POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria Informatica

Tesi di Laurea Magistrale

Analisi multivariata di serie temporali mediante contrastive learning

Relatore

Prof. Luca CAGLIERO

Corelatore

Prof. Francesco VACCARINO

Candidato

Nicola BRAILE

Luglio 2022

Sommario

Nella società odierna la produzione di dati temporali è estremamente diffusa. Questo formato dei dati è comune in molti ambiti, a partire da dati finanziari, IoT, sensori in ambito industriale, monitoraggio della salute fino a dati di profilazione dei clienti. La serie sono composte da letture eseguite ad intervalli regolari di una o piu variabili e si distinguono in due categorie: se è presente un'unica variabile si parla di serie univariata altrimenti si tratta di serie multivariata. Nel caso univariato la variabile in osservazione dipende soltanto dai valori passati mentre nel caso multivariato la dipendenza si estende anche alle altre variabili.

Le serie temporali multivariate derivano da fenomeni complessi che non possono essere descritti soltanto da modelli univariati. Infatti questi ultimi non possiedono la capacità di catturare le dipendenze tra le diverse variabili che compongono la serie. I modelli multivariati, considerando le relazioni tra le variabili indipendenti, riescono a trarre conclusioni più accurate e vicine alla realtà. Una delle tecniche di apprendimento auto supervisionato allo stato dell'arte nel rapresentation learning applicato alle serie temporali è il Contrastive Learning. Nell'apprendimento auto supervisionato si cerca di imparare la struttura dei dati attraverso l'uso di un task ausiliario. Nel contrastive learning il task è quello del confronto tra un dato ed una versione aumentata dello stesso. Se la rete è in grado di avvicinare queste coppie ed allontanarle da tutte le altre allora dimostra di saper estrarre delle caratteristiche che sono indipendenti dalla trasformazione applicata e quindi ottenere una rappresentazione che racchiude una semantica. Nello svolgimento di questo lavoro è stato utilizzato il framework TS2Vec, che propone una variante del Contrastive Learning capace di estrarre una rappresentazione della serie temporale a diversi livelli di semantica, confrontando in modo gerarchico lungo la dimensione del tempo e delle istanze.

La tesi inizialmente si focalizza sull'esplorazione delle performance di TS2Vec in diversi scenari, tra cui classificazione di serie temporali e anomaly detection. Poichè si tratta di una soluzione poco flessibile nella gestione di serie multivariate , sono state proposte e valutate tre soluzioni alternative sempre basate sul Contrastive Learning. Tutti i test di classificazione sono stati condotti sulla collezione di dataset UCR ed UEA mentre i test di anomaly detection sono stati condotti sui dataset

Yahoo e KPI. Le tre architetture proposte presentano performance di classificazione migliori rispetto al modello TS2Vec in un'alta percentuale dei dataset multivariati mentre presentano performance comparabili o di poco superiori nel caso univariato. Per quanto riguarda l'anomaly detection i risultati sono inferiori alle aspettative.

Ringraziamenti

alla mia famiglia

Indice

Elenco delle tabelle VIII

Elenco delle figure IX

1 Algoritmi di machine learning 1
1.1 Fondamenti di representation learning . . . 2
1.1.1 Generative . . . 8
1.1.2 Adversarial . . . 9
1.1.3 Contrastive . . . 11

2 Classificazione serie temporali e rilevamento anomalie 16
2.1 Definizione di serie temporali . . . 17
2.2 Classificazione di serie temporali . . . 18
2.2.1 Basate su distanza o features . . . 18
2.2.2 Reti neurali . . . 21
2.3 Rilevazione anomalie nelle serie temporali . . . 24
2.4 Caratterizzazione delle serie temporali . . . 27

3 Ts2vec 30
3.1 Descrizione dell'architettura . . . 33
3.2 Estensioni al caso multivariato . . . 33
3.2.1 Multi Layer Perceptron (MLP) . . . 35
3.2.2 Fully Convolutional (FC) . . . 36
3.2.3 ResNet . . . 37

4 Esperimenti 39
4.1 Dataset . . . 39
4.2 Metriche . . . 45
4.3 Setup degli esperimenti . . . 47
4.4 Statistiche dataset . . . 47
4.5 Risultati classificazione . . . 53

4.6 Analisi dei risultati di rilevamento anomalie 63

5 Conclusioni e sviluppi futuri 64

Bibliografia 66

Elenco delle tabelle

4.1 Caratteristiche dataset UCR 43
4.2 Caratteristiche dataset UEA 44
4.3 Grid search iperparametri 47
4.4 Statistiche dataset univariato UCR 49
4.4 Statistiche dataset univariato UCR 50
4.4 Statistiche dataset univariato UCR 51
4.5 Statistiche dataset multivariato UEA 52
4.6 Confronto architetture proposte e TS2Vec per dataset UCR. *%dataset* indica il rapporto dei dataset in cui l'accuratezza è maggiore, uguale o inferiore a quella di TS2Vec, rispetto ai 124 dataset dell'archivio UCR. ± avg% indica l'incremento medio dell'accuratezza rispetto a TS2Vec. 53
4.7 Confronto architetture proposte e TS2Vec per dataset UEA. *%dataset* indica il rapporto dei dataset in cui l'accuratezza è maggiore, uguale o inferiore a quella di TS2Vec, rispetto ai 30 dataset dell'archivio UEA. ± avg% indica l'incremento medio dell'accuratezza rispetto a TS2Vec. 55
4.8 Confronto architetture proposte con tutti i modelli di benchmark. *%dataset* indica il rapporto dei dataset in cui l'accuratezza è maggiore rispetto a tutte le altre. 55
4.9 Confronto accuratezze tra UEA** ed UEA. UEA** esclude i quattro dataset critici per faeftures *DuckDuckGeese, InsectWingbeat, PEMS-SF, FaceDetection* 59
4.10 Confronto accuratezze tra UCR** ed UCR. UCR** esclude i dataset con numero di classi superiore a 15 59
4.11 Le accuratezze delle tre architetture a confronto con altri metodi sull'archivio UCR 62
4.12 Le accuratezze delle tre architetture a confronto con altri metodi sull'archivio UEA 63
4.13 Risultati rilevamento anomalie su dataset Yahoo, KPI 63

Elenco delle figure

1.1 Modelli deep learning [3] . . . 3
1.2 Esempi di task ausiliario [3] . . . 6
1.3 Categorie algoritmi self-supervised [3] . . . 7
1.4 Architetture self-supervised [3] . . . 8
1.5 Input parziali metodi Adversarial [3] . . . 10
1.6 Deep InfoMax . . . 13
1.7 Architettura SimCLR . . . 14

2.1 Categorie misure di distanza [56] . . . 19
2.2 Confronto distance based e features based [64] . . . 21
2.3 Classificazione serie temporali [54] . . . 22
2.4 Anomalie puntuali [75] . . . 25
2.5 Anomalie contestuali [75] . . . 26
2.6 Anomalie collettive [75] . . . 26

3.1 Distribuzioni serie temporali [85] . . . 31
3.2 Funzionamento ts2vec [85] . . . 34
3.3 Architettura MLP [87] . . . 35
3.4 Architettura FCN [54] . . . 36
3.5 Architettura Resnet [54] . . . 38

4.1 UCR matrice di correlazione . . . 56
4.2 UEA matrice di correlazione . . . 57

Capitolo 1

Algoritmi di machine learning

Il termine intelligenza artificiale è usato in modo generale per intendere la capacita di una macchina ad avere un qualche comportamento umano come il pensiero, la pianificazione o l'apprendimento. Il Machine learning è un sottoinsieme dell'intelligenza artificiale che si occupa dell'apprendimento automatico di un sistema. E' una tecnologia che sarà sempre più presente nella nostra vita poichè riesce a sfruttare abilita che superano quelle umane. In particolare la capacita di elaborazione di grandi quantità di informazioni e la successiva estrazione di pattern che descrivono un qualche fenomeno molto complesso. Con queste tecniche è possibile distillare dai dati un pezzo di conoscenza che sarà poi utilizzato per finalizzare una decisione o qualche altro comportamento da parte del nostro sistema.

In questo contesto imparare significa mettersi nella condizione di svolgere un' attività futura dopo aver seguito una preparazione preliminare ovvero l'elaborazione di alcune informazioni. In base al tipo di attività da eseguire si differenziano diverse metodologie di apprendimento Figura 1.1, la più comune è il supervised learning [1].

Apprendimento supervisionato e non supervisionato I dati da elaborare sono correlati da un'informazione aggiuntiva che descrivere il target da raggiungere.

Queste informazioni possono essere di vario genere come identificativi delle classi, un insieme di coordinate per algoritmi di detection oppure una maschera di pixel per tecniche di segmentazione. La struttura di questi dati è molto variegata e dipende dall'applicazione che vogliamo far eseguire al sistema.

Nella classificazione [2] il meccanismo di base è quello di fare predizioni sui dati di training e confrontare il risultato con l'etichetta corretta. In caso di errore si corregge il sistema affinchè non sbagli la volta seguente. Possiamo dire che le etichette svolgono il ruolo di supervisore. Il problema di questo metodo è la carenza di dati annotati. La generazione delle etichette nella maggior parte dei casi non avviene in corrispondenza della generazione dei dati ma in un secondo momento e quasi sempre richiede l'intervento di un operatore.

Il processo per etichettare i dati è molto costoso ad esempio Scale.ai, un'azienda che etichetta dati, richiede circa 6 dollari per immagine e considerando che un dataset valido contiene almeno 10k immagini, si raggiungono velocemente cifre importanti [3]. Anche se la maggior parte dei dati non presenta etichetta questo non esclude la possibilità di sfruttare la grande fonte di informazioni disponibile. Il clustering è uno degli algoritmi unsupervised (Figura 1.1) che cerca di raggruppare i dati secondo le caratteristiche che li accomunano[4, 1] in cui l'obiettivo è capire le proprietà del meccanismo che ha generato i dati [5, 1].

1.1 Fondamenti di representation learning

I modelli di machine learning non sono applicati direttamente sui dati ma su una loro rappresentazione. Questa dev'essere utile a mettere in evidenza le caratteristiche di maggior rilievo su cui il modello verrà applicato. Per questo motivo la qualità della rappresentazione è di fondamentale importanza poichè strettamente legata alle performance del metodo.

Il feature engineering si occupa di manipolare opportunamente i dati grezzi al fine di estrapolare la miglior rappresentazione adatta per un task ed è un processo che sfrutta una conoscenza a priori dei dati.

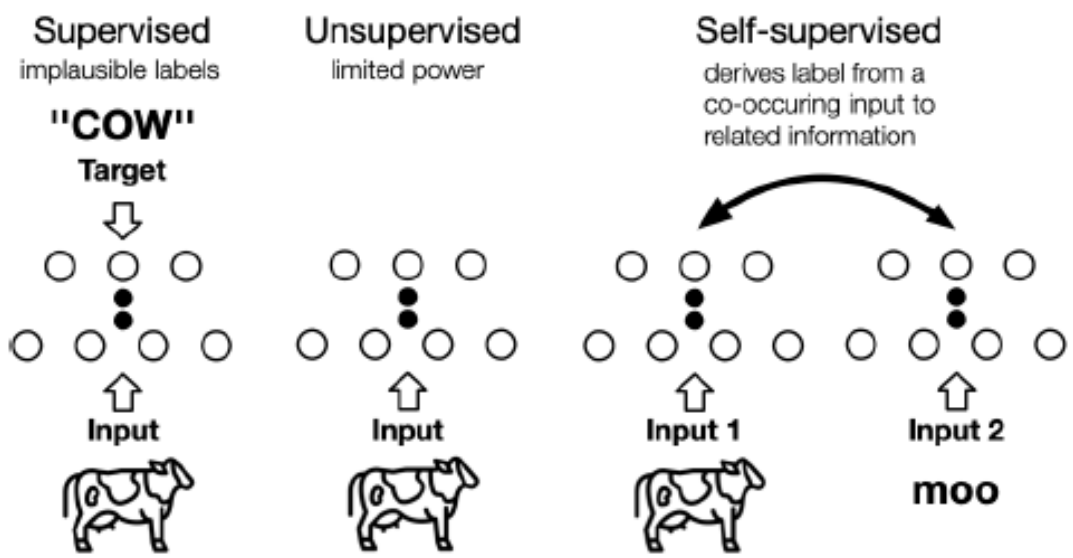


Figura 1.1: Modelli deep learning [3]

Un esempio può essere un classificatore di email spam, questo non lavora sul testo della mail ma su una rappresentazione bag of words [6] in cui le parole sono state rimpiazzate da un numero che descrive la loro frequenza nel testo. In questo modo si cerca di descrivere l'importanza delle parole. Questa trasformazione dell'input può essere adatta per questa specifica applicazione ma non è sufficiente in altri contesti in cui si vorrebbe che la rappresentazione del testo considerasse anche le relazioni tra le parole o l'ordinamento. Sarebbe opportuno includere questo tipo di processamento all'interno dell'algoritmo di apprendimento e renderlo capace di estrarre le caratteristiche migliori indipendentemente dal task da eseguire. Questo è lo scopo del representation learning.

Proprietà e prior knowledge Uno degli aspetti fondamentali del representation learning è quello di sfruttare la prior knowledge inclusa nei dati,proprietà che non essendo specifiche per un determinato task possono essere usate per diversi scopi. Alcuni esempi di prior knowledge sono :

- Regolarità : Si assume che le proprietà di due dati molto simili presentino una

coerenza nell'output e non una differenza brusca. Ciò è una conseguenza del principio della generalizzazione locale della funzione da apprendere.

- Organizzazione gerarchica delle caratteristiche : I concetti che descrivono il mondo che ci circonda possono essere espressi secondo una gerarchia in cui il livello di astrazione aumenta man mano che si sale. Questo punto è ben definito nelle architetture convolutive in cui il livello di astrazione delle features estratte aumenta progressivamente con la profondità della rete. I concetti astratti non cambiano con piccole variazioni dell'input quindi catturano una semantica. Tutto ciò si realizza tramite l'uso dei livelli di pooling che riassumono le informazioni nei punti in cui sono applicati e permettono ai filtri convolutivi di considerare porzioni dell'input sempre maggiori e quindi di aumentare l'astrazione.

- Pluralità delle caratteristiche esplicative : Un fattore di variazione è una proprietà dei dati che viene riconosciuta in modo coerente tra un insieme di dati come ad esempio il colore di un oggetto in un'immagine. Sulla base dell'assunzione di espressività ed invarianza della rappresentazione vorremmo una netta separazione tra di fattori di variazione perchè sono in grado di spiegare singole caratteristiche del dato in input.

  Se consideriamo una sequenza di dati in input, sono pochi i fattori che tendono a cambiare in modo indipendente tra loro nella distribuzione. Al fine di separare quanto possibile questi fattori è doveroso usare enormi quantità di dati ed essere capaci di estrapolare rappresentazioni solide rispetto alle complesse distribuzioni che generano i dati. Sfortunatamente i fattori di variazione utili, non sono immediati da determinare a priori e dipendono dal task da svolgere. Inoltre ci sono vari scenari in cui i fattori possono essere utilizzati da diversi task che ne richiedono un sottoinsieme, questo incoraggia maggiormente una buona capacità di separazione tra i fattori di variazione.

- Coerenza spaziale e temporale : Dati che presentano una continuità temporale o spaziale dovrebbero presentare un cambbio lieve nei fattori di interesse. Questo prior forza una rappresentazione lievemente diversa quando la variazione temporale o spaziale è piccola.

- Caratteristiche condivise tra task : In scenari come transfer learning, domain adaptation o multi task learning il task da eseguire dipende ad un sotto insieme delle caratteristiche estratte ed una rappresentazione comune ai metodi presenta una migliore capacità di generalizzazione.

- Dipendenze lineari tra caratteristiche : Se la rappresentazioen è di ottima qualità allora è possibile separare i fattori di variazione tramite un modello lineare.

Tutti queste informazioni a priori di alto livello servono al modello come strumento aggiuntivo per individuare caratteristiche invarianti nei dati,ortogonali tra loro, al fine di migliorare il task in analisi [7, 8].

Apprendimento auto supervisionato Le nuove tecniche che presentano ottime performance nel representation learning sono quelle del Self-supervised learning (Figura 1.1). Spesso viene confuso con il metodo unsupervised ma c'è una differenza importante ovvero il modello unsupervised cerca di scoprire le caratteristice dei dati mentre il self supervised si focalizza nel recuperare delle informazioni e imparare le occorrenze delle caratteristiche che sono presenti insieme nei dati. Di base è un apprendimento senza etichetta ma viene trasformato in uno supervisionato dove l'etichetta è stata generata a partire dal dato attraverso un processo semi automatico. Successivamente bisogna predire una parte del dato a partire da altre sue parti. La parte mancante potrebbe essere incompleta, trasformata oppure appositamente corrotta. Lo scopo di questo apprendimento è quello di imparare a recuperare le parti mancanti del dato di partenza. Risolvere questo compito ausiliario comporta essere in grado di apprendere pattern nascosti nei dati. Questo

metodo sfrutta le informazioni già presenti nei dati nello stesso modo in cui si usa l'etichetta per un metodo supervisionato.

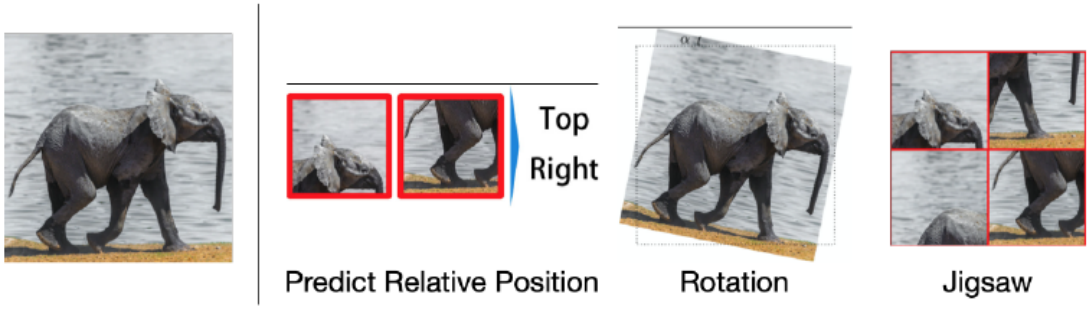


Figura 1.2: Esempi di task ausiliario [3]

Ci sono tantissimi compiti ausiliari, alcuni cercano di trasformare un'immagine in partenza in scala di grigi in una colorata [9, 10, 11] altri cercano di ricostruire un puzzle [12] oppure predire la rotazione di un'immagine [13] cercando di far apprendere alla rete alcune semantiche dei dati ad esempio la direzione in cui si sviluppa un albero.

Un altro compito consiste nel dividere un'immagine in 4 parti e contare le primitive visuali di ognuna, contare e confrontare il risultato con le primitive trovate nell intera immagine [14]. Un altro task è quello di estrarre due patch random da un immagine e cercare di predire la posizione del primo rispetto al secondo, in questo modo il modello deve essere in grado di riconoscere oggetti e le loro parti [15].

Anche nei video questa tecnica è molto utile ad esempio nel capire la corretta sequenza dei frame [16] per cercare di estrapolare informazioni sui pattern di movimento, aumentando di molto le performance nei modelli usati per action recognition.

Tutti questi task hanno in comune il compito di fornire un segnale supervisionato molto forte che sia in grado di far apprendere le caratteristiche semantiche dei dati. Il sistema ottenuto con questo tipo di allenamento ha suscitato molto interesse perchè presenta delle performance molto vicine al caso supervisionato.

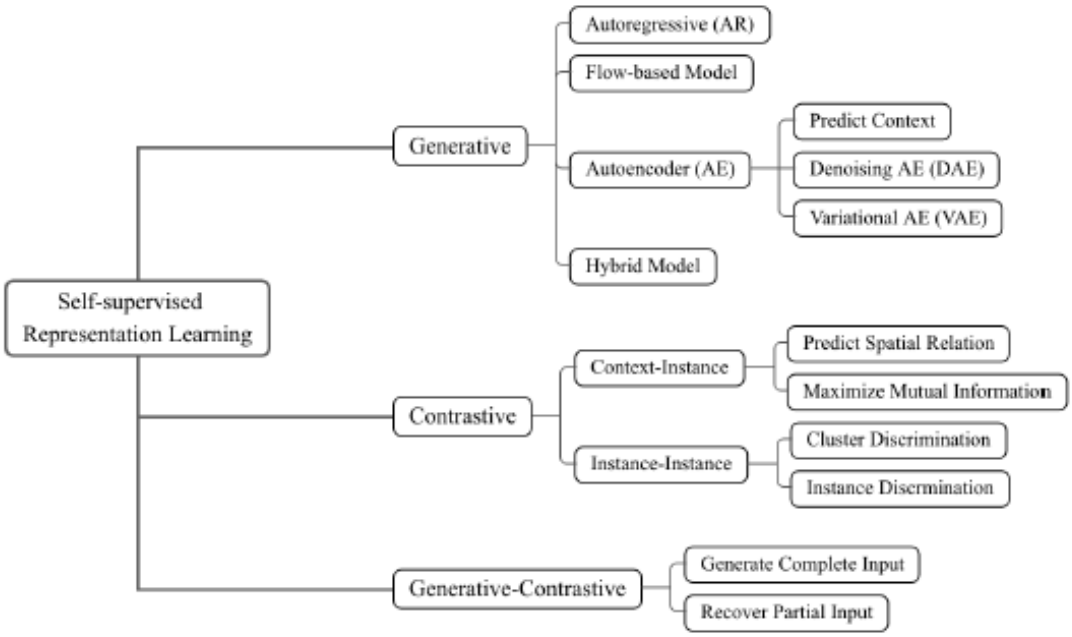


Figura 1.3: Categorie algoritmi self-supervised [3]

Categorie metodi auto supervisionati Esistono tre categorie di self-supervised learning (Figura 1.3) ovvero Generative, Contrastive e Generative-Contrastive (Adversarial)[3, 17] che si differenziano per la funzione obiettivo e l'architettura.

La categoria Generative prevede un encoder per estrarre le features semantiche da cui ricostruire il dato di partenza tramite un decoder e misurare l'errore di ricostruzione. Quella Contrastive allena un encoder per estrarre le features da due versioni diverse dello stesso input e misura le similarità, si basa sul fatto che la semantica non cambia. La categoria Adversarial , tipica delle GAN [18] , cerca di produrre un'immagine che sia realistica, sempre tramite un encoder-decoder e successivamente viene data in input , insieme ad un'immagine reale, ad un discriminatore che deve distinguerle correttamente.L'obiettivo è quello di riuscire ad ingannare il discriminatore.

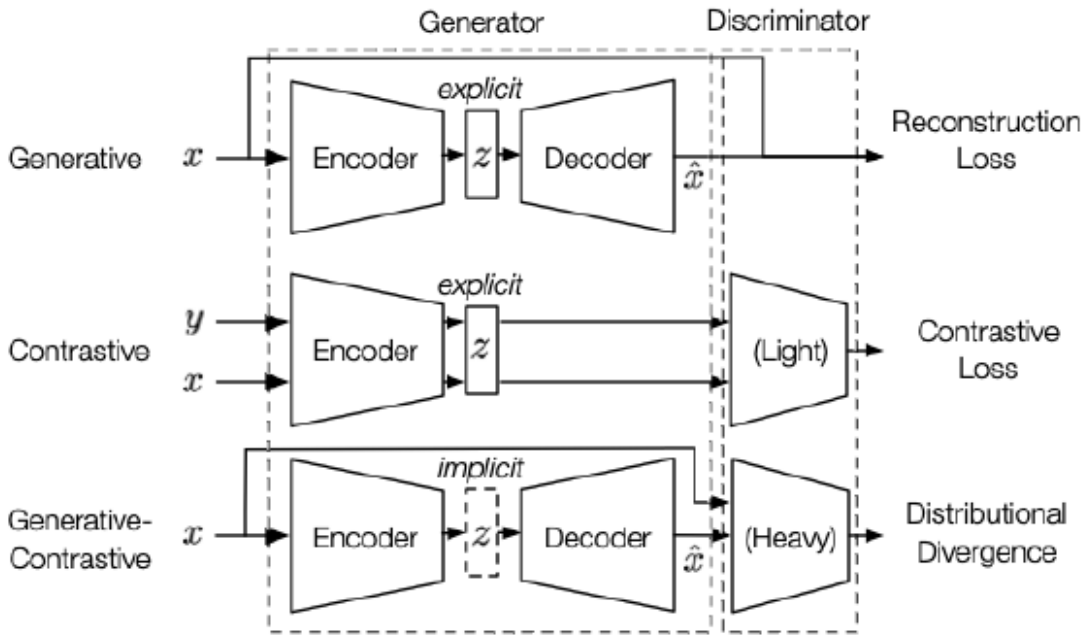


Figura 1.4: Architetture self-supervised [3]

1.1.1 Generative

I modelli generativi (Figura 1.4) cercano di determinare in modo esplicito la distribuzione che ha generato i dati e possono essere suddivisi in auto-encoders, modelli auto-regressivi, flow-based o modelli ibridi. Nei modelli AR si cerca di predire un dato a partire dai dati precedentemente osservati. La probabilità congiunta è data dalla somma delle precedenti N probabilità condizionali dove N determina l'ordine del modello e la probabilità di ogni variabile è indipendente dalle altre. Questi modelli sono stati applicati in molti campi come in NLP [19] dove grazie alla formulazione auto regressiva XLNet massimizza il likelihood imparando contesti bidirezionali e superando in alcuni casi le performance di BERT [20]. Anche nel contesto immagini (PixelCNN, PixelRNN [21, 22]) i metodi AR cercano di modellare l'immagine pixel per pixel dove i pixel a destra sono generati su condizioni applicate sui pixel in alto a sinistra. Il vantaggio di questi metodi è che riescono a rappresentare dipendenze contestuali ma solo lungo una direzione.

Poichè stimare una distribuzione in modo esplicito è molto difficile, i metodi Flow Label cercano di stimare la densità dei dati in input in maniera progressiva attraverso la disposizione di una serie di funzioni che cercano di descrivere le caratteristiche dei dati. In generale data una certa distribuzione latente dell'input, si cerca di stimare una funzione invertibile e differenziabile che sia in grado di descriverla.

I modelli auto encoding sono i piu performanti ed hanno come scopo quello di stimare la distribuzione che ha generato i dati tramite la ricostruzione di una parte del dato in input a partire da una sua versione corrotta. Sono implementati tramite una rete neurale feed-forward composta da un encoder ed un decoder e sono allenati in modo da minimizzare l'errore di ricostruzione (L2 loss). Se l'errore è minimo allora le features latenti caratterizzano l'input in maniera corretta.

Il vantaggio di questi metodi è che dopo aver allenato il modello, possiamo scartare il decodificatore ed usare il codificatore al fine di estrarre i fattori di varianza dai dati e darli in input al task finale, senza fare assunzioni a priori su quale possa essere. Una funzione obiettivo generativa porta uno svantaggio implicito ovvero la rappresentazione estratta è modellata su informazioni di basso livello e questo entra in conflitto con la maggior parte dei task di machine learning che necessitano rappresentazioni distribuite, invarianti e di alto livello [3, 17].

1.1.2 Adversarial

I metodi Adversarial fondono i metodi contrastive e generative, sfruttando una funzione obiettivo discriminativa in cui si cerca di avvicinare la distribuzione imparata dal modello con quella target, che ha generato i dati. Ciò viene fatto in modo implicito senza focalizzarsi sulla ricostruzione del dato in input e deriva dall'idea di superare i problemi legati ai metodi generativi.

Come mostrato in Figura 1.4, l'architettura presenta una sequenza di encoder e decoder, come nel caso generative, ed infine un discriminatore. Il ramo del generatore serve a imporre una rappresentazione che permetta la ricostruzione dell'input ma questo rende l'allenamento molto instabile. Si possono dividere in

due categorie chiamate Input completo oppure parziale e si distinguono nel modo in cui utilizzano l'input.

I metodi ad input completo usano il dato integro e cercano di estrapolare quanta piu informazione possibile. Tra i primi spicca la rete GAN [18] in cui l'allenamento viene descritto come se fosse un gioco tra due players. Un player (generatore) ha il compito di generare dati finti mentre l'altro (discriminatore) cerca di distinguerli dai dati reali, da qui il nome Adversarial.

Nel momento in cui il discriminatore non riesce a distinguere se un dato appartiene alla distribuzione reale o quella imparata allora l'encoder ha imparato in maniera implicita la distribuzione reale dei dati. I modelli simili alla GAN [23, 24] dimostrano performance nettamente superiori a quelle dei modelli Generative quindi è immediato pensare che il representation learning ne tragga dei vantaggi ma non è cosi a causa dell'apprendimento implicito di queste reti.

I metodi con input parziale applicano il metodo Adversarial in modo da aiutare il task di representation learning. Invece di ricostruire l'input nella sua interezza,forniscono al modello solo una parte e si aspettano di recuperare quella mancante. Questo metodo è simile al contrastive ma viene condotto in maniera adversarial.

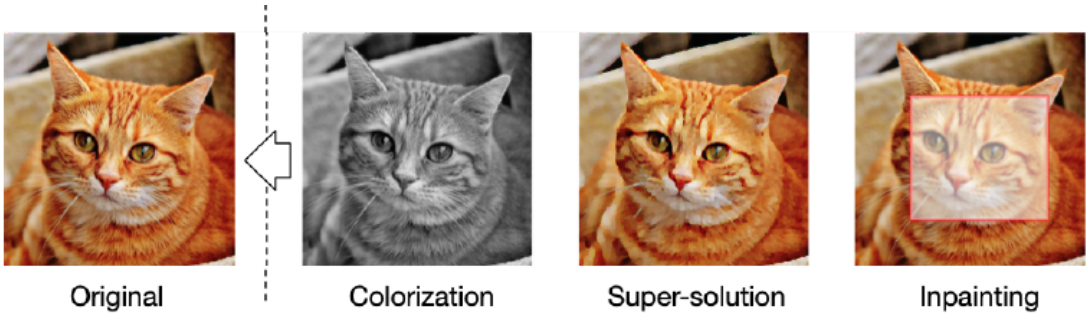


Figura 1.5: Input parziali metodi Adversarial [3]

In Colorization [25] il task da risolvere è quello di predire due dei tre canali di un'immagine, che parte da una scala di grigi. Inpainting [26] richiede al modello di risolvere un task in cui deve predire una parte arbitraria dell'immagine, successivamente il discriminatore la distingue da quella reale.

Super-solution [27] segue la stessa idea dei casi precedenti ma parte da un input sfocato e cerca di aumentarne la risoluzione. Nonostante questi metodi migliorino la rappresentazione estratta, il contrastive learning è di molto superiore. Inoltre i metodi adversarial tendono ad essere instabili durante il training ed hanno applicazioni limitate in ambito NLP.

1.1.3 Contrastive

I modelli statistici sono divisi in generativi e discriminativi. Data la distribuzione congiunta P(X,Y) dell input X e del target Y , il modello generativo calcola P(X|Y) ovvero cerca di modellare la distribuzione dei dati tramite le informazioni note a priori tramite Bayes mentre il modello discriminativo calcola P(Y|X) basandosi solamente sui dati di allenamento.

Anche se inizialmente i modelli generativi erano considerati l'unica soluzione per il representation learning, il contrastive learning ha dimostrato il contrario tramite modelli discriminativi.

L'idea di base è quella di imparare per mezzo del confronto [28, 3]. In particolare si cerca di avvicinare le rappresentazioni estratte da sample positivi mentre si allontanano le rappresentazioni dei sample negativi. Il ruolo dell'etichetta nel contrastive viene sostituito dalle distribuzioni di sample negativi e positivi.

Questo obiettivo è descritto nella funzione di costo 1.1.3 InfoNCE [29] che è un caso più generale della Noise Contrastive Estimation [30] in cui sono considerati molti campioni negativi e mostra il framework generale per il contrastive. La funzione f è un encoder che estrare le caratteristiche dai sample aumentati e può cambiare in base al contesto applicativo. Se la rete è in grado di svolgere questo task allora dimostra di saper estrarre delle caratteristiche che sono indipendenti dalla trasformazione applicata e quindi ottenere una rappresentazione che racchiude una semantica.

$$l = \mathbb{E}_{x,x^+,x^-}[-\log(\frac{e^{f(x)^T f(x^+)}}{e^{f(x)^T f(x^+)} + \sum_{k=1}^{K} e^{f(x)^T f(x^k)}})] \tag{1.1}$$

Esistono due macro categorie di metodi contrastive e si distinguono nella scelta dell'elemento con cui confrontare un dato in analisi.Si chiamano contrasto istanza-istanza e contrasto istanza-contesto.

Metodi contesto-istanza I metodi istanza-contesto si basano sull'ipotesi che una rappresentazione estratta abbia una relazione con il contesto in cui è stata trovata, quindi si cerca di modellare una possibile relazione tra le caratteristiche locali e quelle globali. Si possono sfruttare due strategie ovvero PRP e MI. Predictive Relative Position (PRP) si focalizza sulla predizione della posizione relativa tra le caratteristiche di un sample con l'assunzione di sfruttare le informazioni spaziali o sequenziali presenti nei dati. Il contesto globale resta un requisito implicito per la predizione di queste relazioni.

Uno dei modelli più noti per questo tipo di task è il Jigsaw puzzle [12] in cui bisogna imparare l'ordinamento corretto di una sequenza di dati ordinata in modo casuale oppure [31, 32] in cui vengono estratte delle coppie di parti del dato e si predice la loro posizione relativa usando come segnale supervisore l'informazione spaziale dell'immagine.

Mutual Information (MI) ha come scopo quello di massimizzare l'associazione tra le rappresentazioni di due dati. Questo metodo presenta una funzione di costo che richiede un costo computazionale elevato quindi si massimizza il limite inferiore con una funzione obiettivo NCE.

Deep InfoMax [33] ne è il primo esempio ed enfatizza l'importanza della struttura nei dati. Incorporare le informazioni spaziali nella funzione di costo ha portato un miglioramento nella rappresentazione estratta ed anche il task da eseguire ne ha tratto vantaggi. Da questa idea si è ispirato il Contrastive Predictive Coding [29].

Dalla Figura 1.6 possiamo notare che dalle features del dato viene estratto un contesto tramite una funzione (Read Out) che ne riassume il contenuto. Successivamente si cerca di avvicinare il contesto con le features che lo hanno generato

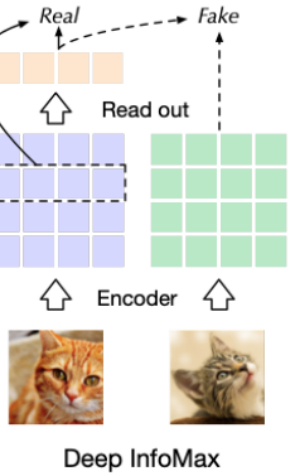


Figura 1.6: Deep InfoMax

(campioni positivi) e si allontana dalle features di altri dati, considerati i campioni negativi.

Metodi istanza-istanza Come descritto in [34] le performance dei metodi sopra citati non dipendono da MI ma da come vengono generati i sample negativi e dalla rappresentazione estraatta dall'encoder. I metodi tra instanze studiano le dirette relazioni tra i dati andando a relazionare le caratteristiche estratte dalle coppie. Empiricamente si dimostrano superiori ai metodi contesto-istanza perchè si focalizzano sulle caratteristiche locali. Se coem esempio consideriamo la classificazione di un input con l'etichetta 'gatto', è immediato notare che al fine di ottimizzare il task bisogna sfruttare solo le caratteristiche del gatto mentre quelle del contesto sono irrilevanti.

Inizialmente l'idea del contrasto tra istanze era quella di generare etichette raggruppando le features estratte tramite un algoritmo di clustering [35] raggiungendo performance comparabili con AlexNet. Si basa sull'ipotesi che le caratteristiche estratte da dati che appartengono alla stessa categoria devono essere vicine anche nello spazio latente. Per fare ciò si applica un algoritmo di clustering sulle rappresentazioni e a seguire un discriminatore deve distinguere se due rappresentazioni

fanno parte dello stesso gruppo o meno. Se fanno parte dello stesso gruppo, il discriminatore non riuscirà a distinguerle quindi le features sono simili. Nonostante il grande successo, questo processo iterativo a due fasi richiede molto tempo per l'allenamento e presenta risultati inferiori rispetto alla tecnica che oggi si utilizza maggiormente e che ho utilizzato nel mio lavoro ovvero la Discriminazione di istanza.

La discriminazione di istanza propone di eseguire il confronto tra diverse versioni del dato in analisi,considerati campioni positivi, ed altri dati considerati campioni negativi. Cambia il modo di confrontare ma l'obiettivo è sempre quello avvicinare le rappresentazioni dei campioni positivi nello spazio latente e allontanare da quelle dei campioni negativi.

In base a quanti campioni negativi vengono selezionati possiamo differenziare le diverse implementazioni CMC [36], InstDisc [37], MoCo [38], SimCLR [39]. MoCo a differenza di tutti gli altri metodi abbandona l'allenamento end to end e per aumentare il numero di campioni negativi utilizza una coda di 65 mila campioni. Un problema di questa rete è la generazione dei campioni positivi non richiede l'applicazione di una trasformazione. Questa strategia risulta troppo semplice per la rete ed impone un limite sulle performance.

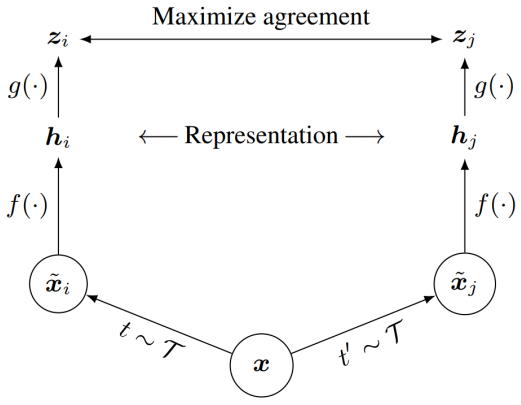


Figura 1.7: Architettura SimCLR

In [39] viene analizzata l'importanza delle strategie per generare i campioni

positivi e negativi e funziona nel seguente modo. Ad insieme di dati viene applicata una trasformazione scelta in modo casuale tra quelle possibili in modo da ottenere un set di dati con cardinalità doppia. In questo modo, fissato un certo dato e la sua versione trasformata, possiamo generare una coppia di campioni positivi mentre tutti gli altri dati sono considerati campioni negativi. La funzione di costo 1.1.3 misura le distanze in entrambi i versi per una certa coppia e questa operazione viene ripetuta per tutte le coppie.

$$\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [l_{2i-1,2i} + l_{2i,2i-1}] \tag{1.2}$$

La funzione l è la InfoNCE presentata in 1.1.3 con l' applicazione di una misura di cosine similarity tra gli output di un livello di proiezione $g(f())$ come mostrato in Figura 1.7.

$$l = -\log \frac{exp(\frac{sim(z_i,z_k)}{\tau})}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} exp(\frac{sim(z_i,z_k)}{\tau})} \tag{1.3}$$

I metodi instance discrimination come [40] dimostrano che la rappresentazione delle caratteristiche rispetta i vincoli per ottenere una buona rappresentazione ovvero l'astrazione e l'indipendenza delle caratteristiche.

Tutte le architetture contrastive non usano il decodificatore quindi si chiamano strutture light-weight avendo un numero di parametri inferiore. Grazie a questo, non è necessario eseguire un fine tuning sul resto della rete al fine di adattare la rappresentazione al task.

Oltre a tutti i vantaggi presentati, questi metodi non si adattano bene nei problemi NLP. Il modo in cui bisogna scegliere i campioni negativi e le trasformazioni da poter applicare hanno un impatto significativo sulle performance ma ancora non sono state esplorate in maniera esaustiva.

Capitolo 2

Classificazione serie temporali e rilevamento anomalie

Ciò che contraddistingue la serie temporale da altre tipologie di dati è la dinamicità nel tempo. Questa rende possibile studiare l'evoluzione delle entità in analisi. L'analisi delle serie temporali mira ad estrapolare le caratteristiche che la descrivono per poter predire un comportamento futuro, rilevare eventuali anomali oppure eseguire un task di classificazione [41, 42, 43].
Inizialmente le serie temporali sono state utilizzate per studi di econometria [42] ma con l'aumento della disponibilità di dati, sono state applicate in moltissimi contesti applicativi come ad esempio il monitoraggio dello stato di salute [44], oppure il sistema di monitoraggio della respirazione durante il sonno dove si stima la frequenza respiratoria a partire dalla frequenza del RSA derivato dalla time series del battito cardiaco [45], o anche il rilevamento anticipato di malattie cardiovascolari che può prevenire una morte prematura causata dal riscontro di un' anomalia nella frequenza cardiaca [46].
Altre applicazioni riguardano la predizione di un guasto al motore,come descritto in [47] o l'analisi dei mercati finanziari in cui [48] è una possibile soluzione per la gestione di serie temporali non lineari e con rumore. Nel capitolo seguente ci

focalizzeremo sull' analisi di serie temporali per il rilevamento di anomalie e la classificazione per quanto riguarda serie temporali univariate e multivariate.

2.1 Definizione di serie temporali

Una serie temporale è il risultato della collezione di una sequenza di misurazioni, eseguite su una o più variabili, ad intervalli di tempo regolari per tutta la finestra di osservazione. Le variabili monitorate possono rappresentare diverse grandezze, in base al dominio applicativo. Un esempio potrebbe essere la lettura della temperatura ogni ora per ogni giorno oppure il numero di click su un sito web su base settimanale.

Per aggiungere un livello di formalità, riporto la definizione di serie temporale secondo Malhotra in [49].

Definizione TS. *Una serie temporale è un vettore* $X = \{x^{(1)}, x^{(2)}, ..., x^{(n)}\}$ *dove ogni elemento* $x^{(t)} \in R^m$ *attinente ad* X *é del tipo* $\{x_1^{(t)}, x_2^{(t)}, ..., x_m^{(t)}\}$. *Nel caso delle serie univariate la variabile osservata è una sola quindi* $m = 1$. *In generale* m *varia in base a quante grandezze misuriamo.*

Le serie possono essere scomposte in un numero finito di componenti, non direttamente osservabili, che le caratterizzano e sono affetti da diversi eventi casuali. Sono state individuate quattro componenti [50, 41, 51] :

- Il trend : Consiste nello sviluppo a lungo termine. Puo essere crescente, decrescente o stabile e può cambiare pendenza nel tempo.
- La ciclicità : Sono le variazioni che si ripetono nel lungo periodo. Un periodo completo si chiama anche Business Cycle. Rispetto alla stagionalità il periodo è molto piu lungo.
- La stagionalità : Pattern che si ripetono nel breve periodo.
- Il disturbo : Descrivere tutte le variazioni che non appartengono alle componenti precedenti e che hanno una natura casuale. Ci si riferisce anche con

il termine Residuo. Sono variazioni imprevedibili, non controllabili e spesso dovute ad errori.

Sotto l'ipotesi di indipendenza tra le componenti, diciamo che la serie che osserviamo è una somma di questi fattori. Per modellare al meglio il processo che genera questi dati temporali, dobbiamo fare delle assunzioni riguardo alla natura delle componenti [43].

2.2 Classificazione di serie temporali

Si parla di classificazione di serie temporali in tutti quei casi in cui il task di classificazione viene eseguito su valori reali, ordinati secondo una dipendenza dal tempo [52, 53] .

Dataset TS. *Un dataset di serie temporali etichettato* $D = \{(X_1, y_1), .., (X_n, y_n)\}$ *è un set di coppie dato-etichetta in cui il dato è una serie temporale univariata o multivariata mentre l'etichetta è un valore discreto che indica la classe di appartenenza del dato associato.*

Classificazione TS. *Data la definizione 2.1 e 2.2, un classificatore allenato su un certo dataset D ha lo scopo di mappare le serie temporali in input sulla corretta distribuzione di probabilità delle classi in output. Nel caso univariato la serie temporale è un vettore di letture mentre nel caso multivariato con m variabili, la serie temporale è una lista di m serie temporali univariate [54, 53].*

2.2.1 Basate su distanza o features

I metodi di classificazione di serie temporali possono essere suddivisi in due gruppi principali ovvero features based e distance based [55].

Basate sulla distanza Nei metodi distance based la classificazione avviene tramite l'etichetta del dato di training meno simile a quello da classificare, dove la similarità è stata definita tramite una misura di distanza. In particolare si calcolano

le distanze tra i dati e successivamente un algoritmo di classificazione basato sul concetto di distanza, come KNN o SVM, produce un'etichetta. I vari algoritmi si differenziano nel mondo in cui viene calcolata la distanza [56].

La maggior parte dei lavori di ricerca entafizza l'importanza della forma della serie e al fine di migliorare la classificazione, cerca di scoprire la misura di distanza più adeguata nel trattare un dato che presenta un ordinamento naturale,una lunghezza variabile ed una componente di rumore.

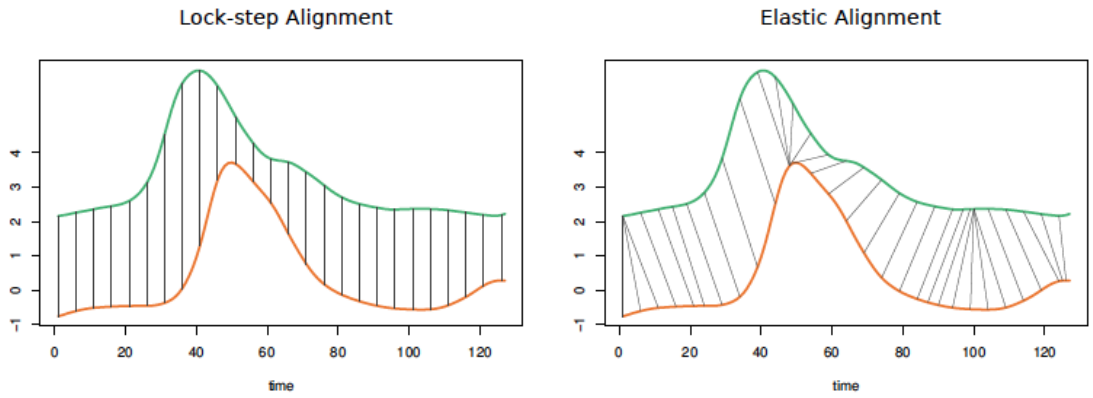


Figura 2.1: Categorie misure di distanza [56]

Le distanze sono divise in misure elastiche o misure a passo fisso Figura 2.1. Il primo caso è quello più naturale da pensare e le distanza si calcola confrontando gli elementi che si trovano nella stessa posizione temporale della serie, ad esempio distanza euclidea. Nel secondo caso le serie vengono allineate tramite un mapping non lineare e si calcola la distanza tra un punto della prima serie e tanti punti della seconda serie. Il caso piu noto è quello del Dynamic Time Warping [57]. Empiricamente si è riscontrato che le performance delle misure sono fortemente dipendenti dal contesto applicativo, non esiste una misura universale per tutti i casi. Nonstante ciò, la DTW accoppiata con 1-NN risulta la migliore, è stata lo standard di riferimento per molto tempo e ancora oggi è considerata un'ottima baseline per confrontare i nuovi metodi [58, 54].

Si potrebbe ipotizzare che le performance di classificazione possano migliorare

sostituendo il K-NN con un classificatore più complesso, in realtà si è scoperto che la performance ottenuta dipende principalmente dalla qualità della misura di distanza adottata I classificatori complessi che usano tali misure, che sono in grado di gestire al meglio la natura temporale del dato, ottengono risultati migliori rispetto ai metodi sopra citati [56]. I nuovi approcci si dividono in due categorie e sfruttano le misure di distanza su classificatori più complessi. La prima categoria basa la classificazione su una nuova rappresentazione della serie, cioè un vettore di caratteristiche, ottenuto tramite l'applicazione di un'opportuna misura di distanza [59]. La seconda categoria sfrutta una misura di similarità il cui prodotto sarà utilizzato in un metodo Kernel [60]. Entrambe le categorie ottengono risultati competitivi con DTW-KNN. [56].

Basate su features Nei metodi feature based la classificazione avviene tramite una rete neurale o un albero decisionale, sulla base di un vettore di rappresentazione ottenuta tramite un metodo di trasformazione che rimuove la dipendenza temporale nel dato. Le varie metodologie per estrapolare la rappresentazione caratterizzano gli algoritmi e servono ad estrapolare le proprietà che meglio descrivono della serie. Tra i metodi più noti fanno parte la trasformata discreta di Fourier, la trasformata discreta Wavelet oppure la decomposizione in autovalori.
Un altro esempio può essere quello di [61] in cui la media, kurtosis, deviazione standard, l'asimmetria vengono utilizzate per calcolare il vettore di rappresentazione per classificare schemi di controllo. Il caso di [62] aggiunge alle misure appena citate la periodicità, la stagionalità, la non linearità ed altre ancora al fine di rappresentare meglio la serie temporale. Esiste un'implementazione anche per il caso multivariato [63]. Tutti questi metodi hanno un grande limite, sono fortemente dipendenti dalla selezione manuale delle caratteristiche in base al tipo di dataset. [64] propone un approccio alternativo in cui l'estrazione delle caratteristiche avviene in maniera estesa tramite migliaia di algoritmi di analisi. Le caratteristiche ottenute vengono opportunamente filtrate per ottenere un vettore di rappresentazione ridotto e su questo viene applicato un algorimo di classificazione Figura 2.2. Tutto

ciò serve a rendere automatico il processo di selezione delle caratteristiche ma resta un approccio che richiede l'intervento umano per determinare il vettore di rappresentazione.

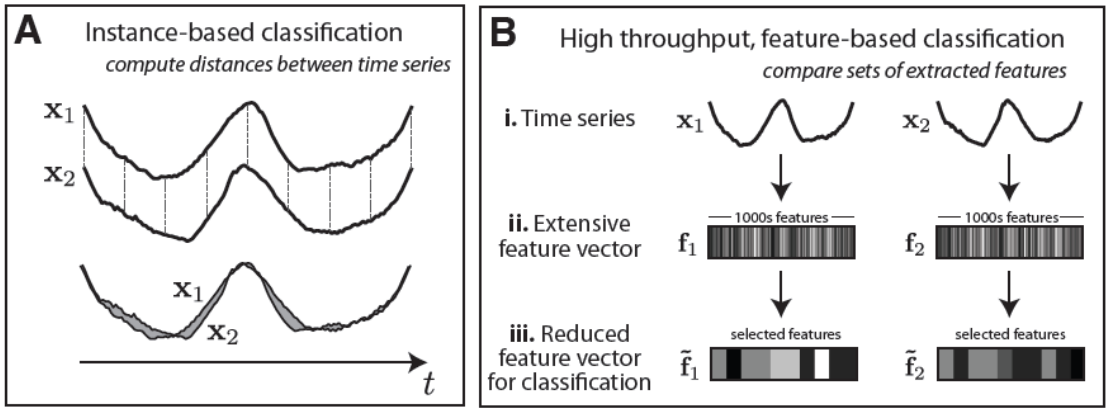


Figura 2.2: Confronto distance based e features based [64]

2.2.2 Reti neurali

Il vantaggio dei metodi sopra descritti è che possono fornire un livello di interpretazione dato che la rappresetanzione della serie è esplicita. Lo svantaggio è che definire le caratteristiche rappresentative della serie introduce un forte bias nel modello [55]. Dato il successo delle reti neurali in Computer vision, è stato proposto un grande numero di lavori riguardo ai problemi di processamento di testi (NLP). Nello stesso modo, data la natura ordinata dei dati NLP, è stata posta molta attenzione nei problemi di classificazione di serie temporali tramite reti neurali. Come riportato in [54] le reti neurali ottengono risutlati superiori ai metodi citati in precedenza.

La Figura 2.3 mostra il framework deep learning generale per la classificazione di serie temporali. Queste architetture sono state costruire per estrarre una rappresentazione gerarchica del dato. Un'architettura deep è una composizione di livelli non lineari, ognuno dei quali lavora su una rappresentazione diversa del dato. Ogni livello è caratterizzato da un numero variabile di parametri che vengono

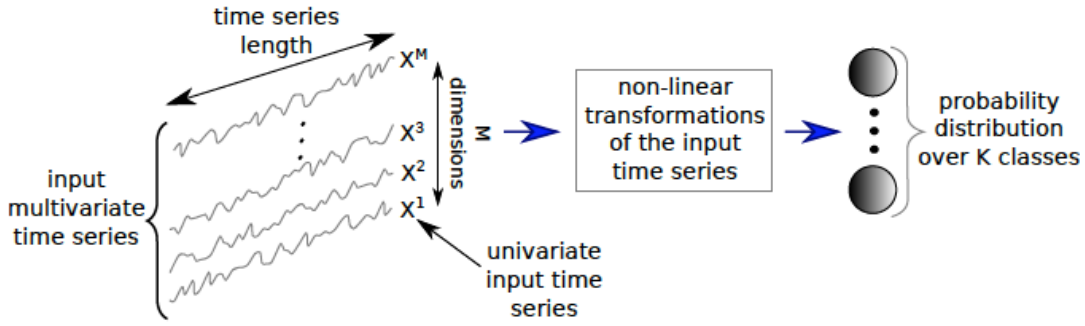


Figura 2.3: Classificazione serie temporali [54]

alterati al fine di ottimizzare la classificazione durante il periodo dell'allenamento. Per classificare un dato bisogna processarlo tramite i livelli e come output si ottiene un vettore con le stime delle probabilità di appartenenza ad ogni classe. La probabilità con valore maggiore sarà tradotta nell'etichetta da assegnare. In questo lavoro ci focalizziamo sulle reti feed forward con allenamento ent to end, in modo da integrare la generazione delle caratteristiche nella struttura della rete. Seguono due esempi di reti neurali che sono utilizzate nella soluzione del lavoro di tesi.

Reti fortemente connesse La rete FC, anche nota come Multi layer percerptron, è la struttura basilare delle reti neurali e ha questo nome poichè ogni neurone di un livello è collegato ad un altro neurone del livello precedente. Queste connessioni son rapprentate dai parametri nei livelli, la cui applicazione avviene tramite la seguente equazione

$$A_l = f(w_l * X + b) \tag{2.1}$$

dove w_l sono i pesi del livello l, b un termine di bias. Un problema di quest'architettura è che ogni campione della serie viene trattato in modo indipendente poichè non applicano nessuna invarianza spaziale. Ogni campione è associato ad un peso quindi si perde l'informazione temporale [54].

Reti convolutive L'esperienza accumulata in ambito visual può essere sfruttata anche nelle serie temporali e [54] ha proposto il primo studio su larga scala degli

approcci deep learning su questo tipo di dato. Una rete convolutiva consiste di vari livelli convolutivi. Ogni livello applica una serie di filtri sulla serie temporale che scorrono per tutta la serie, generando una trasformazione non lineare del dato. Nel caso di serie univariate il filtro è unidimensionale mentre nel caso multivariato il filto ha una profondità pari al numero di variabili della serie. Una formula generale per applicare la convoluzione è data dalla seguente equazione

$$C_t = f(w * X_{t-l/2:t+l/2} + b) \forall t \in [1, T] \tag{2.2}$$

dove f è una funzione non lineare, tipicamente una ReLu, la serie X ha lunghezza T, w è un filtro di lunghezza l e b un bias. Ogni livello convolutivo applicando piu filtri è come se mappasse in uno spazio a piu dimensioni dove la profondità è pari al numero di filtri applicati. La lunghezza della serie può cambiare e dipende dai parametri del filtro come la larghezza, lo stride, il padding. Si vorrebbe che ogni livello convolutivo estrapolasse diverse rappresentazioni indipendenti che caratterizzano la serie, questo è il motivo per cui si applicano tanti filtri e sono tutti diversi tra loro. In questo modo, ogni campione della serie viene analizzato piu volte, considerando i suoi vicini. Grazie a questa proprietà chiamata condivisione dei pesi la rete CNN sarà in grado di ottenere filtri che sono invarianti lungo la dimensione del tempo.[55] Avendo integrato il processo di estrazione delle features nella rete, i valori di questi filtri saranno ottimali per la classificazione su un certo dataset e saranno dedotti in maniera automatica tramite l'allenamento. La convoluzione dev'essere seguita da un classificatore al fine di imparare filtri che siano in grado di discriminare. Generalmente si antepone al classificatore un livello di pooling locale o globale. Il pooling locale serve a ridurre la dimensione dei dati ed estrapolare una rappresentazione man mano piu astratta mentre quello globale ha come unico scopo quello di agevolare il classificatore nel task riducendo i parametri. Infine l'ultimo livello sulla base della rappresentazione prodotta dai filtri convolutivi fornisce un vettore di probabilità per la classificazione come per il caso FC [54]. Il vantaggio della convoluzione è quello di incorporare dei pattern lungo la direzione del tempo, in contrasto con la logica puntuale delle reti FC [55].

La reti neurale allo stato dell'arte è Ts2Vec e sarà presentata nel Capitolo 3

2.3 Rilevazione anomalie nelle serie temporali

Rilevare un'anomalia è un task critico che serve a prevenire una perdita la cui natura dipende dal dominio di applicazione. Queste tecniche trovano un utilizzo intensivo in molti campi applicativi [65, 66, 67, 68] come rilevamento di frodi per carte di credito, assicurazioni o applicazioni medicali, rilevamento di utenti non autorizzati in sicurezza informatica, rilevamento di guasti per la sicurezza dei sistemi e infine sorveglianza militare per attivita ostili [69, 70, 71, 72]. Tra le molte definizioni di anomalia quella di Grubbs dice che un'osservazione anomala, o outlier, è quella che sembra deviare notevolmente dagli altri membri del campione in cui si verifica [73]. Ancora oggi non è stata scelta una definizione universale [74], possiamo definirle eccezzioni, osservazioni discordanti, sorprese, peculiarità o contaminazioni. In questo lavoro farò riferimento con il termine generico anomalia. Le anomalie possono essere dati non voluti dovuti al rumore di una misurazione o ad un errore oppure dati generati da eventi rari ma che sono di interesse ad esempio una frode su carta prepagata. Entrambi i casi presentano valori diversi dal caso normale ma hanno semantiche completamente diverse.

I metodi di rilevamento anomalie possono essere supervisionati quando per ogni anomalia è presente la corrispondente etichetta, non supervisionati quando le anomalie non hanno un'etichetta associata e semi supervisionati quando sono disponibili le etichette dei dati non anomali mentre mancano quelle delle anomalie.

Tipi di anomalie Date le definizioni di serie temporali univariate e multivariate in 2.1 possiamo distinguere le anomali in tre categorie [75] :

- Anomalie puntuali: Un'anomalia puntuale globale è un campione della serie che si discosta in modo significativo da tutti gli altri campioni della serie. E' un'anomalia puntuale locale quando si discosta in modo significativo soltanto dai suoi punti vicini. Possono essere univariati o multivariati in base a quante

variabili della serie sono presenti. Notare che un'anomalia globale è anche locale ma non vale il contrario. Nella Figura 2.4 possiamo notare un'anomalia locale nella serie univariata come anche nelle variabili 2 e 3 della serie multivariata. Inolte la serie multivariata presenta due anomalie nella variabile1.

- Anomalie di contesto: Le anomalie di contesto riguardano l'andamento anomalo di una sottosequenza della serie. Anche in questo caso valgono le considerazioni fatte precedentemente sulla località e globalità dell'anomalia (Figura 2.5).

- Anomalie collettive: Questa categoria di anomalie riguarda solo il caso multivariato poichè si considera anomala l'intera serie temporale quindi una determinata componente tra le m possibili (Figura 2.6).

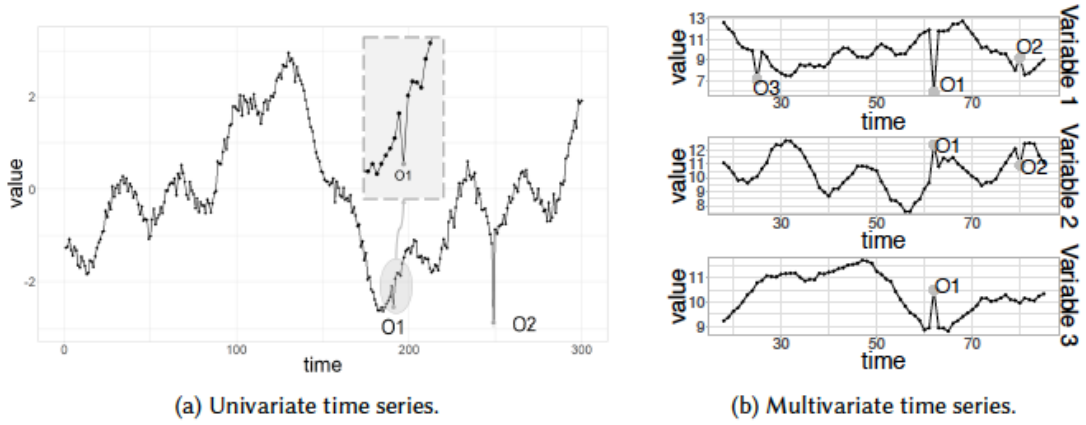


Figura 2.4: Anomalie puntuali [75]

Metodi di rilevamento anomalie I metodi per rilevare le anomalie puntuali si dividono in due categorie che differiscono nel considerare o meno l'ordinamento temporale della serie. In particolare i metodi che non considerano l'ordinamento vengono applicati su un riordinamento casuale di una finestra della serie, ottenendo gli stessi risultati. Il dato è da considerarsi anomalo se la distanza dal valore atteso è maggiore di una certa soglia. Nel caso in cui si considerano solo i punti passati si

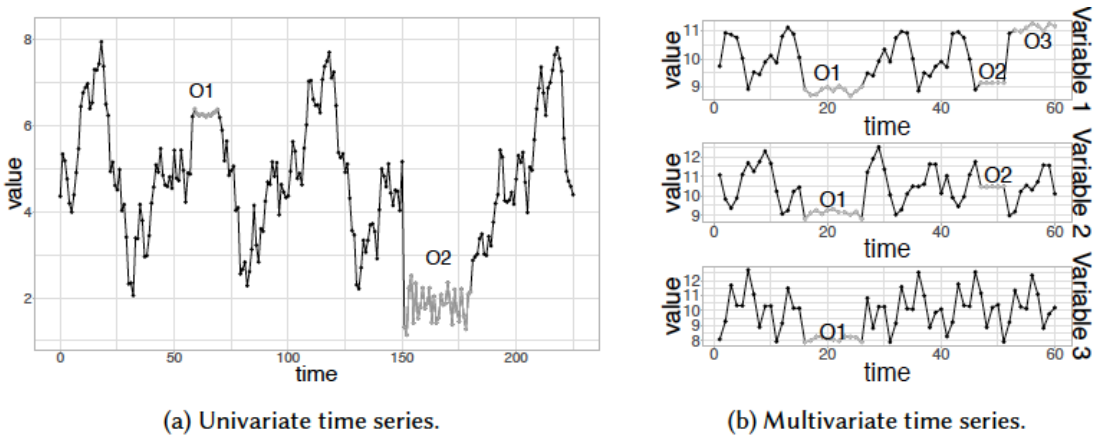


Figura 2.5: Anomalie contestuali [75]

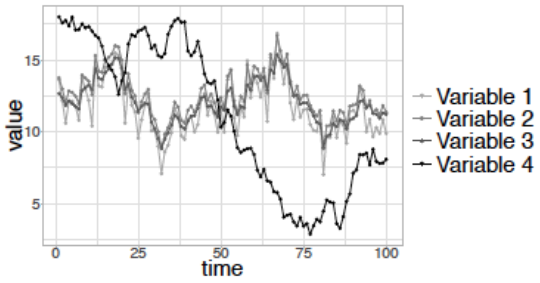


Figura 2.6: Anomalie collettive [75]

parla di predizione mentre se si considerano anche i punti futuri si parla di stima [75].

Per quanto riguarda i metodi per le anomalie di contesto consideriamo le sequenze a lunghezza fissa. Poichè bisogna confrontare sequenze, questi metodi richiedono l'applicazione di una tecnica di discretizzazione al fine di confrontare una rappresentazione dei dati e non i valori grezzi. A differenza dei metodi per anomalie puntuali in questi algoritmi l'ordinamento è di fondamentale importanza. Le squenze da analizzare vengono calcolate sulla base di un filtro che scorre per l'intera serie quindi il numero di sequenze dipende dalla lunghezza del filtro. L'idea più comune consiste nel misurare una finestra con tutte quelle possibili della serie e applicare una misura di distanza per rilevare pattern diversi [76].

Per la terza categoria non esistono dei metodi ben noti perchè sono ancora oggetto di studio ma una possibilità è quella di applicare una tecnica di riduzione della dimensionalità come PCA per ottenere un set di variabili non correlate. Le anomalie sono rivelate quando la devizione della serie è lontana dalla regione a densità massima nello spazio PCA definito dalle prime due componenti [75].

2.4 Caratterizzazione delle serie temporali

In questa sezione riporto tutte le statistiche utilizzate per esaminare le serie temporali, sia per il caso univariato che multivariato.

Sbilanciamento per etichetta (IRLbl) Data una classe λ calcola il rapporto tra gli elementi della classe maggioritaria e gli elementi della classe λ. Vale 1 nel caso in cui λ sia la classe maggioritaria e valori maggiori per le altre classi [77].

$$IRLbl(\lambda) = \frac{\max_{\lambda' \in L}(\sum_{i=1}^{m} h(\lambda', Y_i))}{\sum_{i=1}^{m} h(\lambda, Y_i)}, h(\lambda, Y_i) = \begin{cases} 1 \in Y_i \\ 0 \notin Y_i \end{cases} \tag{2.3}$$

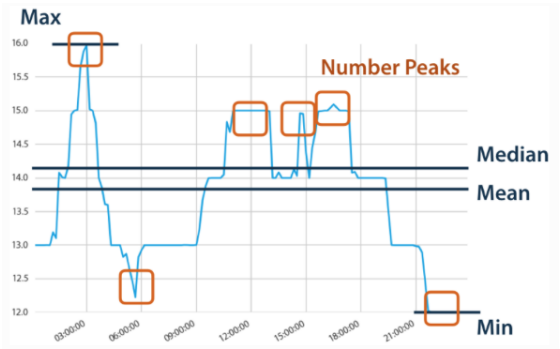
Sbilanciamento medio (MeanIR) Dato un dataset multi etichetta, calcola lo sbilanciamento medio delle classi [77].

$$MeanIR = \frac{1}{q} \sum_{\lambda \in L} IRLbl\lambda \tag{2.4}$$

Coefficiente di variazione (CVIR) Misura la similarità dei livelli di sbilanciamento tra le etichette. Serve per indicare se ci sono grandi differenze tra le frequenze delle classi. Per un dataset perfettamente bilanciato il CVIR è nullo e aumenta in modo proporzionale allo sbilanciamento delle classi [77].

$$CVIR = \frac{IRLlb\sigma}{MeanIR}, IRLlb\sigma = \sqrt{\sum_{\lambda \in L} \frac{(IRLlb\lambda - MeanIR)^2}{q-1}} \tag{2.5}$$

Libreria TsFresh Le misure seguenti sono state selezionate tra quelle disponibili nella libreria tsfresh [1]. Tsfresh è una collezione di pacchetti compatibili con il linguaggio Python e con Pandas. Forniscono una serie di strumenti utili per eseguire l'ingegnerizzazione delle caratteristiche su dati sequenziali o serie temporali. Attraverso un processo automatico le caratteristiche estratte possono essere usate a scopo descrittivo o usate per algoritmi di machine lerning che eseguono classificazione o regressione su serie temporali.



Valore medio Calcola il valore medio della serie.

$$M = \frac{1}{n-1} \sum_{i=1,..,n-1} |x_{i+1} - x_i| \tag{2.6}$$

Valore massimo e minimo Calcola il valore massimo e minimo della serie.

Kurtosis Calcola il kurtosis della serie con i coefficienti G2 del momento standardizzato Fisher-Pearson.

Numero di picchi Stima il numero di picchi della serie tramite la Ricker wavelet con ampiezza da 1 a n.

[1] https://www.tsfresh.com

Energia Calcola l'energia della serie.

$$E = \sum_{i=1,..,n} x_i^2 \tag{2.7}$$

Statistica C3 Calcola la non linearità della serie tramite la statistica C3 come riportato in [78].

$$E = \frac{1}{n - 2lag} \sum_{i=1}^{n-2lag} x_{i+2lag} \cdot x_{i+lag} \cdot x_i \tag{2.8}$$

Autocorrelazione media e varianza Applica una funzione di aggregazione sulla risultato dell'autocorrelazione in base ad un certo lag. Come funzione di aggregazione è stata utilizzata la media e la varianza.

$$R(l) = \frac{1}{(n-l)\sigma^2} \sum_{t=1}^{n-l} (X_t - \mu)(X_{t+l} - \mu) \tag{2.9}$$

$f_{agg}(R(1), .., R(m))$ for $m = max(n, maxlag)$

Welch Density Trasforma la serie nel dominio della frequenza e successivamente calcola la potenza spettrale a diverse frequenze.

Pendenza regressione La serie temporale viene suddivisa in sequenze, per ognuna calcola una regressione lineare e li compara con la regressione lineare calcolata per l'intera serie. Su questi viene selezionato un parametro della regressione, in questo caso la pendenza.

Cid CE Stima la complessità della serie temporale attraverso la seguente formula come descritto in [79].

$$\sqrt{\sum_{i=1}^{n-1} (x_i - x_{i-1})^2} \tag{2.10}$$

Entropia Calcola un'approssimazione dell'entropia secondo gli algoritmi [80, 81].

Capitolo 3

Ts2vec

Il framework allo stato dell'arte per estrarre rappresentazioni dalle serie temporali è Ts2Vec. I metodi [82, 83] si focalizzano ad estrarre una rappresentazione della serie che caratterizza l'intera finestra di osservazione, presentando dei limiti nei task di rilevamento anomalie o predizione. Questi task non ottengono risultati soddisfacenti con rappresentazioni ottenute sull'intera serie poichè si richiede di inferire dei campioni o sequenze brevi. Un altro metodo [84] considera informazioni contestuali con diversa granularità però non estrae rappresentazioni che possano essere indipendenti dal fattore di scala, quindi non è in grado di ottenere vari livelli di semantica della rappresentazione. Un ulteriore problema è che le rappresentazioni di serie temporale traggono ispirazione dall'esperienza guadagnata in ambito visual,basandosi su alcune assunzioni non valide in questo contesto [85]. La rete ts2vec propone una soluzione ai problemi sopra citati tramite un framework basato sul contrastive learning in grado di estrarre rappresentazioni a diversi livelli di semantica, confrontando in modo gerarchico lungo la dimensione del tempo e delle istanze. Le peculiarità della rete sono due : Consistenza contestuale e Contrasto gerarchico.

Consistenza contestuale Come già trattato nella sottosezione 1.1.3, il modo in cui vengono determinati i sample positivi ha una forte influenza sulle performance del modello. I modi principali sono :

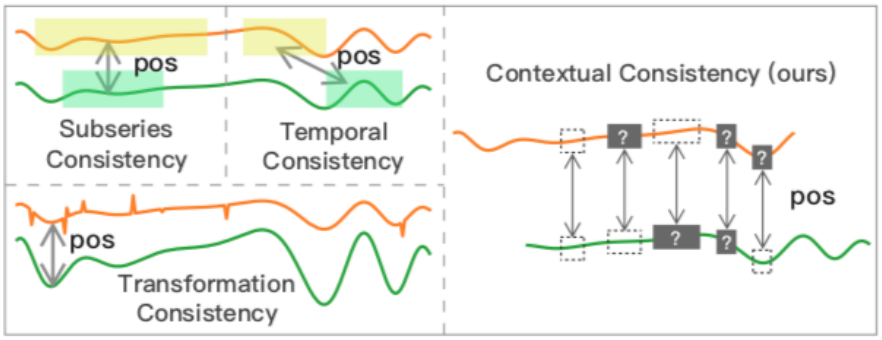
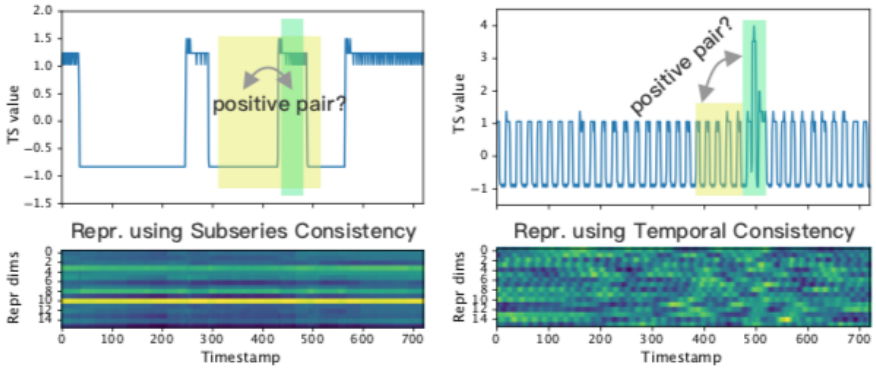


Figure 2: Positive pair selection strategies.



(a) Level shifts. (b) Anomalies.

Figura 3.1: Distribuzioni serie temporali [85]

- Consistenza della sottoserie : Data una serie, si campiona un segmento e da questo si campiona un ulteriore sotto segmento. Si avvicinano le rappresentazioni le rappresentazioni ottenute dai segmenti (vedi Subseries consistency in Figura 3.1).

- Consistenza temporale : Data una serie si campionano due sequenze adiacenti e si avvicinano le rappresentazioni (vedi Temporal consistency in Figura 3.1) per incoraggiare la continuità locale.

- Consistenza di transformazione : data una serie, si generano diverse versioni tramite varie trasformazioni e si avvicinano le rappresentazioni al fine di

renderle invarianti alle trasformazioni (vedi Transofrmation consistency in Figura 3.1).

Dalla Figura 3.1 è immediato notare che queste strategie non funzionano correttamente poichè si basano su assunzioni che perdono di significato quando applicate alle serie temporali. Ts2vec considera come coppie positive le rappresentazioni degli gli stessi campioni ma in due contesti diversi, dove il contesto è generato applicando una maschera e un crop casuale sulla serie in input. In questa maniera si incoraggia la ricostruzione del campione stesso ma in contesti diversi. Data una serie in input, viene applicato un crop casuale che genera due sottosequenze con parziale sovrapposizione. Il risultato viene dato in input al Projection Layer e successivamente si applica una maschera secondo una distribuzione di Bernulli con probabilità 0.5. L'obiettivo del contrastive è avvicinare le rappresentazioni delle parti in comune alle due sottosequenze.

Contrasto gerarchico Questa funzione obiettivo ha lo scopo di estrarre la semantica a diversi livelli. Partendo dalle rappresentazioni del singolo campione, si applica un filtro di pooling ad ogni rappresentazione lungo l'asse temporale e si calcola un elemento della funzione di costo complessiva. Si ripete il procedimento in maniera ricorsiva cosi da estrarre man mano caratteristiche di piu alto livello, fino ad arrivare alla rappresentazione a livello istanza cioè considerando l'intera serie. Al fine di estrarre una rappresentazione del contesto, la loss ricorsiva applica il contrasto sia lungo la direzione temporale che delle istanze.

Funzione obiettivo

$$L_{dual} = \frac{1}{NT} \sum_i \sum_t (l^{i,t}_{temp} + l^{i,t}_{inst}) \tag{3.1}$$

$$l^{(i,t)}_{temp} = -log \frac{exp(r_{i,t} \cdot r'_{i,t})}{\sum_{t' \in \Omega} (\exp(r_{i,t} \cdot r'_{i,t'}) + \mathbb{1}_{t \neq t'} \exp(r_{i,t} \cdot r_{i,t'}))} \tag{3.2}$$

$$l^{(i,t)}_{inst} = -\log \frac{\exp(r_{i,t} \cdot r'_{i,t})}{\sum_{j=1}^{B} (\exp(r_{i,t} \cdot r'_{j,t}) + \mathbb{1}_{i \neq j} \exp(r_{i,t} \cdot r_{j,t}))} \tag{3.3}$$

Le funzioni $r_{i,t}$ e $r'_{i,t}$ sono le rappresentazioni di due trasformazioni diverse ma dello stesso campione x_i. I parametri i e t sono rispettivamente l'indice della serie e il timestamp. B denota il batchsize. La classificazione ha lo scopo di determinare l'etichetta associata all'intera serie quindi la rete aggrega le rappresentazioni a partire dal singolo campione fino ad arrivare alla rappresentazione dell'intera serie. Su questa rappresentazione si applica un classificatore SVM con kernel RBF.

3.1 Descrizione dell'architettura

L'architettura è riportata nella Figura 3.2. L'elemento principale della rete è l'encoder, composto da trer componenti. La prima componente è un livello fully connected chiamato Input Projection Layer ed ha il compito di mappare la serie in input in uno spazio a dimensione maggiore. Segue un livello Timestamp Masking, che alla rappresentazione ottenuta precedentemente, applica delle maschere in modo casuale. Infine si applica un modulo convolutivo, composto da dieci residual blocks, al fine di estrarre rappresentazioni di contesto per ogni campione della serie. Ogni blocco ha un parametro di dilazione che cresce in modo esponenziale con la profondità del blocco nel modulo.

3.2 Estensioni al caso multivariato

Nei casi reali è molto probabile che il monitoraggio di un'entità richieda l'impiego di più sorgenti di informazione dato che i sistemi reali presentano un comportamento complesso. Le serie multivariate consentono di registrare questi andamenti e la scelta di un modello opportuno è fondamentale al fine di modellare le relazioni complesse che intercorrono sia tra le variabili, sia lungo l'asse temporale [55].

Quando si tratta di analizzare serie multivariate, l'approccio più semplice adottato è un metodo di ensemble ovvero applicare un certo modello in maniera separata per ognuna delle m variabili della serie. Per ogni variabile si ottiene un modello e la predizione finale è data dalla congiunzione dei risultati di ogni modello attraverso

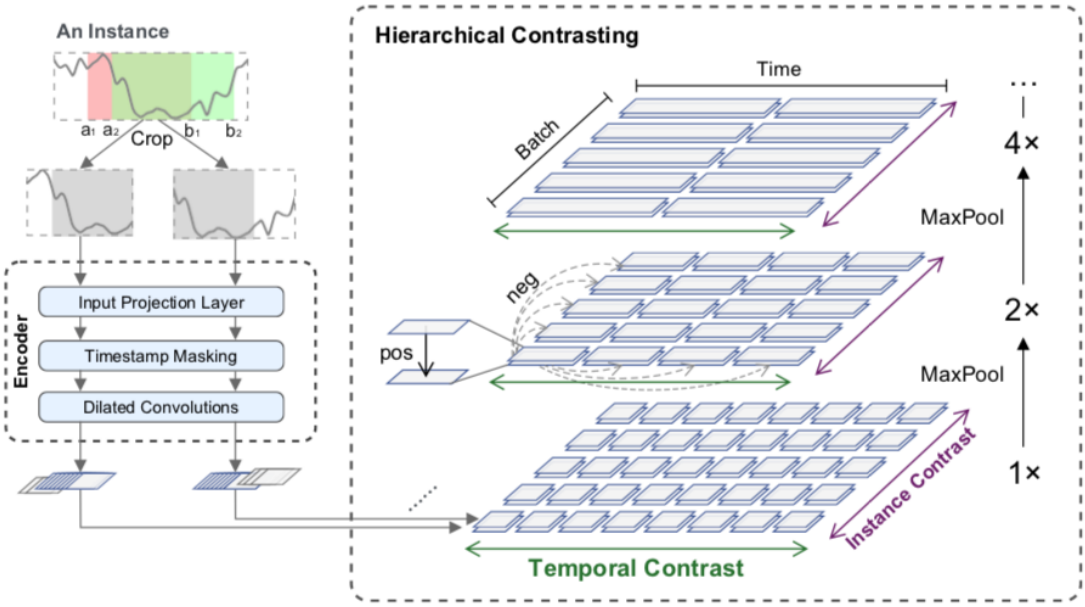


Figura 3.2: Funzionamento ts2vec [85]

una strategia di votazione, ad esempio votazione per maggioranza [58]. Un'altra tecnica [86] consiste nel trasformare la serie multivariata in una univariata attraverso una funzione di cross-correlazione dei vettori con stessa dipendenza temporale.

Gli algoritmi presentati nella sezione 2.2 possono essere applicati anche alle serie multivariate sfruttando una strategia ensemble. Oltre ad evidenziare le caratteristiche temporali, la difficoltà aggiuntiva per il caso multivariato risiede nell'evidenziare caratteristiche discriminanti per il task, che bisogna estrapolare dalle relazioni che intercorrono tra le variabili della serie[58]. Questo obiettivo non può essere raggiunto tramite i metodi di ensemble, causando una perdita di informazioni [75]. La mia analisi si focalizza sull'uso delle reti neurali.

TS2Vec gestisce in modo analogo l'analisi delle serie multivariate e univariate, ottenendo un'accuratezza media del 83% per l'archivio UCR e del 70% per l'archivio UEA. E' evidente che le perfomance sui dataset multivariati sono al di sotto di quelle sui dataset univariati. L'obiettivo del lavoro di tesi è quello di aumentare l'accuratezza media nel caso multivariato. La soluzione proposta segue lo stesso approccio della rete di riferimento sostituendo il livello di Input Projection con

una rete neurale più complessa, all'inizio della pipeline che implementa la logica Contrastive. Vengono esplorate tre architetture alternative, MLP, FCN e ResNet, considerate un forte riferimento nella classificazione di serie temporali e presentate in [87].

3.2.1 Multi Layer Perceptron (MLP)

La rete MLP Figura 3.3 è una sequenza di tre blocchi fortemente connessi, intervallati da una ReLu. Il numero dei parametri nei livelli è dipendente dalla lunghezza della serie in input. Le implicazioni di usare questa struttura sono state presentate nella sezione 2.2.2. Ogni livello è preceduto da un livello di dropout con probabilità rispettivamente di 0.1, 0.2, 0.2 e 0.3 [87]. Ogni blocco esegue le relazioni

$$\tilde{x} = f_{dropout,p(x)}, y = W \cdot \tilde{x} + b, h = ReLU(y) \tag{3.4}$$

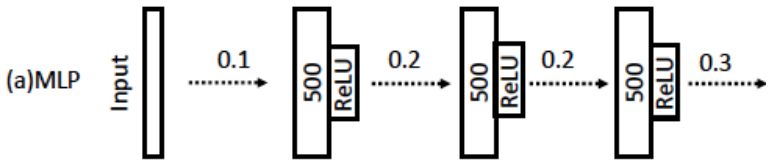


Figura 3.3: Architettura MLP [87]

Il vantaggio nell'utilizzo di questa rete consiste nella sua semplicità strutturale. Questa architettura presenta anche una serie di svantaggi. Il numero di parametri in input è strettamente dipendente dalla lunghezza serie temporale, quindi ogni campione viene processato attraverso un singolo neurone per livello, in maniera indipendente dagli altri. In questo modo non viene applicata nessuna invarianza spaziale, perdendo le informazioni che sono da associare alla dimensione temporale [54]. Nonostante i molteplici svantaggi, non si poteva escluderla a priori dato che la rete fornisce un riferimento base come mostrato nelle diverse pubblicazioni [87, 52, 54, 58].

3.2.2 Fully Convolutional (FC)

La rete FCN Figura 3.4 è composta da tre livelli. Ogni livello è una sequenza di tre elementi ovvero un livello convolutivo, un livello di normalizzazione ed un livello non lineare ReLu. La configurazione base per questi livelli non modifica la lunghezza della serie quindi lo stride è uno e non c'è padding. I filtri applicati sono 128, 256 e 128 rispettivamente nel primo, secondo e terzo livello, con kernel size pari a 8,5,3. Le implicazioni sull'uso di reti convolutive sono state approfondite nella sezione 2.2.2. Ogni livello esegue le seguenti relazioni [87].

$$y = W * x + b, s = BN(y), h = ReLU(s) \tag{3.5}$$

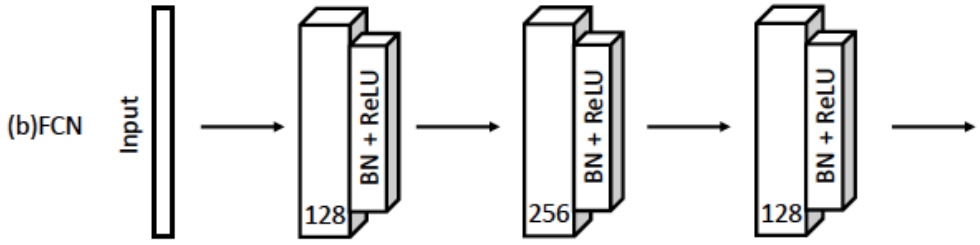


Figura 3.4: Architettura FCN [54]

Il vantaggio principale di questa architettura risiede nello sfruttamento delle proprietà della convoluzione. I segnali temporali sono corredati di rumore e solitamente si applica una moving average pesata ma resta il problema di determinare i parametri corretti basandosi sul dominio applicativo. Uno degli effetti della convoluzione è che esegue una forma di regolarizzazione sui dati. Poichè è parte integrante dell'architettura, non c'è bisogno di specificare nessun parametro. I parametri saranno aprresi durante la fase di allenamento e saranno propri del dataset elaborato [88]. Il vantaggio della convoluzione è quello di estrarre una rappresentazione lungo la direzione del tempo, tramite l'ampiezza di un filtro ,in contrasto con la logica puntuale delle reti FC. L'applicazione di un filtro 1D-CNN

genera una trasformazione non lineare della serie a partire dal prodotto di un insieme di punti della serie ed un insieme di valori di un filtro. Il procedimento si ripete facendo scivolare il filtro lungo la serie. Per ogni livello convolutivo l'applicazione del numero di filtri permette di determinare il numero di rappresentazioni estratte dal segnale in input. Questo processo genera varie rappresentazioni del segnale indipendenti tra di loro.

I dati sono caratterizzati da alcune proprietà strutturali, indipendenti dal task da eseguire, ad esempio la shift invariance. Tramite la condivisione dei pesi dovuta ai filtri, usando la convoluzione è possibile sfruttare queste informazioni note a priori. Un esempio può essere la similarità ovvero dati uguali saranno mappati nella stessa rappresentazione oppure invarianza della traslazione ovvero gli stessi dati avranno la stessa rappresentazione a prescindere dalla loro posizione nella serie. Queste architetture sono state costruite per estrarre una rappresentazione gerarchica del dato. A differenza delle reti FC presentano un numero minore di parametri ma la profondità della rete causa problemi di overfitting e di convergenza del modello.

3.2.3 ResNet

La rete ResNet Figura 3.5 è uno stack di tre blocchi FCN. Essendo la soluzione più profonda si aggiungono delle connessioni per poter permettere al gradiente di influenzare i primi livelli della rete durante la fase di allenamento. Ogni blocco viene indicato anche con il nome residuo. Lo stack di livelli convolutivi dovrebbe estrarre caratteristiche più astratte anche se comporta un costo maggiore nell'allenamento. Oltre a questo problema, c'è un rischio di overfitting quando una rete presenta molti parametri. I parametri dei livelli convolutivi sono gli stessi usati in FCN [54].

$$h1 = Block1(x), h2 = Block2(h1), h3 = Block3(h2), y = h3 + x, \hat{h} = ReLU(y) \tag{3.6}$$

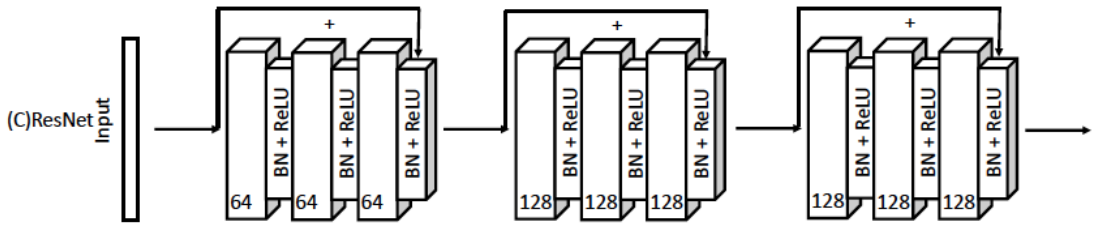


Figura 3.5: Architettura Resnet [54]

Il motivo per cui vogliamo analizzare questa architettura è la sua profondità. Con una rete molto profonda le performance dovrebbero aumentare perchè si apprende una funzione non lineare molto più complessa rispetto alle altre soluzioni. Queesto fa si che la rete sia in grado di apprendere una rappresentazione migliore e sia in grado di discriminare meglio tra le classi. Ogni livello di un'architettura deep lavora su una rappresentazione diversa del dato. In particolare l'astrazione della rappresentazione aumenta con la profondità della rete. Generando rappresentazioni che conservano una semantica di più alto livello è possibile imparare al meglio i dati. Questo vantaggio può diventare uno svantaggio nel caso in cui non ci siano abbastanza dati di training a disposizione. In questo scenario la rete entra in condizione di overfitting e perde la capacità di generalizzazione.

Capitolo 4

Esperimenti

4.1 Dataset

UCR L'archivio UCR [89] contiene 128 dataset di serie univariate appartenenti a diversi contesti e che possono essere suddivise in sette macro categorie : Bordo immagini, letture di sensori, spettrografie, movimenti, ECG, dispositivi elettrici e dati simulati. Dalle statistiche collezionate possiamo notare una significativa varianza nella lunghezza della serie con un massimo di 2700 campioni per il dataset HandOutLines e un minimo di 24 campioni per il dataset ItalyPowerDemand. Un altro parametro che è da tenere in considerazione perchè ha sicuramente un impatto sull'accuratezza é il numero di campioni nello split di training. Il valore massimo di campioni per l'allenamento è 8900 nel Dataset ElectricDevices. Riteniamo che 16 campioni per il caso minimo (dataset DiatomSizeReduction) siano poco significativi per l'allenamento della rete. Circa la metà dei dataset presenta un set per l'allenamento di circa 50 campioni. Per 40 dataset ci sono soltanto due classi e si arriva a 60 classi per il dataset ShapesAll. I dataset sono stati utilizzati tutti per gli esperimenti ma i dataset ElectricDevices, DodgerLoopDay, DodgerLoopGame, DodgerLoopWeekend sono stati esclusi dall'analisi a causa di errori nella fase di test [54, 58]. Gli utlimi tre dataset sono stati esclusi anche nell'analisi di Ts2vec.

Name	CVIR	#classes	#train_s	window_sz
Adiac	0.353	37	390	176
ArrowHead	0.0	3	36	251
Beef	0.0	5	30	470
BeetleFly	0.0	2	20	512
BirdChicken	0.0	2	20	512
Car	0.217	4	60	577
CBF	0.204	3	30	128
ChlorineConcentration	0.467	3	467	166
CinCECGTorso	0.499	4	40	1639
Coffee	0.0	2	28	286
Computers	0.0	2	250	720
CricketX	0.141	12	390	300
CricketY	0.073	12	390	300
CricketZ	0.14	12	390	300
DiatomSizeReduction	0.986	4	16	345
DistalPhalanxOutlineCorrect	0.368	2	600	80
DistalPhalanxOutlineAgeG	1.026	3	400	80
DistalPhalanxTW	0.623	6	400	80
Earthquakes	0.905	2	322	512
ECG200	0.537	2	100	96
ECG5000	1.582	5	500	140
ECGFiveDays	0.307	2	23	136
FaceAll	0.0	14	560	131
FaceFour	0.502	4	24	350
FacesUCR	0.613	14	200	131
FiftyWords	0.81	50	450	270
Fish	0.113	7	175	463
FordA	0.036	2	3601	500
FordB	0.033	2	3636	500
GunPoint	0.057	2	50	150
Ham	0.065	2	109	431
HandOutlines	0.39	2	1000	2709
Haptics	0.342	5	155	1092
Herring	0.309	2	64	512
InlineSkate	0.275	7	100	1882
InsectWingbeatSound	0.0	11	220	256
ItalyPowerDemand	0.021	2	67	24
LargeKitchenAppliances	0.0	3	375	720
Lightning2	0.471	2	60	637

Name	CVIR	#classes	#train_s	window_sz
Lightning7	0.396	7	70	319
Mallat	0.712	8	55	1024
Meat	0.0	3	60	448
MedicalImages	0.764	10	381	99
MiddlePhalanxOutlineCorrect	0.415	2	600	80
MiddlePhalanxOutlineAgeG	0.67	3	400	80
MiddlePhalanxTW	0.582	6	399	80
MoteStrain	0.0	2	20	84
NonInvasiveFetalECGTh1	0.113	42	1800	750
NonInvasiveFetalECGTh2	0.113	42	1800	750
OliveOil	0.473	4	30	570
OSULeaf	0.478	6	200	427
PhalangesOutlinesCorrect	0.427	2	1800	80
Phoneme	0.846	39	214	1024
Plane	0.284	7	105	144
ProximalPhalanxOutlineC	0.5	2	600	80
ProximalPhalanxOutlineAG	0.514	3	400	80
ProximalPhalanxTW	0.769	6	400	80
RefrigerationDevices	0.0	3	375	720
ScreenType	0.0	3	375	720
ShapeletSim	0.0	2	20	500
ShapesAll	0.0	60	600	512
SmallKitchenAppliances	0.0	3	375	720
SonyAIBORobotSurface1	0.566	2	20	70
SonyAIBORobotSurface2	0.262	2	27	65
StarLightCurves	0.611	3	1000	1024
Strawberry	0.404	2	613	235
SwedishLeaf	0.126	15	500	128
Symbols	0.331	6	25	398
SyntheticControl	0.0	6	300	60
ToeSegmentation1	0.0	2	40	277
ToeSegmentation2	0.0	2	36	343
Trace	0.171	4	100	275
TwoLeadECG	0.061	2	23	82
TwoPatterns	0.058	4	1000	128
UWaveGestureLibraryX	0.075	8	896	315
UWaveGestureLibraryY	0.075	8	896	315
UWaveGestureLibraryZ	0.075	8	896	315
UWaveGestureLibraryAll	0.075	8	896	945

Name	CVIR	#classes	#train_s	window_sz
Wafer	1.14	2	1000	152
Wine	0.074	2	57	234
WordSynonyms	0.62	25	267	270
Worms	0.471	5	181	900
WormsTwoClass	0.227	2	181	900
Yoga	0.123	2	300	426
ACSF1	0.0	10	100	1460
AllGestureWiimoteX	0.0	10	300	500
AllGestureWiimoteY	0.0	10	300	500
AllGestureWiimoteZ	0.0	10	300	500
BME	0.0	3	30	128
Chinatown	0.0	2	20	24
Crop	0.0	24	7200	46
EOGHorizontalSignal	0.013	12	362	1250
EOGVerticalSignal	0.013	12	362	1250
EthanolLevel	0.0	4	504	1751
FreezerRegularTrain	0.0	2	150	301
FreezerSmallTrain	0.0	2	28	301
Fungi	0.0	18	18	201
GestureMidAirD1	0.0	26	208	360
GestureMidAirD2	0.0	26	208	360
GestureMidAirD3	0.0	26	208	360
GesturePebbleZ1	0.073	6	132	455
GesturePebbleZ2	0.042	6	146	455
GunPointAgeSpan	0.01	2	135	150
GunPointMaleVersusFemale	0.073	2	135	150
GunPointOldVersusYoung	0.062	2	136	150
HouseTwenty	0.0	2	40	2000
InsectEPGRegularTrain	0.596	3	62	601
InsectEPGSmallTrain	0.529	3	17	601
MelbournePedestrian	0.014	10	1194	24
MixedShapesRegularTrain	0.0	5	500	1024
MixedShapesSmallTrain	0.0	5	100	1024
PickupGestureWiimoteZ	0.0	10	50	361
PigAirwayPressure	0.0	52	104	2000
PigArtPressure	0.0	52	104	2000
PigCVP	0.0	52	104	2000
PLAID	0.721	11	537	1344
PowerCons	0.0	2	180	144

Name	CVIR	#classes	#train_s	window_sz
Rock	0.0	4	20	2844
SemgHandGenderCh2	0.0	2	300	1500
SemgHandMovementCh2	0.0	6	450	1500
SemgHandSubjectCh2	0.0	5	450	1500
ShakeGestureWiimoteZ	0.0	10	50	385
SmoothSubspace	0.0	3	150	15
UMD	0.0	3	36	150

Tabella 4.1: Caratteristiche dataset UCR

UEA L'archivio UEA [90] consiste di una collezione di 30 dataset di serie temporali multivariate. Anche questo archivio contiene dataset con caratteristiche molto diverse tra loro con variazioni sensibili. La dimensione del set per l' allenamento principalemnte si aggira attorno a qualche centinaia di campioni ma presenta dei valori minimi intorno alla decina (12 per il dataset StendWalkJump) fino ad un massimo di 25000 campioni per il dataset InsectWingBeat. Il numero di classi tendenzialmente è un valore sotto la decina con un minimo di due classi ed un massimo di 39 classi per il dataset PhonemeSpectra. Il numero di variabili per ogni serie si aggira tra sei ed otto ma in alcuni casi si arriva a 1345 come per DuckDuckGeese e 963 per il dataset PEMS-SF. La lunghezza della serie è molto variabile ed è ben distribuita tra i valori 8 per il dataset PenDigits e circa 18000 campioni per il dataset EigenWorms [54, 58].

Questi due archivi sono i più utilizzati in letteratura,sono un ottimo punto di partenza ma hanno molti limiti. I dataset sono tendenzialmente piccoli, correlati e la lunghezza cosi diversa delle serie non è rappresentativa dei contesti reali per le serie multivariate [58]. A differenza di altre pubblicazioni in cui gli esperimenti sono stati eseguiti selezionando solo i dataset con stessa lunghezza della finestra, in questo lavoro sono stati testati tutti i dataset al fine di analizzare ogni possibile configurazione.

Name	CVIR	#classes	#train_samples	window_size	#features
ArticularyWordRecognition	0.0	25	275	144	9
AtrialFibrillation	0.0	3	15	640	2
BasicMotions	0.0	4	40	100	6
CharacterTrajectories	0.126	20	1422	182	3
Cricket	0.0	12	108	1197	6
DuckDuckGeese	0.0	5	50	270	1345
EigenWorms	0.471	5	128	17984	6
Epilepsy	0.096	4	137	206	3
ERing	0.0	6	30	65	4
EthanolConcentration	0.008	4	261	1751	3
FaceDetection	0.0	2	5890	62	144
FingerMovements	0.009	2	316	50	28
HandMovementDirection	0.0	4	160	400	10
Handwriting	0.509	26	150	152	3
Heartbeat	0.624	2	204	405	61
JapaneseVowels	0.0	9	270	29	12
Libras	0.0	15	180	45	2
LSST	1.637	14	2459	36	6
MotorImagery	0.0	2	278	3000	64
NATOPS	0.0	6	180	51	24
PEMS-SF	0.136	7	267	144	963
PenDigits	0.042	10	7494	8	2
PhonemeSpectra	0.0	39	3315	217	11
RacketSports	0.105	4	151	30	6
SelfRegulationSCP1	0.011	2	268	896	6
SelfRegulationSCP2	0.0	2	200	1152	7
SpokenArabicDigits	0.0	10	6599	93	13
StandWalkJump	0.0	3	12	2500	4
UWaveGestureLibrary	0.0	8	120	315	3
InsectWingbeat	0.0	10	25000	22	200

Tabella 4.2: Caratteristiche dataset UEA

KPI Il dataset KPI [91] consiste di serie temporali univariate con anomalie etichettate riguardo diversi KPI, collezionate da grandi compagnie internet come eBay o Sogou. E' un dataset pubblico che viene utilizzato come benchmark per testare algoritmi di rilevamento anomalie ed è stato rilasciato da AIOPS data competition. Alcune serie temporali sono state generate da un campionamento eseguito ogni minuto mentre serie sono state campionate ogni 5 minuti.

Yahoo Il dataset Yahoo [92], messo a disposizione tramite webscope data sharing program, è stato creato per testare algoritmi di rilevamento anomalie e migliorare la sicurezza degli utenti Yahoo. Consiste di serie temporali univariate con anomalie etichettate. Descrivono il traffico dei servizi offerti da yahoo e sono costituiti da una parte di informazioni reali e una parte sintetica. Le etichette delle anomalie sono state generate dallo stesso processo che ha creato i dati sintetici mentre le anomalie del traffico reale sono state etichettate ad esperti di dominio.

4.2 Metriche

Accuratezza, precisione, richiamo, Fscore Per quanto riguarda gli esperimenti di classificazione la metrica utilizzata è l'accuratezza mentre per riprodurre i risultati di rilevamento anomalie sono state utilizzate precisione, richiamo ed F-score dato lo sbilanciamento delle classi [93] . Dato un dataset D con N campioni e due classi con rispettive etichette positiva e negativa. Definiamo la variabile True Positive (TP) come il numero di elementi classificati correttamente con etichetta positiva. La stessa definizione vale per la classe negativa (TN). Definiamo False Positive e Falsi Negativi (FN,FP) le variabili che conteggiano il numero di elementi classificati con una etichetta ma che si sono rivelati appartenenti all'altra classe.

$$Accuratezza = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

$$Precisione = \frac{TP}{TP + FP} \tag{4.2}$$

$$Richiamo = \frac{TP}{TP + FN} \tag{4.3}$$

$$Fscore = 2 \cdot \frac{Precisione \cdot Richiamo}{Precisione + Richiamo} \tag{4.4}$$

Sbilanciamento classi I metodi di classificazione sono stati progettati attorno all'assunzione di una uniformità nella distribuzione delle classi. Nel caso in cui il dataset sia sbilanciato, bisogna scegliere una tecnica di sovra campionamento o sotto campionamento al fine di migliorare il bilanciamento. Per il sotto campionamento la tecnica più semplicE consiste nel rimuovere in modo casuale un insieme di dati dalle classi con una frequenza maggiore di una soglia prestabilita e si chiama random undersampling. Cosi facendo c'è una perdita di informazioni. Per il sovra campionamento esiste una tecnica analoga ma con lo scopo opposto ovvero vengono selezionati casualmente dei campioni della classe minoritaria e si creano dei duplicati. Questa tecnica, chiamata random oversampling, altera la distribuzione delle classi. Un'alternativa al random oversampling è la tecnica chiamata SMOTE [94].

SMOTE è una tecnica di sovra campionamento geometrico basata sul knn. Crea campioni "sintetici" selezionando casualmente un campione di una classe di minoranza e scegliendo i suoi k campioni più vicini della stessa classe.

Quindi crea un nuovo campione selezionando in modo casuale nella retta che collega due campioni. Questa tecnica migliora l'overfitting rispetto al random oversampling, inoltre non c'è una perdita di informazioni. Non funziona bene se usata su dati con un'alta dimensionalità e può introdurre rumore aumentando la sovrapposizione di classi diverse. SMOTE mostra che una combinazione del metodo di sovra campionamento della classe minoritaria ed il sotto campionamento della classe maggioritaria può ottenere prestazioni del classificatore migliori rispetto al solo sovra campionamento.

In questo lavoro di tesi per quanto riguarda il setting del rilevamento anomalie non è stata utilizzata nessuna tecnica per il bilanciamento dei dataset al fine di riprodurre gli esperimenti di TS2Vec nelle condizioni originali.

Per quanto riguarda il problema di classificazione, come mostrato nella colonna CVIR della Tabella 4.4 e Tabella 4.5, tutti i dataset presentano un coefficiente prossimo allo zero, sia nel caso multivariato che in quello univariato, indicando un ottimo livello di bilanciamento dei dataset. Per questo motivo non è stata

applicata alcuna tecnica di bilanciamento. Solo tre dataset tra tutti quelli presenti sia in UCR che UEA presentano un coefficiente maggiore di uno ma questo non è un problema in quanto il CVIR non ha un limite superiore e un dataset è da considerarsi sbilanciato solo quando il CVIR presenta valori molto maggiori di quelli riscontrati nelle statistiche [77].

4.3 Setup degli esperimenti

Dopo aver riprodotto gli esperimenti di Ts2vec, sono state testate le tre architetture proposte su tutti i dataset degli archivi UEA/UCR. Per ogni dataset è stata applicata una ricerca dei miglior iperparametri.

Learning rate	Weight decay	hidden_dims	repr_dim
0.1	0.001	32	220
0.05	0.01	64	320
0.01	0.1	128	420
0.007	0.2		
0.005			
0.001			

Tabella 4.3: Grid search iperparametri

Per ogni esperimento, l' accuratezza trovata è pari alla media di cinque esecuzioni diverse al fine di ridurre il bias dovuta all'inizializzazione dei pesi. Per ogni dataset, la separazione dei dati in dati di training e test non è stata cambiata rispetto a quella proposta da entrambi gli archivi. Per il caso della classificazione sono stati eseguiti 150 esperimenti per il caso multivariato e circa 600 esperimenti per il caso univariato. Gli esperimenti sono stati eseguiti su Google COLAB.

4.4 Statistiche dataset

Nelle tabelle Tabella 4.4 e Tabella 4.5 sono riportate le statistiche delle serie temporali, calcolate secondo quanto presentato nella sezione 2.4. Queste misure

sono state utilizzate al fine di scoprire correlazioni tra le caratteristiche delle serie e lo scarto dei risultati di classificazione rispetto alla rete di confronto Ts2vec.

Tabella 4.4: Statistiche dataset univariato UCR

Name	CVIR	#C	TS	WS	max	min	energy	kurtosis	c3	A_M	A_V	entropy	slope	cid_ce	W_D	fft_ent	peaks
Adiac	0.353	37	390	176	2.63	-1.99	68250.0	7.3192	0.1429	0.9141	0.0049	0.1674	-0.0	20.7349	0.1339	0.136	993
ArrowHead	0.0	3	36	251	2.55	-2.26	9000.0	9.7565	-0.2012	0.9354	0.0026	0.1638	-0.0	7.0395	0.438	0.2607	134
Beef	0.0	5	30	470	3.72	-3.29	14070.0	10.5769	-0.2302	0.9386	0.0019	0.1445	-0.0	11.5494	0.7686	0.4297	123
BeetleFly	0.0	2	20	512	2.51	-2.52	10220.0	14.5991	0.1822	0.8741	0.0094	0.2547	-0.0002	11.0278	1.0355	0.5151	214
BirdChicken	0.0	2	20	512	2.12	-2.82	10220.0	10.8904	-0.0443	0.9706	0.0005	0.1029	-0.0	6.0998	0.1788	0.2264	89
Car	0.217	4	60	577	1.99	-2.21	34560.0	15.1909	-0.0468	0.9894	0.0001	0.0521	-0.0	5.512	0.0379	0.1812	204
CBF	0.204	3	30	128	3.24	-2.32	3810.0	7.2256	0.1462	0.6638	0.0067	1.2283	-0.0001	39.8842	2.6363	0.7582	192
ChlorineConcentration	0.467	3	467	166	7.44	-11.84	77055.0	10.2207	-0.0075	0.2597	0.053	0.9938	0.0	199.5143	6.0246	2.3737	4716
CinCECGTorso	0.499	4	40	1639	10.54	-8.59	65520.0	23.8846	-0.5866	0.9393	0.0022	0.0862	-0.0	18.1639	0.5127	0.4512	378
Coffee	0.0	2	28	286	2.18	-2.06	7980.0	6.0834	-0.3715	0.9159	0.003	0.2076	-0.0	10.5273	0.7418	0.4297	82
Computers	0.0	2	250	720	21.6	-3.75	179749.1	5.5811	1.4641	0.7185	0.006	0.1968	0.0	242.4753	2.851	1.0527	4182
CricketX	0.141	12	390	300	11.49	-4.77	116610.0	5.3926	0.1783	0.6721	0.0185	0.5353	-0.0	153.8557	4.4332	1.0269	2775
CricketY	0.073	12	390	300	6.84	-9.77	116610.0	5.0913	-0.3216	0.7123	0.019	0.5536	0.0	128.2341	4.8676	1.0202	2889
CricketZ	0.14	12	390	300	11.92	-4.76	116610.0	5.4533	0.2047	0.6666	0.0182	0.4811	-0.0	156.0822	4.2621	1.0669	2850
DiatomSizeReduction	0.986	4	16	345	1.98	-1.77	5504.0	10.2974	0.2172	0.9742	0.0004	0.0778	-0.0	3.153	0.0341	0.1812	42
DistalPhalanxOC	0.368	2	600	80	2.45	-2.16	47400.0	7.2889	-0.0034	0.6475	0.0561	0.4772	0.0001	47.1323	33.8551	0.6368	991
DistalPhalanxOutAG	1.026	3	400	80	2.06	-1.99	31600.0	7.076	0.0092	0.6405	0.0584	0.4342	0.0001	38.8245	35.8506	0.6368	661
DistalPhalanxTW	0.623	6	400	80	2.06	-1.99	31600.0	7.2562	0.011	0.6387	0.0594	0.3669	-0.0	38.2647	36.1716	0.6763	767
Earthquakes	0.905	2	322	512	7.86	-0.89	164542.0	25.9295	0.0083	0.0197	0.0001	0.6558	-0.0	559.9878	2.2491	2.9321	8348
ECG200	0.537	2	100	96	4.2	-2.62	9500.0	6.5064	-0.4651	0.6563	0.0256	0.6136	0.0	35.0289	3.5852	1.1715	316
ECG5000	1.582	5	500	140	4.06	-5.8	69500.0	6.4735	-0.1221	0.5372	0.056	0.4054	0.0	92.3885	10.2359	1.2704	1618
ECGFiveDays	0.307	2	23	136	5.42	-6.51	3105.0	12.6272	0.2751	0.287	0.0575	0.2426	0.0003	30.4942	1.9408	2.1181	103
FaceAll	0.0	14	560	131	4.88	-4.48	72800.0	14.436	-0.0204	0.0054	0.1276	0.9991	0.0	164.1069	8.4504	2.3905	1015
FaceFour	0.502	4	24	350	5.91	-4.69	8376.0	5.6614	0.1792	0.44	0.113	0.6707	-0.0001	34.0351	16.5618	1.084	244
FacesUCR	0.613	14	200	131	8.74	-3.96	26000.0	11.5041	-0.0547	0.0023	0.1155	0.9024	-0.0	108.0923	9.7203	2.6	610
FiftyWords	0.81	50	450	270	5.02	-2.35	121050.0	20.1625	0.9488	0.8396	0.0136	0.2424	0.0	42.8735	3.0922	0.7376	1901
Fish	0.113	7	175	463	2.13	-1.95	80850.0	19.294	0.1685	0.9836	0.0002	0.0658	0.0	9.716	0.0535	0.1812	598
FordA	0.036	2	3601	500	5.06	-4.62	1796899.0	17.7934	-0.0009	0.1713	0.2625	0.6149	-0.0	425.5162	59.7752	0.9777	25528
FordB	0.033	2	3636	500	5.09	-5.54	1814364.0	19.3136	0.0013	0.1007	0.2819	0.6217	0.0	461.4904	34.867	1.0685	22941
GunPoint	0.057	2	50	150	2.05	-2.37	7450.0	8.0391	0.3345	0.9326	0.0027	0.1392	-0.0	7.6065	1.0162	0.4256	74
Ham	0.065	2	109	431	8.03	-2.05	46870.0	13.0932	0.5952	0.6181	0.0579	0.3601	0.0	46.6736	8.8075	1.2239	960
HandOutlines	0.39	2	1000	2709	2.09	-3.22	2707999.0	18.0104	-0.7196	0.9991	0.0	0.0135	0.0	14.6369	0.0011	0.136	7579
Haptics	0.342	5	155	1092	3.12	-11.15	169105.0	5.6205	0.4277	0.9273	0.0001	0.0732	0.0	121.4257	0.1575	0.2264	768
Herring	0.309	2	64	512	2.13	-2.19	32704.0	18.5925	0.2116	0.9812	0.0002	0.0768	0.0	6.8071	0.0851	0.2264	226
InlineSkate	0.275	7	100	1882	4.34	-2.26	188100.0	9.0096	0.9747	0.9971	0.0	0.0278	0.0	12.3721	0.0195	0.1812	437
InsectWingbeatSound	0.0	11	220	256	6.42	-1.08	56100.0	12.5201	1.0088	0.6762	0.0473	0.208	0.0	46.2936	8.4053	1.0907	944
ItalyPowerDemand	0.021	2	67	24	2.42	-1.99	1541.0	10.1282	0.2588	0.0441	0.1992	0.8474	0.0002	20.6634	45.2875	0.9585	8
LargeKitchenAppInc	0.0	3	375	720	26.8	-1.58	269625.0	6.1344	1.1044	0.4925	0.0382	0.0608	0.0	276.7626	7.636	2.0716	1545
Lightning2	0.471	2	60	637	23.13	-1.4	38160.0	6.7249	1.0217	0.6976	0.0062	0.3376	0.0	99.4538	3.1897	1.3425	414
Lightning7	0.396	7	70	319	17.41	-1.78	22260.0	7.3855	0.7522	0.6677	0.0049	0.4994	0.0	91.6293	2.3042	1.0075	516
Mallat	0.712	8	55	1024	2.76	-1.61	56265.0	7.5625	0.4758	0.9694	0.0004	0.1027	-0.0	16.2135	0.2373	0.3808	297
Meat	0.0	3	60	448	3.39	-1.54	26820.0	7.8822	1.2043	0.9645	0.0007	0.0787	-0.0	12.233	0.2122	0.3165	117
MedicalImages	0.764	10	381	99	7.22	-2.39	37338.0	7.8469	0.098	0.4528	0.0676	0.3363	-0.0	74.4704	10.9615	1.6363	958
MiddlePhalanxOC	0.415	2	600	80	2.07	-1.66	47400.0	6.843	-0.0383	0.6785	0.0529	0.3099	0.0001	40.6252	34.5103	0.5567	901
MiddlePhalanxOutAG	0.67	3	400	80	1.92	-1.64	31600.0	6.5992	-0.0339	0.6755	0.0538	0.3115	0.0001	33.3833	35.0828	0.5567	618
MiddlePhalanxTW	0.582	6	399	80	1.92	-1.72	31521.0	6.7966	-0.0261	0.6696	0.0554	0.3021	-0.0	33.2666	35.6817	0.5567	616
MoteStrain	0.0	2	20	84	2.47	-8.41	1660.0	9.2957	0.0015	0.5236	0.0162	0.4725	-0.0013	31.3969	3.9152	0.9409	40
NonInvFetalECGT1	0.113	42	1800	750	4.79	-5.73	1348200.0	10.1843	0.4737	0.9217	0.0038	0.1223	0.0	91.6055	0.1986	0.4512	6258
NonInvFetalECGT2	0.113	42	1800	750	4.68	-5.42	1348200.0	11.4078	0.6646	0.9286	0.0032	0.1203	-0.0	87.5266	0.215	0.3956	6342
OliveOil	0.473	4	30	570	3.72	-1.0	17070.0	21.4395	1.0731	0.9013	0.0053	0.1453	-0.0	12.2324	2.1824	0.6191	82
OSULeaf	0.478	6	200	427	3.67	-3.16	85200.0	10.899	0.0131	0.8993	0.0056	0.2302	-0.0	31.8129	0.8919	0.4852	1398
PhalangesOutlinesC	0.427	2	1800	80	2.45	-2.16	142200.0	7.2404	-0.0106	0.6598	0.0572	0.4772	0.0	74.2696	36.2902	0.6024	2741

Tabella 4.4: Statistiche dataset univariato UCR

Name	CVIR	#C	TS	WS	max	min	energy	kurtosis	c3	A_M	A_V	entropy	slope	cid_ce	W_D	fft_ent	peaks
Phoneme	0.846	39	214	1024	8.09	-13.32	218922.0	6.3675	-0.0926	0.4739	0.0362	0.9777	-0.0	286.1484	7.012	2.1135	8096
Plane	0.284	7	105	144	2.91	-2.11	15015.0	9.8622	0.353	0.6774	0.0535	0.4333	0.0	22.475	11.5893	0.7644	497
ProximalPOutlCorr	0.5	2	600	80	1.9	-1.48	47400.0	6.9492	0.0099	0.6533	0.0628	0.2667	0.0001	40.542	40.5045	0.5315	849
ProximalPOutlAG	0.514	3	400	80	1.9	-1.48	31600.0	6.861	0.0199	0.6436	0.0656	0.2708	0.0001	33.722	41.9182	0.5675	563
ProximalPhalanxTW	0.769	6	400	80	1.9	-1.48	31600.0	6.959	0.0174	0.6464	0.0651	0.2679	-0.0	33.2023	41.6064	0.5315	620
RefrigerationDevices	0.0	3	375	720	5.59	-5.47	269625.0	6.9108	0.0436	0.3213	0.069	0.2975	0.0	319.8261	10.5748	2.4949	7028
ScreenType	0.0	3	375	720	26.8	-2.92	269624.9	5.2396	0.9719	0.7375	0.0063	0.2748	-0.0	286.1425	3.3064	1.0999	7136
ShapeletSim	0.0	2	20	500	1.89	-1.81	9980.0	26.8691	0.0192	0.0166	0.0002	1.7831	0.0001	138.8183	3.0294	3.3969	400
ShapesAll	0.0	60	600	512	2.82	-3.22	306600.0	12.4389	-0.0896	0.9711	0.0005	0.077	0.0	31.9327	0.1792	0.3165	2401
SmallKitchenAppl	0.0	3	375	720	26.8	-5.07	269625.0	16.0414	0.0387	0.0799	0.014	0.0844	0.0	567.3163	4.1607	3.9972	3052
SonyAIBORobotSurf1	0.566	2	20	70	3.63	-2.73	1380.0	10.1284	0.0268	0.19	0.1155	0.8738	0.0005	19.4392	4.7395	1.5807	7
SonyAIBORobotSurf2	0.262	2	27	65	4.23	-4.14	1728.0	19.182	0.0866	-0.0263	0.087	0.9783	0.0001	34.0042	1.7849	2.6527	4
StarLightCurves	0.611	3	1000	1024	5.29	-2.68	1023000.0	11.8566	0.3964	0.9939	0.0	0.0268	-0.0	32.3824	0.0344	0.2264	2484
Strawberry	0.404	2	613	235	3.68	-2.33	143442.0	10.3761	1.0679	0.9223	0.0032	0.2045	0.0	35.7646	1.0147	0.4404	1586
SwedishLeaf	0.126	15	500	128	3.22	-3.41	63500.0	7.9911	0.1308	0.7331	0.0319	0.4782	-0.0	48.3334	1.9644	0.5905	1615
Symbols	0.331	6	25	398	2.2	-2.31	9925.0	5.8487	0.0368	0.9361	0.0014	0.0812	-0.0	14.3275	0.7154	0.4704	56
SyntheticControl	0.0	6	300	60	2.41	-2.45	17700.0	14.8223	0.0026	0.267	0.0259	1.6829	0.0005	124.7216	1.5076	1.5273	246
ToeSegmentation1	0.0	2	40	277	3.93	-3.58	11040.0	7.3449	0.3028	0.7322	0.0306	0.4215	0.0	22.2097	6.0625	0.9046	257
ToeSegmentation2	0.0	2	36	343	3.93	-2.64	12312.0	7.4019	0.5971	0.8716	0.0079	0.2981	0.0001	16.3544	1.8329	0.5935	175
Trace	0.171	4	100	275	3.97	-2.22	27400.0	7.3959	-0.7876	0.891	0.0032	0.0703	0.0	27.9662	1.5386	0.5465	200
TwoLeadECG	0.061	2	23	82	1.87	-3.15	1863.0	7.5227	-0.1998	0.6843	0.0386	0.2878	-0.0001	9.9405	4.6878	0.732	37
TwoPatterns	0.058	4	1000	128	1.94	-1.94	126999.9	6.6492	0.0039	0.3348	0.0952	0.6873	-0.0	217.1148	16.9048	1.9894	6629
UWaveGestureLibX	0.075	8	896	315	4.43	-4.44	281344.0	10.2593	0.0388	0.9379	0.0019	0.1345	-0.0	50.1427	0.8822	0.4404	2252
UWaveGestureLibY	0.075	8	896	315	7.65	-4.1	281343.9	8.6958	0.0976	0.9517	0.001	0.1191	-0.0	50.5113	0.6705	0.3956	1878
UWaveGestureLibZ	0.075	8	896	315	4.78	-3.55	281344.0	9.0871	0.0681	0.9432	0.0015	0.1227	-0.0	51.9238	0.7615	0.4404	2087
UWaveGestLibAll	0.075	8	896	945	7.63	-4.43	845823.9	7.1529	0.0687	0.9393	0.0016	0.1265	-0.0	107.1139	0.8146	0.4404	7628
Wafer	1.14	2	1000	152	11.79	-3.05	151000.0	5.6544	-0.0554	0.6259	0.0262	0.2119	0.0	188.8313	5.5843	1.1294	2844
Wine	0.074	2	57	234	3.2	-1.94	13281.0	5.8017	0.8233	0.7461	0.0216	0.1851	-0.0	30.6868	3.9503	0.9198	202
WordSynonyms	0.62	25	267	270	5.0	-2.26	71823.0	20.4147	0.9386	0.8378	0.0139	0.2533	0.0	33.1385	3.3412	0.7161	1148
Worms	0.471	5	181	900	4.86	-4.31	162719.0	7.2061	0.0269	0.9432	0.0015	0.2302	0.0	44.7061	0.5438	0.4512	1496
WormsTwoClass	0.227	2	181	900	4.86	-4.31	162719.0	7.2061	0.0269	0.9432	0.0015	0.2302	0.0	44.7061	0.5438	0.4512	1496
Yoga	0.123	2	300	426	2.41	-2.42	127500.0	10.4441	-0.0958	0.9692	0.0005	0.1071	0.0	23.2109	0.1697	0.2715	1127
ACSF1	0.0	10	100	1460	12.43	-1.35	146000.0	24.0511	0.955	-0.0049	0.2649	0.3465	-0.0	592.8896	0.1335	0.2607	5452
AllGestureWiimoteX	0.0	10	300	500	4.27	-4.85	27184.51	8.8558	-0.0845	0.5933	0.0736	0.2146	0.0	91.7785	2.3707	1.1263	1301
AllGestureWiimoteY	0.0	10	300	500	4.22	-3.11	17103.27	8.8195	0.0871	0.6664	0.0473	0.1597	0.0	86.1176	1.0637	1.1488	1119
AllGestureWiimoteZ	0.0	10	300	500	4.31	-4.81	46571.31	7.5392	0.3219	0.8115	0.0138	0.1136	0.0	73.4123	1.0365	0.9409	566
BME	0.0	3	30	128	1.37	-1.4	1585.167	13.6208	0.0021	0.8943	0.0045	0.1549	0.0001	7.486	0.8734	0.6435	35
Chinatown	0.0	2	20	24	1891.0	3.0	330474264	5.719	355825603	0.2167	0.237	0.6074	-0.9232	8.2433	235112	0.4465	0
Crop	0.0	24	7200	46	0.97	-1.01	83055.665	5.2716	0.1371	0.7491	0.0166	0.4504	-0.0	180.782	0.1483	1.1656	3468
EOGHorizontalSignal	0.013	12	362	1250	446.32	-1110.8	9738526218	10.9067	-1023505	0.9922	0.0	0.0243	0.0005	28.1137	1582.63	0.3165	1469
EOGVerticalSignal	0.013	12	362	1250	2540.7	-923.7	17675839474	7.6348	32806987	0.9921	0.0	0.0076	0.0027	33.8225	3958.48	0.3956	1554
EthanolLevel	0.0	4	504	1751	2.08	-0.99	882504.0	21.6875	0.4865	0.9992	0.0	0.0136	0.0	7.5566	0.0003	0.2264	1882
FreezerRegularTrain	0.0	2	150	301	5.02	-2.23	45000.0	5.062	-0.6729	0.9048	0.0025	0.0719	0.0	41.1564	1.1633	0.5315	269
FreezerSmallTrain	0.0	2	28	301	1.43	-2.23	8400.0	5.2576	-0.7622	0.904	0.0025	0.0645	-0.0	17.6169	1.1188	0.5567	52
Fungi	0.0	18	18	201	80.79	-1.49	434427.2341	30.7558	2063.3564	0.6675	0.0479	0.1183	-0.0016	11.2839	861.8	1.0792	28
GestureMidAirD1	0.0	26	208	360	282.72	-207.21	187462196.9	6.2677	1369.3406	0.9424	0.0012	0.0844	0.0002	36.1138	1390.4	0.3956	408
GestureMidAirD2	0.0	26	208	360	375.36	0.0	1597682040	5.6012	4879456	0.9442	0.0009	0.0566	0.001	40.0035	9271.7	0.4256	332
GestureMidAirD3	0.0	26	208	360	225.74	-239.5	155348284.5	5.9959	66901	0.9622	0.0004	0.0565	0.0005	32.0956	988.1387	0.3956	318
GesturePebbleZ1	0.073	6	132	455	39.99	-39.99	2639117.5225	7.3627	-255.6523	0.6804	0.007	0.2078	0.0	116.7377	85.9481	1.5534	658
GesturePebbleZ2	0.042	6	146	455	39.99	-39.99	2540646.8011	7.2588	-215.897	0.6655	0.0076	0.2078	-0.0001	130.8509	79.5574	1.7032	788
GunPointAgeSpan	0.01	2	135	150	1418.4	228.04	9387524360	7.487	411631059	0.9596	0.0006	0.0872	0.0726	15.095	38339.4	0.3956	77
GunPointMaleVersusF	0.073	2	135	150	1418.4	224.5	9426001732	8.3865	415618699	0.9671	0.0004	0.0755	-0.1545	12.8785	33216.2	0.3956	82

Tabella 4.4: Statistiche dataset univariato UCR

Name	CVIR	#C	TS	WS	max	min	energy	kurtosis	c3	A_M	A_V	entropy	slope	cid_ce	W_D	fft_ent	peaks
GunPointOldVersusY	0.062	2	136	150	1416.1	228.22	9000977443	8.371	389005277	0.971	0.0004	0.1	0.3009	11.6553	31379.4	0.4404	79
HouseTwenty	0.0	2	40	2000	24929.0	27.0	107255945317	5.2272	2377078496	0.7529	0.0074	0.1648	0.0195	132.2404	3105868	1.0965	445
InsectEPGRegularT	0.596	3	62	601	2.73	-0.5	84027.3349	9.33	5.4534	0.9996	0.0	0.0813	-0.0009	2.6265	0.0091	0.4256	105
InsectEPGSmallTrain	0.529	3	17	601	2.9	-0.47	24257.604	8.6447	5.8839	0.9986	0.0	0.0024	-0.0032	2.4128	0.038	0.4297	31
MelbournePedestrian	0.014	10	1194	24	9682.0	0.0	44207036724	6.9205	1616047518	0.4393	0.0719	0.507	0.0138	79.8312	39912754	0.9515	492
MixedShapesRegTrain	0.0	5	500	1024	3.66	-2.95	511500.0	12.721	-0.1061	0.9895	0.0001	0.0663	0.0	27.4994	0.0748	0.2264	2181
MixedShapesSTrain	0.0	5	100	1024	2.75	-2.89	102300.0	14.7899	-0.1195	0.9903	0.0001	0.0663	0.0	11.0196	0.0627	0.2264	430
PickupGestWiimoteZ	0.0	10	50	361	2.62	-0.35	6499.0414	6.0493	0.35	0.9104	0.0024	0.1705	-0.0002	22.7213	0.2745	0.612	87
PigAirwayPressure	0.0	52	104	2000	17.97	-0.91	6685954.5354	10.37	307.9725	0.9952	0.0	0.0136	0.0001	16.508	0.5154	0.2607	1604
PigArtPressure	0.0	52	104	2000	119.92	8.88	974091435.9	15.7607	339474.84	0.9681	0.0006	0.0812	0.0002	26.6318	64.2847	0.3848	831
PigCVP	0.0	52	104	2000	73.11	-79.01	3756355.4243	9.9578	103.6822	0.8683	0.0045	0.1496	-0.0001	74.2824	10.5806	1.2482	1159
PLAID	0.721	11	537	1344	460.85	-22.06	197558761.5	4.9033	37372.59	0.9416	0.0005	0.0382	0.0001	177.956	164.7381	0.6685	4266
PowerCons	0.0	2	180	144	7.8	0.04	62815.9979	6.2598	4.7483	0.6928	0.0184	0.5007	0.0001	54.3243	5.4103	1.4704	228
Rock	0.0	4	20	2844	94.06	0.0	89551837.3	12.6983	102637.1	0.9969	0.0	0.0153	-0.001	6.7499	26.4742	0.3507	210
SemgHandGenderCh2	0.0	2	300	1500	2238.57	0.01	332961563.7	15.5313	29140.4125	0.5884	0.0003	0.8128	0.0	579.6009	604.6	2.2538	9686
SemgHandMovemCh2	0.0	6	450	1500	2238.57	0.01	501215824.9	16.2896	30249.344	0.6045	0.0003	1.0333	-0.0	696.4204	557.80	2.2016	14565
SemgHandSubjectCh2	0.0	5	450	1500	2238.57	0.01	509674234.5	15.7475	31006.3991	0.6052	0.0003	1.2604	-0.0	695.5292	561.27	2.1427	14537
ShakeGestWiimoteZ	0.0	10	50	385	4.15	-4.58	8770.0745	7.8816	0.2913	0.6412	0.0437	0.3264	-0.0	35.704	2.535	1.1743	221
SmoothSubspace	0.0	3	150	15	1.04	0.0	815.1021	13.3748	0.1588	0.0419	0.0048	1.6731	-0.0	61.0596	0.0978	2.5474	41
UMD	0.0	3	36	150	1.5	-1.37	2071.4409	13.8966	0.0638	0.8967	0.0043	0.142	0.0003	8.7724	0.8865	0.6752	40

Tabella 4.5: Statistiche dataset multivariato UEA

Name	#C	T_S	W_S	#ft	CVIR	max	min	mean	std	energy	kurtosis	c3	aut_m	aut_v	slope	cid_ce	w_d	fft_ent	peaks
ArticularyWordRec	25	275	144	9	0.0	4.8887	-5.0996	0.5291	1.1183	39325.0014	5.6712	-0.0366	0.8222	0.0048	-0.0	64.7887	1.2702	0.2013	4641.1111
AtrialFibrillation	3	15	640	2	0.0	2.4849	-2.2195	-0.0007	0.2479	446.7492	10.934	0.0009	0.2041	0.0772	-0.0	54.7539	0.3954	1.0188	625.0
BasicMotions	4	40	100	6	0.0	34.8662	-27.822	-0.0953	2.0388	92113.576	16.5936	3.1563	0.0477	0.063	-0.0117	63.8113	32.602	1.0864	1304.6667
CharacterTrajectories	20	1422	182	3	0.126	4.3051	-4.2832	0.292	0.446	176971.6626	12.5134	-0.3506	0.8204	0.0176	0.0	69.236	1.0334	0.3207	7951.0
Cricket	12	108	1197	6	0.0	10.682	-7.6967	-0.2369	0.6124	129167.9755	13.4086	0.2044	0.8962	0.0041	0.0	62.9574	1.4919	0.2799	9300.6667
DuckDuckGeese	5	50	270	1345	0.0	499.18	0.0	0.3647	1.2577	83469.1217	9.3892	81.0744	0.433	0.016	0.0	83.0649	21.4104	0.897	303646.3
EigenWorms	5	128	17984	6	0.471	1042.31	-455.32	0.0614	5.0923	67557230.2749	6.737	8999.3252	0.9832	0.0001	-0.0	108.9223	6.7685	0.1834	126882.16
Epilepsy	4	137	206	3	0.096	3.59	-3.59	-0.1039	0.837	22004.7336	14.5747	-0.0808	0.4293	0.0188	-0.0001	130.5173	0.3908	0.806	3285.6667
ERing	6	30	65	4	0.0	2.5979	-3.1144	-0.1606	1.0156	1920.0	7.3184	0.0857	0.6451	0.0428	0.0004	12.1435	4.7767	0.425	73.75
EthanolConcentr	4	261	1751	3	0.008	65531.0	-111.45	1271.2838	762.6305	1014858955685	37.9882	6571115041	0.9994	0.0	-0.0253	4.9758	48932.39	0.0907	3079.0
FaceDetection	2	5890	62	144	0.0	24.3269	-24.3278	0.0	0.9992	364560.0	10.2639	0.0398	0.2632	0.0192	0.0	578.0092	3.9029	1.0362	3042453.1
FingerMovements	2	316	50	28	0.009	205.1	-178.6	30.2825	44.1264	44605701.9921	4.6042	177656.5818	0.7339	0.0126	0.0024	59.3424	7297.90	0.311	6941.8214
HandMovementDir	4	160	400	10	0.0	935.516	-758.34	0.9294	167.7923	1805704606	7.7879	129373.3645	0.5353	0.0275	0.0017	110.5256	134037.6	0.6442	21040.7
Handwriting	26	150	152	3	0.509	9.2384	-10.642	0.0	0.9967	22649.9905	8.1731	-0.0475	0.1299	0.1438	-0.0	91.1741	19.7094	0.8899	1551.3333
Heartbeat	2	204	405	61	0.624	33.662	0.0	0.0824	0.2085	17997.9855	8.633	0.4951	0.4678	0.0274	-0.0	153.3212	1.4262	1.1729	72516.24
JapaneseVowels	9	270	29	12	0.0	2.2031	-1.8528	-0.0113	0.2192	788.815	6.1421	0.0177	0.396	0.0896	0.0001	42.3631	0.909	0.2371	1267.9167
Libras	15	180	45	2	0.0	1.0	0.0	0.5143	0.1788	2402.8164	6.195	0.1654	0.6222	0.0479	0.0001	28.4583	0.193	0.443	55.0
LSST	14	2459	36	6	1.637	64062.0	-40036.0	33.5317	855.78	76589611827	7.8625	8581408571.8	0.4108	0.0286	-0.028	188.43	3537221	0.8256	18669.5
MotorImagery	2	278	3000	64	0.0	77.5938	-42.4375	0.6425	9.3708	83409938.8459	9.7756	862.1069	0.9676	0.0004	0.0	71.5202	45.4898	0.2543	421920.95
NATOPS	6	180	51	24	0.0	2.5364	-2.7678	-0.3918	0.5474	8468.3917	7.4582	-0.5895	0.666	0.0452	-0.0	23.71	2.4309	0.4524	1033.25
PEMS-SF	7	267	144	963	0.136	1.0	0.0	0.0531	0.0444	195.5323	5.0081	0.0007	0.8009	0.0081	-0.0	66.6227	0.0053	0.2583	328674.6
PenDigits	10	7494	8	2	0.042	100.0	0.0	50.702	34.5256	225638392.5	20.9246	111182.6344	-0.01	0.0611	0.0	262.4108	1884.18	0.782	2629.5
PhonemeSpectra	39	3315	217	11	0.0	126.76	0.0	1.2791	1.241	13831183.3996	6.2859	230.1658	0.5123	0.0375	0.0	435.4414	59.1951	0.8379	115146.4
RacketSports	4	151	30	6	0.105	34.8742	-34.8715	0.008	5.7536	168123.484	19.3097	6.0571	0.0506	0.0158	0.0053	77.5095	137.0029	1.6493	1608.5
SelfRegulationSCP1	2	268	896	6	0.011	125.84	-74.69	25.9155	23.4156	292986704.0458	7.9975	64481.0315	0.9521	0.0008	-0.0011	51.125	690.5749	0.3248	5771.5
SelfRegulationSCP2	2	200	1152	7	0.0	113.88	-45.72	16.73	13.892	108976466.9388	9.5932	13558.5634	0.9077	0.0033	-0.0001	64.6454	651.642	0.4707	7293.5714
SpokenArabicDigits	10	6599	93	13	0.0	9.3416	-12.957	-0.1896	0.8371	770637.2731	7.829	-0.776	0.4858	0.0424	-0.0	422.7551	4.8774	0.4764	153732.4
StandWalkJump	3	12	2500	4	0.0	5.76	-3.61	0.008	0.4946	7511.0817	25.642	0.2927	0.7836	0.0253	0.0	24.0954	1.6167	0.4766	1180.75
UWaveGestureLib	8	120	315	3	0.0	4.1372	-3.3944	-0.0	0.9984	37680.0021	9.5376	0.053	0.943	0.0016	0.0	18.9583	0.7999	0.2006	602.6667
InsectWingbeat	10	25000	22	200	0.0	106.5366	-116.08	0.0002	0.2908	189998.2959	31.2676	0.0001	-0.0004	0.0029	0.0	1048.4913	0.6715	1.6895	3252326.7

4.5 Risultati classificazione

In questa sezione vengono riportati i risultati ottenuti su UCR ed UEA dalle tre architetture proposte. Oltre alla baseline TS2Vec sono stati riportati i risultati di altri modelli utili a fini di benchmark. Ognuno di questi modelli presenta delle somiglianze con TS2Vec, le uniche eccezioni sono il metodo basato su trasformazione TST [95] e il metodo DTW (Dynamic time Warping) che non è basato su rete neurale ma sulla distanza, come descritto in precedenza nella sottosezione 2.2.1.

La rete T-Loss [84] è un metodo contrastive dove le coppie di campioni positivi sono generate dal campionamento casuale di sotto sequenze della serie in analisi. Anche la rete TNC [83] è basata sul contrastive learning e definisce il concetto di vicinato di un segnale basandosi sull'informazione a priori di similarità, incoraggia la rete a mantenere costante la rappresentazione per un certo vicinato di segnali e allontanarla da quella di altri segnali. L'ultimo metodo contrastive è TS-TCC [96] in cui si applica un task di predizione cross-view sulle coppie aumentate dei dati.

	Migliore		Uguale	Peggiore	
UCR	%dataset	+avg%	%dataset	%dataset	-avg%
MLP	35.48%	+2.42%	11.29%	53.22%	-3.71%
FCN	33.87%	+2.65%	17.74%	48.38%	-4.57%
RESNET	21.77%	+2.75%	9.67%	68.54%	-6.49%

Tabella 4.6: Confronto architetture proposte e TS2Vec per dataset UCR. *%dataset* indica il rapporto dei dataset in cui l'accuratezza è maggiore, uguale o inferiore a quella di TS2Vec, rispetto ai 124 dataset dell'archivio UCR. ± avg% indica l'incremento medio dell'accuratezza rispetto a TS2Vec.

Commenti classificazione UCR rispetto a TS2Vec La Tabella 4.6 riassume i risultati del confronto tra le accuratezze ottenute dalle tre reti proposte e la baseline TS2Vec per quanto riguarda l'archivio UCR. Si nota che le tre reti superano l'accuratezza di TS2Vec sul 35.48%, 33.87%, 21.77% dei dataset rispettivamente per MLP, FCN e ResNet, con un incremento dell'accuratezza media del 2.5%. Nel caso in cui le performance sono peggiori della baseline notiamo che ResNet è la rete che peggiora sul maggior numero

di dataset e con un peggioramento medio del -6.49%, a seguire MLP con un peggioramento medio del -3.71% e FCN con un peggioramento medio del -4.57%. Si nota che, anche se apparentemente MLP sembra essere migliore di FCN, se si considera la colonna *Uguale*, che riporta la percentuale di dataset in cui le accuratezze coincidono, FCN è maggiore rispetto a MLP. Quindi considerando l'insieme delle colonne *Uguale* e *Migliore* possiamo affermare che FCN presenta risultati migliori o comparabili con la baseline, su un numero maggiore di dataset rispetto a MLP e ResNet.

Commenti classificazione UEA rispetto a TS2Vec Confrontando i risultati della Tabella 4.7 la prima cosa che si nota è che il numero di dataset in cui le accuratezze sono uguali è diminuito in maniera sensibile per tutte le tre soluzioni proposte. Un altro aspetto importante è che nel caso dei dataset in cui si performa meglio rispetto alla baseline, l'accuratezza media è aumentata di un fattore tre mentre l'accuratezza media nei dataset con peformance minori ha subito un incremento molto lieve. Anche in questa analisi la rete FCN supera MLP se tra i dataset consideriamo anche quelli in cui le performance sono uguali. Ad ogni modo basta guardare le accuratezze medie dei due metodi per notare che il miglioramento medio di FCN è maggiore rispetto a MLP e che il peggioramento medio di FCN è minore di MLP. Questo è sufficiente a concludere che FCN sembra essere migliore. Un ultimo aspetto di grande importanza riguarda la ResNet. Come per il caso di UCR, anche con UEA le perfomance sono le più sfavorevoli sia nella colonna *Migliore* che in quella *Peggiore*. Sono sempre meno i dataset in cui migliora e sempre di più quelli in cui peggiora. Questa osservazione fornisce un indizio sulle scarse qualità di questa rete.

	Migliore		Uguale	Peggiore	
UEA	%dataset	+avg%	%dataset	%dataset	-avg%
MLP	40%	+6.78%	0%	60%	-5.72%
FCN	40%	+7.36%	3.33%	56.66%	-5.21%
RESNET	26.66%	+7.28%	6.66%	66.66%	-6.81%

Table 4.7 continued from previous page

	Migliore	Uguale	Peggiore

Tabella 4.7: Confronto architetture proposte e TS2Vec per dataset UEA. *%dataset* indica il rapporto dei dataset in cui l'accuratezza è maggiore, uguale o inferiore a quella di TS2Vec, rispetto ai 30 dataset dell'archivio UEA. ± avg% indica l'incremento medio dell'accuratezza rispetto a TS2Vec.

Commenti classificazione UEA-UCR rispetto a tutti i benchmark Nella Tabella 4.8 sono state riportate le percentuali di dataset in cui le reti proposte superano tutti gli altri metodi. Considerando il caso UCR, la rete MLP , contrariamente ad ogni aspettativa ha un'accuratezza migliore sul 15.32% dei dataset, a seguire FCN con 11.29% e infine ResNet con 4.83% . Per quanto riguarda UEA la classifica si riordina secondo i risultati riscontrati precedentemente, ponendo al primo posto la rete FCN con un numero di dataset pari al 26.6% , segue MLP al 20% e ResNet al 10%. E' da notare che anche in questo caso, le percentuali dei dataset su cui si ha un miglioramento per l'archivio UEA, in generale sono aumentate rispetto al caso univariato. Per le reti FCN e ResNet sono più che raddoppiate. Questo risultato è in accordo con le accuratezze medie riscontrate nelle tabelle precedentemente analizzate. Includendo nel confronto anche TS2Vec si nota che la percentuale di dataset su cui predomina è invariata per entrambi gli archivi.

UCR	%dataset	UEA	%dataset
MLP	15.32%	MLP	20%
FCN	11.29%	FCN	**26.66%**
RESNET	4.83%	RESNET	10%
TS2VEC	**20.16%**	TS2VEC	20%

Tabella 4.8: Confronto architetture proposte con tutti i modelli di benchmark. *%dataset* indica il rapporto dei dataset in cui l'accuratezza è maggiore rispetto a tutte le altre.

Analisi dei dataset Per comprendere al meglio le accuratezze, i dataset sono stati analizzati con le misure descritte nella sezione 2.4. Le statistiche sono state utilizzate per misurarne la correlazione con lo skew delle accuratezze di ogni architettura proposta.

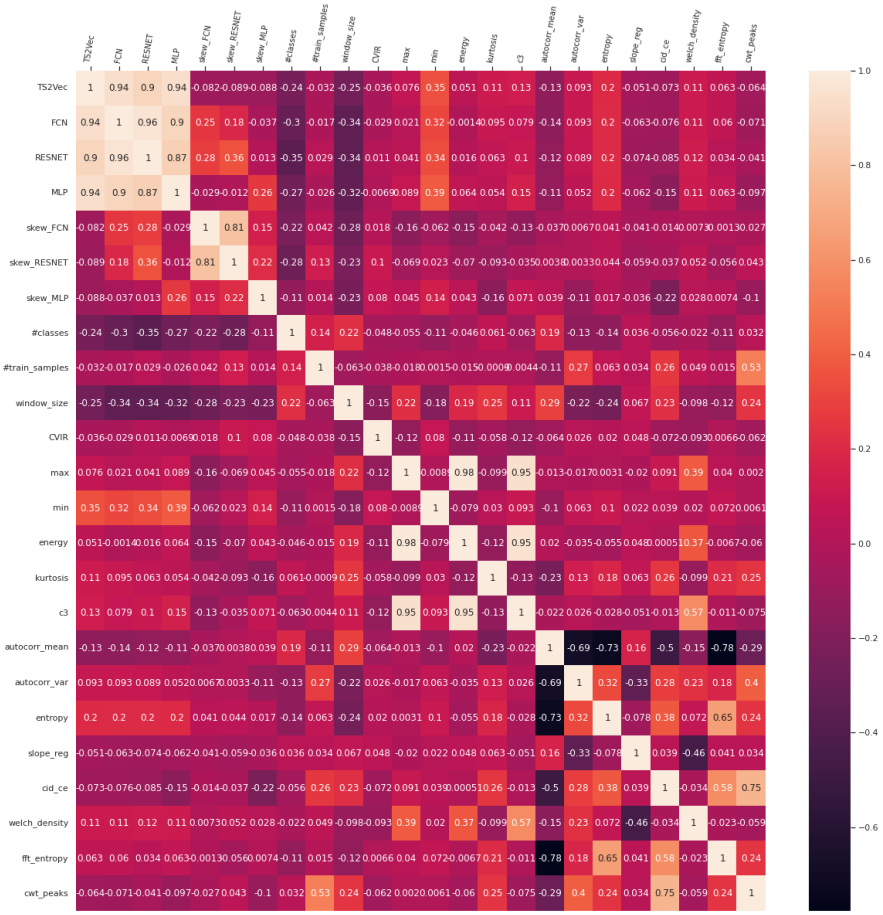


Figura 4.1: UCR matrice di correlazione

Lo skew è inteso come lo scarto tra il valore di accuratezza di una delle tre architetture rispetto al valore di baseline della rete TS2Vec.

I risultati sono mostrati in Figura 4.1 e Figura 4.2. In generale non sono emersi valori di correlazione forti abbastanza da poter giustificare le accuratezze ottenute. Considerando il caso univariato si nota una lieve correlazione negativa con il numero di classi per tutte e tre le architetture proposte. Allo stesso modo per il caso multivariato si nota una lieve

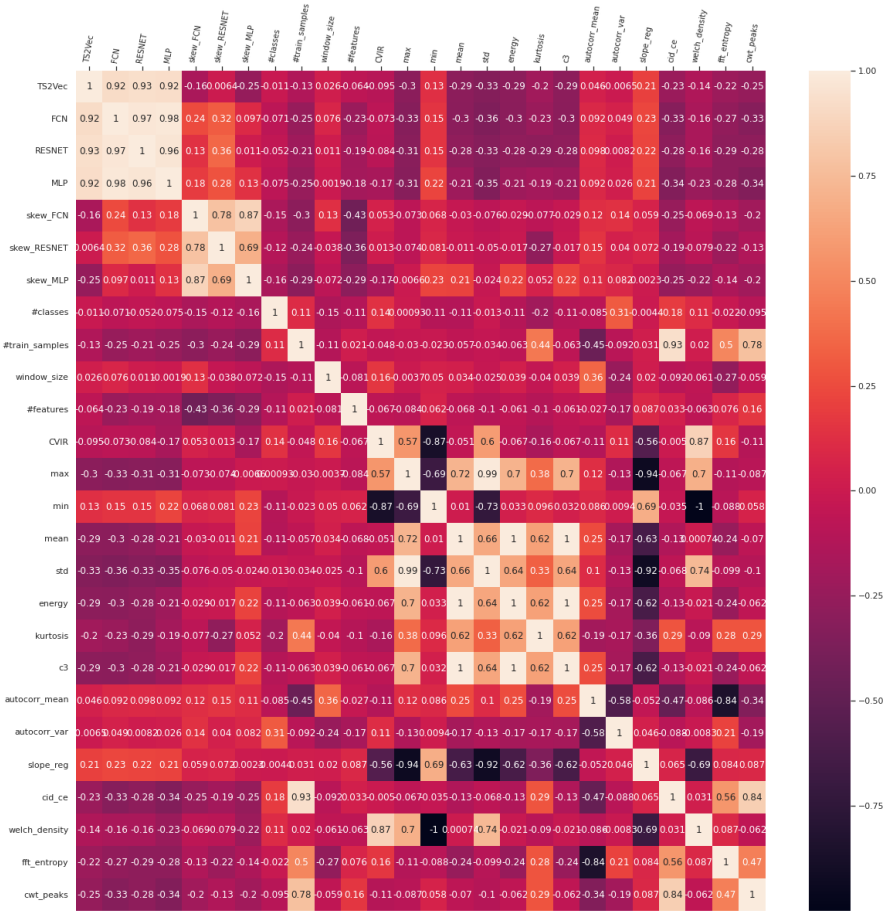


Figura 4.2: UEA matrice di correlazione

correlazione negativa con il numero di features della serie temporale.

Analizzando i dataset sulla base di queste due caratteristiche e tenendo in considerazione le accuratezze ho notato che alcuni dataset di entrambi gli archivi sarebbero da considerare outliers, in quanto presentano delle caratteristiche molto diverse rispetto alla media.

Nei lavori di classificazione di serie temporali [58, 54] gli esperimenti sono stati condotti

sugli stessi archivi UEA-UCR selezionando a priori un sottoinsieme omogeneo dei dataset disponibili. Ispirato da questa idea ho deciso di condurre un'analisi equivalente sui risultati ottenuti escludendo alcuni dataset sulla base dei due parametri trovati dalla matrice di correlazione.

In particolare per l'archivio univariato sono stati esclusi i dataset che presentano un numero di classi molto elevato (maggiori di 15 classi) e ho riscontrato i dati ottenuti nella Tabella 4.10.

Per l'archivio multivariato sono stati esclusi i dataset che presentano un numero di faetures superiore a 100 (*DuckDuckGeese, PEMS-SF, InsectWingbeat, FaceDetection*). Questi dataset hanno un numero di caratteristiche (rispettivamente di 1345, 963, 200, 144) che non rappresenta una casistica reale. Le accuratezze su questi dataset si discostano molto, sia in negativo che in positivo, dalla media dei risultati. Un numero cosi elevato di features vanifica l'intenzione di mappare la serie in una rappresentazione a maggior dimensionalità, la dimensione dei filtri convolutivi utilizzati non è adatta a gestire questi casi. Nei dataset con poche features il mapping avviene in uno spazio a dimensionalità superiore mentre in questi casi la dimensionalità si riduce (ad esempio si passa da 1345 features a 128 dei livelli latenti) causando una perdita di informazioni. Con queste considerazioni non si vuole affermare che la rete performa bene solo su dataset con pochissime caratteristiche, infatti i dataset *MotorImagery* e *HeartBeat* sono due esempi di dataset con circa 60 features e con accuratezze superiori alla baseline.

Escludendo questi dataset dall'analisi e calcolando l' accuratezza media per ogni rete proposta e per la rete di baseline si ottengono i risultati della Tabella 4.9. In accordo con quanto precedentemente discusso, la rete FCN raggiunge i risultati migliori, superando la baseline dell' **1.4%**. La rete MLP presenta risultati equivalenti alla baseline mentre ResNet si conferma la rete peggiore.

	TS2Vec	FCN	RESNET	MLP
UEA**	0.722	**0.736**	0.706	0.724
UEA	0.704	**0.704**	0.678	0.696

Table 4.9 continued from previous page

	TS2Vec	FCN	RESNET	MLP

Tabella 4.9: Confronto accuratezze tra UEA** ed UEA. UEA** esclude i quattro dataset critici per faeftures *DuckDuckGeese, InsectWingbeat, PEMS-SF, FaceDetection*

	TS2Vec	FCN	RESNET	MLP
UCR**	**0.845**	0.835	0.813	0.834
UCR	**0.830**	0.817	0.791	0.819

Tabella 4.10: Confronto accuratezze tra UCR** ed UCR. UCR** esclude i dataset con numero di classi superiore a 15

Name	MLP	FCN	RESNET	TS2Vec	T-Loss	TNC	TS-TCC	TST	DTW
EthanolLevel	0.542	0.692	0.714	0.468	0.434	0.392	0.376	0.298	0.276
FreezerSmallTrain	0.868	0.985	0.925	0.87	0.956	0.991	0.989	0.922	0.753
Beef	0.733	0.833	0.766	0.767	0.667	0.733	0.6	0.5	0.633
Wine	0.777	0.925	0.925	0.87	0.992	0.994	0.994	0.991	0.574
GestureMidAirD1	0.63	0.661	0.592	0.608	1.0	0.527	0.753	0.366	0.569
GestureMidAirD2	0.538	0.515	0.438	0.469	0.608	0.431	0.369	0.208	0.608
GestureMidAirD3	0.361	0.338	0.369	0.292	0.546	0.362	0.254	0.138	0.323
FaceAll	0.776	0.812	0.8	0.771	0.707	0.7	0.686	0.676	0.808
Ham	0.704	0.752	0.752	0.714	0.98	0.967	0.993	0.827	0.467
EOGHorizontalSignal	0.486	0.577	0.52	0.539	0.722	0.738	0.742	0.71	0.503
ItalyPowerDemand	0.961	0.96	0.956	0.925	0.597	0.549	0.415	0.266	0.95
LargeKitchenAppliances	0.824	0.877	0.829	0.845	0.954	0.928	0.955	0.845	0.795
SmallKitchenAppliances	0.749	0.762	0.744	0.731	0.848	0.788	0.773	0.733	0.643
MiddlePhalanxTW	0.597	0.61	0.59	0.584	0.656	0.643	0.63	0.617	0.506
DistalPhalanxOutlineCorrect	0.764	0.786	0.775	0.761	0.775	0.754	0.754	0.728	0.717
MiddlePhalanxOutlineAgeG	0.662	0.655	0.649	0.636	0.825	0.818	0.818	0.753	0.5
ToeSegmentation1	0.934	0.934	0.872	0.917	0.987	1.0	0.99	0.49	0.772
UWaveGestureLibraryAll	0.927	0.946	0.931	0.93	0.757	0.721	0.69	0.655	0.892
FiftyWords	0.756	0.786	0.751	0.771	0.884	0.789	0.863	0.543	0.69
DistalPhalanxOutlineAgeG	0.741	0.741	0.741	0.727	0.727	0.741	0.755	0.741	0.77
DistalPhalanxTW	0.683	0.712	0.676	0.698	0.676	0.669	0.676	0.568	0.59
InlineSkate	0.394	0.429	0.443	0.415	0.594	0.594	0.594	0.594	0.384
UWaveGestureLibraryX	0.797	0.809	0.805	0.795	0.999	1.0	0.999	0.466	0.728
Chinatown	0.982	0.979	0.988	0.965	0.993	0.973	0.933	0.76	0.957
PowerCons	0.983	0.972	0.977	0.961	0.555	0.495	0.445	0.419	0.878
ECG200	0.94	0.93	0.87	0.92	0.94	0.83	0.88	0.83	0.77

Table 4.11 continued from previous page

Name	MLP	FCN	RESNET	TS2Vec	T-Loss	TNC	TS-TCC	TST	DTW
HandOutlines	0.921	0.932	0.924	0.922	0.724	0.752	0.743	0.524	0.881
ECG5000	0.94	0.945	0.941	0.935	0.933	0.937	0.941	0.928	0.924
FreezerRegularTrain	0.989	0.995	0.982	0.986	0.382	0.424	0.486	0.26	0.899
Computers	0.66	0.668	0.676	0.66	0.664	0.684	0.704	0.696	0.7
CricketZ	0.823	0.8	0.715	0.792	0.708	0.682	0.713	0.403	0.754
TwoLeadECG	0.991	0.992	0.956	0.986	0.99	1.0	1.0	1.0	0.905
SemgHandMovementCh2	0.821	0.866	0.817	0.86	0.89	0.882	0.837	0.725	0.584
Mallat	0.929	0.919	0.855	0.914	0.795	0.767	0.685	0.411	0.934
ProximalPhalanxOutlineAgeG	0.853	0.839	0.853	0.834	0.859	0.866	0.873	0.77	0.805
MedicalImages	0.782	0.793	0.777	0.789	0.95	0.917	0.883	0.9	0.737
MoteStrain	0.856	0.865	0.784	0.861	0.591	0.571	0.61	0.506	0.835
MixedShapesRegularTrain	0.922	0.92	0.918	0.917	0.944	0.942	0.949	0.741	0.842
ScreenType	0.408	0.413	0.426	0.411	0.515	0.565	0.563	0.483	0.397
Strawberry	0.962	0.964	0.964	0.962	0.964	0.968	0.967	0.949	0.941
NonInvasiveFetalECGThorax2	0.901	0.939	0.932	0.938	0.878	0.898	0.898	0.471	0.865
MixedShapesSmallTrain	0.855	0.862	0.863	0.861	0.905	0.911	0.855	0.879	0.78
BeetleFly	0.8	0.9	0.85	0.9	0.8	0.85	0.8	1.0	0.7
BirdChicken	0.95	0.8	0.75	0.8	0.85	0.75	0.65	0.65	0.75
CBF	1.0	1.0	1.0	1.0	0.983	0.983	0.998	0.898	0.997
Coffee	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.821	1.0
CricketX	0.8	0.782	0.705	0.782	0.713	0.623	0.731	0.385	0.754
Earthquakes	0.748	0.748	0.748	0.748	0.748	0.748	0.748	0.748	0.719
ECGFiveDays	1.0	1.0	0.997	1.0	1.0	0.999	0.878	0.763	0.768
GunPoint	0.986	0.98	0.98	0.98	0.793	0.733	0.815	0.507	0.907
NonInvasiveFetalECGThorax1	0.912	0.93	0.914	0.93	0.851	0.825	0.843	0.768	0.79
OliveOil	0.833	0.9	0.9	0.9	0.919	0.912	0.913	0.832	0.833
Plane	0.971	1.0	1.0	1.0	0.276	0.18	0.242	0.139	1.0
ShapeletSim	1.0	1.0	0.944	1.0	0.416	0.509	0.419	0.419	0.65
StarLightCurves	0.969	0.969	0.966	0.969	0.889	0.834	0.907	0.745	0.907
Trace	1.0	1.0	1.0	1.0	0.9	0.831	0.877	0.615	1.0
TwoPatterns	1.0	1.0	1.0	1.0	0.999	0.993	0.976	0.871	1.0
BME	0.989	0.993	0.98	0.993	0.723	0.646	0.689	0.447	0.9
GunPointAgeSpan	0.996	0.987	0.996	0.987	0.899	0.316	0.43	0.38	0.918
GunPointMaleVersusFemale	1.0	1.0	1.0	1.0	0.994	0.984	0.994	0.991	0.997
GunPointOldVersusYoung	1.0	1.0	1.0	1.0	0.997	0.994	0.997	1.0	0.838
InsectEPGRegularTrain	1.0	1.0	1.0	1.0	0.933	0.782	0.79	0.815	0.872
InsectEPGSmallTrain	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.735
ShakeGestureWiimoteZ	0.944	0.94	0.86	0.94	0.853	0.771	0.753	0.484	0.86
CricketY	0.741	0.748	0.725	0.749	0.728	0.597	0.718	0.467	0.744
DiatomSizeReduction	0.99	0.983	0.947	0.984	0.984	0.993	0.977	0.961	0.967
Fish	0.937	0.925	0.88	0.926	0.732	0.653	0.653	0.525	0.823
Wafer	0.997	0.997	0.997	0.998	0.896	0.903	0.692	0.475	0.98

Table 4.11 continued from previous page

Name	MLP	FCN	RESNET	TS2Vec	T-Loss	TNC	TS-TCC	TST	DTW
SwedishLeaf	0.924	0.939	0.923	0.941	0.954	0.951	0.965	0.916	0.792
SyntheticControl	0.996	0.993	0.983	0.997	0.963	0.885	0.916	0.786	0.993
UWaveGestureLibraryY	0.711	0.714	0.723	0.719	0.785	0.781	0.733	0.569	0.634
InsectWingbeatSound	0.622	0.624	0.586	0.63	0.371	0.378	0.347	0.287	0.355
WordSynonyms	0.681	0.67	0.625	0.676	0.815	0.759	0.778	0.5	0.649
SmoothSubspace	0.978	0.973	0.953	0.98	0.92	0.82	0.86	0.76	0.827
UMD	0.66	0.993	0.986	1.0	0.96	0.913	0.953	0.827	0.993
Crop	0.741	0.748	0.75	0.756	0.951	0.977	0.983	0.936	0.665
SonyAIBORobotSurface2	0.903	0.862	0.836	0.871	0.902	0.804	0.899	0.724	0.831
AllGestureWiimoteZ	0.722	0.737	0.648	0.746	0.726	0.699	0.741	0.423	0.643
FacesUCR	0.93	0.913	0.87	0.924	0.92	0.659	0.773	0.511	0.905
SemgHandSubjectCh2	0.866	0.94	0.924	0.951	0.789	0.593	0.613	0.42	0.727
Symbols	0.952	0.963	0.932	0.976	0.914	0.88	0.923	0.738	0.95
Worms	0.727	0.688	0.675	0.701	0.691	0.63	0.531	0.422	0.584
Yoga	0.882	0.872	0.857	0.887	0.792	0.727	0.753	0.584	0.837
MelbournePedestrian	0.948	0.943	0.94	0.959	1.0	1.0	1.0	1.0	0.791
Lightning2	0.819	0.852	0.803	0.869	0.789	0.776	0.848	0.595	0.869
ProximalPhalanxOutlineCorr	0.89	0.869	0.852	0.887	0.99	1.0	1.0	0.933	0.784
Adiac	0.785	0.741	0.741	0.762	0.675	0.726	0.767	0.55	0.604
PhalangesOutlinesCorrect	0.804	0.787	0.792	0.809	0.76	0.723	0.723	0.545	0.728
ToeSegmentation2	0.93	0.869	0.846	0.892	0.939	0.864	0.93	0.807	0.838
UWaveGestureLibraryZ	0.757	0.747	0.735	0.77	0.71	0.697	0.641	0.348	0.658
EOGVerticalSignal	0.397	0.48	0.441	0.503	0.605	0.442	0.401	0.373	0.448
ProximalPhalanxTW	0.8	0.8	0.795	0.824	0.844	0.854	0.839	0.854	0.761
SemgHandGenderCh2	0.956	0.938	0.953	0.963	0.58	0.58	0.6	0.68	0.802
WormsTwoClass	0.766	0.779	0.766	0.805	0.727	0.623	0.753	0.455	0.623
Haptics	0.503	0.496	0.467	0.526	0.922	0.93	0.724	0.735	0.377
ACSF1	0.58	0.87	0.71	0.9	0.837	0.812	0.791	0.83	0.64
AllGestureWiimoteX	0.758	0.747	0.7	0.777	0.9	0.73	0.73	0.76	0.716
MiddlePhalanxOutlineCorrect	0.828	0.807	0.814	0.838	0.75	0.754	0.747	0.632	0.698
FordA	0.932	0.903	0.905	0.936	0.891	0.817	0.817	0.72	0.555
Meat	0.933	0.916	0.933	0.95	0.951	0.871	0.922	0.713	0.933
FordB	0.786	0.759	0.761	0.794	0.928	0.902	0.93	0.568	0.62
Lightning7	0.821	0.821	0.8	0.863	0.869	0.869	0.836	0.705	0.726
SonyAIBORobotSurface1	0.885	0.861	0.841	0.903	0.677	0.725	0.691	0.592	0.725
PLAID	0.512	0.519	0.506	0.561	0.788	0.649	0.615	0.596	0.84
Phoneme	0.303	0.266	0.218	0.312	0.784	0.787	0.804	0.773	0.228
Herring	0.625	0.593	0.593	0.641	0.49	0.474	0.396	0.357	0.531
AllGestureWiimoteY	0.778	0.744	0.72	0.793	0.763	0.703	0.697	0.259	0.729
PigCVP	0.58	0.75	0.58	0.812	0.928	0.808	0.524	0.774	0.154
RefrigerationDevices	0.584	0.525	0.541	0.589	0.771	0.81	0.8	0.78	0.464
ShapesAll	0.89	0.836	0.825	0.902	0.672	0.589	0.683	0.489	0.768

Table 4.11 continued from previous page

Name	MLP	FCN	RESNET	TS2Vec	T-Loss	TNC	TS-TCC	TST	DTW
ChlorineConcentration	0.761	0.764	0.746	0.832	0.749	0.76	0.753	0.562	0.648
PigArtPressure	0.971	0.894	0.802	0.966	0.51	0.413	0.38	0.12	0.245
ArrowHead	0.834	0.777	0.748	0.857	0.766	0.703	0.737	0.771	0.703
CinCECGTorso	0.794	0.747	0.815	0.827	0.713	0.669	0.671	0.508	0.651
GesturePebbleZ1	0.901	0.837	0.552	0.93	0.285	0.292	0.177	0.154	0.791
Car	0.9	0.733	0.616	0.833	0.833	0.683	0.583	0.55	0.733
PickupGestureWiimoteZ	0.84	0.72	0.68	0.82	0.86	0.813	0.735	0.828	0.66
FaceFour	0.965	0.829	0.818	0.932	0.786	0.766	0.813	0.504	0.83
Fungi	0.994	0.844	0.67	0.957	0.933	0.982	0.979	0.92	0.839
HouseTwenty	0.949	0.798	0.798	0.916	1.0	1.0	1.0	1.0	0.924
OSULeaf	0.822	0.723	0.702	0.851	0.867	0.833	0.8	0.8	0.591
GesturePebbleZ2	0.898	0.727	0.632	0.873	0.919	0.378	0.395	0.5	0.671
Rock	0.66	0.5	0.5	0.7	0.9	0.933	0.961	0.911	0.6
PigAirwayPressure	0.581	0.341	0.355	0.63	0.74	0.62	0.6	0.24	0.106

Tabella 4.11: Le accuratezze delle tre architetture a confronto con altri metodi sull'archivio UCR

Name	MLP	FCN	RESNET	TS2Vec	T-Loss	TNC	TS-TCC	TST	DTW
AtrialFibrillation	0.466	0.4	0.2	0.2	0.133	0.133	0.267	0.067	0.200
PEMS-SF	0.849	0.867	0.867	0.682	0.676	0.699	0.734	0.74	0.711
EthanolConcentration	0.414	0.292	0.273	0.308	0.205	0.297	0.285	0.262	0.323
Heartbeat	0.775	0.77	0.741	0.683	0.741	0.746	0.751	0.746	0.717
StandWalkJump	0.533	0.6	0.333	0.467	0.333	0.4	0.333	0.267	0.200
MotorImagery	0.54	0.58	0.59	0.51	0.58	0.5	0.61	0.5	0.500
FaceDetection	0.53	0.541	0.538	0.501	0.513	0.536	0.544	0.534	0.529
BasicMotions	1.0	1.0	0.975	0.975	1.0	0.975	1.0	0.975	0.975
FingerMovements	0.49	0.49	0.57	0.48	0.58	0.47	0.46	0.56	0.530
SpokenArabicDigits	0.993	0.981	0.958	0.988	0.905	0.934	0.97	0.923	0.963
PhonemeSpectra	0.238	0.226	0.196	0.233	0.222	0.207	0.252	0.085	0.151
SelfRegulationSCP1	0.815	0.846	0.877	0.812	0.843	0.799	0.823	0.754	0.775
Epilepsy	0.963	0.963	0.956	0.964	0.971	0.957	0.957	0.949	0.964
PenDigits	0.988	0.982	0.976	0.989	0.981	0.979	0.974	0.56	0.977
CharacterTrajectories	0.991	0.99	0.988	0.995	0.993	0.967	0.985	0.975	0.989
Libras	0.861	0.844	0.822	0.867	0.883	0.817	0.822	0.656	0.870
UWaveGestureLibrary	0.9	0.865	0.837	0.906	0.875	0.759	0.753	0.575	0.903
RacketSports	0.848	0.894	0.842	0.855	0.855	0.776	0.816	0.809	0.803
ArticularyWordRecognition	0.976	0.926	0.936	0.987	0.943	0.973	0.953	0.977	0.987
HandMovementDirection	0.324	0.31	0.351	0.338	0.351	0.324	0.243	0.243	0.231
ERing	0.848	0.874	0.929	0.874	0.133	0.852	0.904	0.874	0.133
SelfRegulationSCP2	0.55	0.577	0.506	0.578	0.539	0.55	0.533	0.55	0.539

Table 4.12 continued from previous page

Name	MLP	FCN	RESNET	TS2Vec	T-Loss	TNC	TS-TCC	TST	DTW
JapaneseVowels	0.937	0.954	0.943	0.984	0.989	0.978	0.93	0.978	0.949
Cricket	0.916	0.958	0.902	0.972	0.972	0.958	0.917	1.0	1.000
Handwriting	0.458	0.538	0.495	0.515	0.451	0.249	0.498	0.225	0.286
EigenWorms	0.77	0.885	0.801	0.847	0.84	0.84	0.779	0.748	0.618
NATOPS	0.838	0.894	0.9	0.928	0.917	0.911	0.822	0.85	0.883
LSST	0.412	0.5	0.468	0.537	0.509	0.595	0.474	0.408	0.551
InsectWingbeat	0.291	0.291	0.291	0.466	0.156	0.469	0.264	0.105	-
DuckDuckGeese	0.38	0.28	0.28	0.68	0.65	0.46	0.38	0.62	0.600

Tabella 4.12: Le accuratezze delle tre architetture a confronto con altri metodi sull'archivio UEA

4.6 Analisi dei risultati di rilevamento anomalie

	Yahoo			KPI		
Rete	F-score	Prec.	Rich.	F-score	Prec.	Rich.
TS2Vec	0.74	0.73	0.76	0.67	0.93	0.53
FCN	0.5	0.62	0.42	0.67	0.85	0.55
ResNet	0.49	0.59	0.42	0.74	0.85	0.65
MLP	0.26	0.06	0.17	0.06	0.24	0.04

Tabella 4.13: Risultati rilevamento anomalie su dataset Yahoo, KPI

Capitolo 5

Conclusioni e sviluppi futuri

L'analisi su un vasto numero di dataset ha fornito una maggiore importanza statistica ai risultati ma come nei lavori di classificazione di serie temporali [54, 58], la scelta opportuna dei dataset offerti dagli archivi UCR-UEA è essenziale al fine di ottimizzare la misura di classificazione. Considerando tutti i dataset disponibili per il caso univariato la rete proposta con accuratezza media più alta, è inferiore rispetto alla baseline. La stessa considerazione vale per il caso multivariato quando si considerano tutti i dataset dell'archivio UEA. La matrice di correlazione non ha mostrato relazioni utili nell'individuare dipendenze tra lo skew delle reti, le statistiche dei dataset e le caratteristiche delle serie temporali. Nonostante ciò, è stata utile nel fornire un suggerimento sulle possibili statistiche del dataset da investigare. Per il caso univariato è stato analizzato il numero di classi mentre per il caso multivariato il numero di features del dataset. Dall'analisi del caso univariato è emerso che le reti proposte non presentano miglioramenti rispetto al modello di baseline. Nel caso multivariato le accuratezze medie tra la rete FCN e la baseline sono uguali. Se si considerano solamente 26 dei 30 dataset forniti da UEA, quindi escludendo i quattro dataset che sono meno rappresentativi nel descrivere un dominio applicativo reale, si nota che sia la baseline sia la rete FCN presentano accuratezze medie superiori. In particolare la rete proposta FCN supera la baseline del **1.4%** e si afferma come migliore soluzione proposta. In queste condizioni anche la rete MLP presenta un'accuratezza pari a quella della baseline.

Tra le soluzioni proposte, la rete che si è rivelata migliore è la FCN, seguita da MLP e

per finire ResNet. Le performance delle reti analizzate sono in accordo con [87]. Le rete convolutiva FCN si è rivelata migliore poichè l'utilizzo dei filtri considera le dipendenze tra diversi istanti di tempo. La rete ResNet è molto simile ad FCN poichè costituita dalla stessa architettura di base ma la profondità ha giocato un ruolo fondamentale nel penalizzare le performance di questa soluzione. Le reti profonde presentano una difficoltà maggiore nella regolarizzazione del modello causando il fenomeno dell'overfitting. Contro ogni aspettativa, la rete MLP presenta una buona performance negli esperimenti condotti e si è dimostrata superiore a ResNet. Come descritto in [87, 54] i livelli di Dropout e ReLu hanno favorito la capacità di generalizzazione della rete.

Futuri sviluppi I valori dei filtri convolutivi utilizzati in questo lavoro sono stati definiti in maniera empirica come suggerito in [87]. Dato che la rete FCN si è rivelata una scelta promettente, bisognerebbe investigare quale sia la miglior configurazione dei filtri (dimensione filtro, stride, padding, dilation) al fine di migliorare la capacità di classificazione. Quest'analisi potrebbe basarsi sullo studio presentato in [97].

Bibliografia

[1] *Machine Learning Algorithms A Review*. https://www.researchgate.net/publication/344717762_Machine_Learning_Algorithms_-A_Review (cit. alle pp. 1, 2).

[2] Òscar Lorente, Ian Riera e Aditya Rana. *Image Classification with Classic and Deep Learning Techniques*. 2021. DOI: 10.48550/ARXIV.2105.04895. URL: https://arxiv.org/abs/2105.04895 (cit. a p. 2).

[3] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang e Jie Tang. «Self-supervised Learning: Generative or Contrastive». In: *IEEE Transactions on Knowledge and Data Engineering* (2021), pp. 1–1. ISSN: 2326-3865. DOI: 10.1109/tkde.2021.3090866. URL: http://dx.doi.org/10.1109/TKDE.2021.3090866 (cit. alle pp. 2, 3, 6–11).

[4] *Unsupervised learning and data clustering*. https://towardsdatascience.com/unsupervised-learning-and-data-clustering-eeecb78b422a (cit. a p. 2).

[5] Osvaldo Simeone. *A Very Brief Introduction to Machine Learning With Applications to Communication Systems*. 2018. arXiv: 1808.02342 [cs.IT] (cit. a p. 2).

[6] Jinman Zhao, Sidharth Mudgal e Yingyu Liang. *Generalizing Word Embeddings using Bag of Subwords*. 2018. DOI: 10.48550/ARXIV.1809.04259. URL: https://arxiv.org/abs/1809.04259 (cit. a p. 3).

[7] Yoshua Bengio, Aaron Courville e Pascal Vincent. *Representation Learning: A Review and New Perspectives*. 2012. DOI: 10.48550/ARXIV.1206.5538. URL: https://arxiv.org/abs/1206.5538 (cit. a p. 5).

[8] Qiyang Hu, Attila Szabó, Tiziano Portenier, Matthias Zwicker e Paolo Favaro. *Disentangling Factors of Variation by Mixing Them*. 2017. DOI: 10.48550/ARXIV.1711.07410. URL: https://arxiv.org/abs/1711.07410 (cit. a p. 5).

[9] Richard Zhang, Phillip Isola e Alexei A. Efros. *Colorful Image Colorization*. 2016. arXiv: 1603.08511 [cs.CV] (cit. a p. 6).

[10] Gustav Larsson, Michael Maire e Gregory Shakhnarovich. *Colorization as a Proxy Task for Visual Understanding*. 2017. arXiv: 1703.04044 [cs.CV] (cit. a p. 6).

[11] Richard Zhang, Phillip Isola e Alexei A. Efros. *Split-Brain Autoencoders: Unsupervised Learning by Cross-Channel Prediction*. 2017. arXiv: 1611.09842 [cs.CV] (cit. a p. 6).

[12] Mehdi Noroozi e Paolo Favaro. *Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles*. 2017. arXiv: 1603.09246 [cs.CV] (cit. alle pp. 6, 12).

[13] Spyros Gidaris, Praveer Singh e Nikos Komodakis. *Unsupervised Representation Learning by Predicting Image Rotations*. 2018. arXiv: 1803.07728 [cs.CV] (cit. a p. 6).

[14] Mehdi Noroozi, Hamed Pirsiavash e Paolo Favaro. *Representation Learning by Learning to Count*. 2017. arXiv: 1708.06734 [cs.CV] (cit. a p. 6).

[15] Carl Doersch, Abhinav Gupta e Alexei A. Efros. *Unsupervised Visual Representation Learning by Context Prediction*. 2016. arXiv: 1505.05192 [cs.CV] (cit. a p. 6).

[16] Ishan Misra, C. Lawrence Zitnick e Martial Hebert. *Shuffle and Learn: Unsupervised Learning using Temporal Order Verification*. 2016. arXiv: 1603.08561 [cs.CV] (cit. a p. 6).

[17] *Machine Learning semi-supervised learning*. https://www.geeksforgeeks.org/ml-semi-supervised-learning/ (cit. alle pp. 7, 9).

[18] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville e Yoshua Bengio. *Generative Adversarial Networks*. 2014. arXiv: 1406.2661 [stat.ML] (cit. alle pp. 7, 10).

[19] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov e Quoc V. Le. *XLNet: Generalized Autoregressive Pretraining for Language Understanding*. 2019. DOI: 10.48550/ARXIV.1906.08237. URL: https://arxiv.org/abs/1906.08237 (cit. a p. 8).

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee e Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2018. DOI: 10.48550/ARXIV.1810.04805. URL: https://arxiv.org/abs/1810.04805 (cit. a p. 8).

[21] Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves e Koray Kavukcuoglu. *Conditional Image Generation with PixelCNN Decoders*. 2016. DOI: 10.48550/ARXIV.1606.05328. URL: https://arxiv.org/abs/1606.05328 (cit. a p. 8).

[22] Aaron van den Oord, Nal Kalchbrenner e Koray Kavukcuoglu. *Pixel Recurrent Neural Networks*. 2016. DOI: 10.48550/ARXIV.1601.06759. URL: https://arxiv.org/abs/1601.06759 (cit. a p. 8).

[23] Andrew Brock, Jeff Donahue e Karen Simonyan. *Large Scale GAN Training for High Fidelity Natural Image Synthesis*. 2018. DOI: 10.48550/ARXIV.1809.11096. URL: https://arxiv.org/abs/1809.11096 (cit. a p. 10).

[24] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou e Alexei A. Efros. *Image-to-Image Translation with Conditional Adversarial Networks*. 2016. DOI: 10.48550/ARXIV.1611.07004. URL: https://arxiv.org/abs/1611.07004 (cit. a p. 10).

[25] Gustav Larsson, Michael Maire e Gregory Shakhnarovich. *Learning Representations for Automatic Colorization*. 2016. DOI: 10.48550/ARXIV.1603.06668. URL: https://arxiv.org/abs/1603.06668 (cit. a p. 10).

[26] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell e Alexei A. Efros. «Context Encoders: Feature Learning by Inpainting». In: (2016). DOI: 10.48550/ARXIV.1604.07379. URL: https://arxiv.org/abs/1604.07379 (cit. a p. 10).

[27] Christian Ledig et al. *Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network*. 2016. DOI: 10.48550/ARXIV.1609.04802. URL: https://arxiv.org/abs/1609.04802 (cit. a p. 11).

[28] Phuc H. Le-Khac, Graham Healy e Alan F. Smeaton. «Contrastive Representation Learning: A Framework and Review». In: *IEEE Access* 8 (2020), pp. 193907–193934. DOI: 10.1109/ACCESS.2020.3031549 (cit. a p. 11).

[29] Aaron van den Oord, Yazhe Li e Oriol Vinyals. *Representation Learning with Contrastive Predictive Coding*. 2018. DOI: 10.48550/ARXIV.1807.03748. URL: https://arxiv.org/abs/1807.03748 (cit. alle pp. 11, 12).

[30] Ciwan Ceylan e Michael U. Gutmann. *Conditional Noise-Contrastive Estimation of Unnormalised Models*. 2018. DOI: 10.48550/ARXIV.1806.03664. URL: https://arxiv.org/abs/1806.03664 (cit. a p. 11).

[31] Carl Doersch, Abhinav Gupta e Alexei Efros. «Unsupervised Visual Representation Learning by Context Prediction». In: (mag. 2015). DOI: 10.1109/ICCV.2015.167 (cit. a p. 12).

[32] Longlong Jing e Yingli Tian. *Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey*. 2019. DOI: 10.48550/ARXIV.1902.06162 (cit. a p. 12).

[33] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler e Yoshua Bengio. *Learning deep representations by mutual information estimation and maximization*. 2018. DOI: 10.48550/ARXIV.1808.06670 (cit. a p. 12).

[34] Michael Tschannen, Josip Djolonga, Paul K. Rubenstein, Sylvain Gelly e Mario Lucic. *On Mutual Information Maximization for Representation Learning*. 2019. DOI: 10.48550/ARXIV.1907.13625 (cit. a p. 13).

[35] Mathilde Caron, Piotr Bojanowski, Armand Joulin e Matthijs Douze. *Deep Clustering for Unsupervised Learning of Visual Features*. 2018. DOI: 10.48550/ARXIV.1807.05520 (cit. a p. 13).

[36] Apurva Kalia, Dilip Krishnan e Soha Hassoun. *Contrastive Multiview Coding for Enzyme-Substrate Interaction Prediction*. 2021. DOI: 10.48550/ARXIV.2111.09467 (cit. a p. 14).

[37] Zhirong Wu, Yuanjun Xiong, Stella Yu e Dahua Lin. *Unsupervised Feature Learning via Non-Parametric Instance-level Discrimination*. 2018. DOI: 10.48550/ARXIV.1805.01978 (cit. a p. 14).

[38] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie e Ross Girshick. *Momentum Contrast for Unsupervised Visual Representation Learning*. 2019. DOI: 10.48550/ARXIV.1911.05722. URL: https://arxiv.org/abs/1911.05722 (cit. a p. 14).

[39] Ting Chen, Simon Kornblith, Mohammad Norouzi e Geoffrey Hinton. *A Simple Framework for Contrastive Learning of Visual Representations*. 2020. DOI: 10.48550/ARXIV.2002.05709. URL: https://arxiv.org/abs/2002.05709 (cit. a p. 14).

[40] Jovana Mitrovic, Brian McWilliams, Jacob Walker, Lars Buesing e Charles Blundell. *Representation Learning via Invariant Causal Mechanisms*. 2020. DOI: 10.48550/ARXIV.2010.07922 (cit. a p. 15).

[41] Terence C. Mills. *Applied Time Series Analysis - A practical guide for modeling and forecasting*. 2019 (cit. alle pp. 16, 17).

[42] Uwe Hassler Gebhard Kirchgässner Jürgen Wolters. *Introduction to Modern Time Series Analysis*. 2007 (cit. a p. 16).

[43] Jin Fan, Ke Zhang, Yipan Huang, Yifei Zhu e Baiping Chen. «Parallel spatio-temporal attention-based TCN for multivariate time series prediction». In: *Neural Computing and Applications* (mag. 2021). ISSN: 1433-3058. DOI: 10.1007/s00521-021-05958-z. URL: http://dx.doi.org/10.1007/S00521-021-05958-Z (cit. alle pp. 16, 18).

[44] Rui Zhao, Ruqiang Yan, Zhenghua Chen, Kezhi Mao, Peng Wang e Robert X. Gao. «Deep Learning and Its Applications to Machine Health Monitoring: A Survey». In: *CoRR* abs/1612.07640 (2016). arXiv: 1612.07640. URL: http://arxiv.org/abs/1612.07640 (cit. a p. 16).

[45] Yutaka Yoshida, Kiyoko Yokoyama e Naohiro Ishii. «Real-time Continuous Estimation of Respiratory Frequency during Sleep based on Heart Rate Time Series». In: *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 2007. DOI: 10.1109/IEMBS.2007.4352373 (cit. a p. 16).

[46] Jagadeeswara Rao Annam e Bapi Raju Surampudi. «AAMI Based ECG Heart-Beat Time-Series Clustering Using Unsupervised ELM and Decision Rule». In: *2016 International Conference on Information Technology (ICIT)*. 2016, pp. 137–141. DOI: 10.1109/ICIT.2016.039 (cit. a p. 16).

[47] Turker Ince, Serkan Kiranyaz, Levent Eren, Murat Askar e Moncef Gabbouj. «Real-Time Motor Fault Detection by 1-D Convolutional Neural Networks». In: *IEEE Transactions on Industrial Electronics* 63.11 (2016), pp. 7067–7075. DOI: 10.1109/TIE.2016.2582729 (cit. a p. 16).

[48] Xu Jiali. «Financial Time Series Prediction Based on Adversarial Network Generated by Attention Mechanism». In: *2021 International Conference on Public Management and Intelligent Society (PMIS)*. 2021, pp. 246–249. DOI: 10.1109/PMIS52742.2021.00061 (cit. a p. 16).

[49] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff e Puneet Agarwal. «Long Short Term Memory Networks for Anomaly Detection in Time Series». In: apr. 2015 (cit. a p. 17).

[50] *Component of Time Series*. URL: https://www.toppr.com/guides/business-mathematics-and-statistics/time-series-analysis/components-of-time-series/ (cit. a p. 17).

[51] *Time Series Forecasting*. URL: https://www.tableau.com/learn/articles/time-series-forecasting (cit. a p. 17).

[52] John Cristian Borges Gamboa. *Deep Learning for Time-Series Analysis*. 2017. DOI: 10.48550/ARXIV.1701.01887 (cit. alle pp. 18, 35).

[53] Zhicheng Cui, Wenlin Chen e Yixin Chen. *Multi-Scale Convolutional Neural Networks for Time Series Classification*. 2016. DOI: 10.48550/ARXIV.1603.06995 (cit. a p. 18).

[54] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar e Pierre-Alain Muller. «Deep learning for time series classification: a review». In: *Data Mining and Knowledge Discovery* 33.4 (mar. 2019), pp. 917–963. DOI: 10.1007/s10618-019-00619-1 (cit. alle pp. 18, 19, 21–23, 35–39, 43, 57, 64, 65).

[55] Tsung-Yu Hsieh, Suhang Wang, Yiwei Sun e Vasant Honavar. *Explainable Multivariate Time Series Classification: A Deep Neural Network Which Learns To Attend To Important Variables As Well As Informative Time Intervals*. 2020. DOI: 10.48550/ARXIV.2011.11631. URL: https://arxiv.org/abs/2011.11631 (cit. alle pp. 18, 21, 23, 33).

[56] Amaia Abanda, Usue Mori e Jose A. Lozano. *A review on distance based time series classification*. 2018. DOI: 10.48550/ARXIV.1806.04509. URL: https://arxiv.org/abs/1806.04509 (cit. alle pp. 19, 20).

[57] Omer Gold e Micha Sharir. *Dynamic Time Warping and Geometric Edit Distance: Breaking the Quadratic Barrier*. DOI: 10.48550/ARXIV.1607.05994 (cit. a p. 19).

[58] Anthony Bagnall, Aaron Bostrom, James Large e Jason Lines. *The Great Time Series Classification Bake Off: An Experimental Evaluation of Recently Proposed Algorithms. Extended Version*. 2016. DOI: 10.48550/ARXIV.1602.01711 (cit. alle pp. 19, 34, 35, 39, 43, 57, 64).

[59] Rohit Kate. «Using dynamic time warping distances as features for improved time series classification». In: *Data Mining and Knowledge Discovery* 30 (mag. 2015). DOI: 10.1007/s10618-015-0418-x (cit. a p. 20).

[60] Steinn Gudmundsson, Thomas Runarsson e Sven Sigurdsson. «Support Vector Machines and Dynamic Time Warping for Time Series». In: lug. 2008, pp. 2772–2776. DOI: 10.1109/IJCNN.2008.4634188 (cit. a p. 20).

[61] Alex Nanopoulos, Rob Alcock e Yannis Manolopoulos. «Feature-based Classification of Time-series Data». In: *International Journal of Computer Research* 10 (gen. 2001), pp. 49–61 (cit. a p. 20).

[62] Xiaozhe Wang, Kate Smith-Miles e Rob Hyndman. «Characteristic-Based Clustering for Time Series Data». In: *Data Min. Knowl. Discov.* 13 (set. 2006), pp. 335–364. DOI: 10.1007/s10618-005-0039-x (cit. a p. 20).

[63] Xiaozhe Wang, Anthony Wirth e Liang Wang. «Structure-Based Statistical Features and Multivariate Time Series Clustering». In: nov. 2007, pp. 351–360. ISBN: 978-0-7695-3018-5. DOI: 10.1109/ICDM.2007.103 (cit. a p. 20).

[64] Ben D. Fulcher e Nick S. Jones. «Highly Comparative Feature-Based Time-Series Classification». In: *IEEE Transactions on Knowledge and Data Engineering* (). DOI: 10.1109/tkde.2014.2316504 (cit. alle pp. 20, 21).

[65] Jiuqi Elise Zhang, Di Wu e Benoit Boulet. «Time Series Anomaly Detection for Smart Grids: A Survey». In: *2021 IEEE Electrical Power and Energy Conference (EPEC)*. 2021, pp. 125–130. DOI: 10.1109/EPEC52095.2021.9621752 (cit. a p. 24).

[66] Ang Zhang, Xiaoyong Zhao e Lei Wang. «CNN and LSTM based Encoder-Decoder for Anomaly Detection in Multivariate Time Series». In: *2021 IEEE 5th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)*. Vol. 5. 2021, pp. 571–575. DOI: 10.1109/ITNEC52019.2021.9587207 (cit. a p. 24).

[67] Jingye Yee, Cheng Yee Low, Natiara Mohamad Hashim, Fazah Akhtar Hanapiah, Ching Theng Koh, Noor Ayuni Che Zakaria, Khairunnisa Johar e Nurul Atiqah Othman. «Systematic Development of Machine for Abnormal Muscle Activity Detection». In: *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. 2021, pp. 1376–1381. DOI: 10.1109/CASE49439.2021.9551525 (cit. a p. 24).

[68] Kyeong-Joong Jeong, Jin-Duk Park, Kyusoon Hwang, Seong-Lyun Kim e Won-Yong Shin. «Two-Stage Deep Anomaly Detection With Heterogeneous Time Series Data». In: *IEEE Access* 10 (2022), pp. 13704–13714. DOI: 10.1109/ACCESS.2022.3147188 (cit. a p. 24).

[69] Jingyu Sun Zhixian Niu Shuping Shi e Xiu He. «A Survey of Outlier Detection Methodologies and Their Applications». In: *Artificial Intelligence and Computational Intelligence* 7002 (2011). DOI: https://doi-org.ezproxy.biblio.polito.it/10.1007/978-3-642-23881-9_50 (cit. a p. 24).

[70] Jim Austin Victoria Hodge. «A Survey of Outlier Detection Methodologies». In: *Artificial Intelligence Review* 22 (2004), pp. 85–126. DOI: 10.1023/B:AIRE.0000045502.10941.a9 (cit. a p. 24).

[71] Qunke Wang, Lanting Fang, Zhenchao Zhu e Jie Huang. «Detection Algorithm of the Mimicry Attack based on Variational Auto-Encoder». In: *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. 2021, pp. 114–120. DOI: 10.1109/DSN-W52860.2021.00029 (cit. a p. 24).

[72] Moisés Felipe Silva, Alessandro Pacini, Andrea Sgambelluri e Luca Valcarenghi. «Learning Long-and Short-Term Temporal Patterns for ML-driven Fault Management in Optical Communication Networks». In: *IEEE Transactions on Network and Service Management* (2022), pp. 1–1. DOI: 10.1109/TNSM.2022.3146869 (cit. a p. 24).

[73] Grubbs. «Procedures for Detecting Outlying Observations in Samples». In: *Technometrics* 11 (1969), pp. 1–21. DOI: http://dx.doi.org/10.1080/00401706.1969.10490657 (cit. a p. 24).

[74] Ander Carreño, Iñaki Inza e Jose Lozano. «Analyzing rare event, anomaly, novelty and outlier detection terms under the supervised classification framework». In: *Artificial Intelligence Review* 53 (giu. 2020). DOI: 10.1007/s10462-019-09771-y (cit. a p. 24).

[75] Ane Blázquez-García, Angel Conde, Usue Mori e Jose A. Lozano. *A review on outlier/anomaly detection in time series data*. 2020. DOI: 10.48550/ARXIV.2002.04236 (cit. alle pp. 24–27, 34).

[76] Jessica Lin, Eamonn Keogh, Ada Fu e Helga van herle. «Approximations to magic: Finding unusual medical time series». In: lug. 2005, pp. 329–334. ISBN: 0-7695-2355-2. DOI: 10.1109/CBMS.2005.34 (cit. a p. 26).

[77] Adane Nega Tarekegn, Mario Giacobini e Krzysztof Michalak. «A review of methods for imbalanced multi-label classification». In: *Pattern Recognition* 118 (2021), p. 107965. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2021.107965. URL: https://www.sciencedirect.com/science/article/pii/S0031320321001527 (cit. alle pp. 27, 47).

[78] Thomas Schreiber e Andreas Schmitz. «Discrimination power of measures for nonlinearity in a time series». In: *Physical Review E* (). DOI: 10.1103/physreve.55.5443. URL: https://doi.org/10.1103%2Fphysreve.55.5443 (cit. a p. 29).

[79] Gustavo Batista, Eamonn Keogh, Oben Tataw e Vinícius Alves de Souza. «CID: An efficient complexity-invariant distance for time series». In: *Data Mining and Knowledge Discovery* 28 (apr. 2013). DOI: 10.1007/s10618-013-0312-3 (cit. a p. 29).

[80] Jennifer Yentes, Nathaniel Hunt, Kendra Schmid, Jeffrey Kaipust, Denise Mcgrath e Nicholas Stergiou. «The Appropriate Use of Approximate Entropy and Sample Entropy with Short Data Sets». In: *Annals of biomedical engineering* 41 (ott. 2012). DOI: 10.1007/s10439-012-0668-3 (cit. a p. 29).

[81] Joshua Richman e Joseph Moorman. «Physiological Time-Series Analysis Using Approximate Entropy and Sample Entropy». In: *American journal of physiology. Heart and circulatory physiology* 278 (lug. 2000), H2039–49. DOI: 10.1152/ajpheart.2000.278.6.H2039 (cit. a p. 29).

[82] Lingfei Wu, Ian En-Hsu Yen, Jinfeng Yi, Fangli Xu, Qi Lei e Michael Witbrock. *Random Warping Series: A Random Features Method for Time-Series Embedding*. 2018. DOI: 10.48550/ARXIV.1809.05259. URL: https://arxiv.org/abs/1809.05259 (cit. a p. 30).

[83] Sana Tonekaboni, Danny Eytan e Anna Goldenberg. *Unsupervised Representation Learning for Time Series with Temporal Neighborhood Coding*. 2021. DOI: 10.48550/ARXIV.2106.00750 (cit. alle pp. 30, 53).

[84] Jean-Yves Franceschi, Aymeric Dieuleveut e Martin Jaggi. «Unsupervised Scalable Representation Learning for Multivariate Time Series». In: (2019). DOI: 10.48550/ARXIV.1901.10738 (cit. alle pp. 30, 53).

[85] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong e Bixiong Xu. *TS2Vec: Towards Universal Representation of Time Series*. 2021. DOI: 10.48550/ARXIV.2106.10466 (cit. alle pp. 30, 31, 34).

[86] Hui Lu, Yaxian Liu, Zongming Fei e Chongchong Guan. «An Outlier Detection Algorithm Based on Cross-Correlation Analysis for Time Series Dataset». In: *IEEE Access* PP (set. 2018), pp. 1–1. DOI: 10.1109/ACCESS.2018.2870151 (cit. a p. 34).

[87] Zhiguang Wang, Weizhong Yan e Tim Oates. «Time series classification from scratch with deep neural networks: A strong baseline». In: mag. 2017, pp. 1578–1585. DOI: 10.1109/IJCNN.2017.7966039 (cit. alle pp. 35, 36, 65).

[88] Ben McKay. «What is convolution intuitively». In: (). URL: https://mathoverflow.net/users/13268/ben-mckay (cit. a p. 36).

[89] Hoang Anh Dau et al. *The UCR Time Series Classification Archive*. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/. Ott. 2018 (cit. a p. 39).

[90] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam e Eamonn Keogh. *The UEA multivariate time series classification archive, 2018*. 2018. DOI: 10.48550/ARXIV.1811.00075 (cit. a p. 43).

[91] Hansheng Ren et al. «Time-Series Anomaly Detection Service at Microsoft». In: *CoRR* abs/1906.03821 (2019) (cit. a p. 44).

[92] Nikolay Laptev, Saeed Amizadeh e Youssef Billawala. *A Benchmark Dataset for Time Series Anomaly Detection*. 2015. URL: https://yahooresearch.tumblr.com/post/114590420346/a-benchmark-dataset-for-time-series-anomaly (cit. a p. 45).

[93] David M. W. Powers. «Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation». In: (2020). DOI: 10.48550/ARXIV.2010.16061. URL: https://arxiv.org/abs/2010.16061 (cit. a p. 45).

[94] N. V. Chawla, K. W. Bowyer, L. O. Hall e W. P. Kegelmeyer. «SMOTE: Synthetic Minority Over-sampling Technique». In: *Journal of Artificial Intelligence Research* (). DOI: 10.1613/jair.953 (cit. a p. 46).

[95] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty e Carsten Eickhoff. *A Transformer-based Framework for Multivariate Time Series Representation Learning*. 2020. DOI: 10.48550/ARXIV.2010.02803. URL: https://arxiv.org/abs/2010.02803 (cit. a p. 53).

[96] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li e Cuntai Guan. *Time-Series Representation Learning via Temporal and Contextual Contrasting*. 2021. DOI: 10.48550/ARXIV.2106.14112. URL: https://arxiv.org/abs/2106.14112 (cit. a p. 53).

[97] Wensi Tang, Guodong Long, Lu Liu, Tianyi Zhou, Michael Blumenstein e Jing Jiang. «Omni-Scale CNNs: a simple and effective kernel size configuration for time series classification». In: (2020). DOI: 10.48550/ARXIV.2002.10061 (cit. a p. 65).