SAPIENZA
UNIVERSITÀ DI ROMA

Modello di Vicsek topologico analizzato con tecniche di Machine Learning

Facoltà di Scienze Matematiche, Fisiche e Naturali

Corso di Laurea in Fisica

Candidato

Francesco D'Amico
Matricola 1844265

Relatori

Prof. Stefano Giagu
.....

Anno Accademico 2020/2021

Tesi non ancora discussa

Modello di Vicsek topologico analizzato con tecniche di Machine Learning
Tesi di Laurea. Sapienza – Università di Roma

Questa tesi è stata composta con LaTeX e la classe Sapthesis.

Email dell'autore: francescoluigidamico@gmail.com

A tutti coloro che difendono tramandano evolvono la cultura

Indice

Bibliografia **21**

Capitolo 1

Introduzione

Nella presente dissertazione si affronta lo studio di un modello matematico alla base della comprensione dei comportamenti collettivi presenti in molte specie viventi. Il modello di Vicsek si pone l'obiettivo di saper generare tali fenomeni naturali utilizzando un set di regole semplice e minimale. Nonostante la sua semplicità di formulazione, i fenomeni generati sono notevoli e complessi; nel seguito saranno affrontati e analizzati solo alcuni di essi. In particolare l'attenzione sarà sulle caratteristiche del modello più affini a quanto trovato in natura per la specie di Storno comune, Sturnus Vulgaris ([Bal+08], [Bia+12]).
Al fine di osservare i comportamenti complessi che emergono dalle regole del modello, è stata implementata una simulazione computazionale. Il numero di particelle simulate è dell'ordine di 10^3, ovvero il numero massimo considerando le risorse computazionali e il tempo a disposizione. Tutte le analisi effettuate sui dati ottenuti sono concordi con quanto atteso dalla teoria. E' stato studiato il comportamento del parametro d'ordine del sistema, la polarizzazione ϕ, tramite la quale è possibile osservare la presenza di una transizione di fase. Anche la più sofisticata funzione di correlazione C(r) mostra il comportamento atteso: un andamento quasi lineare, e una lunghezza di correlazione ξ che cresce linearmente con la taglia L del sistema.
Infine, è stata addestrata una rete neurale al fine di analizzare il sistema. Non è di grande interesse il risultato stesso dell'analisi, in quanto non è stata ottenuta qualche nuova comprensione della fisica che non fosse già nota prima di cominciare l'addestramento. La cosa più interessante è proprio la capacità della rete neurale di imparare a svolgere tali analisi, anche in caso di grandezze computazionalmente difficili da calcolare o addirittura laddove non sia noto affatto un algoritmo risolutivo.
E' doveroso sottolineare che nella presente dissertazione tutti i dati sono simulati, e questa è una semplificazione notevole poiché si hanno conoscenze a priori che non si avrebbero studiando dati veri ottenuti sul campo. Ma questo non significa che sia impossibile ripetere quanto ottenuto su dati veri, anzi, la naturale evoluzione di quanto fatto nella presente dissertazione consiste proprio nel tentativo di utilizzare tali metodi di Machine Learning nello studio di dati reali ottenuti sul campo.

Capitolo 2

Il modello di Vicsek

2.1 Definizione

Il modello di Vicsek nasce come un set di regole semplici in grado di generare il fenomeno del moto collettivo, presente in molte specie viventi. Ne sono esempio banchi di pesci, mandrie di mammiferi, stormi di uccelli, sciami di insetti. Sono presenti anche a livello microscopico, come ad esempio il batterio Bacillus Subtilis. La caratteristica principale che differenzia gli organismi viventi dalle particelle inanimate, e quindi dai tipici modelli che studia la fisica, è la loro capacità di consumare energia a piacimento per produrre movimento rispetto al substrato in cui vivono.
La definizione originale di modello di Vicsek [Vic+95] consiste in una cella bidimensionale di lato L con condizioni periodiche al bordo, riempita di un numero N di particelle puntiformi. Queste sono Self-Driven, ovvero decidono autonomamente la direzione di moto, senza conservare energia, momento o quantità di moto. Hanno tutte una velocità in modulo uguale e costante nel tempo. Lo stato iniziale è caratterizzato da posizioni e velocità casuali. Ad ogni step di tempo discreto, vengono aggiornate la posizione $\boldsymbol{x_i}$ e la direzione angolare θ di ciascuna particella secondo le equazioni:

$$\boldsymbol{x_i}(t+1) = \boldsymbol{x_i}(t) + \boldsymbol{v_i}(t)\Delta t \tag{2.1}$$

$$\theta(t+1) = \langle\theta(t)\rangle + \Delta\theta \tag{2.2}$$

dove la media $\langle\cdot\rangle$ è eseguita sui vicini secondo una data metrica, e $\Delta\theta$ un rumore uniforme $\Delta\theta \in [-\eta/2, +\eta/2]$. Ci sono tre parametri liberi: entità del rumore η, densità di particelle ρ e modulo della velocità delle particelle v. Questa è la formulazione generale del modello: note le equazioni del moto discrete che segue ogni particella, e le caratteristiche spaziali del sistema preso in considerazione come dimensionalità e condizioni al bordo, il sistema è completamente definito. Nonostante la semplicità di formulazione, il modello è in grado di generare fenomeni complessi e di variegata tipologia.

2.1.1 Metrica topologica e rumore vettoriale

Nello studio effettuato, ci si è concentrati su particolari caratteristiche del modello di Vicsek che sono state riscontrate in natura nella specie dello storno comune, lo Sturnus Vulgaris [Bal+08]. La metrica del modello di Vicsek originale è euclidea:

data una particella, questa interagisce con tutte quelle che si trovano all'interno di un cerchio di raggio r. Invece si è scelta la metrica topologica, cioè ogni particella interagisce con le sue prime m vicine. Inoltre le equazioni sono definite in modo vettoriale, al fine di estendere in 3 dimensioni la definizione originale che ne prevedeva 2. Le equazioni diventano:

$$\boldsymbol{x_i}(t+1) = \boldsymbol{x_i}(t) + \boldsymbol{v_i}(t)\Delta t \tag{2.3}$$

$$\boldsymbol{v_i}(t+1) = \frac{\sum_j n_{ij}\boldsymbol{v_j}(t) + \eta m \boldsymbol{e_i}(t)}{||\sum_j n_{ij}\boldsymbol{v_j}(t) + \eta m \boldsymbol{e_i}(t)||} \tag{2.4}$$

dove m è il numero di primi vicini che ciascuna particella tiene in considerazione per decidere la sua $\boldsymbol{v_i}(t+1)$ ed n_{ij} è la "matrice di vicinanza" che vale 1 se la particella j fa parte dei primi m vicini di i e zero viceversa; si sceglie inoltre $n_{ii} = 1$. L'errore è implementato in modo vettoriale: η è l'intensità dell'errore, ed $\boldsymbol{e_i}(t)$ è un vettore di norma unitaria con distribuzione angolare uniforme: è creato in modo pseudocasuale per ogni particella e ad ogni istante t. Cioè, ogni particella commette in media lo stesso errore a calcolare la velocità di ciascun suo primo vicino. Ovvero all'aumentare del numero di vicini seguiti, aumenta anche l'errore totale commesso. E' ragionevole che sia davvero così per uccelli reali, ma è comunque una assunzione; la cosa più importante è che tale implementazione ha comunque le stesse proprietà asintotiche del "rumore scalare" [Gin16], come ad esempio quello del modello di Vicsek originale.

2.2 Analisi del modello

2.2.1 Grandezze di interesse

Un parametro fondamentale del sistema è la polarizzazione della velocità delle particelle

$$\boldsymbol{\phi}(t) = \frac{1}{N}\sum \boldsymbol{v_i}(t) \tag{2.5}$$

il cui modulo $\phi(t) = |\boldsymbol{\phi}(t)|$ è il parametro d'ordine. Il rumore η è invece il parametro di controllo al variare del quale si studia il comportamento del sistema. Esso svolge il ruolo tipico della temperatura in un sistema di atomi o molecole, ovvero inserisce fluttuazioni randomiche che contrastano la tendenza delle particelle di mostrare un comportamento collettivo, come l'allineamento degli spin in un ferromagnete.
Un'altra grandezza di interesse del modello è la funzione di correlazione C(r). Data la velocità media dello stormo, cioè la polarizzazione vettoriale $\boldsymbol{\phi}$, si definiscono le deviazioni $\delta\boldsymbol{v_i} = \boldsymbol{v_i} - \boldsymbol{\phi}$, da cui la funzione di correlazione

$$C(r) = \langle \frac{\sum_{ij} \delta\boldsymbol{v_i} \cdot \delta\boldsymbol{v_j}\, \delta(r - r_{ij})}{\sum_{ij} \delta(r - r_{ij})} \rangle \tag{2.6}$$

dove la media $\langle\cdot\rangle$ si intende su varie realizzazioni, la $\delta(r - r_{ij})$ è la delta di Dirac opportunamente discretizzata, e r_{ij} è il modulo della distanza tra le particelle i e j. Per gli stormi di Sturnus Vulgaris, ovvero per il modello di Vicsek topologico è stato osservato, sia nelle simulazioni sia nei dati reali, che la forma di tale funzione correlazione è circa lineare decrescente, con uno zero [Cav+10]. Per cui per uno

stormo di dimensione finita, preso un uccello, esiste un dominio di esemplari con velocità correlata, e un altro con velocità anticorrelata (da non intendere come una assenza di correlazione). La posizione dello zero della funzione di correlazione, cioè la distanza ξ tale che $C(r = \xi) = 0$, è definita "Lunghezza di correlazione".
Tale lunghezza di correlazione si è vista crescere linearmente con la dimensione di uno stormo di Sturnus Vulgaris [Cav+10]. E' possibile dimostrare che per stormi di taglia finita la relazione $\xi \propto L$ è una diretta conseguenza della rottura spontanea della simmetria continua angolare [Gin16]. Si dimostra che la forma funzionale attesa della funzione di correlazione è

$$C(r) = \frac{1}{r^{\gamma}} g(\frac{r}{\xi}) \tag{2.7}$$

dove g è una funzione adimensionale di scaling, e tale che g(1)=0. Questo significa che non vi è una scala di lunghezza tipica per il fenomeno, eccetto quella data dalla dimensione dello stormo L. Il valore di γ misurato risulta molto piccolo: $\gamma = 0.19 \pm 0.08$ [Cav+10]. Pertanto la funzione di correlazione C(r) è quasi una retta, specialmente a valori di r piccoli e minori di ξ.

2.2.2 Caratteristiche e proprietà

Differentemente dai tipici modelli studiati dalla meccanica statistica, l'energia totale e la quantità di moto del sistema nel modello di Vicsek non sono conservate, e il sistema non è invariante sotto trasformazioni di velocità Galileiane (il substrato dissipativo è il sistema di riferimento fermo e speciale rispetto agli altri). Per queste caratteristiche si dice che è "lontano dall'equilibrio termodinamico" [Gin16]. L'unica quantità conservata è il numero di particelle presenti.
Il fenomeno principale è la rottura spontanea della simmetria angolare. Quando il rumore η è inferiore ad una soglia critica, il sistema da una situazione caotica, ovvero velocità orientate casualmente e un valore della polarizzazione $\boldsymbol{\phi} \approx 0$, sceglie autonomamente una direzione privilegiata verso cui si direziona $\boldsymbol{\phi}$, il cui modulo ϕ diventa di ordine 1. Tale scelta non è intuibile a priori, e tutte le direzioni sono ugualmente probabili. Il fatto che l'orientazione di ogni particella può variare in modo continuo nell'angolo solido implica che la simmetria angolare sia continua. Pertanto, in corrispondenza del rumore η critico, è presente una transizione di fase tra stato ordinato polarizzato e stato caotico.

2.2.3 Confronto con altri modelli della fisica statistica

E' possibile fare una comparazione tra il modello di Vicsek e quello di Ising. E' stato già osservato come il rumore svolge il ruolo della temperatura in un ferromagnete. Il modulo della polarizzazione $\boldsymbol{\phi}$ descritta in Eq. 2.5 è a tutti gli effetti l'analogo della magnetizzazione, sia per come è definita, sia per il ruolo che svolge. Entrambe sono il parametro d'ordine, ovvero quello da cui comprendiamo lo stato macroscopico del sistema, e dal quale osserviamo la transizione di fase.
La differenza più importante è che nel modello di Ising le particelle sono ferme e disposte in una griglia, mentre in quello di Vicsek si muovono, e per di più le stesse leggi di attrazione e repulsione che regolano l'orientazione agiscono indirettamente

anche sul moto. Ovvero, la matrice di vicinanza che determina quali particelle interagiscono varia in modo dinamico e molto complesso a causa delle leggi del moto. Questa è una caratteristica assente nei ferromagneti, dove la matrice di vicinanza è fissata e quindi indipendente dalle leggi che governano l'orientazione degli spin.

Capitolo 3

La simulazione

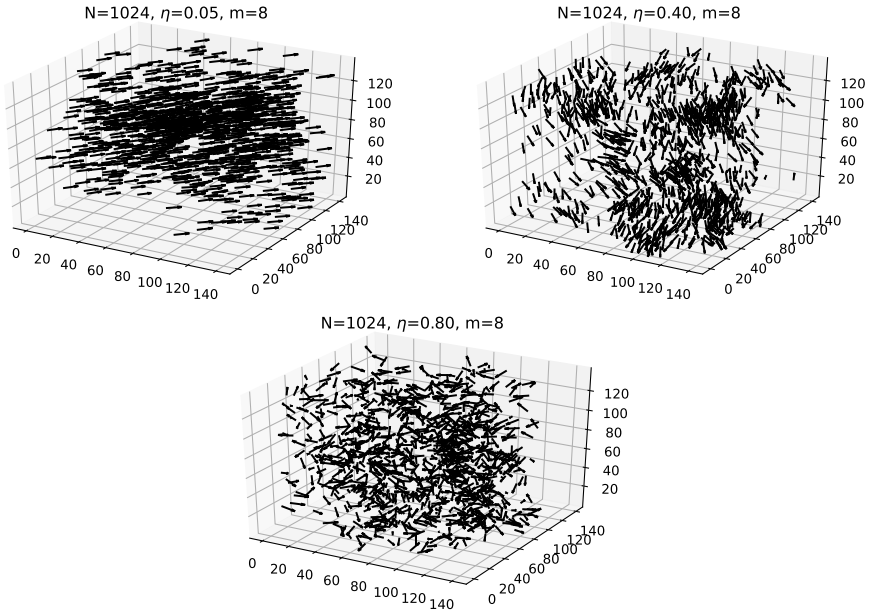

Figura 3.1. Esempi di stormi simulati e termalizzati al variare del parametro di controllo η. Le frecce nere rappresentate sono le velocità delle particelle. In alto a sinistra: Stormo fortemente polarizzato simile ai reali stormi di Sturnus Vulgaris, $\eta = 0.05$. In alto a destra: E' ancora presente il moto collettivo, anche se più disordinato. $\eta = 0.40$. In basso: è avvenuta la transizione di fase, pertanto la polarizzazione è praticamente nulla e il sistema ha perso il comportamento collettivo.

3.1 Caratteristiche studiate

Si simula il modello di Vicsek topologico con il duplice obiettivo di analizzare il sistema al variare dei suoi parametri, e di ottenere le nuvole di punti necessarie per l'addestramento dell'architettura DGCNN.
In particolare si studia il sistema esplorando lo spazio dei parametri nei dintorni di quanto osservato in natura per stormi di Sturnus Vulgaris [Bia+12]:

1. L: taglia lineare del sistema. Si sceglie una densità di particelle ρ fissata in modo da ottimizzare la velocità di esecuzione. E' possibile sceglierla a piacimento poiché il comportamento di un modello topologico non dipende dal valore assoluto delle distanze. A causa di questa scelta, il numero totale di particelle $N = \rho L^3$ dipende esclusivamente dal valore di L. Si scelgono valori di L tali che $N \in [371, 1024]$.

2. m: numero di primi vicini "visti" da ogni particella. Coerentemente con quanto osservato sperimentalmente $m \approx 7.8$ [Bia+12], si scelgono i valori $m = \{6, 7, 8, 9\}$.

3. η: entità dell'errore commesso. Si scelgono valori tali che la polarizzazione ϕ risulti un valore prossimo a 1, poiché negli stormi naturali è stato misurato $\phi \in (0.85, 0.98)$.

3.2 Risultati ottenuti

La prima grandezza studiata è la polarizzazione ϕ in funzione del parametro di controllo η, l'entità dell'errore vettoriale. Nella Fig. 3.2 si osserva l'andamento di ϕ al variare di m e della taglia N. Il grafico ottenuto è coerente con quanto presente in [Gin16]. Si sottolinea che, come descritto in Eq. 2.4, l'errore η è moltiplicato per m: pertanto è interessante osservare in Fig. 3.2 (a) che all'aumentare di m migliora comunque la capacità del sistema di polarizzarsi.

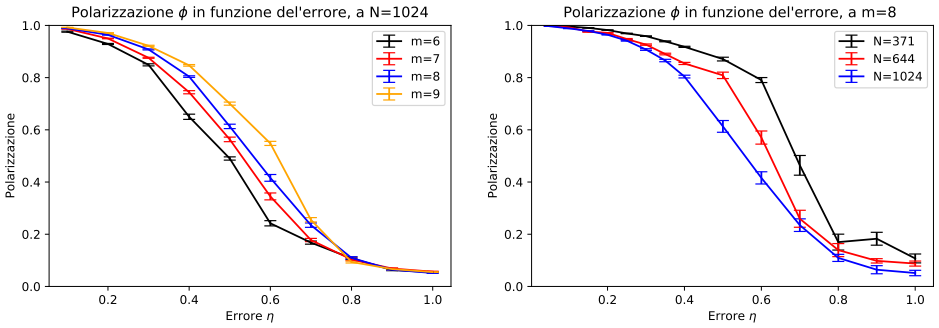

Figura 3.2. a) L'andamento è simile per tutti i casi, ma all'aumentare di m aumenta la capacità del sistema di resistere al rumore e polarizzarsi. b) Studio dell'andamento asintotico della curva di polarizzazione a taglie N crescenti.

Si studia successivamente la funzione di correlazione C(r), e in particolare la lunghezza di correlazione ξ (Grafico 3.3). Come atteso dalla teoria e dalle osservazioni sul campo di stromi reali, si ottiene un andamento lineare di ξ rispetto alla taglia L del sistema. Pertanto è verificato che la funzione C(r) è un'invariante di scala. Invece al variare degli altri parametri ξ risulta pressoché costante, fintanto che l'errore non è tale da distruggere il comportamento collettivo del sistema ($\eta > 0.4$).

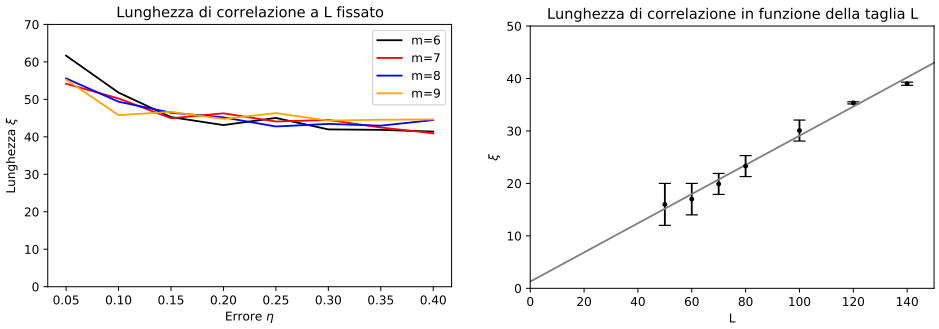

Figura 3.3. a) E' interessante osservare che non si evidenzia una dipendenza da m della lunghezza di correlazione ξ, il quale è anche poco dipendente dall'errore fino a che il sistema mostra comportamento collettivo ($\eta < 0.4$). b) Si ottiene $\xi \propto L$, come atteso dalla teoria e osservato in natura. I punti sperimentali sono stati ottenuti mediando su vari errori η a m=8.

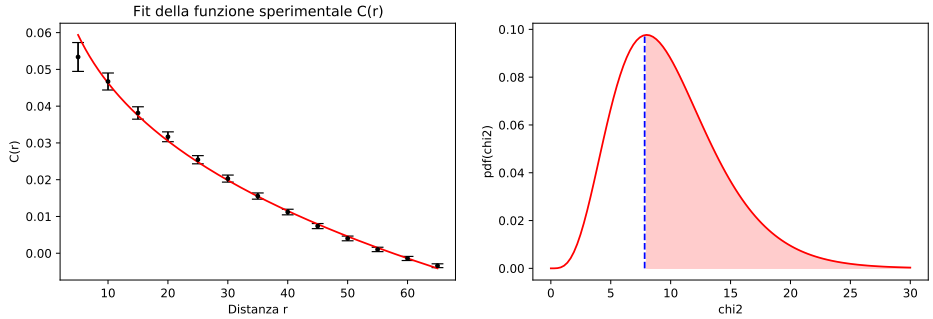

Figura 3.4. Fit sulla funzione C(r) descritta in Eq. 2.7 dei dati ottenuti con L=140, η=0.2, m=8. Si ottiene $\gamma = (0.12 \pm 0.02)$. b) PDF del chi quadro del fit: in rosso l'area corrispondente al p-value, la linea blu corrisponde all'ascissa del χ^2 misurato. Si ottiene χ^2/gdl=0.78, p-value=0.78, segno di un'ottima bontà del Fit effettuato.

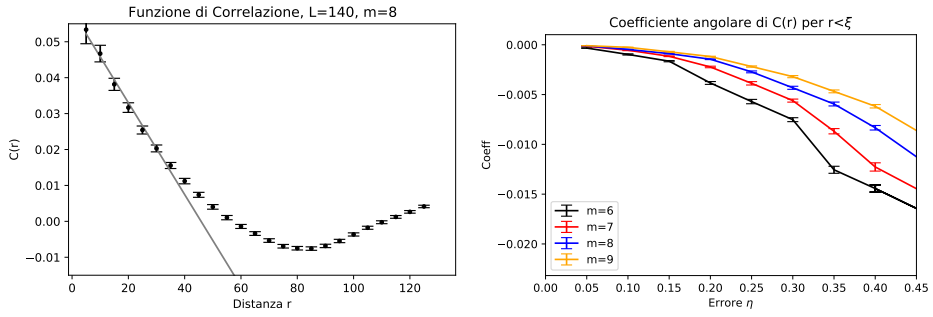

Figura 3.5. a) Esempio di funzione di correlazione (L=140, m=8, $\eta = 0.2$). Per $r < \xi$ la funzione ha un comportamento pressoché lineare, come previsto dall'Eq. 2.7 e dal fatto che $\gamma \approx 0$. Si definisce il coefficiente angolare α di tale retta "Coefficiente di Correlazione". Per r maggiori si evidenzia un atteso discostamento dalla linearità, e per r>L/2 la correlazione torna ad aumentare a causa delle condizioni periodiche al bordo. b) Studio del coefficiente di correlazione α, coefficiente angolare della retta che approssima C(r), a taglia L fissata.

E' presente un esempio di funzione di correlazione completa in Fig. 3.5. La parte di funzione di correlazione di interesse fisico è quella per $r < \xi$, poiché successivamente dominano effetti di taglia finita. Nell'intervallo di interesse, la funzione è approssimata da una retta, come atteso dall'Eq. 2.7 nel caso di esponente $\gamma \approx 0$. Si ottiene tramite fit $\gamma = (0.12 \ \pm 0.02)$ considerando solamente la zona $r < \xi$ (Fig. 3.4) . Il risultato è compatibile con quanto misurato sperimentalmente in stormi reali $\gamma_{exp} = (0.19 \pm 0.08)$ [Cav+10].
Per ultimo, in Fig. 3.5 si studia al variare dei parametri il coefficiente angolare α, che chiamiamo "Coefficiente di Correlazione", della retta che approssima C(r) per $r < \xi$. Il fatto che $\alpha \approx 0$, specialmente per errori η piccoli, indica la presenza di una forte correlazione nel sistema. Di nuovo si osserva che all'aumentare di m, migliora la capacità del sistema di mostrare caratteristiche collettive contrastando il rumore.

Capitolo 4

Studio con tecniche di Machine Learning

4.1 Il Machine Learning

Il machine learning è un insieme di tecniche algoritmiche di analisi dati. Si basa su idee del secolo scorso, ma solo nell'ultima decade ha raggiunto livelli ineguagliabili in certi ambiti, grazie al notevole sviluppo delle componenti hardware che accelerano il loro funzionamento. Ma presenta due grandi limitazioni: rispetto ad altri metodi di analisi dati, il machine learning è caratterizzato da un elevatissimo costo computazionale e dalla necessita di enormi quantità di dati. Quindi spesso è la scelta migliore laddove si hanno a disposizione grandi moli di dati, e nei casi in cui il problema sia di difficile soluzione, ad esempio quando non è noto un algoritmo risolutivo.
Nonostante la vasta tipologia di algoritmi che è possibile racchiudere in questa classe, il loro obiettivo è molto simile. In un algoritmo tradizionale, introduciamo dei dati e delle regole (ovvero scriviamo un programma), e otteniamo in output delle risposte, come ad esempio delle grandezze calcolate. In un algoritmo di machine learning, inseriamo i dati e le risposte che già conosciamo, e l'output del programma consiste nelle regole da applicare ai dati per trovare le risposte che abbiamo fornito.[1] Questa fase prende il nome di "addestramento". Scopo ultimo è dare al programma dei dati nuovi e mai visti, e ottenere i risultati corretti. La grande novità apportata da tali metodi è la loro capacità di imparare in modo autonomo a svolgere quei compiti che noi non sapremmo formalizzare tramite algoritmi.

4.1.1 Le reti neurali artificiali

Le reti neurali sono ad oggi la classe di algoritmi di Machine Learning con le migliori prestazioni in una vasta gamma di problemi. Si basano sulla presenza di una grande quantità di elementi semplici collegati tramite un grafo, chiamati neuroni in analogia con una rete neurale biologica.
La rete di neuroni più semplice, ma comunque molto utilizzata, è chiamata "fully connected". I neuroni sono divisi in strati, i layers, e ogni neurone dell'l-esimo

[1] Si precisa che nel "machine learning non supervisionato", non si comunicano al programma nemmeno le risposte, ma qui e in seguito escluderemo questa tipologia dalla trattazione.

layer è collegato con tutti quelli dei layers l+1 e l-1. Il segnale viaggia nella rete in modo discreto: parte dal layer di input, e passa ad ogni step temporale allo strato successivo. Preso un neurone dello strato l, questo riceve in input tanti numeri reali x_i quanti sono i neuroni nel layer l-1. A sua volta, esso calcola il numero reale

$$a = \sum_i w_i x_i + b \tag{4.1}$$

e invia a tutti i neuroni del successivo layer il numero reale di output

$$y = \phi(a) \tag{4.2}$$

- La funzione $\phi(a)$ prende il nome di "funzione di attivazione", e viene scelta in modo da ottimizzare le prestazioni. Ad esempio a volte $\phi(a) = tanh(a)$, ma la più utilizzata è detta "ReLU", ovvero $\phi(a) = max(0, a)$.

- I numeri reali w_i e b si chiamano "pesi della rete". L'addestramento della rete consiste unicamente nell'apprendimento degli opportuni pesi di ogni neurone. In reti come quelle utilizzate in questo studio, il numero di pesi è dell'ordine di 10^6.

Una rete intera addestrata può essere immaginata come una funzione molto complessa F tale che

$$\hat{Y} = F(X) \tag{4.3}$$

dove X è l'input, ad esempio una immagine di pixels o nel caso del presente studio 1024 vettori di 6 dimensioni, e $\hat{Y}$ è un output il più "vicino" possibile alla risposta corretta Y, chiamata "target". Qualsiasi cosa sia Y, è necessario quindi che sia stata fornita alla rete una metrica dalla quale valutare la nozione di vicinanza tra il target Y e l'output della rete $\hat{Y}$. Prendendo come esempio lo studio compiuto più avanti in questo capitolo, Y è un numero reale nella task di regressione del coefficiente di correlazione α, mentre invece nella task di classificazione del numero m di vicini, Y potrebbe essere un intero che corrisponde al "nome" di una classe, oppure un vettore nella codifica one-hot (descritta nel paragrafo successivo).

4.1.2 La Loss Function e l'addestramento di una rete

Visti gli elementi di base che costituiscono un ANN (Artificial Neural Network), ora è possibile presentare l'algoritmo che aggiorna i pesi della rete in modo opportuno, nella fase di addestramento o "training". Come prima cosa si definisce "loss function" la già citata metrica che misura la distanza tra gli oggetti Y ed $\hat{Y}$. In una task di regressione, dato che Y ed $\hat{Y}$ sono numeri reali, una scelta comune è

$$MSE = \frac{1}{n} \sum_{j=1}^{n} (\hat{Y} - Y)^2 \tag{4.4}$$

ovvero la deviazione quadratica media. Nella classificazione, spesso si usa la codifica "one-hot": se sono presenti c classi, Y è un vettore di c componenti, tutte zero tranne

la componente che indica la classe corretta, la quale vale 1. La loss function è allora la "categorical crossentropy" L, definita come

$$L = -\sum_{i=1}^{c} Y_i log(\hat{Y}_i) \tag{4.5}$$

L'addestramento della rete consiste nella ricerca dei pesi che minimizzano la loss function. Ovvero, l'addestramento di un ANN non è nient'altro che la ricerca del minimo di una funzione scalare definita in uno spazio con moltissime dimensioni, pari al numero di pesi (tipicamente dell'ordine di 10^6). Il metodo di ricerca del minimo è l'algoritmo che prende il nome di "back-propagation". Una trattazione esaustiva esula dagli intenti di questa breve introduzione; in modo generico, l'algoritmo può essere distinto in due fasi:

1. Forward phase: a pesi fissati, dato un input X, si ottiene l'output della rete $\hat{Y} = F(X)$.

2. Backward phase: si calcola l'errore Δ che ha commesso la rete, ovvero la distanza tra il target Y ed $\hat{Y}$ secondo la metrica data dalla loss function. Tale errore Δ viene propagato all'indietro a partire dall'ultimo layer, e si ottiene per ogni peso di ogni neurone una misura di quanto esso debba essere cambiato al fine di rendere $\hat{Y}$ più vicino al target Y.

Per ultimo, è importante citare le due principali problematiche che si incontrano nell'addestramento di un ANN.
Per prima cosa, sono presenti dei parametri "esterni" alla rete, ovvero non apprendibili da essa. Ad esempio il numero di neuroni che deve avere un dato layer, oppure la scelta della loss function, sono caratteristiche che devono essere decise dall'addestratore sulla base di esperienze e tentativi. La gran parte del tempo speso nell'addestramento di un ANN consiste proprio nel cosidetto "tuning degli iperparametri", ovvero nella scelta dei parametri che producono le migliori prestazioni.
Infine, la presenza di un numero di pesi enorme comporta seri problemi di overfitting. Ovvero, una rete neurale tende a "memorizzare" le relazioni tra gli input X e i target Y, invece di comprendere la reale dipendenza $Y = F(X)$. Per cui è sempre presente il rischio che ad ottime prestazioni durante la fase di training, corrispondano pessime prestazioni della rete nei confronti di nuovi input X mai visti. Dato che ci interessa proprio la capacità di una ANN di generalizzare, ovvero di fornire risposte corrette di fronte a dati nuovi X e di cui non sappiamo già la risposta corretta Y, questa tematica è di fondamentale importanza. Esistono varie tecniche per mitigare questo problema: la più semplice e importante è suddividere il dataset in due parti:

1. Training set: la gran parte dei dati sono utilizzati per addestrare la rete.

2. Test set: una piccola frazione del dataset non viene mai presentata alla ANN durante l'addestramento. E' su questa che si valutano le reali prestazioni dell'algoritmo.

4.1.3 L'architettura CNN

Sono molti i problemi in cui l'input è costituito da una griglia regolare: ad esempio una immagine composta da una struttura fissata di pixels, o un rilevatore di particelle fatto

da una griglia di scintillatori, o ancora una struttura cristallina di spin. L'architettura CNN è specializzata nell'analizzare in modo efficace queste tipologie di strutture di dati con un numero minimo di parametri da addestrare. Rispetto alle reti fully connected, funziona sfruttando delle assunzioni:

- l'input è una griglia (struttura spaziale simmetrica)
- le caratteristiche di una immagine rimangono le stesse indipendentemente dalla loro posizione (invarianza per traslazione)
- una feature complessa può essere rilevata riconoscendo alcune sub-features di cui è composta (composizionalità delle features)
- spesso basta una piccola porzione dell'input per determinare una feature (località)

Sfruttare queste assunzioni riduce notevolmente il costo computazionale dell'architettura CNN, che implica migliori prestazioni a parità di potenza di calcolo. Purtroppo nel caso delle Points Clouds prodotte dal modello di Vicsek la prima assunzione non è valida, per cui è necessario l'uso della più sofisticata architettura DGCNN, che comunque basa il proprio funzionamento su idee simili a quelle delle reti CNN.

4.2 Il modello DGCNN

I problemi in cui sono presenti Points Clouds sono molti e di grande interesse, dalla ricostruzione di oggetti tridimensionali a problemi di meccanica statistica, come ad esempio il Modello di Vicsek. La principale difficoltà nell'applicazione di metodi di Machine Learning risiede nell'assenza di una struttura regolare nei dati, per cui non è possibile utilizzare architetture come le CNN. Una conseguenza importante è che la PointCloud è invariante sotto permutazione dell'ordine di due punti, ovvero non esiste un ordinamento univoco e oggettivo, contrariamente ad una griglia ordinata. Il primo modello a gestire direttamente l'irregolarità intrinseca nelle Points Clouds, senza passare per rappresentazioni intermedie, è stato PointNet [Qi+17]. Ha raggiunto l'invarianza sotto scambio di punti operando su ciascuno di essi separatamente, e applicando funzioni simmetriche per accumulare le features mantenendo tale invarianza.
Estensioni di PointNet lavorano su vicinati di punti invece che su ciascun singolo, migliorando la capacità di trovare features locali. Ma per mantenere l'invarianza sotto scambio, trattano i punti in modo indipendente a livello locale. Questo però distrugge le relazioni geometriche tra i punti, introducendo una limitazione non superabile. Per migliorare ancora di più le prestazioni è necessario tenere conto delle strutture locali di punti.

4.2.1 L'operazione di Edge Convolution

Per risolvere questo problema, si utilizza l'operazione di EdgeConv [Wan+19], che ha il pregio di catturare strutture geometriche locali, mantenendo l'invarianza sotto permutazione. L'operazione di EdgeConv costruisce grafi locali, permettendo di ottenere Edge Features invece di Point Features. Ovvero, invece che lavorare sui

singoli punti, si lavora su grafi locali di vicinanza, sui quali si eseguono operazioni di convoluzione. Il modello DGCNN proposto dal lavoro originale consiste principalmente nell'architettura PointNet, ma con la funzione di EdgeConv come funzione di accumulazione delle features.
Al contrario delle CNN, i grafi che connettono i punti non sono fissati, ma bensì vengono ricreati ad ogni layer. Cioè ad ogni layer, il set dei primi vicini di ogni punto cambia. Quindi la prossimità nello spazio delle Features cambia da quella nello spazio euclideo iniziale. Si raggiunge così una diffusione non locale dell'informazione, caratteristica importante del modello.
Si chiami PointCloud $\boldsymbol{X} = \{\boldsymbol{x_1}, ..., \boldsymbol{x_n}\}$ l'insieme dei punti $\boldsymbol{x_i} \subset R^F$. Nel caso di particelle tridimensionali, F=6, racchiudendo sia le coordinate che le velocità. Si definiscano $\boldsymbol{V} = \{1, ..., n\}$ i nodi ed $\boldsymbol{E} \subset \boldsymbol{V} \times \boldsymbol{V}$ gli archi che li collegano. Si definisce Grafo l'insieme $\boldsymbol{G = (V, E)}$ l'insieme dei vertici e degli archi che li connettono. Nella fattispecie si sceglie $\boldsymbol{G}$ come il grafico k-NN: ogni nodo è connesso da archi ai suoi k primi vicini, incluso se stesso.
Si definiscono "Edge Features" le grandezze

$$\boldsymbol{e_{ij}} = h_{\boldsymbol{\Theta}}(\boldsymbol{x_i}, \boldsymbol{x_j}) \tag{4.6}$$

dove $h(\cdot, \cdot) : R^F \times R^F \Rightarrow R^{F'}$ è una funzione non lineare dipendente da un set di parametri addestrabili $\boldsymbol{\Theta}$. Chiamiamo $\boldsymbol{O}[\cdot]$ l'operazione di aggregazione simmetrica sotto permutazione, che agisce su tutte le Edge Features collegate a un dato nodo. Ad esempio $\boldsymbol{O}$ potrebbe essere la sommatoria o il massimo. Si può ora definire l'operazione di Edge Convolution applicata all'i-esimo vertice come

$$\boldsymbol{x_i'} = \boldsymbol{O}[h_{\boldsymbol{\Theta}}(\boldsymbol{x_i}, \boldsymbol{x_j})] \tag{4.7}$$

Si sottolinea che tale operazione di aggregazione $\boldsymbol{O}$ applicata a un nodo i, tiene conto solamente dei suoi primi vicini j. Data una PointCloud F-dimensionale, l'operazione di EdgeConv applicata a tutti i vertici produce una nuova PointCloud F'-dimensionale, ma con lo stesso numero di punti.
La scelta della funzione $h(\cdot, \cdot)$ è cruciale nel comportamento dell'architettura. PointNet può essere visto come un caso speciale di DGCNN, in cui $h_{\boldsymbol{\Theta}}(\boldsymbol{x_i}, \boldsymbol{x_j}) = h_{\boldsymbol{\Theta}}(\boldsymbol{x_i})$, ovvero si considerano solo i punti e non la struttura locale del loro vicinato. La scelta compiuta da [Wan+19] è stata

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x_i}, \boldsymbol{x_j} - \boldsymbol{x_i}) \tag{4.8}$$

Tale scelta ha il pregio di combinare informazioni globali (la posizione $\boldsymbol{x_i}$ del vertice i da cui si diramano gli archi), con informazioni locali grazie a $\boldsymbol{x_j} - \boldsymbol{x_i}$.
In dettaglio, la funzione h scelta è

$$\boldsymbol{e'_{ijm}} = ReLU(\boldsymbol{\Theta_m} \cdot (\boldsymbol{x_j} - \boldsymbol{x_i}) + \boldsymbol{\Phi_m} \cdot \boldsymbol{x_i}) \tag{4.9}$$

dove $\boldsymbol{\Theta_m}, \boldsymbol{\Phi_m}$ sono set di parametri addestrabili, e la funzione di accumulazione $\boldsymbol{O}$ viene decisa

$$\boldsymbol{x_i'} = max\{\boldsymbol{e'_{ijm}}\} \tag{4.10}$$

dove l'operazione di massimo viene calcolata su tutti i j primi vicini.

4.2.2 Dynamic Graph Convolution

Nelle classiche CNN, il grafo di input è fissato. Invece nel caso del Dynamic Graph CNN (DGCNN), ad ogni layer il grafo viene ricalcolato in base ai k primi vicini *nello spazio delle Features*. Questa precisa caratteristica permette di ottenere relazioni globali di diametro pari all'intera PointCloud. Per cui, ad ogni layer l, ci sarà un differente grafo $\boldsymbol{G^l = (V^l, E^l)}$. In sostanza il modello impara come costruire il grafo di vicinanza piuttosto che prenderlo come fissato.

4.3 Applicazione dell'architettura DGCNN al modello di Vicsek

4.3.1 Il Dataset

Il Dataset utilizzato nello studio contiene 7360 eventi, ciascuno costituito da una PointCloud di F=6 dimensioni: le prime tre sono le coordinate cartesiane tridimensionali, le altre tre le velocità. Quindi alla DGCNN vengono date le 3 coordinate cartesiane come posizioni, e tutte le F=6 dimensioni come Features. I valori di m simulati sono m=$\{6,7,8,9\}$, numeri interi vicini a m=7.8 misurato in natura [Bia+12]. E' una approssimazione il fatto che ogni particella segua un numero di vicini fissato per tutti e costante ad ogni passo, ma si è seguita questa strada per semplicità. Un miglioramento semplice potrebbe consistere nell'estrarre da una opportuna distribuzione di probabilità il numero di vicini seguiti da un esemplare ad un dato istante di tempo; una opzione più complessa potrebbe basarsi sul campo visivo dell'uccello, ovvero su quanti altri uccelli riesce effettivamente a vedere ad ogni istante di tempo.
Per ciascun valore di m, si è campionato in modo uniforme l'intervallo dell'errore $\eta \in [0.05,0.40]$. In tutto sono state fatte termalizzare 320 simulazioni indipendenti, e ciascuna è stata campionata 23 volte a intervalli di tempo abbastanza grandi da ottenere configurazioni sensibilmente differenti. Per uno studio ancora più rigoroso la cosa migliore sarebbe prendere una sola PointCloud da ciascun sistema termalizzato, per poi ricominciare da capo, o altrimenti attendere tra due campionamenti un tempo dell'ordine di quello di termalizzazione. Ma per ragioni di limitatezza di risorse computazionali, è stato preferito disporre di 7360 eventi diversi piuttosto che averne solamente 320 completamente indipendenti. In questo modo il Dataset contiene un numero di PointClouds simile a quello dello studio originale della DGCNN [Wan+19].
A ciascuna PointCloud sono assegnate 3 labels: la polarizzazione ϕ e il coefficiente di correlazione α, misurate in modo numerico, e il numero di vicini m, parametro noto a priori poiché dato in input a ciascuna simulazione. Le prime due si studiano in modalità regressione, la terza tramite classificazione.

4.3.2 Task di regressione

Si addestra un modello DGCNN ad analizzare le PointClouds e ad ottenerne due misure: la polarizzazione ϕ e il coefficiente di correlazione α. Entrambe queste grandezze sono calcolabili tramite algoritmi numerici, la prima in modo molto semplice, la seconda in modo più articolato e computazionalmente costoso. Poiché sono

grandezze numeriche e non categoriche, si addestra la rete in modalità regressione, e la misura della precisione della regressione è data dalla MSE (Mean Squared Error).

La polarizzazione ϕ è una misura che dipende esclusivamente dalle velocità delle particelle, e non dalla loro posizione (Eq. 2.5). Pertanto, differentemente dagli altri casi, si utilizzano solamente le tre componenti delle velocità sia come posizioni, che come features. Così facendo si semplifica il lavoro della rete DGCNN, che non deve analizzare informazioni non direttamente utili al fine della misura.

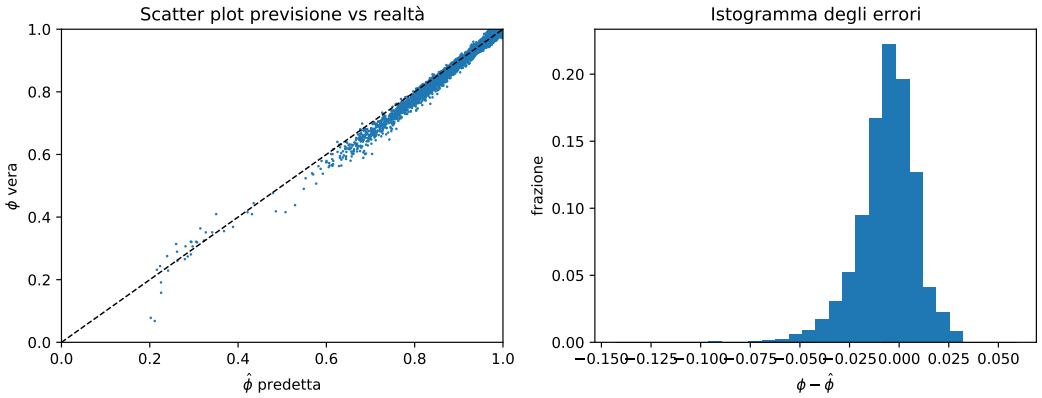

Figura 4.1. a) Dallo Scatter plot è possibile osservare come il modello ha imparato a prevedere in modo preciso e accurato la polarizzazione degli stormi del Test Set. Le misure si addensano sulla linea tratteggiata nera y=x. b) La precisione si valuta dalla $MSE = 2.6 \cdot 10^{-4}$, per cui la deviazione standard sul Test Set è $SD = 5 \cdot 10^{-2}$. Poiché le polarizzazioni valgono circa 1, l'errore medio è dell'ordine dello 0.5%.

Il coefficiente di correlazione α, dato che si ottiene a partire dalla funzione di correlazione C(r) (Eq. 2.6), dipende sia dalle posizioni che dalle velocità. Pertanto vengono utilizzate tutte le F=6 componenti di ciascun punto della PointCloud.

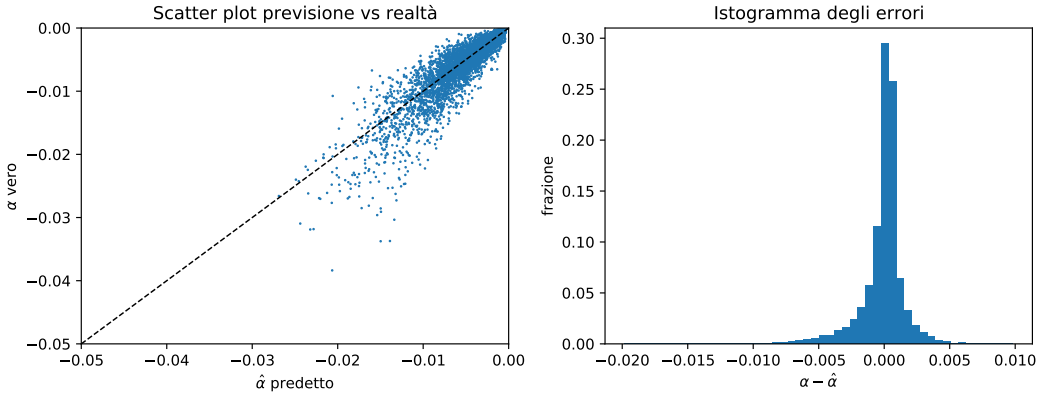

Figura 4.2. a) Lo scatter plot mostra che il modello ha imparato a prevedere il valore di α, ma a differenza del caso della polarizzazione, la misura è rumorosa, ovvero accurata ma non precisa. b) Si ottiene sul Test Set $MSE = 3.4 \cdot 10^{-6}$, ovvero una deviazione standard $SD = 1.8 \cdot 10^{-3}$. Ad esempio per un valore di α tipico di -0.01, questo implica un errore medio di circa il 15-20%.

4.3.3 Task di classificazione

A differenza delle due grandezze ϕ ed α descritte in precedenza, la misura di m (numero di vicini ai quali si allinea un uccello, Eq. 2.4) non è facilmente calcolabile. Sono state ottenute misure su stormi reali utilizzando metodologie articolate e molto differenti tra di loro. Ad esempio è stato trovato $m = (6.5 \pm 0.9)$ [Bal+08], oppure $m = 7.8$ utilizzando un modello a massima entropia [Bia+12].
Nel caso preso in esame, poiché sono stati simulati 4 valori di m interi, si addestra la rete in modalità classificazione, dove ciascun valore m rappresenta una classe.

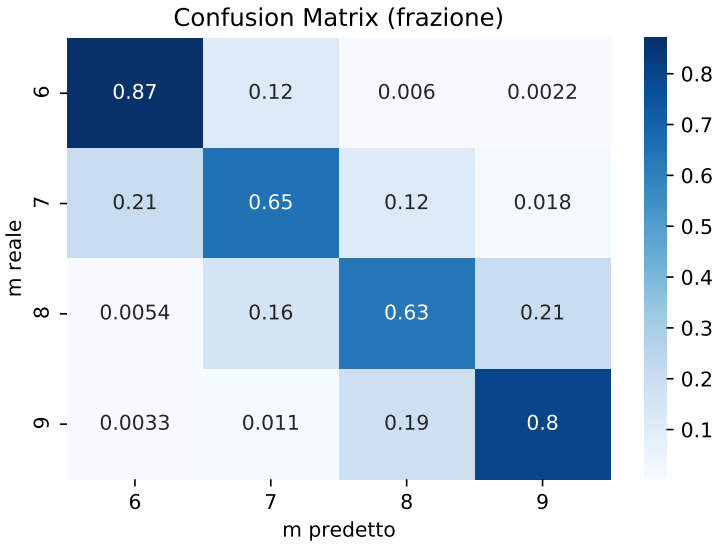

Figura 4.3. Confusion Matrix sul Test Set. Ciascun numero indica la frazione di eventi che sono stati classificati in una determinata classe. Viene predetto il valore di m corretto tra il 65% e l'85% delle volte. E' notevole il fatto che il 99% degli eventi sono stati classificati o nella classe corretta, oppure in una immediatamente adiacente.

Capitolo 5

Conclusioni

Il modello di Vicsek è il più semplice modello matematico in grado di spiegare i comportamenti collettivi biologici osservabili in natura. Al modello base è possibile aggiungere regole di funzionamento, ma ogni aggiunta implica l'assunzione che il sistema biologico segua davvero tale regola. Pertanto uno studio più approfondito richiede una analisi più sofisticata, e una attenta comparazione dei risultati ottenuti con i dati sul campo.
Anche nella semplicità della sua originaria formulazione, il modello di Vicsek mostra comportamenti fortemente non triviali [Gin16]. Inoltre la raccolta dati sul campo ha complicazioni di natura tecnica, in particolare risulta complicato il tracciamento tridimensionale di ogni particella ad ogni istante di tempo.
Nel presente studio si è preso in esame il modello di Vicsek topologico con le regole di base, e si è analizzato lo spazio dei parametri nei dintorni di quanto misurato in natura per la specie Sturnus Vulgaris. Dall'implementazione numerica si sono ottenuti risultati concordi sia con quanto dimostrato algoritmicamente nello studio originale [Vic+95], sia con quanto osservato in natura in stormi reali [Bia+12]. Sono verificati sia comportamenti più semplici, come la transizione di fase tra sistema polarizzato e sistema caotico, sia comportamenti non triviali come la proporzionalità tra lunghezza di correlazione ξ e taglia L.
Infine l'applicazione di metodi di Machine Learning, nella fattispecie del modello DGCNN, ha mostrato una interessante capacità di compiere misure numericamente complicate o addirittura non completamente attuabili senza l'utilizzo di assunzioni o teorie sofisticate, come la determinazione del numero di vicini m seguiti da ogni particella.
Ci sono molte evoluzioni possibili al presente studio. Ovviamente complicare le regole della simulazione numerica in modo adeguato renderebbe gli stormi simulati più simili a quelli reali. Inoltre una maggiore potenza di calcolo consentirebbe di avere eventi completamente indipendenti, ad esempio 7360 simulazioni indipendenti invece di 320 da cui campionare 23 eventi ciascuna. Ma l'evoluzione più interessante potrebbe essere applicare ai dati ottenuti sul campo il modello già addestrato tramite PointClouds simulate. In particolare, la misura di m numero di vicini è la più adatta poiché per le altre esistono metodi algoritmici chiari e ben definiti. In questo modo si potrebbe accostare ai vari metodi utilizzati in passato ([Bal+08], [Bia+12]) un modo nuovo e completamente differente di effettuare tale misura su stormi reali.

Bibliografia

[Vic+95] Tamás Vicsek et al. “Novel Type of Phase Transition in a System of Self-Driven Particles”. In: *Physical Review Letters* 75.6 (ago. 1995), pp. 1226–1229. ISSN: 1079-7114. DOI: 10.1103/physrevlett.75.1226. URL: http://dx.doi.org/10.1103/PhysRevLett.75.1226.

[Bal+08] M. Ballerini et al. “Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study”. In: *Proceedings of the National Academy of Sciences* 105.4 (gen. 2008), pp. 1232–1237. ISSN: 1091-6490. DOI: 10.1073/pnas.0711437105. URL: http://dx.doi.org/10.1073/pnas.0711437105.

[Cav+10] A. Cavagna et al. “Scale-free correlations in starling flocks”. In: *Proceedings of the National Academy of Sciences* 107.26 (giu. 2010), pp. 11865–11870. ISSN: 1091-6490. DOI: 10.1073/pnas.1005766107. URL: http://dx.doi.org/10.1073/pnas.1005766107.

[Bia+12] W. Bialek et al. “Statistical mechanics for natural flocks of birds”. In: *Proceedings of the National Academy of Sciences* 109.13 (mar. 2012), pp. 4786–4791. ISSN: 1091-6490. DOI: 10.1073/pnas.1118633109. URL: http://dx.doi.org/10.1073/pnas.1118633109.

[Gin16] Francesco Ginelli. “The Physics of the Vicsek model”. In: *The European Physical Journal Special Topics* 225.11-12 (nov. 2016), pp. 2099–2117. ISSN: 1951-6401. DOI: 10.1140/epjst/e2016-60066-8. URL: http://dx.doi.org/10.1140/epjst/e2016-60066-8.

[Qi+17] Charles R. Qi et al. *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*. 2017. arXiv: 1612.00593 [cs.CV].

[Wan+19] Yue Wang et al. *Dynamic Graph CNN for Learning on Point Clouds*. 2019. arXiv: 1801.07829 [cs.CV].