di **PointNet** [42], **PointNet++** [43], **DGCNN** [44] e **Point-MAE** [45]. Quest'ultimo, in particolare, nasce come l'unione di **autoencoder mascherato** [46] e dell'architettura **Transformer** [47].

2.7.1 PointNet

Solitamente, a causa del formato irregolare delle point cloud, la maggior parte dei ricercatori trasforma tali dati in griglie voxel 3D o raccolte di immagini prima di inviarle ad una rete neurale. Questo procedimento rende i dati voluminosi e causa spesso problemi al loro processamento (e.g., presenza di artefatti di quantizzazioni che possono oscurare le naturali invarianze dei dati). Le nuvole di punti, invece, essendo strutture di dati semplici ed unificate permettono di evitare le complessità presenti in altre rappresentazioni 3D e dunque rendono più facile il meccanismo di apprendimento sulle stesse.

PointNet [42], invece, prende direttamente in ingresso nuvole di punti rispettando l'invarianza di permutazione dei punti stessi ed esegue elaborazioni robuste a perturbazioni e/o corruzioni dei dati affinché se ne ottenga una rappresentazione in uno spazio ausiliario per poi restituire in uscita un set di valori che possono assumere significati diversi a seguito dell'obiettivo da

perseguire (probabilità di ogni classe per la classificazione e segmenti di point cloud per la segmentazione).

PointNet nel dettaglio

Concretamente, PointNet riceve in ingresso una nuvola di punti in cui ciascun elemento è rappresentato dalle sue tre coordinate (x, y, x). Ciascun punto verrà elaborato dalla rete in maniera indipendente attraverso un meccanismo di max pooling il quale permette di imparare funzioni/criteri di ottimizzazione che selezionino elementi informativi codificando il motivo della loro scelta. A valle, i valori ottimali precedentemente appresi vengono aggregati per generare il descrittore globale rappresentate l'intero oggetto e si procede ad utilizzare tale codifica per risolvere problemi di classificazione o segmentazione.

L'architettura implementata permette a PointNet di elaborare una sorta di riassunto della point cloud, paragonabile allo scheletro dell'oggetto che stiamo analizzando, per poi concentrarsi su tale artefatto per ulteriori analisi. Tale comportamento risulta robusto alla presenza di punti anomali o punti mancanti.

Di seguito è rappresentata l'architettura del modello PointNet (Figura 2.1).

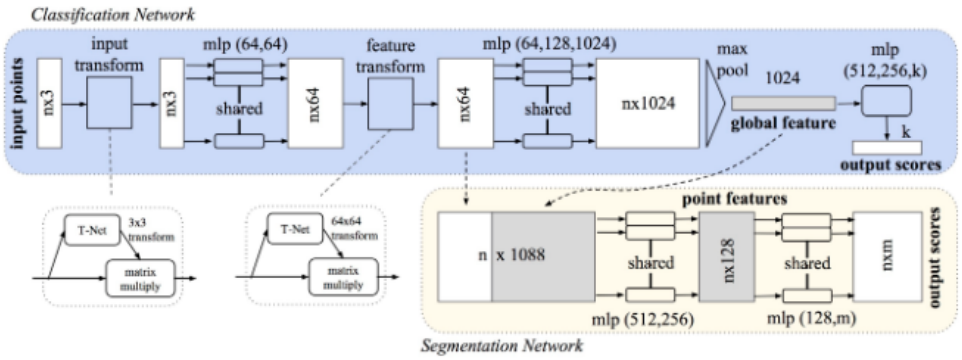


Figura 2.1: Architettura di PointNet

2.7.2 PointNet++

PointNet++ [43] si pone come versione migliore di PointNet.
Infatti, PointNet non cattura il valore informativo delle strutture locali delle point cloud, limitando la sua capacità di riconoscere schemi a grana fine e dunque l'abilità di generalizzare durante l'analisi scene complesse. Viceversa, PointNet++, costruita come rete neurale gerarchica in cui si applica ricorsivamente il modello PointNet su vari partizionamenti della nuvola di punti, permette di apprendere e pesare caratteristiche locali. Quest'ultime vengono contestualizzate alla scala utilizzata, identificando quindi un modello in grado di operare in modo adattivo a schemi locali multi-scala.

Gli autori di PointNet++ hanno mostrato che la rete è in grado di apprendere informazioni da set di punti in modo efficiente e robusto, ottenendo risultati significativamente migliori rispetto allo stato dell'arte preesistente.

PointNet++ nel dettaglio

L'idea alla base di PointNet++ è semplice. Inizialmente si suddivide l'insieme di punti in regioni locali sovrapposte utilizzando una metrica di distanza nello spazio di rappresentazione. Similmente alle reti convoluzionali, si estraggono caratteristiche locali catturando strutture geometriche per ciascuna regione. Esse vengono ulteriormente raggruppate in unità più grandi e trasformate per produrre caratteristiche di livello superiore. Questo procedimento si ripete fino a quando non si ottengono le caratteristiche dell'intero set di punti (caratteristiche globali).

Il modello adottato per apprendere caratteristiche locali, come citato precedentemente, risulta essere PointNet in quanto robusto al danneggiamento dei dati ed efficace per l'estrazione di caratteristiche semantiche di un insieme non ordinato di punti.

Di seguito (Figura 2.2) è rappresentata l'architettura di PointNet++ delineata nell'articolo di giornale.

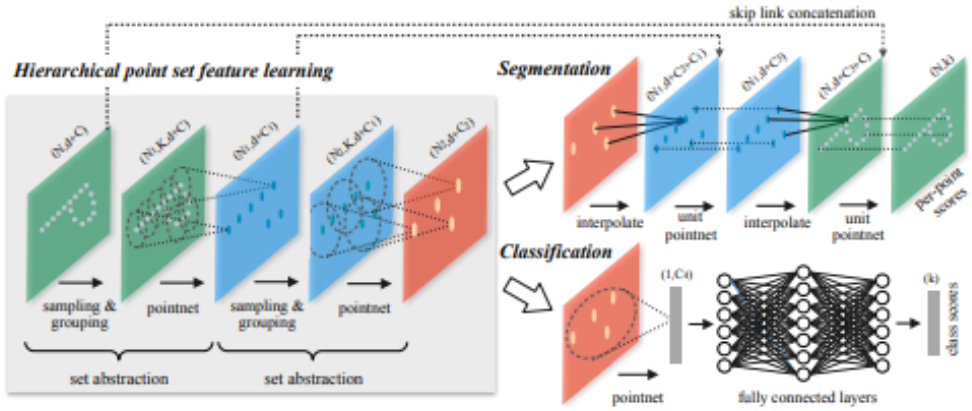


Figura 2.2: Architettura di PointNet++

2.7.3 DGCNN

Gli autori di DGCNN [44] propongono l'utilizzo del meccanismo di convoluzione adottato nelle reti neurali convoluzionali (CNN) per l'analisi di immagini nel contesto di nuvole di punti, considerando il travolgente successo raggiunto dalle CNN.

Essi infatti sostengono che, data la mancanza intrinseca di informazioni topologiche all'interno di point cloud, l'operazione di convoluzione risulta interessante per recuperare tali dati ed arricchire il potere di rappresentazione degli oggetti.

DGCNN nel dettaglio

DGCNN si compone di moduli definiti **EdgeConv** adatti a svolgere operazioni di alto livello basate su CNN per nuvole di punti. Edge Conv agisce sui grafici calcolati dinamicamente ad ogni livello della rete, è differenziabile e può essere inserito in architetture esistenti. Inoltre, presenta proprietà interessanti quali, ad esempio, l'abilità di incorporare informazioni sul vicinato locale catturando la struttura geometrica e la possibilità di essere impilato per apprendere proprietà globali della forma.

Di seguito (Figura 2.3) è rappresentata l'architettura di DGCNN delineata nell'articolo di giornale.

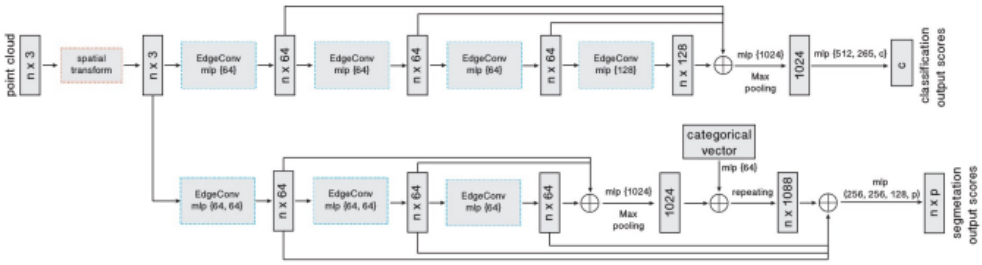


Figura 2.3: Architettura di DGCNN

2.7.4 Transformer

Il modello Transformer ideato nell'articolo di giornale **Attention Is All You Need** [47] risulta essere altresì interessante per lo scopo di questo progetto.

Transformer è adottato nell'analisi di sequenze di dati (e.g., NLP, serie temporali), sostitutivo a reti neurali ricorrenti o convoluzionali. Tuttavia, in molti studi è stata validata la sua applicazione anche nel settore della Computer Vision (e.g., ViT [48]).

Esso si basa su una configurazione del tipo **autoencoder (encoder-decoder)** arricchita dal meccanismo di attenzione, eliminando completamente qualsiasi elemento ricorrente o operazione di convoluzione. La sue struttura permette un alto livello di parallelizzazione con conseguente abbattimento di tempi per la fase di addestramento.

Transformer nel dettaglio

La parte di **encoder** è composta da una pila di N strati identici. Ogni strato ha due sotto-strati. Il primo di essi attua un meccanismo di auto-attenzione a più teste, mentre il secondo è una semplice rete feed-forward fully connected per la gestione della posizione. Attorno a ciascuno dei due

strati viene applicata una connessione residua seguita dalla normalizzazione dello strato. Questo si traduce in:

$$output\_sotto\_strato = norm\_strato(x + sotto\_strato(x))$$

dove $sotto\_strato(x)$ è la funzione implementata dal sotto-strato stesso.

La parte di **decoder** è anch'essa composta da una pila di N strati identici. Oltre ai due sotto-strati di ogni strato dell'encoder, il decoder introduce un terzo sotto-strato che esegue il meccanismo di attenzione a più teste sull'output prodotto dalla fase di codifica. Similmente all'encoder, si utilizzano connessioni residue attorno a ciascuno dei sotto-strati seguite dalla normalizzazione dello strato. Inoltre, il meccanismo di auto-attenzione del decoder è leggermente diverso a quello dell'encoder per impedire l'influenza delle posizioni successive. Questa modifica, assieme al fatto che gli embedding in uscita sono sfalsati di una posizione, assicura che le previsioni per la posizione i-esima dipendando solo dagli output noti con posizioni inferiori alla i-esima. Di seguito (Figura 2.4) è rappresentata l'architettura del Transformer.

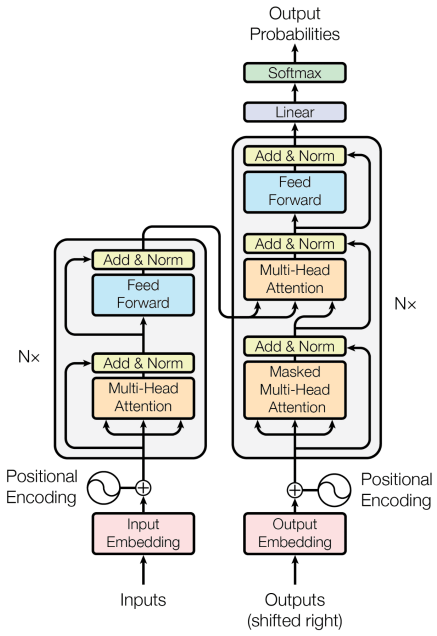


Figura 2.4: Architettura di Transformer

Meccanismo di attenzione

Il **meccanismo di attenzione** può esser descritto come una mappatura di una query e di un insieme di coppie chiave-valore su un output, dove la query, le chiavi, i valori e l'output sono tutti vettori. L'output è calcolato come somma pesata dei valori, dove il peso assegnato a ciascun valore è calcolato tramite una funzione di compatibilità della query con la chiave corrispondente.

Il meccanismo di attenzione adottato all'interno del Transformer è rappresentato come segue (Figura 2.5), considerando Q, K, V come query, chiavi e valori rispettivamente.

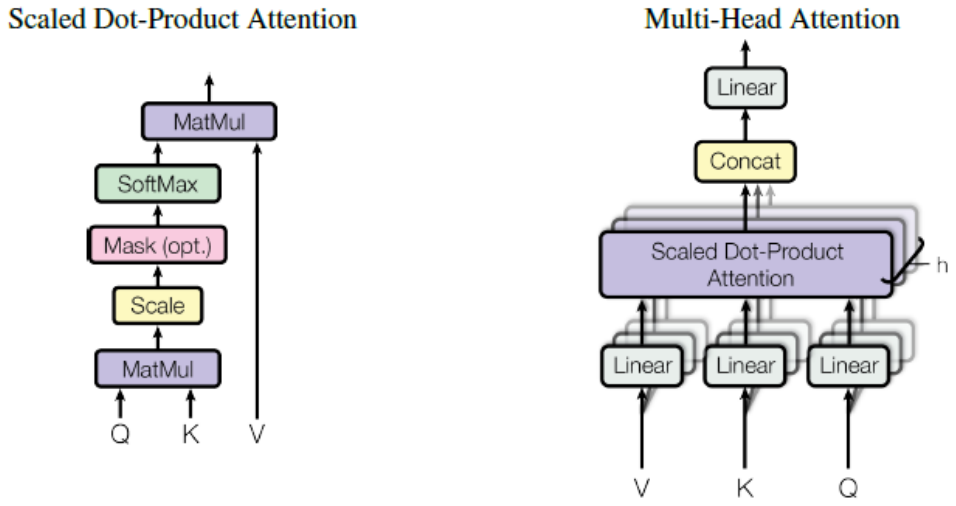


Figura 2.5: Attenzione in Transformer

2.7.5 Autoencoder mascherato

Tra i recenti studi relativi a modelli di apprendimento auto-supervisionati in visione artificiale spicca l'autoencoder mascherato [46].

Esso si basa sul mascheramento di patch casuali di un'immagine e su un'architettura autoencoder asimmetrica per la ricostruzione di tali parti. L'addestramento permette al modello di riconoscere ed apprendere caratteristiche delle immagini al fine di utilizzare tale conoscenza per ulteriori obiettivi (e.g., classificazione).

Autoencoder mascherato nel dettaglio

A differenza dei classici autoencoder, l'autoencoder mascherato adotta un design asimmetrico costituito da un encoder operante solo sull'input visibile (senza considerare la parte mascherata) ed un decoder leggero che ricostruisce l'intero input a partire dalla codifica uscente dall'encoder e dalle parti mascherate.

La parte di **encoder** è costituita da un'architettura ViT [48] applicata solamente alle parti visibili (non mascherate). Come un ViT standard, l'encoder codifica le parti attraverso una proiezione lineare con l'aggiunta di embedding di posizione ed elabora l'insieme risultante tramite una serie di blocchi Transformer. Eseguendo la fase di codifica esclusivamente sulle parti visibili dell'immagine si ha la possibilità di gestire grandi quantità di dati in una breve durata di tempo.

Il **decoder**, invece, riceve in ingresso l'intero set composto da parti codificate e da parti mascherate. Ciascuna parte mascherata è un vettore condiviso e appreso che indica la presenza di una parte mancante da prevedere. Ad ogni elemento del set vengono aggiunti embedding di posizione affinché le parti mascherate abbiano informazioni riguardo alla locazione nell'immagine. Il decoder viene utilizzato solo durante la fase di ricostruzione dell'immagine e anch'esso fa utilizzo di un'altra serie di blocchi Transformer. Infatti, solamente l'encoder è utilizzato per produrre rappresentazioni delle immagini per succcessive elaborazioni. Questo permette di progettare l'architettura del decoder in modo flessibile, essendo essa indipendente dall'encoder.

In Figura 2.6 è raffigurato il processo di ricostruzione in autoencoder mascherato.

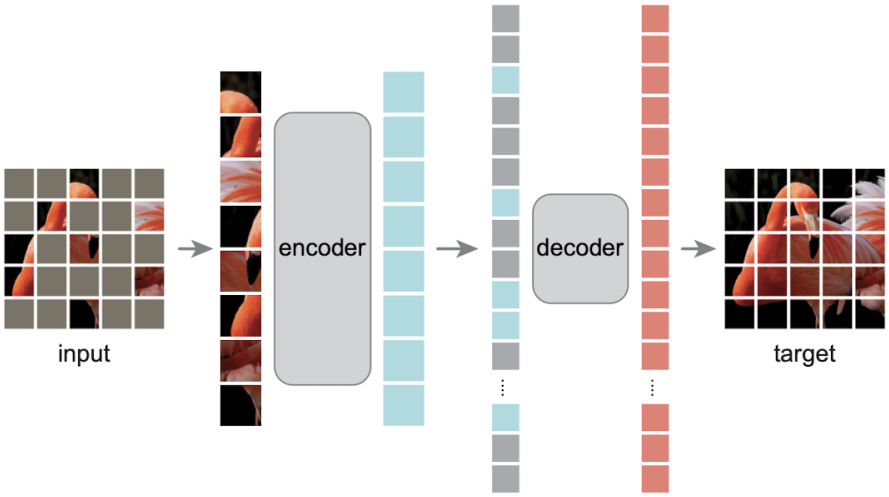


Figura 2.6: Ricostruzione in autoencoder mascherato

2.7.6 Point-MAE

Point-MAE, implementato nell'articolo di giornale **Masked Autoencoders for Point Cloud Self-supervised Learning** [45], consiste in un modello di deep learning basato su Transformer con tecnica di mascheramento. Questa architettura permette un'adeguata gestione delle sfide poste dalle nuvole di punti quali, ad esempio, la perdita di informazioni sulla posizione e la densità di informazioni irregolare.

Point-MAE nel dettaglio

Concretamente, si divide la point cloud in più patch di punti irregolari utilizzando FPS (Farthest Point Sampling) per identificare i centri e KNN (K-Nearest Neighbors) per generare le patch. Successivamente, si procede

con il mascheramento casuale della maggior parte di esse, per poi procedere alla trasformazione in token (visibili e mascherati). I token visibili vengono codificati da un encoder in grado di evidenziare le caratteristiche latenti di ognuno, mentre i token mascherati ricevono l'aggiunta di dati relativi alla posizione affinché la loro ricostruzione risulti ben localizzata.

In secondo luogo, i token codificati e quelli mascherati vengono gestiti da un decoder unito ad una testa di predizione a singolo strato per produrre patch mascherate ricostruite, le quali permettono di calcolare valori di perdita se confrontate con le patch mascherate di ground truth.

L'architettura del modello Point-MAE è delineata in Figura 2.7.

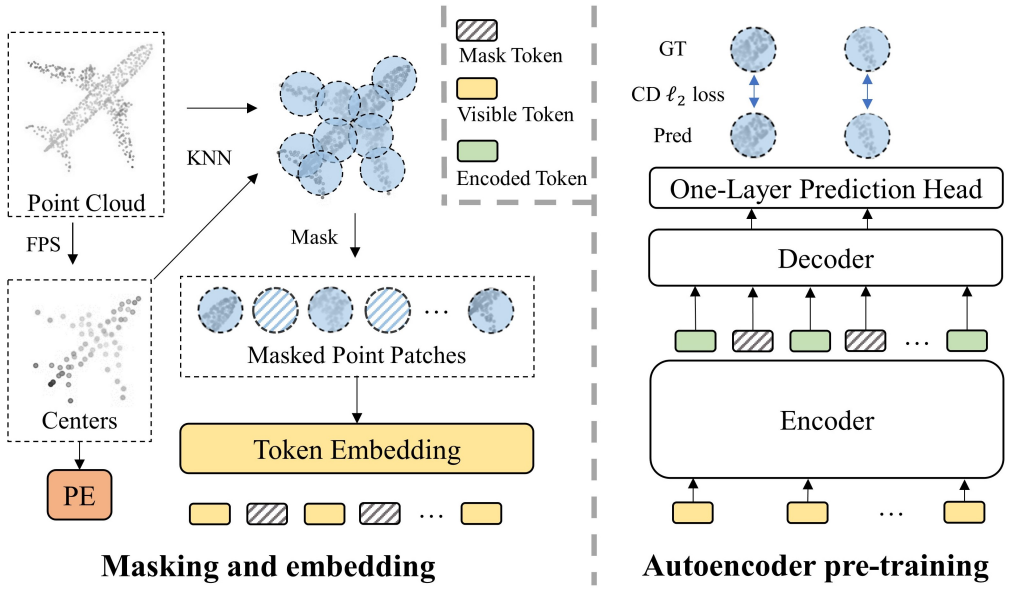


Figura 2.7: Architettura di Point-MAE

Capitolo 3

RefRec

Come delineato nell'introduzione, le fondamenta di questo progetto risiedono nel paper **RefRec: Pseudo-labels Refinement via Shape Reconstruction for Unsupervised 3D Domain Adaptation** [1]. In esso, gli autori identificano una soluzione brillante per il problema di adattamento a nuovi domini non supervisionato (UDA) relativo alla classificazione di nuvole di punti. Infatti, l'approccio adottato in RefRec differisce dal percorso standard costituito dall'apprendimento multi-task per allineare le caratteristiche tra i domini basandosi invece sull'analisi di pseudo-annotazioni e auto-addestramento. Le idee principali per rendere efficace l'auto-addestramento sono:

i Raffinamento di pseudo-annotazioni rumorose facendo corrispondere i descrittori di forma che vengono appresi tramite l'addestramento non supervisionato di ricostruzione della forma su entrambi i domini.

ii Un nuovo protocollo di auto-addestramento che apprende i confini decisionali specifici del dominio e riduce l'impatto negativo di campioni target etichettati in modo errato e la variabilità intraclasse all'interno del dominio

3.1 Introduzione

E' importante ragionare correttamente su dati geometrici 3D come nuvole di punti o mesh per affrontare molti problemi di Computer Vision, i quali sono fondamentali per gestire propriamente applicazioni come guida autonoma,

percezione robotica e realtà aumentata. In particolare, un'abilità richiesta da un sistema intelligente è il riconoscimento in una categoria semantica di un insieme di punti rappresentanti la superficie di un oggetto di una scena. La classificazione delle forme è stata inizialmente gestita utilizzando caratteristiche artigianali [49, 50], mentre, con in progressi in Deep Learning, le proposte recenti vertono sull'uso di reti neurali profonde per apprendere caratteristiche dalle stesse [42, 43, 51, 52, 53, 54].

Sebbene i risultati ottenuti tramite approcci basati sui dati sono impressionanti, l'enorme quantità di dati annotati richiesta necessita di molto tempo per la raccolta degli stessi nonché di spazio sufficiente per la memorizzazione. Tipicamente, si fa utilizzo di dataset sintetici contenenti modelli CAD 3D per collezionare velocemente una grande quantità di dati. Tuttavia, benché i modelli addestrati su questi dataset risultano molto efficaci sulla classificazione di nuovi dati sintetici come testimoniato in [55, 56], essi non riescono a trasferire le loro prestazioni a scenari del mondo reale [55] dove i dati sono solitamente catturati da sensori RGB-D o LiDAR [57, 58]. Questa condizione limita fortemente la distribuzione di metodi di deep learning 3D in applicazioni del mondo reale. L'apprendimento non supervisionato a nuovi domini (UDA) mira a colmare la mancata capacità di trasferimento delle conoscenze [59] acquisite in un dominio annotato definito sorgente o source in un secondo dominio definito target o di destinazione.

UDA nasce nella visione artificiale 2D, dove sono stati proposti molteplici metodi [60]. Tra questi, l'approccio più diffuso l'allineamento globale delle distribuzioni di caratteristiche tra il dominio source e target. Questo paradigma è sfruttato anche in caso di UDA con dati 3D utilizzando esplicitamente appositi modelli e funzioni di perdita per allineare le caratteristiche [39] oppure in modo implicito attraverso la risoluzione di compiti auto-supervisionati sui domini source e target con un encoder condiviso [41, 40]. Muovendosi da un dataset sintetico ad un reale l'allineamento delle features può portare a soluzioni non ottimali a causa delle grandi differenze tra i due domini. Infatti, le scansioni di oggetti reali risultano spesso parziali e carenti di parti significative a causa di occlusioni dovute al disordine presente nelle scene. Inoltre, durante la fusione di più scansioni 2.5D [61] per la creazione di una forma 3D completa, si verificano spesso errori di registrazione, i quali, uniti al rumore interno del sensore, producono oggetti meno chiari e geometricamen-

te coerenti rispetto ai modelli CAD. Il classificatore, quindi, dovrà fondare la sua decisione su nuovi spunti diversi da quelli appresi sul dominio sorgente sintetico in cui sono presenti forme complete e chiare al fine di riconoscere correttamente i campioni target.

Per consentire al classificatore di apprendere questi nuovi spunti direttamente dal dominio target, in RefRec (Refinement via Reconstruction) si fa affidamento sulle pseudo-annotazioni [62], ovvero predizioni sul dominio target ottenute tramite inferenza di un modello precedentemente addestrato su dominio source, utilizzate come supervisione rumorosa per l'addestramento di un classificatore sul dominio di destinazione. Tale processo viene identificato con il nome di auto-addestramento [62, 63]. Tuttavia, il classificatore addestrato su dominio sorgente potrebbe produrre pseudo-annotazioni su dominio target errate a cause dello spostamento del dominio. Pertanto, si introducono fasi di perfezionamento delle pseudo-annotazioni prima (offline) e durante (online) l'auto-addestramento. Entrambe le fasi si basano sull'idea che forme simili dovrebbero condividere etichette simili. Per trovare forme simili si abbinano i descrittori di forma globali, ovvero gli embedding prodotti dall'encoder data una forma in ingresso.

Al fine di ottenere un encoder che produca embedding ragionevoli ovvero che forniscano una rappresentazione compatta e distintiva delle strutture geometriche dei dati in ingresso si utilizza una tecnica efficace di addestramento basata sulla ricostruzione delle nuvole di punti come provato in proposte recenti di descrittori di forma locale e globale [64, 65, 66]. Questo processo può includere anche il dominio target in quanto avviene in modo non supervisionato.

Con l'ausilio degli embedding nella fase offline si procede alla riassegnazione delle pseudo-annotazioni dei campioni target in cui il classificatore addestrato nel dominio sorgente mostra una bassa confidenza, mentre nella fase online si pesano le pseudo-annotazioni in base alla somiglianza dell'embedding della forma in input con il suo prototipo [67], ovvero il descrittore globale medio sul dominio target calcolato per ogni classe. L'utilizzo di prototipi calcolati tramite embedding derivanti da ricostruzione comprendente anche il dominio target permette di evitare lo spostamento di dominio sostenuto quando si adopera il classificatore addestrato su dominio source come fatto dai pre-

cedenti metodi 2D [68]. Inoltre, nella fase online, viene sfruttato anche il protocollo di addestramento standard dei metodi UDA 2D basati su mean teacher [69] per migliorare la qualità delle pseudo-annotazione man mano che l'apprendimento progredisce.

I contributi di RefRec possono essere riassunti in:

1. Utilizzo di un nuovo approccio per risolvere UDA nel caso di nuvole di punti, il quale differisce nettamente delle proposte esistenti in letteratura basate sull'apprendimento multi-task.

2. Dimostrare come i descrittori globali appresi attraverso la ricostruzione delle forme possano essere utilizzati efficacemente sia per perfezionare in modo offline che online le pseudo-annotazioni.

3. Mostrare come tecniche efficaci per UDA 2D, come classificatori specifici del dominio e supervisione del mean teacher, possano essere utilizzati con successo su dati 3D.

4. Ottenere prestazioni all'avanguardia sui parametri di riferimento standard utilizzati per valutare i progressi in UDA per la classificazione delle nuvole di punti.

3.2 Dataset

I dataset contenenti le nuvole di punti per le fasi di addestramento e di testing adottati da RefRec sono sia relativi ad oggetti sintetici (ModelNet40 [70], ShapeNet [71]) che reali (ScanNet [72]). Nel dettaglio, sono state utilizzate delle versioni ridotte dei dataset citati denominate ModelNet-10 (M), ShapeNet-10 (S), ScanNet-10 (S*) come avviene in PointDAN [39] la cui visualizzazione e le cui informazioni sono presenti in Figura 3.1 e Tabella 3.1 rispettivamente. Questi tre dataset condividono le stesse 10 categorie di oggetti per permettere la valutazione di metodi di Domain Adaptation.

ModelNet-10 (M) contiene 10 categorie di modelli CAD 3D estratti dalle 40 categorie di Modelnet40 unendo le classi che presentano la stessa struttura come ad esempio il comodino ed il mobile.
ShapeNet-10 (S) contiene 10 categorie di modelli CAD 3D estratti dalle

55 categorie di ShapeNet. ShapeNet contiene più campioni e i suoi oggetti hanno una maggiore varianza nella struttura rispetto a ModelNet. Si applica un campionamento uniforme per raccogliere i punti di ShapeNet in superficie, che, rispetto a ModelNet, potrebbe perdere qualche punto marginale.
ScanNet-10 (S*) isola 10 categorie di oggetti del mondo reale scansionate e ricostruite presenti in ScanNet. Gli oggetti presenti perdono spesso alcune parti a causa di occlusioni dell'ambiente circostante rendendo questo dominio molto impegnativo.

Dataset	bathtub	bed	bookshelf	cabinet	chair	lamp	monitor	plant	sofa	table	Total
M Train	106	515	572	200	889	124	465	240	680	392	4,183
M Test	50	100	100	86	100	20	100	100	100	100	856
S Train	599	167	310	1,076	4,612	1,620	762	158	2,198	5,876	17,378
S Test	85	23	50	126	662	232	112	30	330	842	2,492
S* Train	98	329	464	650	2,578	161	210	88	495	1,037	6,110
S* Test	26	85	146	149	801	41	61	25	134	301	1,769

Tabella 3.1: Numero di esempi in M, S, S*

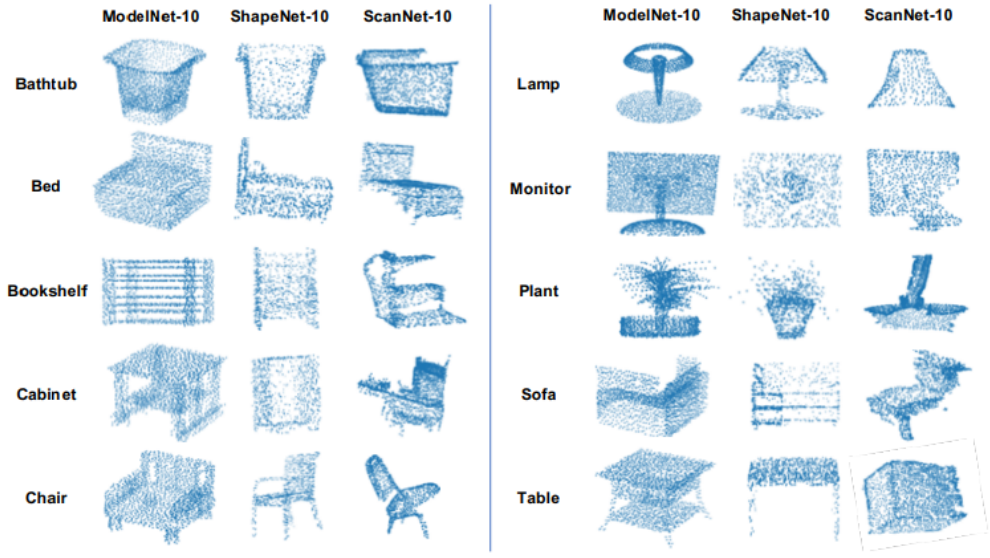


Figura 3.1: Campioni di nuvole di punti M, S, S*

3.3 Passo 1: warmup per generazione di pseudo-annotazioni

Nel passo iniziale di RefRec si ha come obiettivo la generazione di **pseudo-annotazioni** iniziali. Esso, definito *warmup* dagli autori del paper, comprende due sotto-passi: **ricostruzione** (Sezione 3.3.1 e **classificazione** (Sezione 3.3.2).
Tale passo risulta importante in quanto migliori saranno le pseudo-annotazioni iniziali, migliore sarà la capacità di predizione del modello sul dominio target dopo le fasi di raffinamento offline, auto-addestramento e raffinamento online.
Si consideri target il dominio con oggetti reali non etichettati e source il dominio con oggetti sintetici etichettati.

3.3.1 Sotto-passo 1: Ricostruzione

Inizialmente, si addestra in modo non supervisionato una rete di ricostruzione composta da un encoder PointNet [42] ed un semplice decoder costituito da 3 strati fully connected per ricostruire la forma originale nei domini target e source utilizzando nuvole di punti aumentate e minimizzando **Chamfer Distance (CD)** ed **Earth Mover's Distance (EMD)** [73] come funzioni di perdita. Utilizzare tecniche di data augmentation è molto importante soprattutto nello scenario **sintetico-reale** per incrementare la capacità di generalizzazione in quanto le nuvole di punti di oggetti reali sono spesso non uniformi e occluse in alcune parti.

Questo addestramento permette di ottenere una funzione di trasformazione dallo spazio 3D iniziale ad uno spazio compatto (spazio embedding) in cui gli oggetti di entrambi i domini sono distribuiti in accordo alla loro struttura e caratteristiche. La scelta di utilizzare dominio source e target durante questo sotto-passo permette di apprendere caratteristiche interessanti di entrambi e di aumentare quindi la capacità di generalizzazione (qualora si escludesse il dominio target, la rete potrebbe ritrovarsi in una situazione di overfitting nel dominio source).

3.3.2 Sotto-passo 2: Classificazione

Addestrato in modello in ricostruzione, si procede con la rimozione del decoder e l'aggiunta di una testa per creare un modello di classificazione. Esso verrà addestrato, in modo supervisionato, sul dominio source.

Terminato l'addestramento, il modello è adesso pronto per la generazione di pseudo-annotazioni ottenute tramite inferenze nel dominio target.

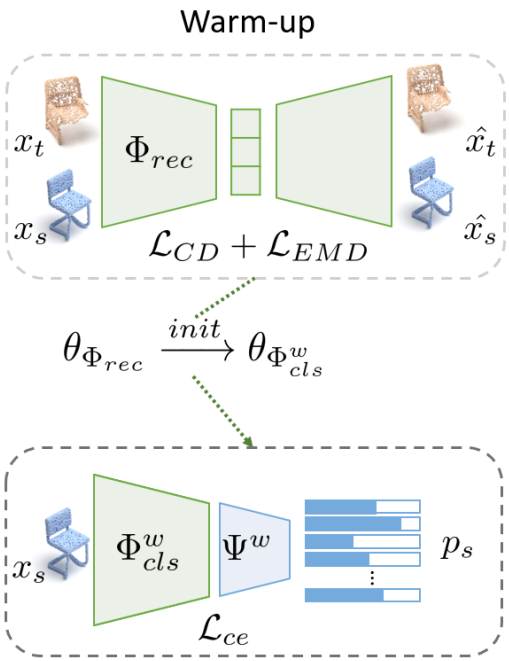


Figura 3.2: RefRec, passo 1

3.4 Passo 2: raffinatezza offline

Ricavate le pseudo-annotazioni (dominio target) al Passo 1 3.3 si procede adesso con la raffinatezza delle medesime attraverso una procedura offline (priva di fasi di addestramento). Questo passo è importante in quanto solamente una parte delle pseudo-annotazioni generate può essere considerata affidabile a causa del divario tra dominio source e target mentre alla maggioranza dei campioni target viene assegnata una classe errata.

Si creano due insiemi definiti H**(hard)** ed ε **(easy)**. L'insieme H conterrà le nuvole di punti che il classificatore ha predetto con un valore di probabilità basso, mentre l'insieme ε conterrà le point cloud predette con un valore di probabilità alto (per discriminare alto e basso non si utilizza un valore di soglia fisso, bensì si considera il 10% di predizioni più alte per ciascuna classe).

Adesso, tramite la rete di ricostruzione del passo 1, si calcola un descrittore (embedding) con dimensione 1024 per ciascuna nuvola di punti. I successivi passaggi verranno svolti nello **spazio dei descrittori** (precedentemente definito spazio compatto) in quanto esso restituisce una visione più significativa degli oggetti. Infatti, oggetti con caratteristiche o strutture simili si troveranno vicini in questo spazio e, viceversa, oggetti di natura differente saranno distanti gli uni dagli altri.

Posizionati i vari oggetti nello spazio dei descrittori, si procede ad analizzare ogni descrittore dell'insieme H. Per ciascuno di essi, si cerca nell'insieme ε il descrittore più vicino. Tale procedimento viene percorso nuovamente dal descrittore appena trovato in ε all'insieme H. Se il descrittore in H ora identificato corrisponde al descrittore iniziale considerato, allora si sposta la nuvola di punti da H a ε impostando la sua pseudo-annotazione a quella del descrittore in ε individuato. Si genera così l'insieme ε_r (raffinato).
Tuttavia, se ciò non avviene, le pseudo-annotazioni delle point cloud restanti in H vengono raffinate utilizzando **KNN**[1]. Infine, qualora non si ottenesse il consenso con KNN, si assegna come pseudo-etichetta la classe del descrittore più vicino in ε_r. Si ottiene così l'insieme H_r.

L'ultima operazione di questo passo consiste nella generazione di **prototipi** per ciascuna classe attraverso una media di classe tra le nuvole di punti dell'insieme ε.

[1]K-Nearest Neighbors: per ciascuna nuvola di punti in H si considera il suo descrittore e si cercano i K descrittori più vicini in ε_r ad esso. Se tramite votazione dei K vicini si raggiunge la maggioranza, si ha il perfezionamento della pseudo-annotazione

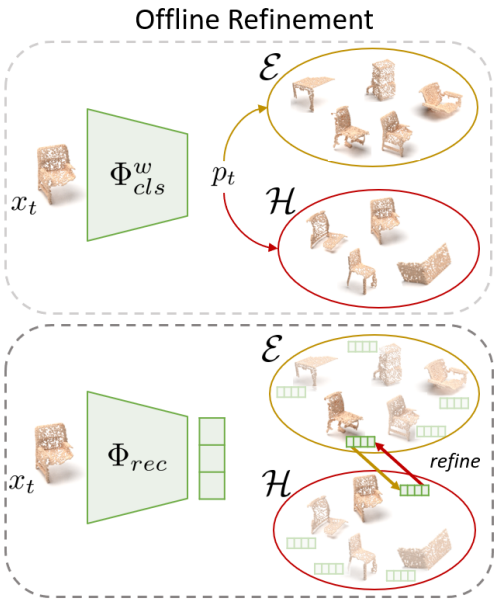


Figura 3.3: RefRec, passo 2

3.5 Passo 3: auto-addestramento e raffinatezza online

L'ultimo passo di RefRec consiste in una fase di auto-addestramento mediante le pseudo-annotazioni raffinate al passo precedente 3.4 al fine di trovare un modello abile nella classificazione di oggetti target. Esso consiste nell'utilizzo del classificatore ottenuto al Passo 1 3.3.2 al quale vengono attaccate due **teste di classificazione**: una per il dominio source e una per il dominio target.

L'auto-addestramento (supervisionato) si svolge dunque seguendo questo percorso: ad ogni passo di addestramento vengono passati alla rete due lotti diversi, uno contenente nuvole di punti relative a dominio sorgente unite ad oggetti appartenenti a ε_r e uno contente point cloud relative solo al dominio target appartenenti sia a ε_r che H_r. Quando la rete analizza lotti della primo tipo disattiva preventivamente la testa di classificazione relativa al dominio target e viceversa per lotti della seconda tipologia.

Tramite questo addestramento si evita che il modello impari dei confini tra

le varie classi per uno specifico dominio. Di fatto la visione contemporanea di due domini costringe la rete ad apprendere due diverse tipologie di confini tra classi e quindi riuscire a generalizzare meglio. Tale procedimento viene svolto perché spesso i domini sono abbastanza diversi tra di loro.

Inoltre, dal calcolo della funzione di perdita durante l'addestramento si ignorano gli oggetti che distano molto dai **prototipi** delle loro classi (ottenuti al passo 2). Tale meccanismo permette di individuare sia errori di assegnazione di pseudo-annotazioni che oggetti anomali ed evitarli.

3.5.1 Raffinatezza online

Sebbene si utilizzino due teste di classificazione specifiche per ciascun dominio, la variabilità intraclasse sul dominio target può comunque influenzare le prestazioni del modello. Per far fronte a questo problema si adotta una una strategia di raffinatezza e ponderazione delle pseudo-annotazioni online. L'intuizione chiave alla base del perfezionamento online è che, con il progredire dell'addestramento, il modello impara meglio a classificare il dominio target e dunque è possibile sfruttare tale conoscenza appena acquisita per migliorare gradualmente le pseudo-annotazioni. A questo proposito, si fa uso di **mean teacher** [69] del modello per ottenere pseudo-annotazioni raffinate online.

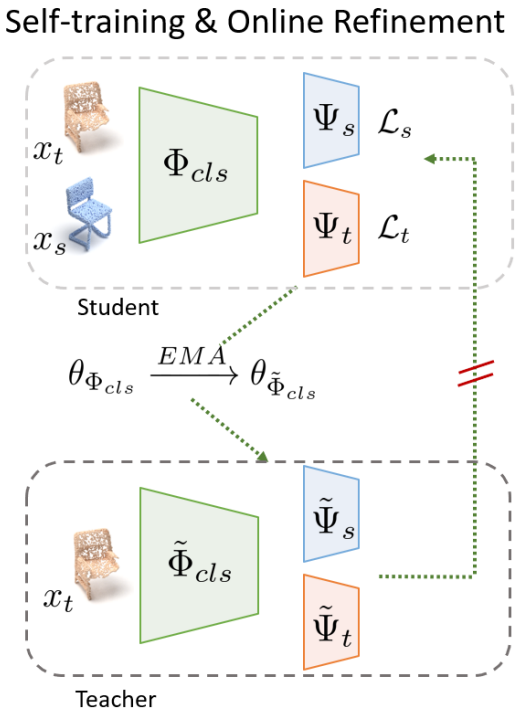


Figura 3.4: RefRec, passo 3

Capitolo 4

Risultati sperimentali

Il progetto mira a incrementare i risultati ottenuti da RefRec attraverso l'analisi dell'architettura e dei processi implementati dagli autori. Si cerca dunque di esaminare la pipeline preesistente e applicare variazioni nei singoli passi al fine di ottenere un modello di classificazione migliore o quantomeno interessante per ulteriori sviluppi.

In questo capitolo si affrontano le varie sperimentazioni intraprese, evidenziando i risultati ottenuti.

4.1 Casi di domain adaptation analizzati

I domini utilizzati per analizzare la pipeline di RefRec sono ModelNet [**?**], in veste di dominio sorgente e ScanNet [**?**] in veste di dominio target. Entrambi i dataset presentano lo stesso numero (10) e la stessa tipologia di classi (bathtub, bed, bookshelf, cabinet, chair, lamp, monitor, plant, sofa, table). L'intero progetto è stato svolto valutando questo specifico scenario di adattamento al dominio.

4.2 Metriche adottate

La metrica utilizzata per analizzare il modello in ogni suo passo è la metrica di **accuratezza**. Essa è calcolata secondo la seguente formula:

$$accuratezza = \frac{numero\ di\ esempi\ corretti}{numero\ di\ esempi\ totali}$$

Come ciascuna metrica, essa può esser analizzata considerando a livello globale ogni predizione a sé (**micro**) oppure valutando dapprima a livello di classe e per poi calcolare una media dei valori ottenuti (**macro**). Queste due visioni restituiscono punti di vista diversi della performance dello stesso modello e permettono uno studio più preciso del medesimo.

E' bene, inoltre, definire questa metrica contestualizzata al processo sotto analisi, differenziandola qualora si parli di accuratezza in **retrieval** oppure accuratezza in **classificazione**.

4.2.1 Accuratezza in retrieval

Attraverso la ricostruzione il modello neurale apprende una funzione per rappresentare le nuvole di punti nello spazio embedding. La qualità di questa funzione si può misurare tramite un'analisi dello spazio embedding definita test retrieval. Essa è delineata quanto segue.

In primo luogo, si passa dallo spazio 3D di rappresentazione delle nuvole di punti allo spazio compatto. Adesso, affinché ogni point cloud sia adeguatamente e coerentemente rappresentata in questo spazio, è necessario che oggetti della stessa classe, e dunque con caratteristiche simili, siano collocati nello stesso vicinato. Viceversa, oggetti appartenenti a classi diverse si troveranno localizzati in posizioni distanti.

Considerando quindi tale premessa, la metrica di accuratezza è calcolata attraverso l'analisi del vicinato di ciascuna nuvola di punti compatta (vicinato ottenuto tramite KNN impostando come K i valori 1,5,10). Se in esso è presente almeno un vicino appartenente alla medesima classe allora la rappresentazione di tale oggetto nello spazio compatto risulta essere corretta.

4.2.2 Accuratezza in classificazione

In classificazione il calcolo dell'accuratezza è gestito analizzando quante classi delle nuvole di punti sono state correttamente predette attraverso il confronto con le annotazioni presenti.

4.3 Analisi: PointNet

Il primo esperimento condotto in questa tesi consiste nel replicare i risultati mostrati da RefRec al fine di verificare che la pipeline sia integra in ogni suo passo. Come introdotto precedentemente, si è preso in considerazione il caso di adattamento al dominio con ModelNet dominio source e ScanNet dominio target. Quindi, sono stati eseguiti i passi di warmup per generazione di pseudo-annotazioni, raffinatezza offline, auto-addestramento e raffinatezza online ed è stato confrontato il valore di accuratezza ottenuto, calcolato sul dataset ScanNet/test, con il valore presente nel articolo di riferimento.

4.3.1 Ricostruzione

Nella fase di ricostruzione, la funzione di perdita adottata è costituita dalla somma di Chamfer Distance e Earth Mover's Distance pesata:

$$loss = loss\_CD + (loss\_EMD * weight\_EMD)$$

dove $weight\_EMD$ è impostato a 0.05 in RefRec.

L'andamento dell'addestramento, in accordo con le due funzioni di perdita citate, è di seguito riportato (Figure 4.1 e 4.2).

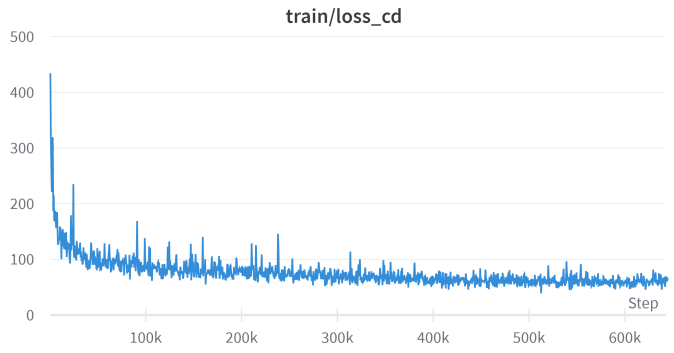


Figura 4.1: PointNet: ricostruzione Chamfer Distance in ricostruzione

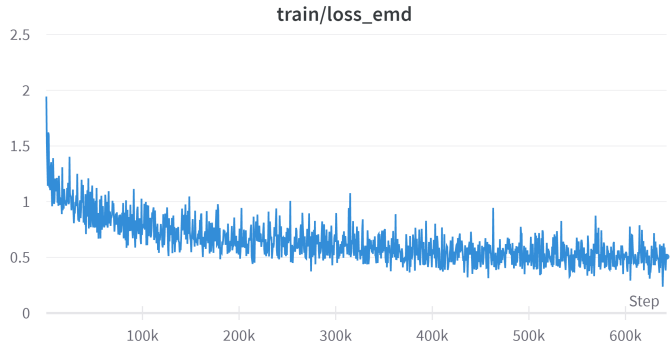
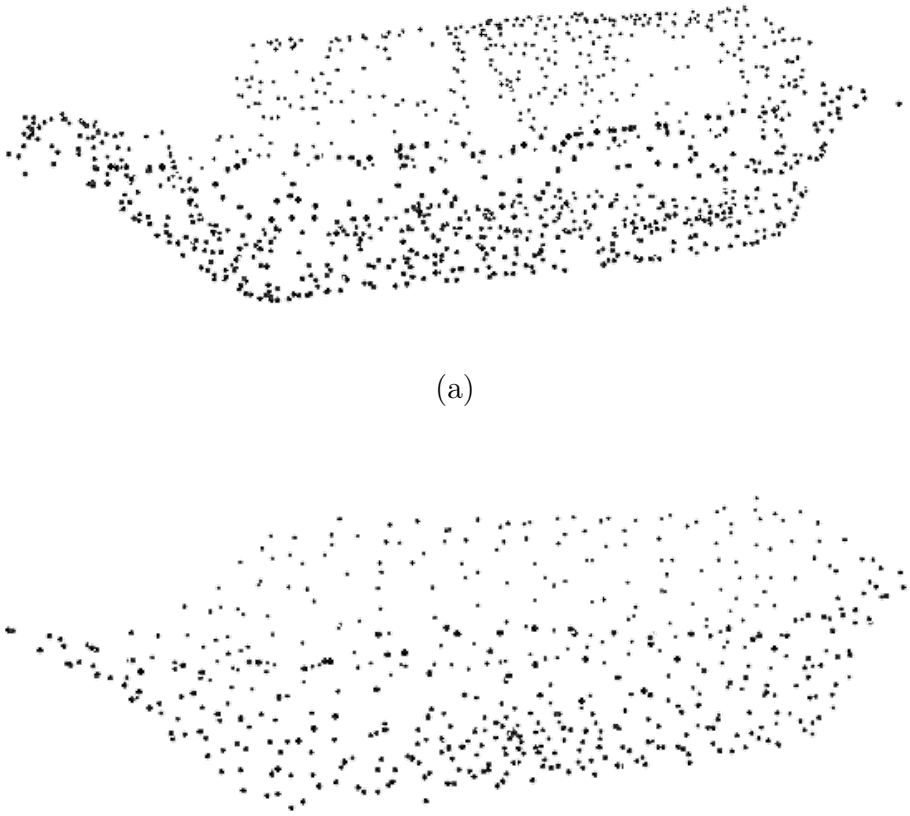


Figura 4.2: PointNet: Earth Mover's Distance in ricostruzione

Terminato l'addestramento, è stato valutato il modello ottenuto qualitativamente e quantitativamente.

Analisi qualitativa

Da un analisi qualitativa (Figure 4.3 e 4.4) è possibile notare che il modello riesce a ricostruire piuttosto adeguatamente le nuvole di punti introducendo solo una minima quantità di artefatti. Infatti, sia nella ricostruzione della vasca da bagno (b) che del mobile (d) sono presenti distribuzioni dei punti in modo più uniforme rispetto alle nuvole di punti originali, riducendo quindi le densità di alcune parti delle stesse.



(a)

(b)

Figura 4.3: PointNet: input (a) e output (b) in ricostruzione

(c)

(d)

Figura 4.4: PointNet: input (c) e output (d) in ricostruzione

Analisi quantitativa

Analizzando quantitativamente i dati di accuratezza in retrieval (ottenuti come esplicato nella Sezione 4.2.1) calcolati su ScanNet/test e presenti in Tabella 4.1 si evince che il modello ha imparato a codificare la maggior parte delle nuvole di punti nello spazio embedding. Infatti proprio in questo spazio, riferendosi solamente al caso K=1, il 72.75% (micro accuratezza) delle nuvole di punti ha il vicino più prossimo appartenente alla stessa classe. Tuttavia, dai valori di macro accuratezza inferiori, si percepisce la presenza di eterogeneità nel calcolo di accuratezza a livello di classe.

Tipo	Top1(%)	Top5(%)	Top10(%)
Micro	72.75	89.88	93.95
Macro	60.02	80.29	88.07

Tabella 4.1: PointNet: accuratezza in retrieval (1)

Tale intuizione trova conferma il valore di accuratezza per ciascuna classe presente nella Tabella 4.2. Le classi *monitor* e *lamp* sono quelle che hanno riscontrato una rappresentazione nello spazio embedding meno significativa.

Tipo - Classe	Top1(%)	Top5(%)	Top10(%)
Macro - bathtub	100.00	100.00	100.00
Macro - bed	75.41	88.57	93.51
Macro - bookshelf	69.53	84.87	91.28
Macro - cabinet	56.49	88.28	93.20
Macro - chair	90.40	97.39	98.51
Macro - lamp	12.04	52.63	82.35
Macro - monitor	48.15	67.69	78.69
Macro - plant	20.45	50.00	59.38
Macro - sofa	48.61	84.48	91.60
Macro - table	79.15	88.99	92.24

Tabella 4.2: PointNet: accuratezza in retrieval (2)

4.3.2 Warmup per generazione di pseudo-annotazioni

Il passo successivo consiste in un addestramento in classificazione sul dominio source e nelle generazione di pseudo-annotazioni sul dominio target

attraverso il modello ottenuto. Di seguito (Figure 4.5 e 4.6) sono riportati i grafici di accuratezza su ModelNet/test e ScanNet/test generati durante la fase di addestramento. E' possibile calcolare i valori di accuratezza sul dataset target poiché, in questo ambito sperimentale, esso è provvisto di annotazioni.

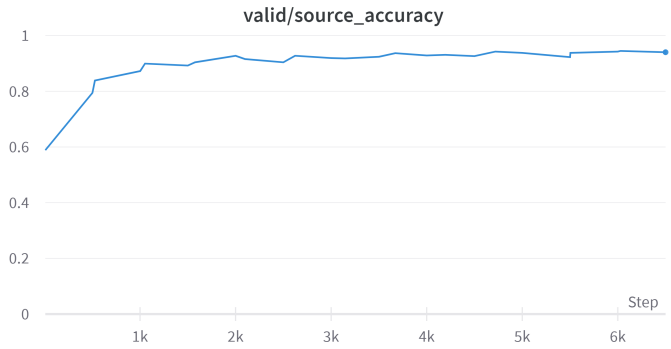


Figura 4.5: PointNet: accuratezza source in warmup

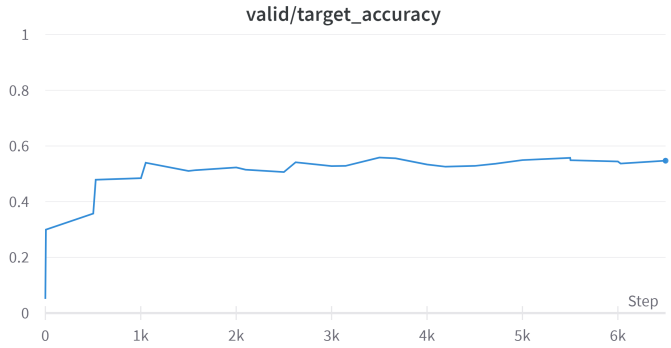


Figura 4.6: PointNet: accuratezza target in warmup

L'accuratezza migliore ottenuta su ScanNet/test risulta essere **53.70%**.

A livello di classe, attraverso la matrice di confusione in Figura 4.7 calcolata su ScanNet/test, si può notare che alcune classi sono molto più facili da riconoscere (e.g., *bathtub, chair, monitor, plant, table*) rispetto ad altre (e.g., *cabinet, lamp*). La diversa abilità nel riconoscere oggetti è dovuta al fatto che alcuni di essi presentano caratteristiche forti e ben distinte, mentre altri posseggono tratti comuni ad altre classi.

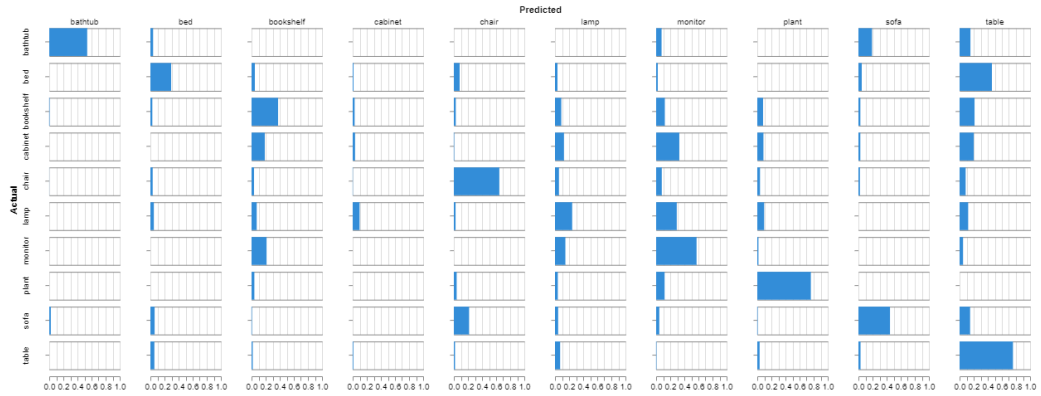


Figura 4.7: PointNet: matrice di confusione in warmup

4.3.3 Raffinatezza offline, auto-addestramento e raffinatezza online

Le pseudo-annotazioni ottenute al passo precedente subiscono un processo di raffinatezza offline al fine di rafforzare le predizioni più incerte. Questo processo fa uso di KNN con K impostato al valore 3 dagli autori di RefRec.

Le pseudo-annotazioni ottenute vengono utilizzate per addestrare nuovamente il modello tramite auto-addestramento utilizzando come punto di partenza i pesi di ricostruzione. Durante questa fase è presente anche una procedura di raffinatezza online gestita attraverso il meccanismo di mean teacher.

Si riportano in Figura 4.8 e 4.9 i grafici di accuratezza su ModelNet/test e ScanNet/test generati durante la fase di auto-addestramento e raffinatezza online.

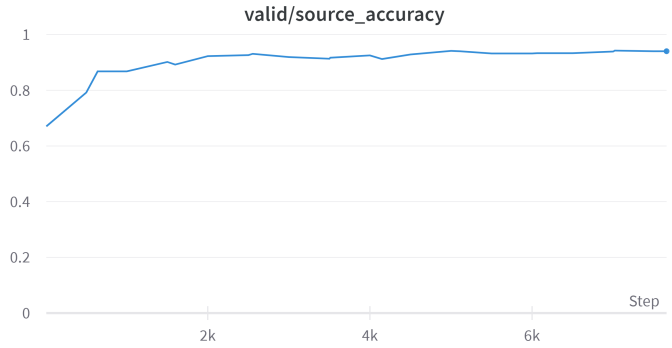


Figura 4.8: PointNet: accuratezza source in auto-addestramento

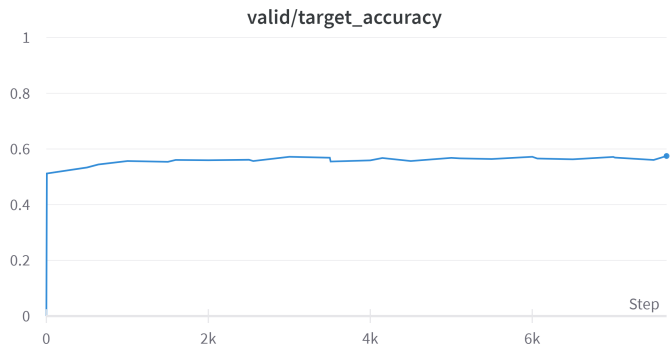


Figura 4.9: PointNet: accuratezza target in auto-addestramento

L'accuratezza migliore ottenuta su ScanNet/test risulta essere **56.92%**, paragonabile al valore **56.5%** presente in RefRec.

Inoltre, in Figura 4.10 è riportata la matrice di confusione ricavata su ScanNet/test nella quale è possibile notare un lieve miglioramento nella predizione di alcune classi (e.g., *lamp*).

4.3.4 K variabile in raffinatezza offline

Una prima modifica alla pipeline di RefRec è stata apportata cambiando il valore di K per il KNN nella raffinatezza offline al fine di raffinare le pseudo-annotazioni quanto più possibile e dunque ottenere un valore di

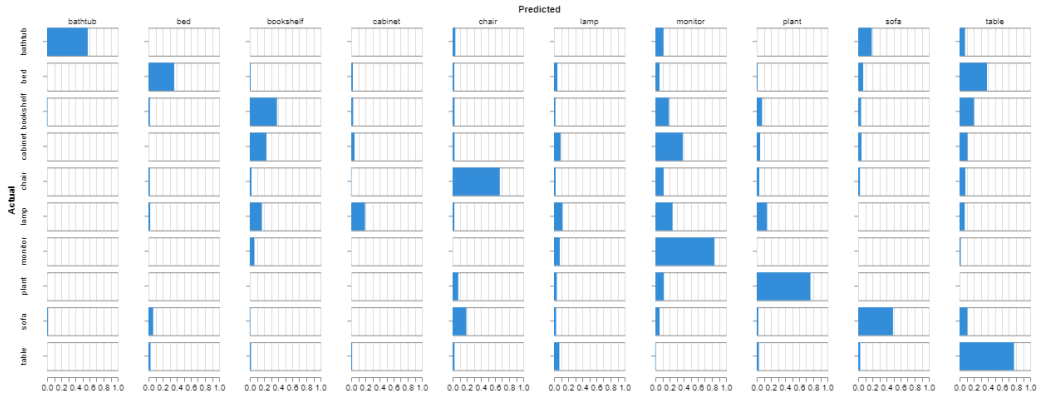


Figura 4.10: PointNet: matrice di confusione in auto-addestramento

accuratezza su ScanNet/test auspicabilmente migliore dopo la fase di auto-addestramento. Tutti i risultati ottenuti dopo l'auto-addestramento con pseudo-annotazioni raffinate con K variabile sono riportati nella Tabella 4.3.

Fase/Accuratezza	K=3	K=5	K=10
Auto-addestramento	56.92%	56.81%	56.98%

Tabella 4.3: PointNet: accuratezza dopo auto-addestramento

Il valore di accuratezza migliore, sebbene di pochi centesimi, è ottenuto dal caso K=10 con una percentuale di **56.98%**.

4.4 Analisi: PointNet++

In questa Sezione si sostituisce il modello di autoencoder utilizzato in RefRec con il modello **PointNet**++ e si vanno ad analizzare le prestazioni durante e alla fine dell'intera pipeline.

4.4.1 Ricostruzione

La funzione di perdita adottata è costituita dalla somma di Chamfer Distance e Earth Mover's Distance pesata:

$$loss = loss\_CD + (loss\_EMD * weight\_EMD)$$

dove $weight\_EMD$ è impostato a 0.05 come in RefRec.

Nelle Figure 4.11 e 4.12 sono riportate le due funzioni di perdita adottate durante la fase di addestramento.

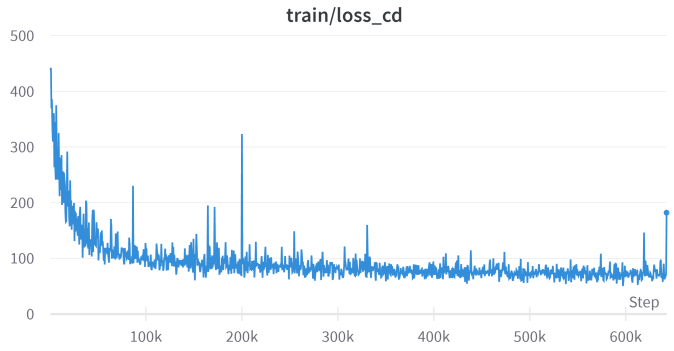


Figura 4.11: PointNet++: ricostruzione Chamfer Distance in ricostruzione

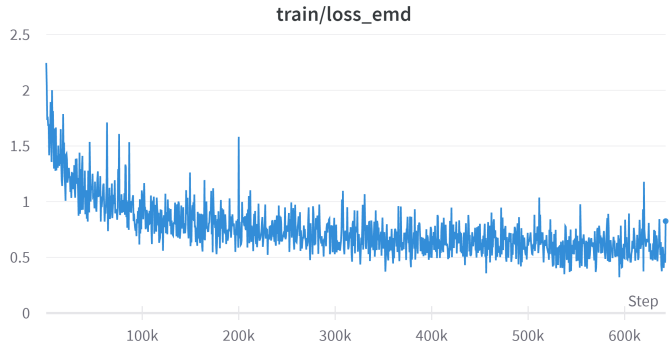


Figura 4.12: PointNet++: Earth Mover's Distance in ricostruzione

Analisi qualitativa

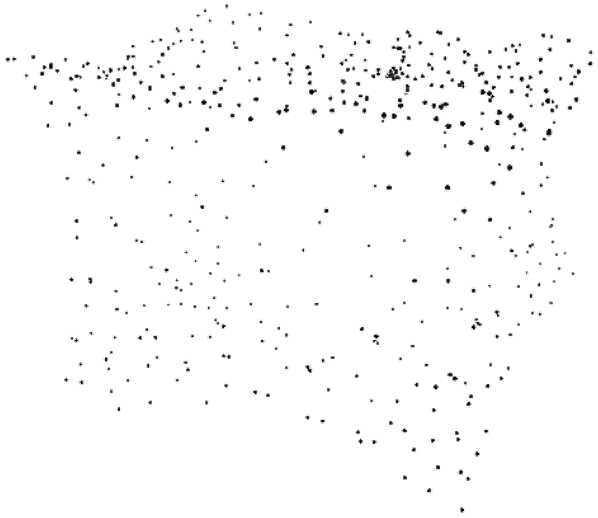
Analizzando le Figure 4.13 e 4.14, è possibile notare che il modello ricostruisce le nuvole di punti con accuratezza inferiore rispetto alla versione PointNet (Figure 4.3 e 4.4). Infatti, concentrandosi sulle immagini (a) e (b) di Figura 4.13, si vede che la vasca da bagno risulta essere deformata e presenta artefatti in zone originariamente assenti di punti. Tale situazione si verifica anche nel caso rappresentato in Figura 4.14, nel quale i piani inferiore e superiore del mobile assumono una forma ondulata (d).

(a)

(b)

Figura 4.13: PointNet++: input (a) e output (b) in ricostruzione

(c)



(d)

Figura 4.14: PointNet++: input (c) e output (d) in ricostruzione

Analisi quantitativa

I dati di accuratezza in retrieval calcolati su ScanNet/test e presenti in Tabella 4.4 si confermano le analisi qualitative svolte. Infatti, ad eccezione di micro accuratezza con K=1, tutti i valori calcolati dimostrano un lieve peggioramento delle prestazioni nella codifica delle nuvole di punti nello spazio embedding rispetto al caso PointNet (Tabella 4.1).

Tipo	Top1(%)	Top5(%)	Top10(%)
Micro	73.09	89.15	93.44
Macro	59.83	78.69	86.16

Tabella 4.4: PointNet++: accuratezza in retrieval (1)

Tale decremento delle abilità del modello si ripercuote anche a livello di classe, confrontando le Tabelle 4.5 e 4.2

Tipo - Classe	Top1(%)	Top5(%)	Top10(%)
Macro - bathtub	100.00	100.00	100.00
Macro - bed	80.00	92.00	93.51
Macro - bookshelf	65.83	88.57	91.16
Macro - cabinet	56.49	85.33	92.00
Macro - chair	92.04	97.37	98.63
Macro - lamp	11.58	45.45	67.65
Macro - monitor	45.45	66.67	73.13
Macro - plant	23.40	44.12	61.29
Macro - sofa	46.04	78.86	91.67
Macro - table	77.42	88.51	92.52

Tabella 4.5: PointNet++: accuratezza in retrieval (2)

4.4.2 Warmup per generazione di pseudo-annotazioni

In questa Sezione sono riportati i grafici (Figure 4.15 e 4.16) di accuratezza su ModelNet/test e ScanNet/test generati durante la fase di addestramento in classificazione sul dominio sorgente.

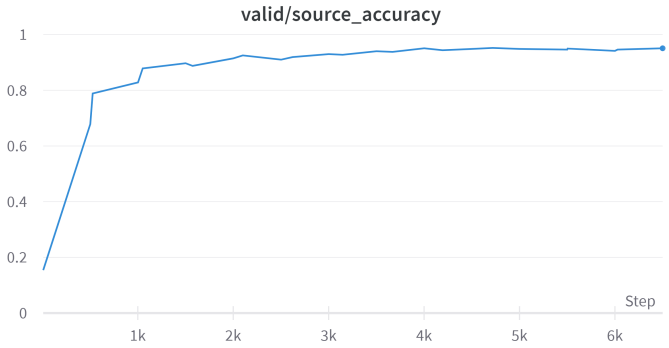


Figura 4.15: PointNet++: accuratezza source in warmup

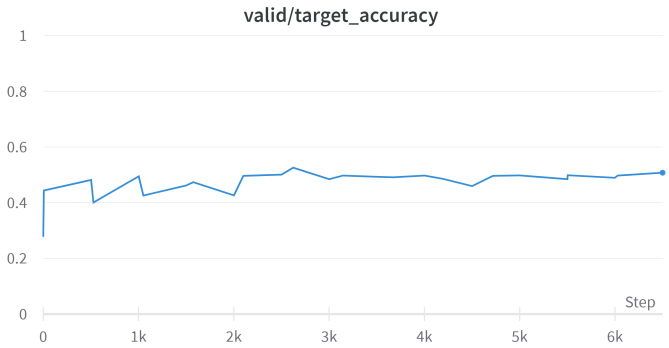


Figura 4.16: PointNet++: accuratezza target in warmup

L'accuratezza migliore ottenuta su ScanNet/test risulta essere **49.63%**.

La matrice di confusione in Figura 4.17, calcolata per il classificatore ottenuto su ScanNet/test, mostra un comportamento simile al caso PointNet (Figura 4.7).

4.4.3 Raffinatezza offline, auto-addestramento e raffinatezza online

In questa Sezione sono analizzati i risultati di accuratezza ottenuti dopo aver eseguito gli ultimi passi della pipeline RefRec con PointNet++.

In particolare, sono stati testati diversi K per il KNN nella fase di raffinatezza

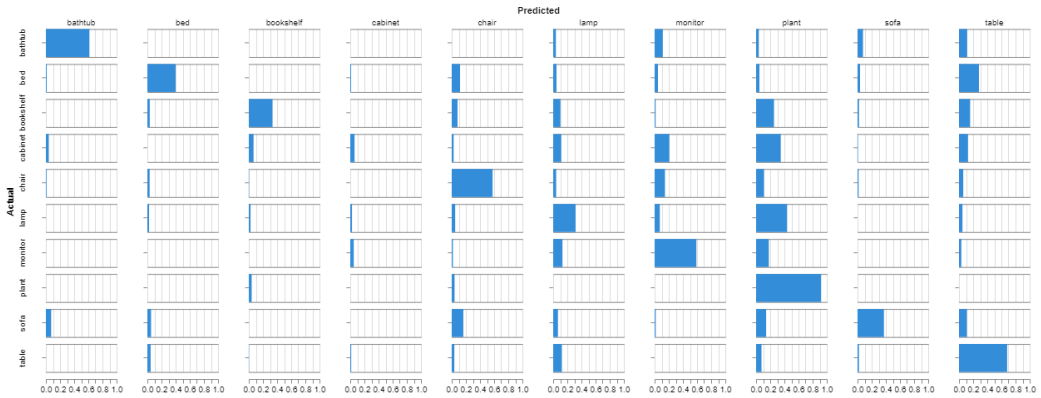


Figura 4.17: PointNet++: matrice di confusione in warmup

offline al fine di identificare il valore che permetta di ottenere la percentuale di accuratezza migliore dopo la fase di auto-addestramento (Tabella 4.6).

Fase/Accuratezza	K=3	K=5	K=10
Auto-addestramento	49.35%	52.74%	53.02%

Tabella 4.6: PointNet++: accuratezza dopo auto-addestramento

L'accuratezza migliore ottenuta su ScanNet/test risulta essere **53.02%** ed è relativa al caso K=10 per il KNN in raffinatezza offline. In più, si nota che il caso K=3 peggiora il risultato di accuratezza ottenuto dopo il warmup (49.63%). Ciò indica che in questo caso la raffinatezza offline ha provveduto sì a raffinare alcune pseudo-annotazioni, ma anche a rovinarne altrettante al punto di decrementare l'accuratezza finale ottenuta.

Si riportano in Figura 4.18 e 4.19 i grafici di accuratezza su ModelNet/test e ScanNet/test generati durante la fase di auto-addestramento e raffinatezza online relativi al caso K=10.

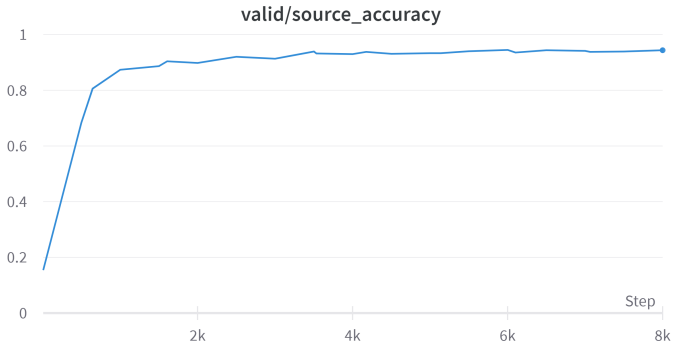


Figura 4.18: PointNet++: accuratezza source in auto-addestramento

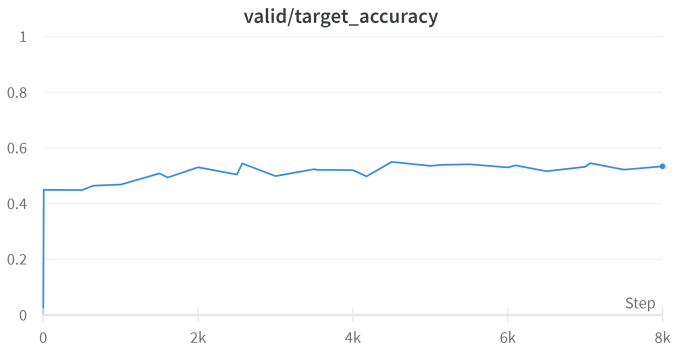


Figura 4.19: PointNet++: accuratezza target in auto-addestramento

Inoltre, in Figura 4.20 è riportata la matrice di confusione ricavata su ScanNet/test nella quale è possibile notare un lieve miglioramento nella predizione di alcune classi (e.g., *monitor, sofa*) rispetto alla fase di warmup in classificazione (Figura 4.17).

4.5 Analisi: DGCNN

Il terzo modello autoencoder utilizzato al posto dell'architettura PointNet all'interno della pipeline RefRec è **DGCNN**.

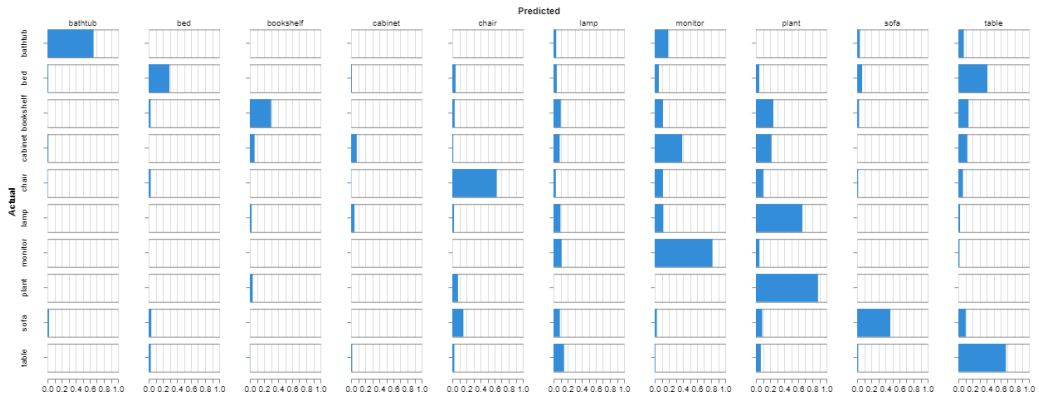


Figura 4.20: PointNet++: matrice di confusione in auto-addestramento

4.5.1 Ricostruzione

La funzione di perdita adottata è costituita dalla somma di Chamfer Distance e Earth Mover's Distance pesata:

$$loss = loss\_CD + (loss\_EMD * weight\_EMD)$$

dove $weight\_EMD$ è impostato a 0.05 come in RefRec.

Nelle Figure 4.21 e 4.22 sono riportate le due funzioni di perdita adottate durante la fase di addestramento.

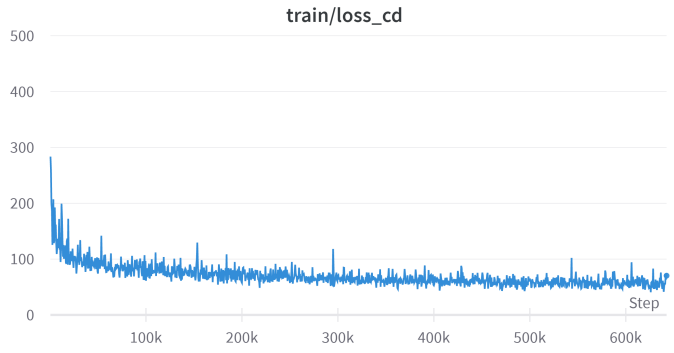


Figura 4.21: DGCNN: Chamfer Distance in ricostruzione

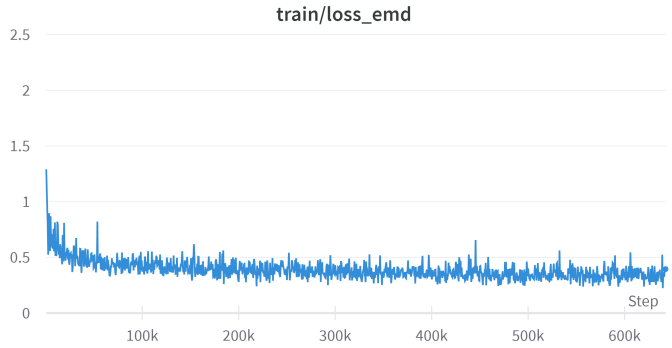


Figura 4.22: DGCNN: Earth Mover's Distance in ricostruzione

Analisi qualitativa

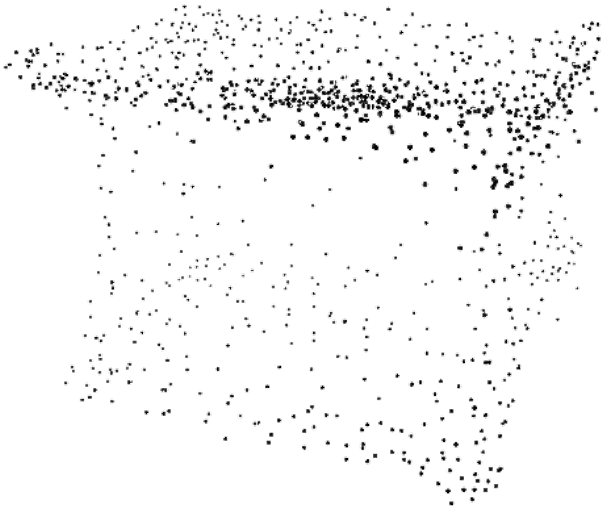
Dalla Figure 4.23 e 4.24 si vede che il modello ricostruisce le nuvole di punti mantenendo pressoché giuste le forme delle stesse. A differenza dei casi PointNet (Figure 4.3 e 4.4) e PointNet++ (Figure 4.13 e 4.14), DGCNN produce zone ad alta densità di punti non presenti nelle nuvole di punti originali (e.g., fondale della vasca da bagno, centro del piano superiore del mobile).

(a)

(b)

Figura 4.23: DGCNN: input (a) e output (b) in ricostruzione

(c)



(d)

Figura 4.24: DGCNN: input (c) e output (d) in ricostruzione

Analisi quantitativa

La Tabella 4.7 mostra i valori di accuratezza in retrieval calcolati su ScanNet/test. Tutti i valori presenti risultato inferiori alle percentuali calcolate con la versione PointNet (Tabella 4.1) e PointNet++ (Tabella 4.4).

Tipo	Top1(%)	Top5(%)	Top10(%)
Micro	69.14	86.77	92.59
Macro	56.78	73.92	83.46

Tabella 4.7: DGCNN: accuratezza in retrieval (1)

Inoltre, anche a livello di classe (Tabella 4.8) i valori ottenuti rispecchiano il peggioramento ottenuto rispetto ai modelli precedenti (Tabelle 4.2 e 4.5)

Tipo - Classe	Top1(%)	Top5(%)	Top10(%)
Macro - bathtub	100.00	100.00	100.00
Macro - bed	78.95	88.73	90.00
Macro - bookshelf	66.10	84.40	91.28
Macro - cabinet	52.78	82.24	93.15
Macro - chair	90.27	96.95	98.39
Macro - lamp	3.88	26.00	67.86
Macro - monitor	44.62	61.11	69.01
Macro - plant	11.11	37.04	47.83
Macro - sofa	42.01	74.80	84.62
Macro - table	78.05	87.93	92.48

Tabella 4.8: DGCNN: accuratezza in retrieval (2)

4.5.2 Warmup per generazione di pseudo-annotazioni

Di seguito sono raffigurati i grafici (Figure 4.25 e 4.26) di accuratezza su ModelNet/test e ScanNet/test generati durante la fase di addestramento in classificazione sul dominio sorgente.

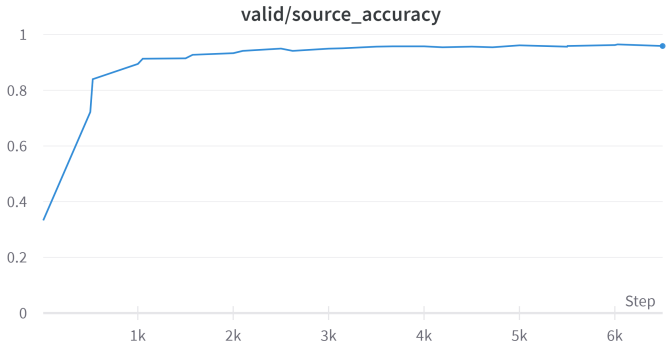


Figura 4.25: DGCNN: accuratezza source in warmup

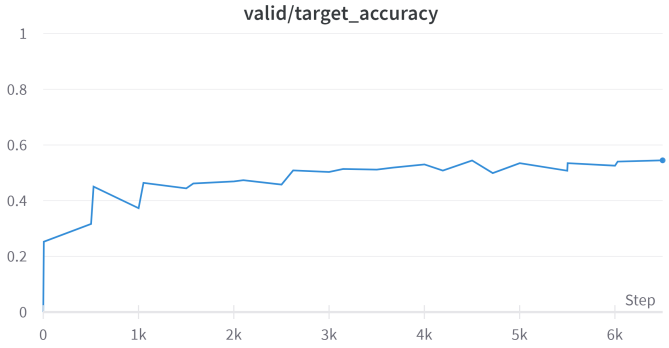


Figura 4.26: DGCNN: accuratezza target in warmup

L'accuratezza migliore ottenuta su ScanNet/test è **54.04%**.

La matrice di confusione in Figura 4.27, calcolata per il classificatore ottenuto su ScanNet/test, è molto simile alle matrici ricavate con PointNet (Figura 4.7) e PointNet++ (Figura 4.17)

4.5.3 Raffinatezza offline, auto-addestramento e raffinatezza online

Gli ultimi passi della pipeline RefRec con DGCNN, eseguiti al variare di K per il KNN nella fase di raffinatezza offline, hanno prodotto i valori in Tabella 4.9.

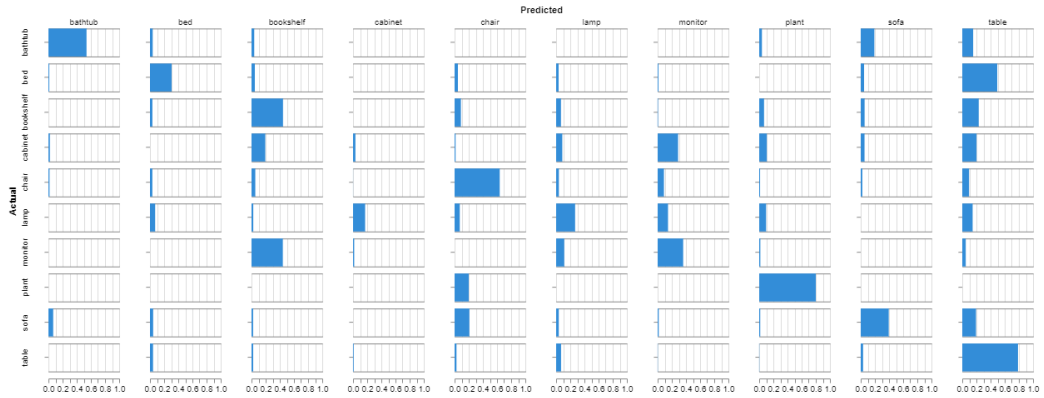


Figura 4.27: DGCNN: matrice di confusione in warmup

Fase/Accuratezza	K=3	K=5	K=10
Auto-addestramento	55.91%	56.87%	56.81%

Tabella 4.9: DGCNN: accuratezza dopo auto-addestramento

L'accuratezza migliore ottenuta su ScanNet/test risulta essere **56.87%** ed è relativa al caso K=5 per il KNN in raffinatezza offline.

Si riportano in Figura 4.28 e 4.29 i grafici di accuratezza su ModelNet/-test e ScanNet/test generati durante la fase di auto-addestramento relativi al caso K=5. Inoltre, in Figura 4.30 è riportata la matrice di confusione ricavata per il caso medesimo su ScanNet/test nella quale è possibile notare un discreto miglioramento nella predizione di alcune classi (e.g., *monitor, sofa*) rispetto alla fase di warmup in classificazione (Figura 4.27).

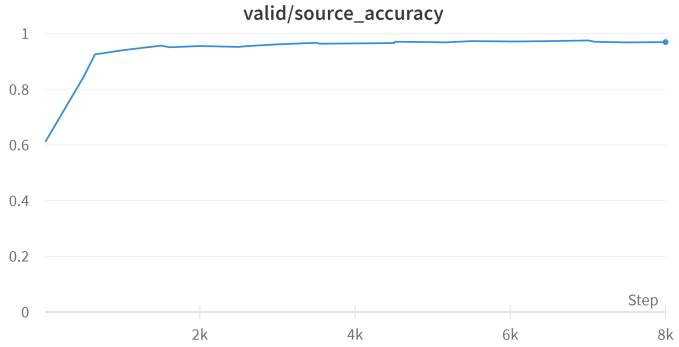


Figura 4.28: DGCNN: accuratezza source in auto-addestramento

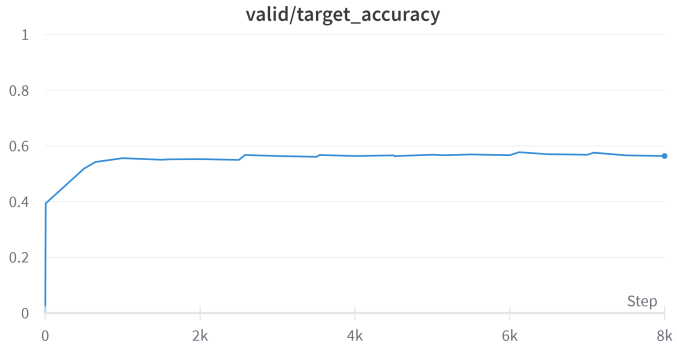


Figura 4.29: DGCNN: accuratezza target in auto-addestramento

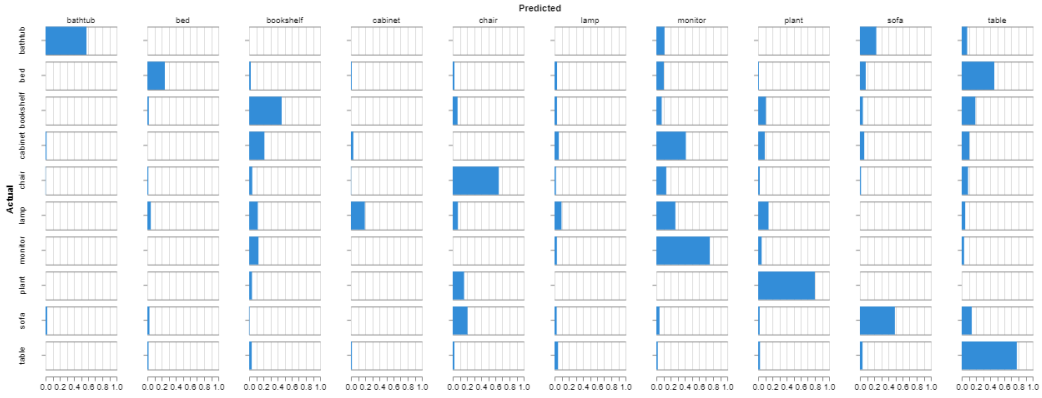


Figura 4.30: DGCNN: matrice di confusione in auto-addestramento

4.6 Analisi: Transformer

Considerando il forte successo raggiunto dall'architettura **Transformer** anche nel settore della Computer Vision ed in particolar modo nell'elaborazione di nuvole di punti, si è deciso di testare tale modello in questo scenario di Domain Adaptation seguendo la pipeline di RefRec.

4.6.1 Configurazione di Transformer

A differenza degli altri modelli, il Transformer presenta molteplici parametri da configurare adeguatamente al contesto di utilizzo e dunque necessita di molti esperimenti da condurre. A tal proposito, considerando l'ingente

quantità di ore necessaria per l'addestramento del modello nei vari passi, la maggior parte di essi sono stati interrotti dopo la fase di warmup in classificazione qualora non fosse stata raggiunta una percentuale di accuratezza sufficientemente interessante per proseguire.

La configurazione base utilizzata, alla quale sono state apportate modifiche nel corso degli esperimenti, e il meccanismo di generazioni di gruppi a partire dalla nuvola di punti (FPS e KNN) corrispondo a ciò che è stato adottato in Point-MAE [45]. Tali scelte sono state fatte considerando che gli autori di tale articolo hanno raggiunto risultati interessanti eseguendo anch'essi una fase preliminare di ricostruzione seguita da fine-tuning in classificazione. L'unica differenza iniziale consiste nell'assenza della tecnica di mascheramento adottata in Point-MAE. Questo primo esperimento con Transformer viene identificato con la sigla *T1*.

I parametri di *T1* base in ricostruzione sono quindi:

- Numero di gruppi: 64
- Dimensione di ogni gruppo: 32
- Dimensione codifica: 384
- Profondità encoder: 12
- Numero di teste encoder: 6
- Profondità decoder: 4
- Numero di teste decoder: 6

4.6.2 Ricostruzione

Nella fase di ricostruzione è stata utilizzata la funzione di perdita Chamfer Distance (CD). In Figura 4.31 è presente il grafico contenente l'andamento di CD durante l'addestramento.

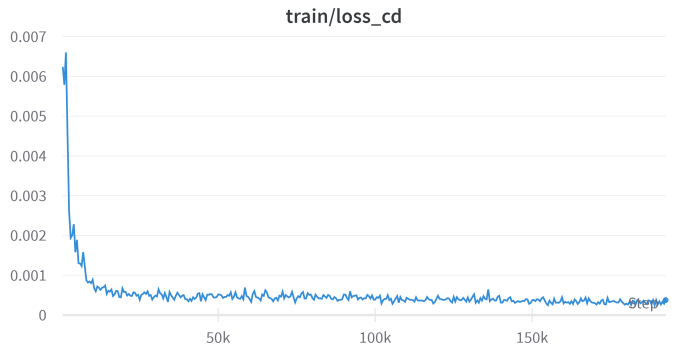
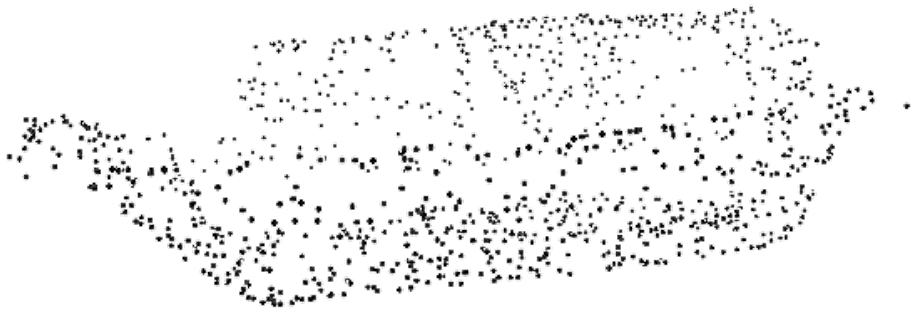


Figura 4.31: T1: Chamfer Distance in ricostruzione

Analisi qualitativa

Dalle Figure 4.32 e 4.33 si vede che il Transformer riesce a ricostruire molto bene le nuvole di punti mantenendo ben intatta la forma degli oggetti e preservando correttamente le densità di alcune parti presenti originariamente.



(a)

(b)

Figura 4.32: T1: input (a) e output (b) in ricostruzione

(c)

(d)

Figura 4.33: T1: input (c) e output (d) in ricostruzione

Analisi quantitativa

Transformer, per sua natura, opera su dati suddivisi in token lungo la pipeline encoder-decoder. L'encoder dunque, per ciascuna nuvola di punti, produce un insieme di token codificati, i quali necessitano di una funzione di aggregazione per ottenere un'unica rappresentazione nello spazio embedding e procedere con il calcolo dell'accuratezza in retrieval. La funzione di aggregazione scelta è la media.

In Tabella 4.10 sono presenti i valori di accuratezza in retrieval calcolati su ScanNet/test. I valori ottenuti sono paragonabili ai valori di PointNet (Tabella 4.1) e lievemente superiori ai valori di DGCNN (Tabella 4.7) e PointNet++ (Tabella 4.4).

Tipo	Top1(%)	Top5(%)	Top10(%)
Micro	70.60	89.60	94.12
Macro	58.50	80.13	88.67

Tabella 4.10: T1: accuratezza in retrieval (1)

A livello di classe, in Tabella 4.11) i valori ottenuti confermano la somiglianza di al modello PointNet (Tabella 4.2).

Tipo - Classe	Top1(%)	Top5(%)	Top10(%)
Macro - bathtub	100.00	100.00	100.00
Macro - bed	80.30	89.86	92.41
Macro - bookshelf	69.23	83.92	91.55
Macro - cabinet	52.26	83.54	89.87
Macro - chair	91.17	97.89	99.13
Macro - lamp	8.33	58.82	83.33
Macro - monitor	39.62	66.67	76.47
Macro - plant	21.57	51.28	70.97
Macro - sofa	42.71	78.52	89.15
Macro - table	79.77	90.82	93.85

Tabella 4.11: T1: accuratezza in retrieval (2)

4.6.3 Warmup per generazione di pseudo-annotazioni

In questa fase, il modello utilizzato per la classificazione è composto da: encoder Transformer per ottenere i gruppi codificati, pila di blocchi Transformer per elaborare i gruppi evidenziando caratteristiche interessati, testa di classificazione composta da strati di tipo Fully Connected, BatchNorm, ReLU e Dropout per ottenere 10 neuroni in uscita, ciascuno rappresentante una classe.

L'intero modello di classificazione con Transformer adottato corrisponde a quanto utilizzato in Point-MAE [45] come architettura per la loro fase di fine-tuning.

In Figura 4.34 è raffigurato il modello appena descritto.

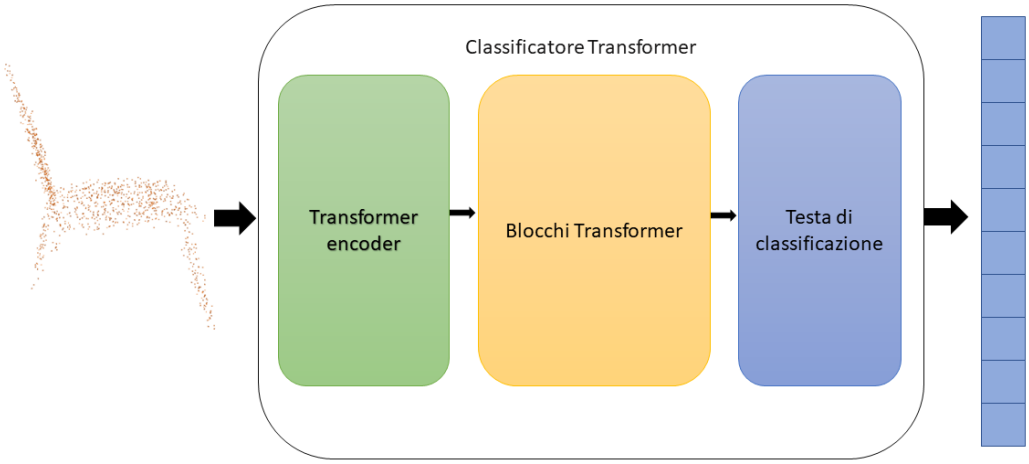


Figura 4.34: Transformer: classificatore

Nei grafici (Figure 4.35 e 4.36) sono presenti i valori di accuratezza su ModelNet/test e ScanNet/test generati durante la fase di addestramento in classificazione sul dominio sorgente.

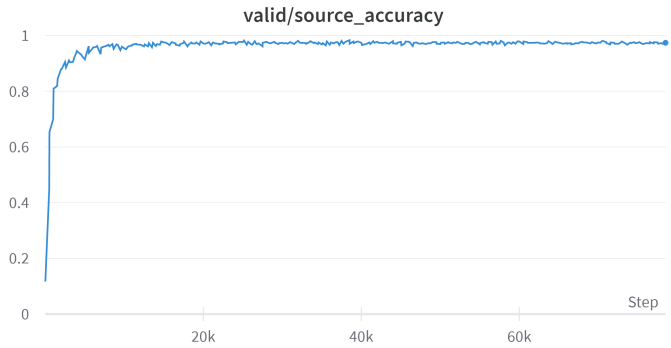


Figura 4.35: T1: accuratezza source in warmup

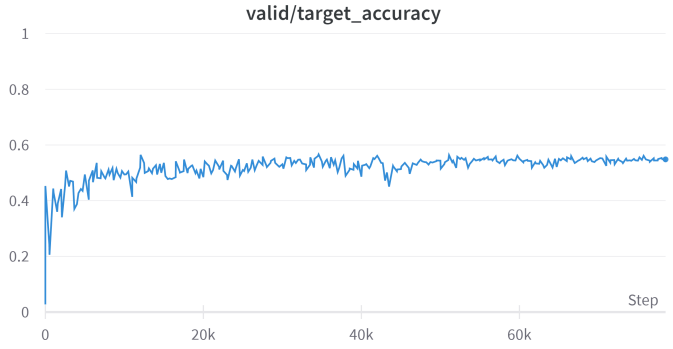


Figura 4.36: T1: accuratezza target in warmup

L'accuratezza migliore ottenuta su ScanNet/test è **53.70%**.

La matrice di confusione in Figura 4.37, calcolata per il classificatore ottenuto su ScanNet/test, è molto simile alle matrici di PointNet++ (Figura 4.17), DGCNN (Figura 4.27) ed in particolar modo di PointNet (Figura 4.7).

4.6.4 Occlusioni delle nuvole di punti

Tutti i casi analizzati sino ad ora eseguono elaborazioni su nuvole di punti occluse per incrementare l'abilità di generalizzazione del modello sulle stesse,

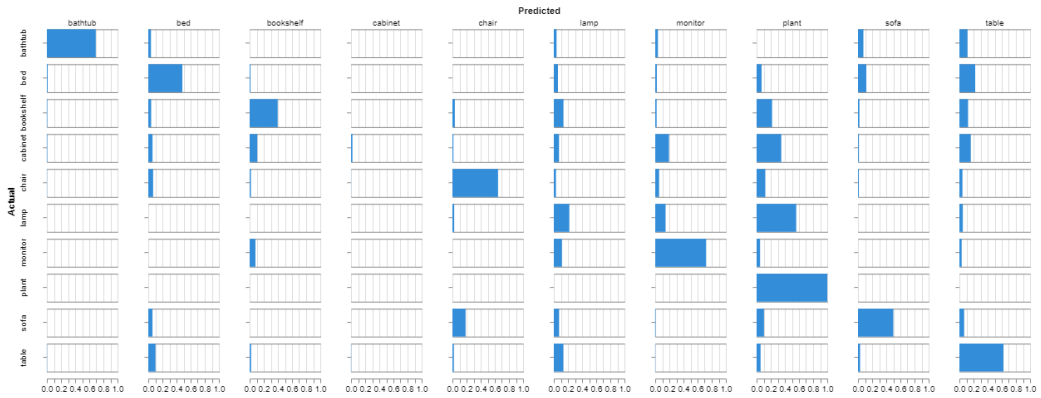


Figura 4.37: T1: matrice di confusione in warmup

disincentivando scenari di **overfitting**. In particolare, la pipeline di RefRec carica nuvole di punti occludendo alcune parti dei dataset sintetici, lasciando invariate le point cloud dei dataset reali. Questo viene fatto poiché i dataset reali sono spesso composti da nuvole con densità di punti molto diversa all'interno e l'occlusione potrebbe rimuovere completamente parti poco dense, perdendo caratteristiche interessanti dell'oggetto.

L'esperimento di Transformer iniziale (*T1*) segue la modalità di occlusione implementata in RefRec ovvero occlusioni su ModelNet abilitate e occlusioni su ScanNet disabilitate. Per comprendere veramente se questa sia la configurazione migliore in relazione alle occlusioni sono stati eseguiti altri tre esperimenti comprendenti ricostruzione e warmup in classificazione, variando presenza o assenza di occlusioni in ModelNet e/o ScanNet. I risultati di accuratezza delle pseudo-annotazioni su ScanNet/test ottenuti dopo la fase di classificazione sono riportati in Tabella 4.12, in cui sono presenti l'esperimento già analizzato *T1* e gli esperimenti *T2* (occlusioni ModelNet disabilitate, occlusioni ScanNet abilitate), *T3* (occlusioni ModelNet abilitate, occlusioni ScanNet abilitate) e *T4* (occlusioni ModelNet disabilitate, occlusioni ScanNet disabilitate).

Metrica/Esperimento	T1	T2	T3	T4
Accuratezza	53.70%	47.31%	55.17%	45.05%

Tabella 4.12: Transformer: accuratezza esperimenti T1-T4

Dalla Tabella 4.12 si evince che la miglior configurazione è data dall'occlusione su entrambi i domini. Infatti, come detto precedentemente, la presenza di occlusioni aiuta la generalizzazione permettendo di ottenere un'accuratezza superiore quando si testa il modello su nuvole di punti mai viste prima. In questo caso specifico, con l'utilizzo di Transformer senza mascheramento, occludere sul dataset reale ScanNet è conveniente poiché non genera forti malus rimuovendo completamente parti caratteristiche delle point cloud.

4.7 Analisi: Transformer mascherato

Il passo seguente consiste nell'introduzione del meccanismo di mascheramento durante la fase di ricostruzione. Questo metodo, adottato dagli autori di Point-MAE [45], permette al modello apprendere caratteristiche degli oggetti ricostruendo solamente alcune parti della nuvola di punti (parti mascherate) attraverso la visione delle parti in chiaro.

4.7.1 Ricostruzione

La ricostruzione iniziale svolta segue la configurazione base impostata in Transformer *T3*, in quanto ha restituito i risultati migliori negli esperimenti *T1-T4*. Il mascheramento imposto è pari al 40% dei gruppi di punti della point cloud indipendentemente dal suo dominio di appartenenza. Questo primo esperimento prende il nome *TM3*.

Nella fase di ricostruzione è stata utilizzata la funzione di perdita Chamfer Distance (CD) il cui andamento durante la fase di addestramento è presente in Figura 4.38.

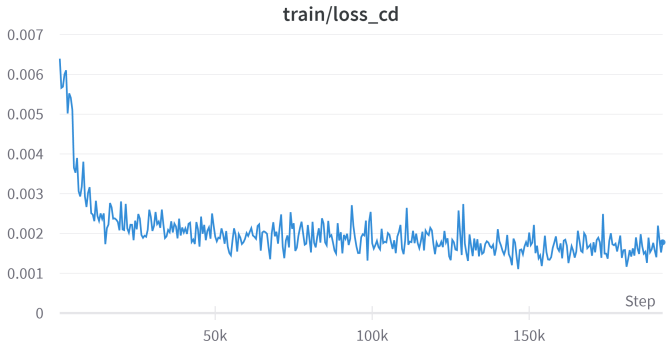
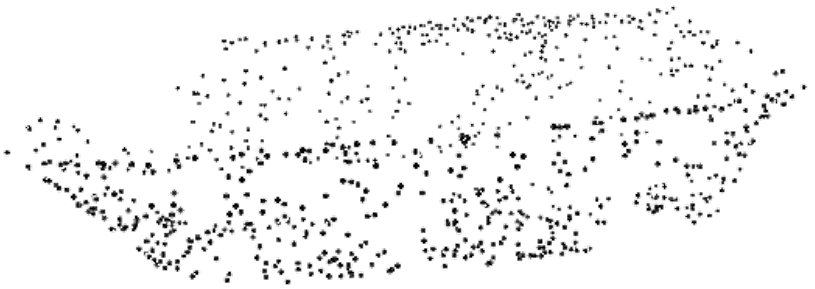


Figura 4.38: TM3: Chamfer Distance in ricostruzione

Analisi qualitativa

Nelle Figure4.39 e 4.40 è possibile analizzare come viene svolta la ricostruzione nel caso *TM3*. Le nuvole di punti in ingresso (a/d) subisce un mascheramento di alcune sue parti (b/e) le quali vengono ricostruite dal Transformer mascherato producendo (c/f). Da un'analisi di (b/e) e (c/f) si percepisce che il modello ha completato molto bene le parti mancanti dalle nuvole di punti, ricreando la giusta forma degli oggetti vasca da bagno e mobile presenti e preservando adeguatamente le loro densità interne. Tale comportamento risulta molto simile al caso di Transformer senza mascheramento (Figure 4.32 e 4.33). Risulta complesso fare ulteriori confronti con quest'ultimo tramite un'analisi qualitativa.

(a)



(b)

(c)

Figura 4.39: TM3: input (a), input mascherato (b) e output (c) in ricostruzione

(d)

(e)

(f)

Figura 4.40: TM3: input (d), input mascherato (e) e output (f) in ricostruzione

Analisi quantitativa

In Tabella 4.13 sono presenti i valori di accuratezza in retrieval calcolati su ScanNet/test utilizzando la media come funzione di aggregazione. I valori ottenuti sono molto simili ai risultati ottenuti nel caso di Transformer senza mascheramento (Tabella 4.10).

Tipo	Top1(%)	Top5(%)	Top10(%)
Micro	71.32	87.12	93.76
Macro	59.22	80.28	87.43

Tabella 4.13: TM3: accuratezza in retrieval

A livello di classe, in Tabella 4.14) i valori ottenuti confermano la vicinanza al Transformer senza mascheramento (Tabella 4.11).

Tipo - Classe	Top1(%)	Top5(%)	Top10(%)
Macro - bathtub	100.00	100.00	100.00
Macro - bed	80.30	90.02	93.14
Macro - bookshelf	68.81	82.94	92.15
Macro - cabinet	52.76	84.04	89.93
Macro - chair	90.57	98.19	98.72
Macro - lamp	9.16	59.54	83.47
Macro - monitor	37.33	66.91	77.27
Macro - plant	21.87	50.29	71.22
Macro - sofa	43.21	79.05	90.17
Macro - table	78.34	90.64	93.71

Tabella 4.14: TM3: accuratezza in retrieval

4.7.2 Warmup per generazione di pseudo-annotazioni

L'intero modello di classificazione con Transformer adottato corrisponde a quanto utilizzato nell'esperimento senza mascheramento (Sezione 4.6.3).

Nei grafici (Figure 4.41 e 4.42) sono riportati i valori di accuratezza su ModelNet/test e ScanNet/test ottenuti durante la fase di addestramento in classificazione sul dominio sorgente.

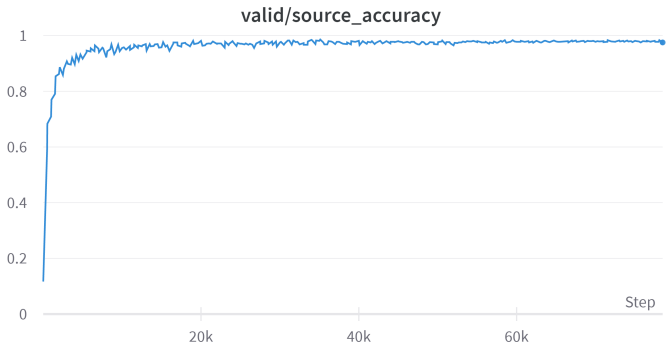


Figura 4.41: TM3: accuratezza source in warmup

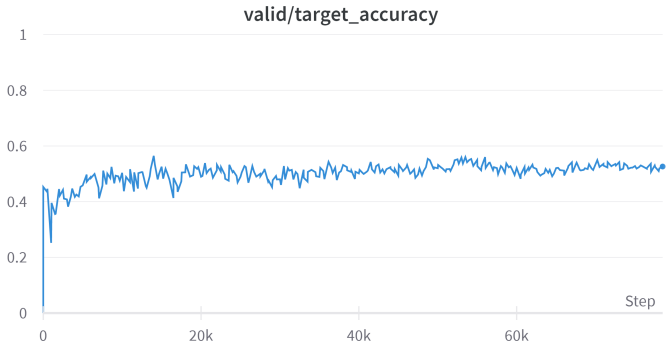


Figura 4.42: TM3: accuratezza target in warmup

L'accuratezza migliore ottenuta su ScanNet/test è **52.18%**.

La matrice di confusione in Figura 4.43, calcolata su ScanNet/test per il classificatore ottenuto, mostra un comportamento molto simile al caso di Transformer senza mascheramento (Figura 4.37).

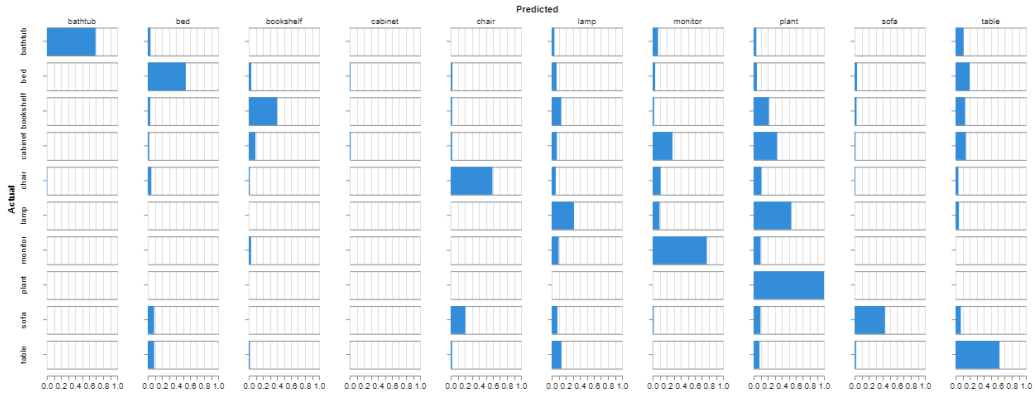


Figura 4.43: TM3: matrice di confusione in warmup

4.7.3 Occlusioni delle nuvole di punti

Come nel caso di Transformer senza mascheramento (Sezione 4.6.4), anche in questo caso sono stati svolti esperimenti per valutare quale sia la configurazione migliore tra occlusioni su ModelNet abilitate/disabilitate e occlusioni su ScanNet abilitate/disabilitate. Denominiamo *TM1* il caso con occlusioni su ModelNet abilitate e ScanNet disabilitate, *TM2* il caso con occlusioni su Modelnet disabilitate e ScanNet abilitate, *TM4* il caso con occlusioni su Modelnet e ScanNet disabilitate.

In Tabella 4.15 sono raggruppati i valori di accuratezza delle pseudo-annotazioni ottenute su ScanNet/test dopo la fase di warmup in classificazione per gli esperimenti *TM1, TM2, TM4* condotti. Inoltre è riportato anche il caso **TM3** precedentemente svolto (Sezione 4.7.2).

Metrica/Esperimento	TM1	TM2	TM3	TM4
Accuratezza	54.04%	40.31%	52.18%	47.99%

Tabella 4.15: Transformer mascherato: accuratezza esperimenti TM1-TM4

E' possibile notare che, in caso di mascheramento, la configurazione migliore risulta essere occlusioni su ModelNet abilitate e occlusioni su ScanNet disabilitate. Questo avviene perché:

- Occludere su ModelNet aiuta il meccanismo di generalizzazione del modello in quanto permette di non apprendere in modo troppo specifico

le nuvole di punti sintetiche di questo dataset, bensì di imparare caratteristiche valide anche per gli oggetti reali di ScanNet permettendo un corretto trasferimento di conoscenza da dominio source a dominio target. Infatti, proprio attraverso le occlusioni si rende l'input di nuvole di punti di ModelNet (dense, complete e sintetiche) molto più vicino all'input di point cloud di ScanNet (sparse, rumorose e reali), riducendo il divario presente tra i due domini.

- Nel caso di ScanNet, invece, è preferibile non occludere poiché essendo esso composto da nuvole di punti sparse e rumorose, l'occlusione unita alla tecnica di mascheramento possono rimuovere molte parti significative di un oggetti e dunque lasciare point cloud in chiaro prive di caratteristiche interessanti. Questo complica il processo di ricostruzione in quanto il Transformer mascherato utilizza proprio la conoscenza derivante dalle parti in chiaro per completare le parti mancanti.

4.7.4 Variazione di mascheramento e gruppi

Appurato che la configurazione del caso *TM1* risulti la migliore nel caso di mascheramento, si procede adesso con l'esecuzione di sei ulteriori esperimenti con la medesima fino alla fase di warmup in classificazione variando percentuale di mascheramento, numero e dimensione di gruppi.
Gli esperimenti svolti, correlati di configurazione, sono:

- TM5: mascheramento 40% ModelNet e ScanNet, 8 gruppi di 256 punti
- TM6: mascheramento 40% ModelNet e ScanNet, 16 gruppi di 128 punti
- TM7: mascheramento 40% ModelNet e 20% ScanNet, 8 gruppi di 256 punti
- TM8: mascheramento 40% ModelNet e 20% ScanNet, 16 gruppi di 128 punti
- TM9: mascheramento 60% ModelNet e 20% ScanNet, 8 gruppi di 256 punti
- TM10: mascheramento 60% ModelNet e 20% ScanNet, 16 gruppi di 128 punti

Metrica/Esperimento	TM5	TM6	TM7	TM8	TM9	TM10
Accuratezza	51.05%	51.44%	52.23%	51.27%	54.83%	55.57%

Tabella 4.16: Transformer mascherato: accuratezza esperimenti TM5-TM10

I valori di accuratezza ottenuti su ScanNet/test sono presenti in Tabella 4.16.

Il caso che ha raggiunto l'accuratezza più alta è il caso *TM10*. Questo è plausibile in quanto una percentuale di mascheramento alta è possibile solo in dataset contenenti nuvole di punti sintetiche poiché i gruppi non mascherati, sebbene in quantità inferiori, contengono comunque caratteristiche interessanti per la ricostruzione delle parti occluse. Tale affermazione ha valenza nel caso opposto riferendosi alla percentuale di mascheramento bassa nei dataset con point cloud reali. Inoltre, relativamente alle configurazioni con mascheramento 60% su ModelNet e 20% su ScanNet, la divisione della nuvola di punti in 16 gruppi contenenti 128 punti ciascuno risulta sperimentalmente migliore rispetto al caso di 8 gruppi di 256 punti.

4.7.5 Raffinatezza offline, auto-addestramento e raffinatezza online

Dopo aver trovato la configurazione migliore all'interno di tutti gli esperimenti comprendenti Transformer con o senza tecnica di mascheramento (Figure 4.12, 4.15, 4.16), identificata nel caso *TM10*, si prosegue con essa per completare la pipeline di RefRec attraverso le fasi di raffinatezza offline delle pseudo-annotazioni, auto-addestramento e raffinatezza online.

Affinché si abbia un unico embedding per ciascuna nuvola di punti sono stati provati diversi metodi di aggregazione dei gruppi codificati tra i quali: media, massimo, concatenazione di media e massimo. Inoltre nella fase di raffinatezza offline sono stati testati molteplici K per il KNN al fine di trovare il più adatto. Tutti gli esperimenti condotti correlati di accuratezza su ScanNet/test raggiunta sono presenti in Tabella 4.17.

Come è possibile notare dalla Tabella 4.17 sono stati portati avanti solamente gli esperimenti con la concatenazione come funzione di aggregazione poiché si è rivelato il metodo migliore per la generazione di embedding, producendo il valore di accuratezza più alto nel caso K=3.

Aggregazione/Accuratezza	K=3	K=7	K=5	K=9
Media	55.27%	X	X	X
Massimo	55.91%	X	X	X
Concat(media, massimo)	56.07%	56.30%	57.60%	56.13%

Tabella 4.17: Transformer mascherato: accuratezza dopo auto-addestramento

Il modello migliore di Transformer, denominato *BT*, per Domain Adaptation relativa ai domini ModelNet (source) e ScanNet (target) è definito dai seguenti parametri in ricostruzione:

- Mascheramento point cloud sintetiche: 60%
- Mascheramento point cloud reali: 60%
- Occlusioni point cloud sintetiche: abilitate
- Occlusioni point cloud reali: disabilitate
- Numero di gruppi: 16
- Dimensione di ogni gruppo: 128
- Dimensione codifica: 384
- Profondità encoder: 12
- Numero di teste encoder: 6
- Profondità decoder: 4
- Numero di teste decoder: 6

Tale modello ha raggiunto un'accuratezza su ScanNet/test di **57.60%** (Tabella 4.17).

Si riportano in Figura 4.44 e 4.45 i grafici di accuratezza su ModelNet/-test e ScanNet/test generati durante la fase di auto-addestramento relativi al caso migliore. Inoltre, in Figura 4.46 è riportata la matrice di confusione del caso medesimo ricavata su ScanNet/test.

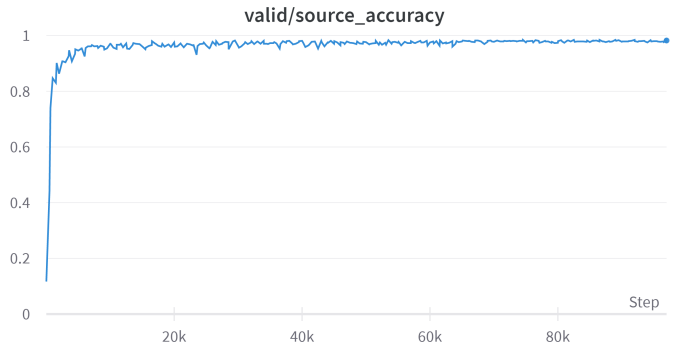


Figura 4.44: BT: accuratezza source in auto-addestramento

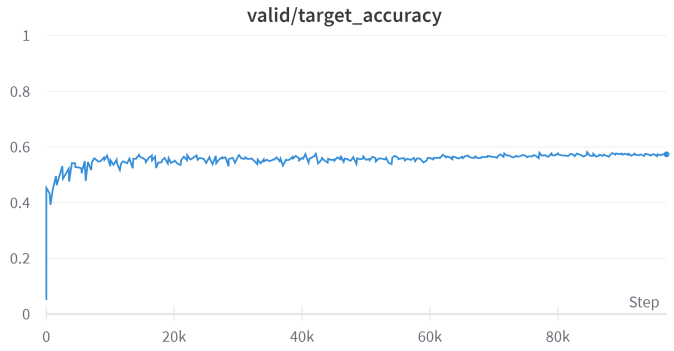


Figura 4.45: BT: accuratezza target in auto-addestramento

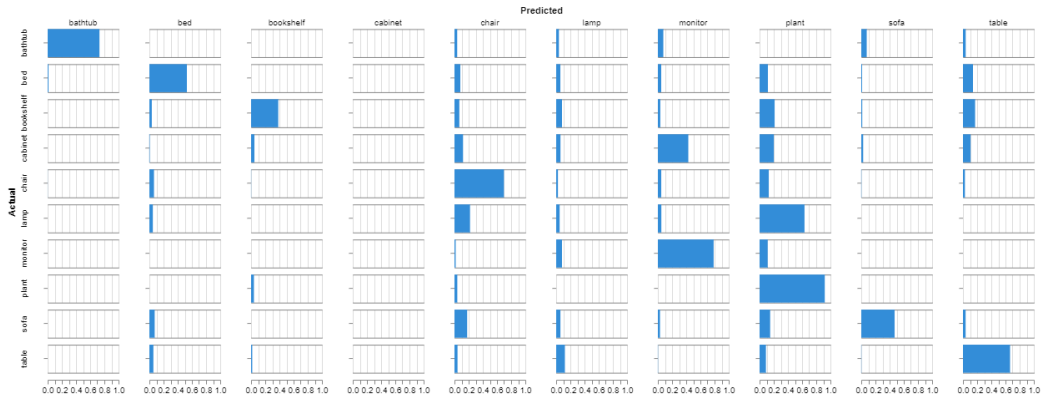


Figura 4.46: BT: matrice di confusione in auto-addestramento

Capitolo 5

Risultati sperimentali a confronto

In questo Capitolo si procede al confronto dei modelli migliori ottenuti dopo la fase di auto-addestramento per ciascun tipo di architettura analizzata. In Figura 5.1 è raffigurato un grafico con le percentuali di accuratezza ottenute su ScanNet/test per ciascun modello.

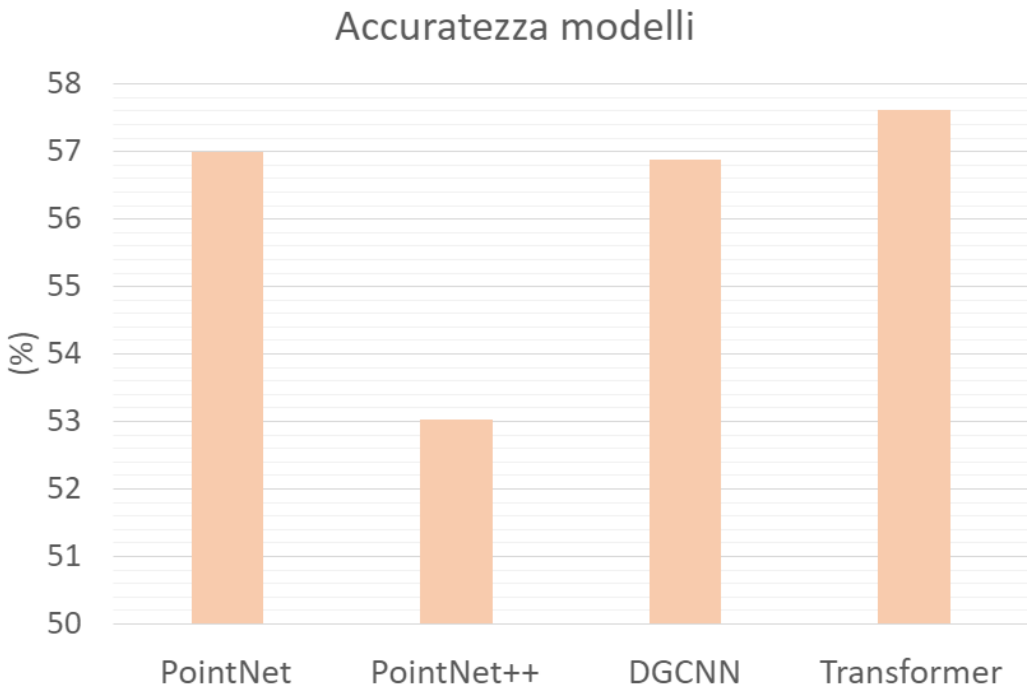


Figura 5.1: Accuratezza dei modelli analizzati

Dalle analisi svolte in questo progetto si evince che il modello più accurato per operare nello scenario di Domain Adaptation con dominio sorgente ModelNet e dominio di destinazione ScanNet mediante la pipeline di RefRec risulta essere il Transformer, il quale ha raggiunto una percentuale di accuratezza su ScanNet/test di **57.60%**, battendo tutti gli altri modelli di PointNet, PointNet++ e DGCNN.

Conclusioni e sviluppi futuri

L'analisi di architetture diverse per la classificazione di nuvole di punti nello scenario di Domain Adaptation studiato ha dimostrato che il Transformer risulta essere il modello più accurato.

E' importante, tuttavia, evidenziare che tale l'architettura di Transformer necessiti di molti parametri da impostare correttamente in relazione al contesto di utilizzo e dunque la sua implementazione in un determinato scenario può richiedere molto tempo. Al contrario, modelli come PointNet, PointNet++ e DGCNN sono molto immediati da inserire in un'architettura esistente e questo può rappresentare un punto di forza notevole in alcune applicazioni.

Sono lasciati a sviluppi futuri eventuali approfondimenti come, ad esempio, la ricerca di una configurazione di parametri migliore oppure la modifica del meccanismo di raffinatezza offline (e.g., concatenazione di KNN di più modelli addestrati diversamente). Un'analisi molto interessante può esser condotta testando il modello Transformer migliore trovato in altri scenari di Domain Adaptation così da verificarne il comportamento al di fuori del caso ModelNet-ScanNet.

Bibliografia

[1] A. Cardace, R. Spezialetti, P. Z. Ramirez, S. Salti, and L. D. Stefano, "Refrec: Pseudo-labels refinement via shape reconstruction for unsupervised 3d domain adaptation," in *2021 International Conference on 3D Vision (3DV)*, IEEE, 2021.

[2] P. Saraswat, "Supervised machine learning algorithm: A review of classification techniques," in *International Conference on Intelligent Emerging Methods of Artificial Intelligence & Cloud Computing* (F. P. García Márquez, ed.), (Cham), pp. 477–482, Springer International Publishing, 2022.

[3] S. Chowdhury and M. P. Schoen, "Research paper classification using supervised machine learning techniques," in *2020 Intermountain Engineering, Technology and Computing (IETC)*, pp. 1–6, 2020.

[4] X. Liu, F. Zhang, Z. Hou, Z. Wang, L. Mian, J. Zhang, and J. Tang, "Self-supervised learning: Generative or contrastive," *CoRR*, vol. abs/2006.08218, 2020.

[5] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018.

[6] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," *CoRR*, vol. abs/2002.05709, 2020.

[7] K. He, H. Fan, Y. Wu, S. Xie, and R. B. Girshick, "Momentum contrast for unsupervised visual representation learning," *CoRR*, vol. abs/1911.05722, 2019.

[8] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance-level discrimination," *CoRR*, vol. abs/1805.01978, 2018.

[9] J. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. Á. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent: A new approach to self-supervised learning," *CoRR*, vol. abs/2006.07733, 2020.

[10] X. Chen, H. Fan, R. B. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *CoRR*, vol. abs/2003.04297, 2020.

[11] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. B. Girshick, "Masked autoencoders are scalable vision learners," *CoRR*, vol. abs/2111.06377, 2021.

[12] Z. Xie, Z. Zhang, Y. Cao, Y. Lin, J. Bao, Z. Yao, Q. Dai, and H. Hu, "Simmim: A simple framework for masked image modeling," *CoRR*, vol. abs/2111.09886, 2021.

[13] C. Wei, H. Fan, S. Xie, C. Wu, A. L. Yuille, and C. Feichtenhofer, "Masked feature prediction for self-supervised visual pre-training," *CoRR*, vol. abs/2112.09133, 2021.

[14] H. Wang, Q. Liu, X. Yue, J. Lasenby, and M. J. Kusner, "Pre-training by completing point clouds," *CoRR*, vol. abs/2010.01089, 2020.

[15] Z. Zhang, R. Girdhar, A. Joulin, and I. Misra, "Self-supervised pretraining of 3d features on any point-cloud," *CoRR*, vol. abs/2101.02691, 2021.

[16] S. Xie, J. Gu, D. Guo, C. R. Qi, L. J. Guibas, and O. Litany, "Pointcontrast: Unsupervised pre-training for 3d point cloud understanding," *CoRR*, vol. abs/2007.10985, 2020.

[17] J. Sauder and B. Sievers, "Self-supervised deep learning on point clouds by reconstructing space," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.

[18] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. J. Guibas, "Representation learning and adversarial generation of 3d point clouds," *CoRR*, vol. abs/1707.02392, 2017.

[19] S. Yan, Z. Yang, H. Li, L. Guan, H. Kang, G. Hua, and Q. Huang, "Implicit autoencoder for point cloud self-supervised representation learning," *CoRR*, vol. abs/2201.00785, 2022.

[20] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu, "Point-bert: Pre-training 3d point cloud transformers with masked point modeling," *CoRR*, vol. abs/2111.14819, 2021.

[21] J. T. Rolfe, "Discrete variational autoencoders," 2016.

[22] W. V. Gansbeke, S. Vandenhende, S. Georgoulis, M. Proesmans, and L. V. Gool, "Learning to classify images without labels," *CoRR*, vol. abs/2005.12320, 2020.

[23] M. Hamilton, Z. Zhang, B. Hariharan, N. Snavely, and W. T. Freeman, "Unsupervised semantic segmentation by distilling feature correspondences," 2022.

[24] G. Izacard, M. Caron, L. Hosseini, S. Riedel, P. Bojanowski, A. Joulin, and E. Grave, "Towards unsupervised dense information retrieval with contrastive learning," *CoRR*, vol. abs/2112.09118, 2021.

[25] H. Alashwal, M. El Halaby, J. J. Crouse, A. Abdalla, and A. A. Moustafa, "The application of unsupervised clustering methods to alzheimer's disease," *Frontiers in Computational Neuroscience*, vol. 13, 2019.

[26] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," 2014.

[27] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," *CoRR*, vol. abs/1608.06019, 2016.

[28] M. Long, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," *CoRR*, vol. abs/1605.06636, 2016.

[29] B. Sun and K. Saenko, "Deep CORAL: correlation alignment for deep domain adaptation," *CoRR*, vol. abs/1607.01719, 2016.

[30] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," *CoRR*, vol. abs/1702.05464, 2017.

[31] Y. Tsai, W. Hung, S. Schulter, K. Sohn, M. Yang, and M. Chandraker, "Learning to adapt structured output space for semantic segmentation," *CoRR*, vol. abs/1802.10349, 2018.

[32] M. Kim and H. Byun, "Learning texture invariant representation for domain adaptation of semantic segmentation," *CoRR*, vol. abs/2003.00867, 2020.

[33] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, "CyCADA: Cycle-consistent adversarial domain adaptation," in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, pp. 1989–1998, PMLR, 10–15 Jul 2018.

[34] M. Chen, H. Xue, and D. Cai, "Domain adaptation for semantic segmentation with maximum squares loss," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2090–2099, 2019.

[35] Y. Chen, W. Li, C. Sakaridis, D. Dai, and L. V. Gool, "Domain adaptive faster R-CNN for object detection in the wild," *CoRR*, vol. abs/1803.03243, 2018.

[36] T. Wang, X. Zhang, L. Yuan, and J. Feng, "Few-shot adaptive faster r-cnn," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[37] X. Wang, Z. Cai, D. Gao, and N. Vasconcelos, "Towards universal object detection by domain attention," *CoRR*, vol. abs/1904.04402, 2019.

[38] K. Saito, Y. Ushiku, T. Harada, and K. Saenko, "Strong-weak distribution alignment for adaptive object detection," *CoRR*, vol. abs/1812.04798, 2018.

[39] C. Qin, H. You, L. Wang, C. J. Kuo, and Y. Fu, "Pointdan: A multi-scale 3d domain adaption network for point cloud representation," *CoRR*, vol. abs/1911.02744, 2019.

[40] A. Alliegro, D. Boscaini, and T. Tommasi, "Joint supervised and self-supervised learning for 3d real-world challenges," *CoRR*, vol. abs/2004.07392, 2020.

[41] I. Achituve, H. Maron, and G. Chechik, "Self-supervised learning for domain adaptation on point-clouds," *CoRR*, vol. abs/2003.12641, 2020.

[42] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *CoRR*, vol. abs/1612.00593, 2016.

[43] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *CoRR*, vol. abs/1706.02413, 2017.

[44] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *CoRR*, vol. abs/1801.07829, 2018.

[45] Y. Pang, W. Wang, F. E. H. Tay, W. Liu, Y. Tian, and L. Yuan, "Masked autoencoders for point cloud self-supervised learning," 2022.

[46] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. B. Girshick, "Masked autoencoders are scalable vision learners," *CoRR*, vol. abs/2111.06377, 2021.

[47] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017.

[48] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *CoRR*, vol. abs/2010.11929, 2020.

[49] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Rotation invariant spherical harmonic representation of 3D shape descriptors," in *Symposium on Geometry Processing*, June 2003.

[50] S. Salti, F. Tombari, and L. di Stefano, "On the use of implicit shape models for recognition of object categories in 3d data," in *Asian Conference on Computer Vision*, 2010.

[51] Z. Liu, H. Hu, Y. Cao, Z. Zhang, and X. Tong, "A closer look at local aggregation operators in point cloud analysis," *CoRR*, vol. abs/2007.01294, 2020.

[52] Q. Xu, X. Sun, C. Wu, P. Wang, and U. Neumann, "Grid-gcn for fast and scalable point cloud learning," *CoRR*, vol. abs/1912.02984, 2019.

[53] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling," *CoRR*, vol. abs/2003.00492, 2020.

[54] Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis," *CoRR*, vol. abs/1904.07601, 2019.

[55] M. A. Uy, Q. Pham, B. Hua, D. T. Nguyen, and S. Yeung, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," *CoRR*, vol. abs/1908.04616, 2019.

[56] A. Goyal, H. Law, B. Liu, A. Newell, and J. Deng, "Revisiting point cloud shape classification with a simple and effective baseline," in *Proceedings of the 38th International Conference on Machine Learning* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 3809–3820, PMLR, 18–24 Jul 2021.

[57] A. Dai, A. X. Chang, M. Savva, M. Halber, T. A. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," *CoRR*, vol. abs/1702.04405, 2017.

[58] B.-S. Hua, Q.-H. Pham, D. T. Nguyen, M.-K. Tran, L.-F. Yu, and S.-K. Yeung, "Scenenn: A scene meshes dataset with annotations," in *International Conference on 3D Vision (3DV)*, 2016.

[59] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," *CVPR 2011*, pp. 1521–1528, 2011.

[60] W. M. Kouw and M. Loog, "A review of domain adaptation without target labels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, pp. 766–785, 2021.

[61] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface re-integration," *CoRR*, vol. abs/1604.01093, 2016.

[62] D.-H. Lee, "Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks," 2013.

[63] Y. Zou, Z. Yu, B. V. K. V. Kumar, and J. Wang, "Domain adaptation for semantic segmentation via class-balanced self-training," *CoRR*, vol. abs/1810.07911, 2018.

[64] Y. Yang, C. Feng, Y. Shen, and D. Tian, "Foldingnet: Interpretable unsupervised learning on 3d point clouds," *CoRR*, vol. abs/1712.07262, 2017.

[65] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "Atlasnet: A papier-mâché approach to learning 3d surface generation," *CoRR*, vol. abs/1802.05384, 2018.

[66] H. Deng, T. Birdal, and S. Ilic, "Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors," *CoRR*, vol. abs/1808.10322, 2018.

[67] P. H. O. Pinheiro, "Unsupervised domain adaptation with similarity learning," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8004–8013, 2018.

[68] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," *CoRR*, vol. abs/1901.00976, 2019.

[69] A. Tarvainen and H. Valpola, "Weight-averaged consistency targets improve semi-supervised deep learning results," *CoRR*, vol. abs/1703.01780, 2017.

[70] Z. Wu, S. Song, A. Khosla, X. Tang, and J. Xiao, "3d shapenets for 2.5d object recognition and next-best-view prediction," *CoRR*, vol. abs/1406.5670, 2014.

[71] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and

F. Yu, "Shapenet: An information-rich 3d model repository," *CoRR*, vol. abs/1512.03012, 2015.

[72] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[73] Y. Rubner, C. Tomasi, and L. Guibas, "The earth mover's distance as a metric for image retrieval," *International Journal of Computer Vision*, vol. 40, pp. 99–121, 01 2000.

Ringraziamenti

Desidero ringraziare il Professore Luigi di Stefano, Adriano Cardace, Pierluigi Zama Ramirez e Riccardo Spezialetti per la disponibilità, i consigli e la professionalità con cui è stato possibile portare a termine questo elaborato.

Colgo inoltre l'occasione per ringraziare famiglia, amici e fidanzata per avermi supportato in momenti difficili ed aver vissuto con me piccoli traguardi di questo percorso universitario.

Un ringraziamento particolare va, però, ai miei coinquilini della casa a Bologna, con i quali ho trascorso giornate dense di emozioni ed ho potuto condividere un'esperienza di vita unica che porterò con me.