Università degli Studi di Firenze

Scuola di Ingegneria - Dipartimento di Ingegneria dell'Informazione

---

Tesi di Laurea Triennale in Ingegneria Informatica

Analisi comparativa di Metriche per la valutazione di modelli generativi

Candidato
Rocco Tescaro

Relatore
Prof. Paolo Frasconi

Correlatori

Anno Accademico 2024

Indice

Introduzione

Negli ultimi anni, i modelli di reti neurali generative hanno registrato notevoli progressi, trovando applicazione in vari ambiti quali la creazione automatica di immagini, la sintesi musicale, la generazione di testi e molto altro. Algoritmi come le *Generative Adversarial Networks (GANs)* hanno prodotto risultati di grande rilievo, aprendo nuove prospettive sia per la ricerca scientifica che per applicazioni industriali e commerciali.

Con l'avanzamento delle capacità di generazione, è emersa la necessità di sviluppare metodi oggettivi e affidabili per la valutazione della qualità dei modelli generativi, andando oltre la soggettività dell'ispezione visiva umana, che sebbene in ultima analisi rimane comunque il metodo più affidabile e diffuso, risulta intrinsecamente non replicabile, è soggetta a variabilità tra diversi osservatori ed è difficilmente scalabile in contesti che richiedono una valutazione in larga scala.

Per rispondere a questa esigenza, inizialmente sono state proposte metriche scalari, basate quindi su singoli indici, tra cui il Frechet Inception Distance (FID), progettate per stimare la somiglianza statistica tra campioni generati e reali. Tuttavia, tali metriche hanno mostrato limiti nel descrivere con precisione proprietà diverse delle distribuzioni dei dati generati, quali la *fidelity* (la somiglianza tra i campioni generati e quelli reali) e la *diversity* (la varietà tra i campioni generati).

Per superare questi limiti, sono state introdotte metriche più complesse, capaci di fornire una valutazione più articolata della qualità dei modelli. Tra queste metriche, particolare attenzione sarà riservata all'analisi di *Improved Precision and Recall*, *Density and Coverage*, *Probabilistic Precision and Recall* e *Precision and Recall Cover*. È importante sottolineare la corrispondenza concettuale tra i termini *fidelity* e *precision*, così come tra *diversity* e *recall*. Tuttavia, nel contesto delle metriche di precision e recall, il calcolo si basa specificamente sulla distanza tra campioni generati e reali, nonché tra ciascun campione e i suoi vicini all'interno dello spazio delle caratteristiche.

Nei capitoli che seguiranno verrà approfondita la definizione di tali metriche e le relative versioni più generali e complesse definite in tempi più recenti, vale a dire le *Precision Recall Curves*. Veranno quindi introdotti gli esperimenti condotti e i relativi risultati.

La nostra ricerca si propone di analizzare e confrontare le metriche di valutazione della qualità dei modelli generativi, vale a dire caratteristiche e limiti come la dipendenza dai diversi iperparametri, la sensibilità alle dimensioni del dataset, la resistenza agli *outliers*, la capacità di discriminare dati generati di altà qualità da quelli di bassa qualità e quindi le possibilità di filtrare i risultati ottenuti.

Capitolo 1

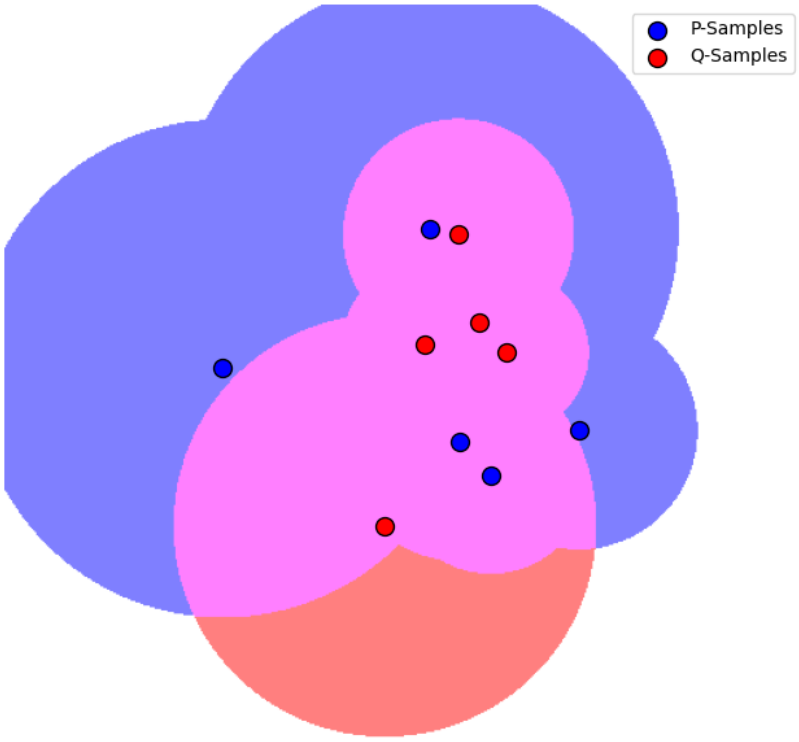
Metriche

Le protagoniste della nostra analisi sono, come anticipato nell'introduzione: *Improved Precision and Recall*, *Density and Coverage*, *Probabilistic Precision and Recall* e *Precision and Recall Cover*. Tutte basate su un calcolo simile dipendente dalla distanza tra campioni generati e reali e la distanza fra i vicini. E utile definire meglio dei concetti appena accennati nell introduzione:

- **Precision**: è la frazione di campioni generati che ricadono nel supporto della distribuzione reale.
- **Recall**: è la frazione di campioni reali che ricadono nel supporto della distribuzione generata.
- **Fidelity**: è la somiglianza tra i campioni generati e quelli reali.
- **Diversity**: è la varietà tra i campioni generati.

Sebbene Fidelity e Diversity siano concetti importanti, non sono direttamente misurabili data la loro astrattezza, si riduce quindi il problema del calcolo della Fidelity e Diversity a quello del calcolo di Precision e Recall, che sono

invece misure più facilmente formalizzabili. Entrambe si basano sul concetto di supporto della distribuzione. Dato un numero limitato di campioni, definire un supporto continuo è un problema non banale. Le diverse metriche si limitano a dare diverse stime di tale supporto. In questo contesto la stima del supporto è detto **manifold**.



1.1 Improved Precision and Recall

Nonostante sia nominalmente dichiarato come "Improved" Precision Recall e quindi sottointendendo un miglioramento ad una versione meno 'potente' di Precision Recall, in realtà la metrica proposta da Sajjadi et al. [**?**] è antecendente a quella che verrà presentata in seguito (chiamata più semplicemente Precision Recall Cover) ed è quindi la forma più semplice di Precision Recall che analizzeremo e dalla quale partiremo.

Sia R il dataset reale e G il dataset generato, identifichiamo con Φ_R e Φ_G le caratteristiche estratte da R e G rispettivamente. Il manifold stimato da R e G è un insieme di ipersfere con un certo raggio e centrate nei vari putni di Φ_R e Φ_G. Il raggio di queste ipersfere è determinato da un iperparametro k che rappresenta il numero dei *nearest neighbors* da considerare. Possiamo quindi definire una funzione binaria che determini se un certo punto appartiene al manifold o meno:

$$f_{ipr}(x, \Phi) = \begin{cases} 1 & \text{if} \quad \exists\ y \in \Phi \text{ such that } ||x - y||_2 \leq ||y - NN_k(y, \Phi)||_2 \\ 0 & \text{otherwise.} \end{cases} \tag{1.1}$$

Dove $NN_k(y, \Phi)$ è il k-th nearest neighbor di y in Φ.

La Precision e la Recall possono essere quindi definite come:

$$Precision(\Phi_R, \Phi_G) = \frac{1}{|\Phi_G|} \sum_{g \in \Phi_G} f_{ipr}(g, \Phi_R) \tag{1.2}$$

$$Recall(\Phi_R, \Phi_G) = \frac{1}{|\Phi_R|} \sum_{r \in \Phi_R} f_{ipr}(r, \Phi_G) \tag{1.3}$$

Come ripeteremo in seguito la metrica risulta dipendente dall'iperparametro k e in particolare pare che per $k = 3$ si ottengano i risultati migliori.

1.2 Density and Coverage

Come vedremo in seguito l'Improved Precision Recall soffre la presenza di outliers nei datasets, vale a dire che il manifold stimato da R e G può essere fortemente influenzato da pochi punti molto distanti dalla distribuzione principale. La Density and Coverage è una metrica che cerca di mitigare questo problema. Invece che contare se le varie ipersfere in Φ_R contengono almeno un punto di Φ_G si conta quanti punti di Φ_G sono contenuti in ciascuna ipersfera di Φ_R. Questo conporta però che la density non è una metrica normalizzata e in certi casi assume valori maggiori di 1.

$$Density(\Phi_R, \Phi_G) = \frac{1}{k|\Phi_G|} \sum_{g \in \Phi_G} \sum_{r \in \Phi_R} \mathbb{1}_{||r-g||_2 \leq ||g-NN_k(g,\Phi_G)||_2} \tag{1.4}$$

Rispetto alle altre metriche in cui si poteva apprezzare la simmetria fra le formule per la stima di fidelity e diversity, in questo caso la Coverage è una metrica completamente diversa dalla Density. Questo perchè rispetto a Φ_G, Φ_R non dovrebbe presentare outliers.

$$Coverage(\Phi_R, \Phi_G) = \frac{1}{|\Phi_R|} \sum_{r \in \Phi_R} \mathbb{1}_{\exists\ g \in \Phi_G \text{ such that } ||r-g||_2 \leq ||r-NN_k(r,\Phi_R)||_2} \tag{1.5}$$

La formula potrebbe sembrare molto simile a quella dell'Improved Precision ma con un'importante differenza, ci chiediamo se esiste almeno un punto di Φ_G per ciascuna ipersfera costruita su Φ_R e non se per ciascun punto di Φ_G esiste una ipersfera che lo contiene. Al contrario della Density, la Coverage è una metrica normalizzata e assume valori compresi tra 0 e 1.

1.3 Precision and Recall Cover

Mantenendo la formalizzazione della sezione precedente si introduce un nuovo iperparametro $k' = Ck$. L'idea è quella di fornire un nuovo elemento

per regolare la dimensione del manifold, e in particolare le aree da considerare trascurabilmente piccole e quelle sufficientemente grandi. L'algoritmo proposto prevede per la Precision di costruire un manifold su Φ_G con ipersfere con raggio pari alla distanza dal k'-th nearest neighbor e contare il numero di ipersfere che contengono almeno k punti di Φ_R, si divide quindi per il numero di ipersfere totali (la dimensione di Φ_G). Per la Recall si procede in modo analogo ma invertendo i ruoli di Φ_R e Φ_G. E' importante notare come la metrica sia fondamentalmente la stessa della Coverage ma non è più sufficiente che esista un unico punto di Φ_R in un ipersfera di Φ_G ma è necessario invece che ce ne siano almeno k.

$$Precision(\Phi_R, \Phi_G) = \frac{1}{|\Phi_G|} \sum_{g \in \Phi_G} \mathbb{1}_{\sum_{r \in \Phi_R} f_{ipr}(r, \Phi_G, \frac{k'}{|\Phi_G|}) \geq \frac{k}{|\Phi_R|}} \tag{1.6}$$

$$Recall(\Phi_R, \Phi_G) = \frac{1}{|\Phi_R|} \sum_{r \in \Phi_R} \mathbb{1}_{\sum_{g \in \Phi_G} f_{ipr}(g, \Phi_R, \frac{k'}{|\Phi_R|}) \geq \frac{k}{|\Phi_G|}} \tag{1.7}$$

In questo caso l'aggiunta di un nuovo parametro, nonostante aumenti la complessità combinatoria di possibili valori assegnabili, si verifica sperimentalmente che mantenendo $C = 3$ la scelta di k e conseguentemente di k' può essere arbitraria ottenendo comunque buoni risultati.

1.4 Probabilistic Precision and Recall

Un altro approccio al problema della stima del manifold è quello di considerare la densità di probabilità delle distribuzioni reali e generate. La formula è la stessa dell'Improved Precision e Recall ma invece che f_{ipr} si considera la probabilità che un punto appartenga al manifold. Questa probabilità è calcolabile suddividendo il supporto in sottosupporti stimati anch'essi. Seguendo

il paper Suhyun Kim et al. [**?**] si considera la seguente formula:

$$P(x \in subSupp(y)) = \begin{cases} 1 - \frac{||x-y||_2}{R} & \text{if} \quad ||x-y||_2 \leq R \\ 0 & \text{otherwise.} \end{cases} \tag{1.8}$$

ne conseguentemente che

$$P(x \in manifold(y)) = 1 - \prod_{i=1}^{N}(1 - P(x \in subSupp(y_i))) \tag{1.9}$$

Chiaramente per la precision itera la x su Φ_G e il manifold su Φ_R e viceversa per la recall. Meno chiaro è cosa si indica con R. L'idea è analoga a quella della Kernel Density Estimation con varianza fissata, si considera un raggio fisso per tutte le ipersfere. La scelta di R è cruciale e sempre seguendo il paper si è scelto $R_{precision} = \frac{a}{|\Phi_R|}\sum_{r\in\Phi_R}||r - NN_k(r, \Phi_R)||_2$. ovvero la media della distanza dal k-th nearest neighbor per ogni punto di Φ_R per la precision e analogamente per la recall. Si introducono quindi nuovi iperparametri come a il cui valore consigliato è 1.2.

$$Precision(\Phi_R, \Phi_G) = \frac{1}{|\Phi_G|}\sum_{g\in\Phi_G} P(g \in manifold(\Phi_R)) \tag{1.10}$$

$$Recall(\Phi_R, \Phi_G) = \frac{1}{|\Phi_R|}\sum_{r\in\Phi_R} P(r \in manifold(\Phi_G)) \tag{1.11}$$

1.5 Precision Recall Curve

Studi più recenti hanno dimostrato che precision e recall sono combinazioni lineari di errori di tipo I e II di un classificatore binario (ottimo). E possibile quindi costruire un limite superiore per la Precision e la Recall con un classificatore binario non ottimo, il metodo permette inoltre di avere un idea più precisa del trade-off tra Precision e Recall, costruendo una curva della quale le metriche viste in precedenza rappresentano solo i valori estremi.

A seguito sono riportate le formule per i classificatori corrispondenti alle metriche viste in precedenza:

$$f_\lambda^{ipr}(x) = \mathbb{1}_{\lambda f_{ipr}(x,\Phi_R) \geq f_{ipr}(x,\Phi_G)} \tag{1.12}$$

$$f_\lambda^{cov}(x) = \mathbb{1}_{\lambda |\{r \in \Phi_R, \text{s.t.} ||r-x||_2 \leq ||r-NN_k(x,\Phi_G)||_2\}| \geq |\{g \in \Phi_G, \text{s.t.} ||g-x||_2 \leq ||g-NN_k(x,\Phi_R)||_2\}|} \tag{1.13}$$

Dal momento che questi classificatori vanno a cotruire un limite superiore, è possibile prendere il minimo fra le curve ottenute con i diversi classificatori per ottenere una stima più precisa. Risulta poi che f_λ^{ipr} e fondamentalmente una KDE con bandwidth variabile.

1.6 Altre metriche e funzioni correlate

Per aggiungere un appendice alla sezione precedente, sono state brevemente sfruttate altre metriche/classificatori accennati nel medesimo paper. In particolare sono citate il knn classifier:

$$f_\lambda^{cov}(x) = \mathbb{1}_{\lambda |\{r \in \Phi_R, \text{s.t.} ||r-x||_2 \leq ||r-NN_k(x,\Phi_G \cup \Phi_R)||_2\}| \geq |\{g \in \Phi_G, \text{s.t.} ||g-x||_2 \leq ||g-NN_k(x,\Phi_G \cup \Phi_R)||_2\}|} \tag{1.14}$$

che è come il cov classifier ma costruisce l'ipersfera su entrambi i dataset e il parzen classifier che invece va a considerare delle iperfere di dimensione fissata (con raggio R come visto nella sezione della Probabilistic Precision Recall). Il parzen classifier è analogo ad un KDE con bandwidth fissata.

Sebbene non sia propriamente una metrica di valutazione per i GAN, la Kernel Density Estimation è un metodo di stima della densità di probabilità di un dataset, ed è stata utilizzata nei nostri esperimenti per osservare alcune caratteristiche dei dataset reali e generati. In particolare sono state sfruttate l'approssimazione di Silverman e nelle fasi di testing anche la stima di Scott.

Per quanto riguarda l'applicazione dei classificatori per la costruzione della Precision Recall Curve, i dataset sono stati divisi in training e testing set. Mentre per gli estrattori di caratteristiche sono state utilizzate metriche informate e non informate (per queste chiariremo meglio in seguito).

TODO mostrare meglio il rapporto pr-curve - manifold e manifold-metriche, spiegare kde, correggere errori di battitura (putni) e aggiungere immagini e bibliografia.

Capitolo 2

Esperimenti

Per la valutazione delle metriche sono stati condotti diversi esperimenti che possono essere suddivisi in due macro-categorie dipendentemente dal tipo di dataset utilizzato:

- **Toy Dataset**: dataset generati artificialmente per testare il comportamento delle metriche in condizioni controllate.
- **Real World Dataset**: dataset reali per testare il comportamento delle metriche in condizioni reali.

Le sezioni seguenti descrivono come sono state implementate le metriche, gli esperimenti così come le distribuzioni di dati utilizzate. La ragione per cui sono stati condotti esperimenti su dateset generati artificilmente (ovvero di matrice matematica, non derivanti dalla realtà o generati da reti neurali) è che in questo modo è stato possibile osservare il comportamento delle metriche in condizioni controllate, ideali e per poter confrontare i risultati ottenuti con quelli attesi presenti nella letteratura. Fanno infatti parte dei test su 'toy dataset', i test di corretta implementazione ovvero un'analisi comparativa delle diverse implementazioni in codice delle metriche. Tali test hanno il fine

di verificare che le metriche restituiscano valori corretti validando gli altri esperimenti.

I dataset reali sono stati utilizzati per testare il comportamento delle metriche in condizioni reali, ovvero per verificare se le metriche si comportano come ci si aspetta in situazioni non ben definite, con distribuzioni di dati non note, ben distanti da quelle ideali. Questo infatti, come vedremo, può sollevare criticità, ad esempio dovute alla assenza di certe categorie di dati o alla presenza di dati non ben distribuiti. Un altro elemento dei dataset reali è che spesso questi non sono di carattere prettamente numerico, nei dataset analizzati ad esempio, abbiamo immagini di farfalle e partiture musicali. Questo comporta la necessità di trasformare i dati per poterli utilizzare, e il processo di **estrazioni di caratteristiche** può equivalere ad una perdita di dati significativi o, al contrario, ad una sovrabbondanza di dati non significativi.

2.1 Toy Dataset

Come detto nell'introduzione di questo capitolo, i Toy Dataset sono dataset generati artificialmente. Questo vuol dire che per la creazione di questi dataset sono state utilizzate delle funzioni matematiche che generano dati in modo casuale, ma controllato. Le distribuzioni di dati utilizzate sono state: distribuzione uniforme e distribuzione normale. Per una certa categoria di esperimenti sono stati poi aggiunti degli outliers, ovvero dei dati che si discostano molto dalla distribuzione principale. Ripercorrendo gli esperimenti presenti in letteratura gli esperimenti condotti volgono a testare l'influenza degli iperparametri delle metriche sulle stesse, la prezenza di outliers, la dimensione dei dataset e un'analisi comparativa delle implementazioni in codice

delle diverse metriche. Quest'ultima in particolare, si suddivide in confronto delle metriche scalari e confronto delle pr-curve.

Per la generazione di dataset con distribuzione uniforme è stata utilizzata la funzione `numpy.random.uniform`, mentre per la generazione di dataset con distribuzione normale è stata utilizzata la funzione `numpy.random.multivariate_normal` di numpy. La prima oltre alla dimensione dei sample e al numero prende come parametri anche il range di valori che i dati possono assumere, mentre la seconda prende come parametri lo shift (che di default è 0). Come media avremo quindi `shift*numpy.ones(dim)` e come covarianza la matrice identità (`numpy.eye(dim)`). Sono state poi adottate una serie di funzioni per facilitare il debugging attraverso la visualizzazione dei dati. In particolare una funzione che mostri dati di due dimensioni con il corrispondente manifold e fuzioni come il **realism score** che permette di valutare la verosimiglianza dei singoli dati generati.

Per ciascun esperimento sono stati prodotti dei grafici che mostrano i risultati ottenuti (fa eccezione solo il test di corretta implementazione delle metriche scalari). Questi grafici sono stati prodotti utilizzando la libreria `matplotlib` di python. Il grafico utilizzato dipende dall'esperimento condotto. Oltre alla produzione di grafici, data la complessità computazionale di alcuni esperimenti, sono stati salvati i risultati in file `.npy` per poterli analizzare in un secondo momento.

2.1.1 Parametro k e dimensione del dataset

Come abbiamo visto, tutte le metriche di cui ci interessa l'analisi si basano sulla distanza dei dati rispetto ai loro vicini. Uno dei parametri più determinanti è l'ordine k del vicino più prossimo. Secondo la letteratura, i valori ottimali di k sono i seguenti: per l'**improved precision recall** $k = 3$,

per la **probabilistic precision recall** $k = 4$, per la **precision recall coverage** $k = \sqrt{\text{len}(P\text{Samples})}$, e per **density** e **coverage** $k = 5$. Inoltre, si attende intuitivamente che l'aumento della dimensione del dataset comporti un incremento dei valori delle metriche, a condizione che vi sia corrispondenza tra le due distribuzioni. L'analisi è stata quindi condotta in relazione alla dimensione del dataset e sulle due distribuzioni anticipate precedentemente: distribuzione uniforme e distribuzione normale. Come per gli esperimenti successivi, i dataset su cui sono state valutate le diverse metriche sono rimasti costanti; il dataset è generato una sola volta per tutte le metriche, evitando così discrepanze nei risultati. I test sono stati effettuati per valori di k variabili da 1 a 10, ovvero $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]$, e per dimensioni del dataset da 500 a 16000, con valori $[500, 1000, 2000, 4000, 8000, 16000]$. Ciascun dato è rappresentato come un vettore in $\mathbb{R}^{64}$.

Per presentare i risultati, sono state utilizzate delle **heatmap**, che mostrano i valori delle metriche in relazione ai valori di k e alla dimensione del dataset, con il rosso indicante valori prossimi a 1 e il verde per valori vicini a 0. Nonostante le heatmap consentano di visualizzare la matrice di valori registrati (con il parametro k in funzione della dimensione dei dati) senza la necessità di un numero eccessivo di grafici (considerando il numero di distribuzioni e metriche), questa densità di valori offre solo un'idea intuitiva delle metriche, poiché non fornisce informazioni precise a causa della difficoltà di collegare il colore del grafico a un valore numerico specifico. Tale tipologia di grafico è stata scelta principalmente per poterla confrontare con gli esperimenti presenti in letteratura.

2.1.2 Dimensione del dataset e dimensione dei dati

In questo caso, l'analisi condotta non ha un riscontro diretto nella letteratura esistente, ovvero non presenta antecedenti (quantomeno per gli articoli presi in analisi). L'obiettivo è determinare come la **dimensione del dataset** possa influenzare la densità dei dati della distribuzione (a dimensione dei dati fissata) e, conseguentemente, il valore delle metriche. I risultati di questa analisi costituiscono una parte fondamentale per le analisi su dati reali, dove la scelta del numero di caratteristiche da considerare può risultare determinante.

In questo esperimento, abbiamo considerato una distribuzione normale ($\mathcal{N}(0, I)$) dei dati con **dimensione del dataset variabile** da 50 a 1600, con valori $[50, 100, 200, 400, 800, 1600]$, e **dimensione dei dati** da 2 a 64, con valori $[2, 4, 8, 16, 32, 64]$. A differenza degli esperimenti precedenti, rappresentati tramite heatmap, qui l'assenza di un riscontro nella letteratura ci ha permesso di utilizzare **grafici a linee bidimensionali** per rappresentare i risultati (data la ridotta dimensionalità, almeno per le dimensioni di interesse, di uno degli iperparametri da regolare, ovvero la dimensione dei dati). Il valore dell'**iperparametro** k è stato scelto in accordo con quanto suggerito nei vari articoli, per garantire la massima efficacia della metrica. Le misurazioni sono state ripetute 25 volte e successivamente mediate. Anche in questo caso, il calcolo è stato effettuato in parallelo.

2.1.3 Outliers

Una delle proprietà più rilevanti da esaminare nelle diverse metriche è la loro **resistenza agli outliers**. In linea con la letteratura, abbiamo analizzato come i valori delle metriche cambiano in presenza di dataset con distribuzione

normale, **senza outliers** e con **outliers** inseriti sia nei dati reali sia in quelli generati.

L'esperimento si è svolto considerando una distribuzione reale fissa $X \sim \mathcal{N}(0, I)$ e una distribuzione generata $Y \sim \mathcal{N}(\mu, I)$, con uno shift della media μ variabile in $[-1, 1]$ (con **step** di `0.05`). In aggiunta, sono stati esaminati due scenari di outliers, in cui un campione estremo a $x = +1$ è stato aggiunto ai dati reali o a quelli generati. Lo spazio di lavoro è stato definito in $\mathbb{R}^{64}$, con vettori reali centrati sull'origine e campioni generati con media variabile lungo la direzione del vettore unitario. Come per gli altri esperimenti condotti, le operazioni sono state svolte in **parallelo**.

Oltre alle metriche di **precision-recall** e **density-coverage** (come nel paper), i test sono stati condotti anche sulle metriche di **probabilistic precision-recall** e **improved precision-recall**. Per ciascuna metrica sono stati utilizzati i valori degli iperparametri suggeriti dai rispettivi articoli.

In assenza di outliers, ci si attende che i valori delle metriche diminuiscano gradualmente man mano che μ si allontana da zero, indicando correttamente la divergenza tra le due distribuzioni.

2.1.4 Comparazione con implementazioni esistenti

Non tutti i papers analizzati presentavano un'implementazione delle metriche in codice. Sono stati svolti dei test confrontando su dataset identici le diverse implementazioni delle metriche presenti in letteratura. Non sono stati possibili confronti diretti per quanto riguarda la **precision-recall coverage**, in quanto non sono state trovate implementazioni in codice, mentre per la **improved precision-recall** sono state confrontate due diverse implementazioni. Sono state scelte tre diverse distribuzioni di dati: distribuzione uniforme, distribuzione normale e distribuzione normale con media in

$3/\sqrt{dim}$. Ciascuna distribuzione è stata generata con dimensione del dataset pari a 10000 e dimensione dei dati pari a 64. I risultati sono stati riportati su un file `log.txt` per poter essere confrontati in un secondo momento. Allo scopo di velocizzare la computazione e ridurre il tempo di esecuzione, i test comparativi per la **probabilistic precision-recall** e la **density-coverage** sono stati eseguiti con ordine $k = 3$, nonostante i valori ottimali suggeriti dalla letteratura siano diversi. Questo hai infatti permesso di calcolare le distanze intraset una sola volta, evitando di ripeterle per ogni valore di k e dato che non eravamo interessati a valutare l'efficacia delle metriche, quanto confrontare le diverse implementazioni. TODO Aggiungere bibliografia per le implementazioni in codice delle metriche.

2.1.5 Riproduzione delle pr-curve

Anche per la riproduzione delle pr-curve sono stati utilizzati dataset generati artificialmente. L'articolo di riferimento per questo esperimento [**?**], non presentava un'implementazione in codice delle metriche, ma solo i risultati ottenuti. Abbiamo quindi replicato le pr-curve per i quattro classificatori presentati nel paper, vale a dire il classificatore **ipr**, **coverage**, **knn** e **parzen**, per due differenti distribuzioni di dati, in particolare distribuzioni normali con media in 0 per i dati reali e $1/\sqrt{dim}$ e $3/\sqrt{dim}$ per i dati generati ($dim = 64$). I classificatori hanno operato su un dataset di 20000 elementi, con 10000 elementi per ciascuna classe. Per gli esperimenti condotti i dati sono stati divisi in training e test set sia operando uno split del 50% (ovvero 5000 punti effettivi per classe) sia senza split. Sono stati inoltre scelti due valori di k ovvero $k = 4$ e $k = \sqrt{n}$ (dove n è il numero di punti nel training set). Osservando i risultati del paper ci si aspetta che delle quattro pr-curve generate, la coverage-curve sia la più estrema, ovvero quella che produce un

risultato migliore (più vicino al classificatore ottimale). Un'altra proprietà attesa è la simmetria delle curve rispetto alla diagonale, questo è dovuto al tipo di distribuzione dei dati utilizzata e al fatto che i training set fossero bilanciati. Dei test preliminari hanno poi mostrato fondamentale la scelta del range di valori e degli step per quanto riguarda la variabile λ (ovvero il parametro che regola la trade-off tra precision e recall). Come consigliato da [**?**], il range di valori è stato generato dalla formula $\tan(\pi/2 * i/(g+1))$ con $i \in [1, g]$ e $g = 1001$ il numero di valori generati. Questa trasformazione consente di esplorare diverse scale di λ con una densità variabile: i valori crescono rapidamente da 0 a 1, variano lentamente vicino a $\pi/2$, e infine aumentano rapidamente verso l'infinito. Questa caratteristica rende la funzione adatta per analizzare con precisione le transizioni critiche della PR-curve in regioni chiave, bilanciando una copertura fine e una rapida esplorazione delle estremità. In fase sperimentale sono state utilizzate altre funzioni per coprire il range di valori di λ, ma la funzione sopra descritta è risultata la più adatta per l'analisi delle curve, e quella che ha prodotto i risultati più simili a quelli presenti in letteratura.

2.2 Real World Dataset

Come anticipato nell'introduzione di questo capitolo, oltre agli esperimenti condotti in ambienti controllati, regolati e basati su dati sintetici, è fondamentale analizzare il comportamento delle metriche in condizioni reali, ovvero su dataset rappresentativi di problemi pratici. Questa fase di sperimentazione consente di testare l'applicabilità delle metriche in contesti che vanno oltre l'ambito strettamente numerico e teorico, avvicinandosi alle condizioni operative in cui tali strumenti dovrebbero operare. In particolare, l'o-

biettivo finale delle metriche studiate è proprio quello di fornire un supporto concreto nell'analisi della qualità dei dati generati, facilitando l'integrazione delle reti generative in applicazioni pratiche.

Gli esperimenti sui dati reali sono stati condotti su due dataset distinti: un set di immagini raffiguranti farfalle e una collezione di partiture musicali di Alessandro Scarlatti, compositore rappresentativo della musica barocca. Questi dataset presentano specificità intrinseche che richiedono l'estrazione di caratteristiche rilevanti dal dominio dei dati (in particolare per le immagini utilizzare i `raw data` sarebbe improponibile data la loro dimensione). Per le immagini delle farfalle si è scelto di lavorare con feature basilari, come istogrammi di colore e saturazione, per valutare l'abilità delle metriche nel rilevare differenze qualitative senza fare ricorso a rappresentazioni complesse o specifiche del dominio. Nel caso delle partiture musicali, invece, le caratteristiche estratte sono state più mirate e informate dal dominio della musica barocca seguendo quanto descritto nella letteratura [**?**]. Sono state utilizzate, ad esempio, informazioni di carattere ritmico e tonali. Questo approccio permette di valutare le metriche su dati complessi con maggiore precisione.

Un ulteriore strumento di analisi utilizzato in questo contesto è la **Kernel Density Estimation** (**KDE**), che si è dimostrata particolarmente utile per ottenere una stima non parametrica della distribuzione dei dati. La **KDE** permette di visualizzare come i dati siano distribuiti nel loro spazio delle caratteristiche, fornendo così un quadro più completo delle relazioni tra i campioni reali e quelli generati. Questa informazione è cruciale per interpretare meglio il comportamento delle **metriche**, soprattutto quando si cerca di identificare regioni di alta o bassa densità che potrebbero indicare rispettivamente dati generati di alta qualità o outlier.

Infine, un obiettivo centrale di questa fase sperimentale è quello di veri-

ficare l'efficacia delle metriche nel discriminare dati generati di alta qualità da dati generati di bassa qualità, fungendo così da filtro ultimo per le reti generative. In questa ottica, le metriche potrebbero operare come strumento di selezione, scartando i dati che non soddisfano determinati standard di qualità e potenzialmente indicando i campioni da rigenerare.

2.2.1 Butterflies

Gli esperimenti condotti sul dataset di immagini di farfalle si sono basati su un'analisi semplice ma efficace delle caratteristiche visive, sfruttando estrattori di caratteristiche basati su istogrammi. In particolare, per ogni immagine sono stati calcolati sei tipi di istogrammi:

- **hue histogram**
- **saturation histogram**
- **value histogram**
- **grayscale histogram** (diverso dal value histogram, in questo caso abbiamo una combinazione lineare dei valori RGB)
- **rgb histogram**
- **hsv histogram**

Ogni istogramma è stato generato utilizzando 256 **bin** (per un totale di 256*3 bin per le rappresentazioni **RGB** e **HSV**), con l'obiettivo di catturare le distribuzioni dei valori cromatici e di intensità nelle immagini.

Per la rilevazione dei **falsi positivi**, è stato utilizzato un classificatore analogo a quello impiegato negli esperimenti sulle **IPR-curve**. In questo caso, le differenze tra istogrammi sono state misurate utilizzando sia la norma

l_1 che la norma l_2 come funzioni di distanza. Il classificatore ha operato su un **k-nearest neighbors** (**k-NN**) con $k = 3$ e $k = \sqrt{n}$, dove n è il numero di punti nel training set. Per valutare le prestazioni del classificatore, gli esperimenti sono stati condotti utilizzando diversi schemi di divisione dei dati: uno split 80-20 per il **training** e il **test**, e un approccio senza divisione, in cui tutti i dati venivano considerati come parte di un unico set per la classificazione.

2.2.2 Scarlatti

Capitolo 3

Conclusioni

...

Bibliografia