PSO su Darwin Dataset

Alfieri Giuseppe; Cannavale Achille; Colacicco Nunziamaria; La Torre Noemi

Università Degli Studi di Cassino e del Lazio Meridionale

Corso Di Laurea Magistrale in Ingegneria Informatica

Abstract: Questo studio analizza l'impatto dei parametri dell'algoritmo Particle Swarm Optimization (PSO) nella selezione delle feature sul dataset DARWIN, il quale raccoglie caratteristiche della scrittura a mano per lo studio dell'Alzheimer. L'obiettivo è valutare l'influenza di tali parametri su convergenza, stabilità ed efficienza computazionale, al fine di identificare un sottoinsieme ottimale di feature. Questo permetterebbe di migliorare la diagnosi riducendo la complessità del modello, offrendo così un approccio più efficace all'analisi dei dati nel contesto medico.

1 Introduzione

In questa tesina analizziamo l'impatto dei parametri dell'algoritmo Particle Swarm Optimization (PSO) per la selezione delle feature sul dataset DARWIN. Questo dataset raccoglie caratteristiche della scrittura a mano per lo studio dell'Alzheimer, offrendo un contesto reale per valutare le prestazioni e la convergenza dell'algoritmo. L'obiettivo è esaminare come i parametri di PSO influenzino la convergenza, la stabilità e l'efficienza computazionale, identificando al contempo un sottoinsieme ottimale di feature per migliorare la diagnosi e ridurre la complessità del modello. Nello specifico, sono state adottate le seguenti strategie:

- **Rappresentazione Binaria**: la selezione delle feature è stata modellata tramite una rappresentazione binaria.
- **Funzione di Fitness**: è stata definita una funzione di fitness basata sulla correlazione di Spearman tra le feature selezionate. A differenza della correlazione di Pearson, quella di Spearman è meno sensibile alla presenza di outlier nel dataset.
- **Dimensione particelle**: sono stati utilizzati due diverse dimensioni del subset di features, ovvero 22 (5% #total features) e 45 (10% #total features). Tuttavia, si è osservato un miglioramento della fitness globale con un subset più compatto. Per questo motivo, le analisi successive sono state condotte utilizzando 22 feature, un valore coerente anche con l'euristica che suggerisce di scegliere un numero di feature pari alla radice quadrata del totale delle feature disponibili.

2 PSO

Il *Particle Swarm Optimization* (PSO) è un algoritmo di ottimizzazione ispirato al comportamento collettivo di sciami di particelle, come stormi di uccelli o banchi di pesci. Proposto da Kennedy e Eberhart nel 1995, il PSO sfrutta un insieme di particelle che esplorano lo spazio delle soluzioni aggiornando le proprie posizioni in base a una funzione obiettivo. Ogni particella adatta la sua traiettoria considerando la propria esperienza passata e quella delle altre particelle, utilizzando le equazioni:

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 (p_i - x_i^t) + c_2 r_2 (g - x_i^t) \quad (1)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (2)$$

dove v_i^t è la velocità della particella i al tempo t, x_i^t è la sua posizione, p_i è la migliore posizione trovata dalla particella, g è la migliore posizione globale trovata dall'intero sciame, e ω, c_1, c_2 sono parametri che regolano l'inerzia e l'influenza dei migliori valori locali e globali. Il nostro algoritmo è stato implementato in maniera tale da eseguire 30 RUN diverse, incrementando il seed e selezionando alla fine solo la migliore esecuzione in termini di global best trovato.

3 Scenari

3.1 Scenario Dimensione Sciame

In questo scenario, sono stati fissati i seguenti parametri:

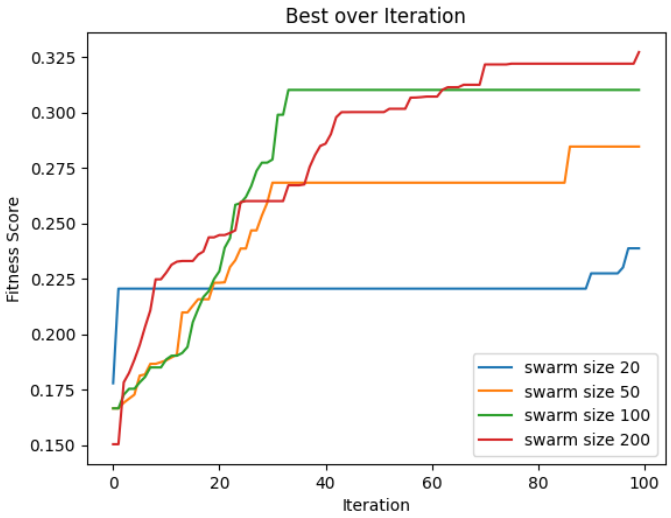

Figura 1: Fitness Scores con diversi Swarm Size

- Inerzia (w) = 0.7
- Coefficiente Cognitivo (c_1) = 2.0
- Coefficiente Sociale (c_2) = 2.0
- Iterazioni Massime = 100

Mentre abbiamo variato le dimensioni dello sciame tra [20, 50, 100, 200, 500]. Abbiamo notato (Figura 1) come all'aumentare delle particelle, la velocità di convergenza diminuisce, arrivando a valori di global best fitness superiori. Inoltre, si nota un progressivo aumento del costo computazionale, sia a livello temporale, partendo da circa $2s$ nel caso di uno *swarm size = 20*, fino ad arrivare a circa $30s$ nel caso di uno *swarm size=200*; e sia dal punto di vista dell'occupazione di memoria, che parte da $150Mb$ nel primo caso, fino ad arrivare a $200Mb$ nel caso di uno *swarm size=200*. Per quanto riguarda il caso di *swarm size = 500*, esso non è stato implementato, poiché un numero di particelle maggiore rispetto al numero di feature nel dataset avrebbe comportato un'uscita inefficiente. Inoltre, abbiamo riscontrato che il set di features selezionate è molto variabile tra una configurazione e l'altra.

3.2 Scenario Coefficienti PSO

In questo scenario è stata fissata la dimensione dello sciame a 100, mentre vengono variati i coefficienti di Inerzia, Cognitivo e Sociale:

- Inerzia (w) : [0.4, 0.7]
- Coefficiente Cognitivo (c_1) : [1.0, 2.5]

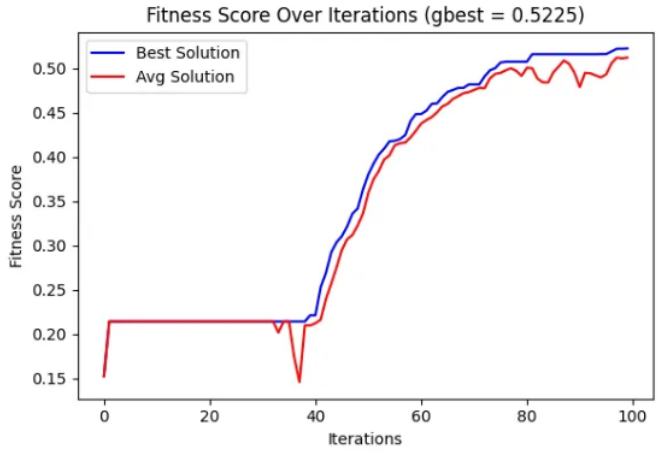

Figura 2: Fitness Score: $w = 0.4$; seed=58

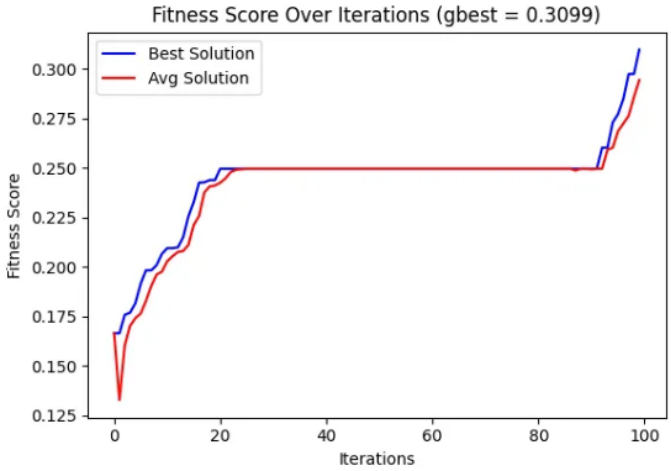

Figura 3: Fitness Score: $w = 0.7$; seed=68

- Coefficiente Sociale (c_2) : [1.0, 2.5]

3.2.1 Inerzia

Come possiamo vedere dai grafici (Figura 2, Figura 3), nel primo caso ($w = 0.4$), l'algoritmo raggiunge uno best score maggiore rispetto al secondo caso ($w = 0.7$). Questo risultato è dovuto al fatto che un valore più basso di inerzia favorisce maggiormente l'exploration dello spazio di ricerca, permettendo alle particelle di muoversi verso posizioni più favorevoli. In particolare, possiamo notare come nel secondo caso, 100 iterazioni non sono sufficienti a raggiungere una fase di convergenza, indicando che l'algoritmo tende a concentrarsi troppo sulle posizioni già esplorate, impedendo una ricerca efficace di soluzioni migliori.

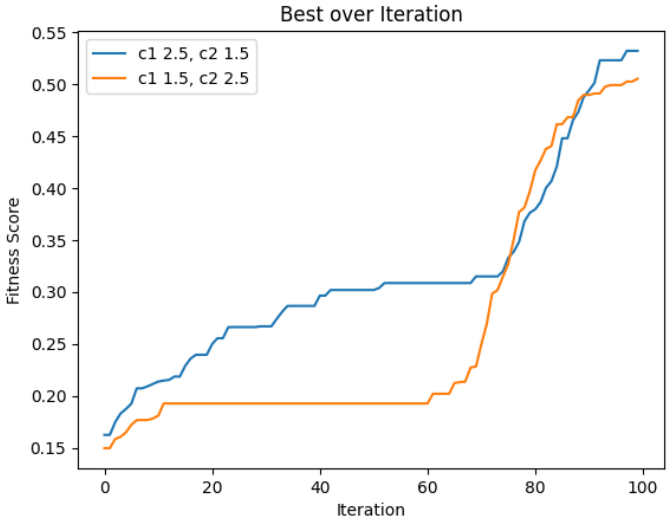

Figura 4: Fitness Score con diversi valori di c_1 e c_2

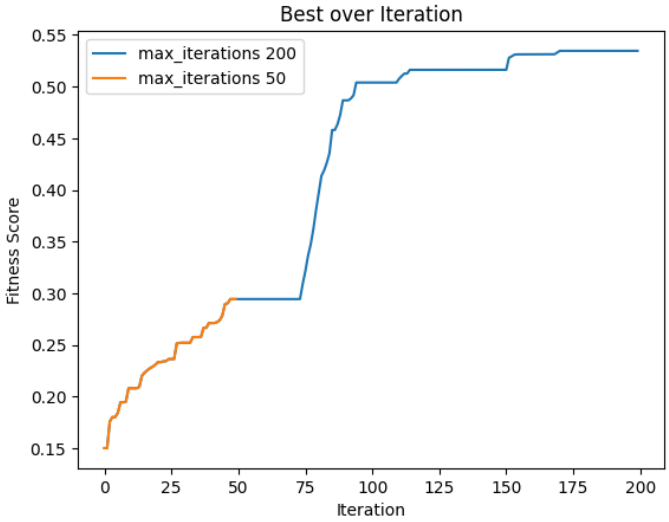

Figura 5: 50 e 200 iterazioni massime; seed=56

3.2.2 Coefficienti Cognitivo e Sociale

Fissando il coefficiente di inerzia ad un valore medio ($w = 0.6$) e uno *swarm size = 100* abbiamo variato i valori dei coefficienti c_1 e c_2. Da come si può evincere dal grafico (Figura 4) usando un coefficiente cognitivo (c_1) più basso di quello sociale (c_2), le particelle tenderanno ad esplorare maggiormente lo spazio delle soluzioni, dando più peso all'esperienza individuale delle singole particelle. Viceversa, usando un coefficiente sociale più grande di quello cognitivo, notiamo come le particelle si muovono in maniera più coordinata tra loro, raggiungendo più presto ad una soluzione, ma esplorando di meno lo spazio delle soluzioni.

3.3 Scenario Criteri di Stop

3.3.1 Iterazioni Massime

In questa prima fase dello scenario sono state variate le iterazioni massime:

- 50
- 200

Come è palese notare dal grafico (Figura 5), le particelle, a fronte di 50 iterazioni, non riescono a raggiungere la convergenza, cosa che viene fatta a seguito di 200 iterazioni, raggiungendo una soluzione molto migliore.

3.3.2 Early Stop e Tolleranza

In questa seconda fase dello scenario sono stati variati i parametri:

- Threshold : [10, 30]
- Tolleranza : [1e-4, 1e-6]

Variando questi parametri, l'algoritmo si fermerà non appena la funzione di Fitness Score rimarrà confinata in un range definito dalla Tolleranza, per un numero di iterazioni pari al Threshold. Dato che il nostro algoritmo è stato impostato in modo tale da selezionare la migliore RUN tra quelle effettuate, dove per migliore si intende quella che raggiunge il best score più alto, verranno sempre selezionate le RUN in cui si raggiungono le 100 iterazioni massime (Figura 6).

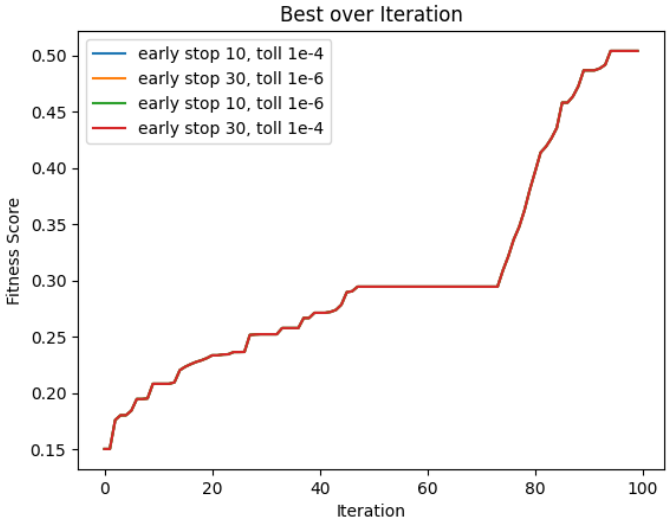

Figura 6: Early Stop

4 Valutazione Finale

In questa ultima sezione, riportiamo i grafici relativi alla RUN, con la conformazione di parametri riportati nelle figure (figure 7, 8), che ha raggiunto il Fitness Score migliore (0.5345):

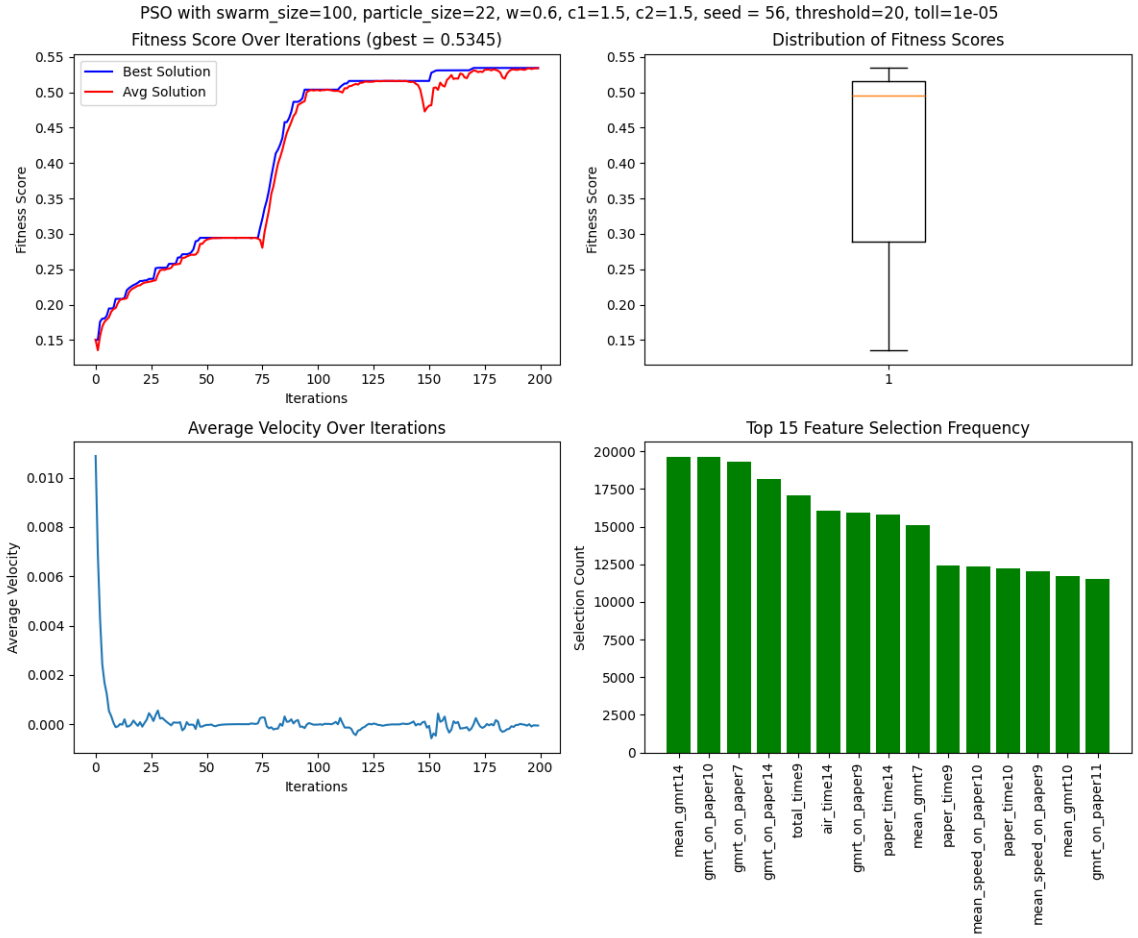

Figura 7: Miglior Esperimento - (a) Fitness Score, (b) Fitness Distribution, (c) Avg Velocity, (d) Top 15 Features

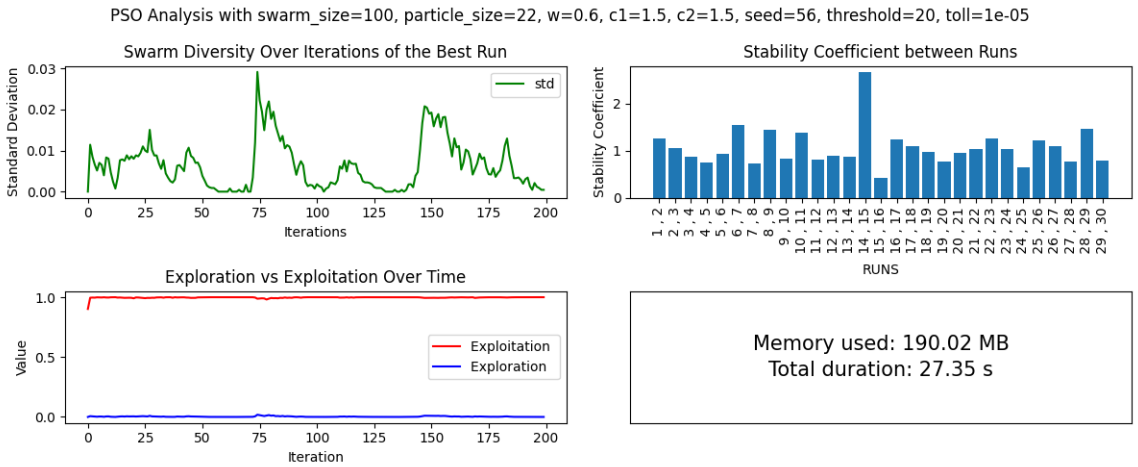

Figura 8: Miglior Esperimento - (a) Swarm Diversity, (b) Stability Coefficient, (c) Exploration Vs Exploitation, (d) Computational Complexity