1. Computação Swarm

A Computação *Swarm* é um conjunto de algoritmos inspirados na natureza, nomeadamente do movimento de animais, como abelhas, formigas, lobos, baleias, entre outros. Esta abordagem permite otimizar redes neurais, tornando-as mais eficientes, por conseguir lidar com problemas de grande escala, uma vez que consegue evitar ótimos locais ao explorar o ambiente de forma global.

1.1. Características

Swarm são compostos por múltiplos agentes, ou partículas, que trabalham em conjunto para encontrar soluções para problemas complexos, apresentam-se de seguida as suas características:

- Descentralização: cada agente toma decisões com base nas informações locais;
- Comportamento emergente: a solução surge da interação entre agentes;
- Adaptação e aprendizagem: o sistema evolui com o tempo, há adaptação conforme as condições do ambiente.

1.2. Tipos

Swarm tem sido usada como uma ferramenta no treino e otimização de redes neurais, especialmente para otimizar parâmetros como os pesos e os Hiper parâmetros do modelo. Segue-se uma descrição de 3 técnicas mais utilizadas, derivadas deste paradigma:

- *Particle Swarm Optimization* (PSO): Baseado em partículas que simulam o movimento de um enxame. Cada partícula é uma solução candidata que se move no espaço, ajusta a sua posição com base na experiência pessoal e na experiência do grupo. O objetivo é minimizar ou maximizar uma função de custo/fitness. É usado para ajustar os pesos das redes neurais, especialmente quando o problema de otimização é complexo e multidimensional;
- *Ant Colony Optimization* (ACO): Baseado no comportamento de formigas que deixam trilhas de feromona para marcar caminhos para recursos. O algoritmo simula esse processo para resolver problemas de otimização combinatória, como o Problema do Caixeiro Viajante;
- *Artificial Bee Algorithm*: Inspirado no comportamento das abelhas que coletam néctar, simula o processo de procura e exploração de soluções ótimas. Equilibra a exploração de novas áreas no espaço e o refinamento das melhores soluções já encontradas, de forma a garantir um balanço eficiente entre diversificação e intensificação.

1.3. Aplicações

Segue-se uma descrição de algumas das principais aplicações de *Swarm* no contexto das redes neurais:

- Otimização dos pesos da rede neural: Com o PSO ou ACO, os pesos da rede podem ser otimizados de forma global, já que esses algoritmos exploram amplamente o espaço de soluções e podem evitar ótimos locais;
- Otimização de Hiper parâmetros: PSO, pode ser usado para procurar a melhor configuração de Hiper parâmetros, sem a necessidade de uma pesquisa exaustiva ou de técnicas de validação cruzada demoradas;

Book: Handbook Of Swarm Intelligence
Article: A Systematic and Meta-Analysis Survey of Whale Optimization Algorithm

- Treino de CNN: PSO pode encontrar a melhor combinação de parâmetros para as camadas convolucionais, de forma a maximizar a precisão e minimizar o erro de treino;
- Treino Paralelo: O uso de múltiplos agentes, ou partículas, permite que múltiplas soluções sejam avaliadas ao mesmo tempo, de maneira a acelerar o processo de otimização.

1.4. Vantagens

- Exploração Global: o espaço é explorado de forma mais eficiente, acaba por conseguir soluções que são ignoradas pelos métodos tradicionais;
- Evita Mínimos Locais: os múltiplos agentes, com o seu comportamento descentralizado, conseguem evitar ficar presos em ótimos locais;

1.5. Desvantagens

- Custo computacional elevado: A necessidade de avaliar múltiplas soluções simultaneamente pode ser intensiva em termos de tempo e recursos computacionais, especialmente para problemas de alta dimensionalidade;
- Convergência Lenta: Embora eficazes na exploração global, os algoritmos de *Swarm* podem apresentar uma convergência mais lenta em comparação a métodos tradicionais, principalmente em problemas simples.

Book: Handbook Of Swarm Intelligence
Article: A Systematic and Meta-Analysis Survey of Whale Optimization Algorithm

2. Swarm Algorithm

O *Whale Optimization Algorithm* (WOA) é um método de otimização baseado no comportamento das baleias. Esse algoritmo foi inspirado pelo processo cooperativo das baleias na captura de presas e pode ser aplicado para resolver problemas de otimização complexos.

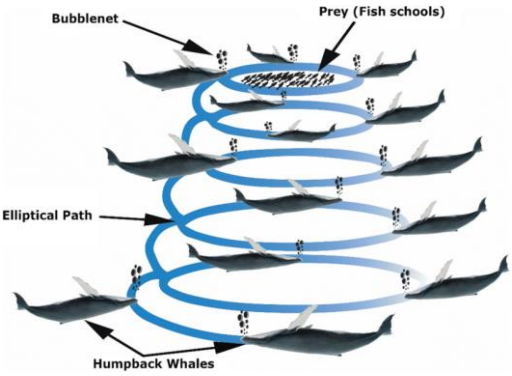

Figura 1 Bubble-net

Cada baleia na população representa uma possível solução para o problema. O movimento das baleias é inspirado pelo movimento em espiral que simula o padrão helicoidal de aproximação das baleias em direção à presa e movimento em direção ao líder, em que as baleias ajustam as suas posições com base na localização de outras que estão mais próximas da solução ótima. A baleia líder, que representa a melhor solução, é identificada e usada como referência para a atualização das posições das restantes baleias. O algoritmo é dividido em três fases: aproximar e cercar a presa, método de ataque *bubble-net* e procura por presas. O WSA utiliza fatores como a taxa de convergência, distância da espiral e parâmetros de ajuste aleatório para controlar a procura global e local.

2.1. Comparação entre WSA e PSO

Embora o WSA compartilhe similaridades com o PSO em termos de inspiração por natureza e procura por soluções globais, há algumas diferenças:

Características	Whale Swarm Algorithm (WSA)	Particle Swarm Optimization (PSO)
Motivação	Movimento em espiral das baleias	Movimento de partículas de um enxame
Exploração	Combina movimento helicoidal	Com base no histórico global e local das partículas
Complexidade Computacional	Maior devido à necessidade de cálculo do movimento espiral	Menor pela simplicidade das fórmulas
Robustez	Mais robusto pelo seu tipo de Exploração	Pode estagnar nos mínimos locais se o problema for mais complexo

2.2. Vantagens

- Eficiência em problemas multimodais: A abordagem helicoidal permite que o WSA evite mínimos locais e explore melhor o ambiente;
- Robustez: Melhor desempenho em ambientes complexos.

2.3. Desvantagens

- Complexidade Computacional: A implementação do movimento em espiral adiciona custo computacional;
- Convergência Lenta: Em alguns casos, o algoritmo pode ser mais lento do que o PSO para problemas simples ou com baixa dimensionalidade.

Book: Handbook Of Swarm Intelligence
Article: A Systematic and Meta-Analysis Survey of Whale Optimization Algorithm

3. Aplicação do algoritmo

Os parâmetros selecionados para a otimização da função de *Ackley* nos algoritmos PSO e WOA são:

- Número de partículas: Foi testado com valores de 5, 15, 30 e 55 partículas/*whales*;
- Número de iterações: Valores de 10, 30, 70 e 150;
- Dimensões: 2D e 3D.

Os parâmetros do PSO incluem:

- w_max (Inércia máxima): 0.9
- w_min (Inércia mínima): 0.4
- c1 (Coeficiente de aprendizado cognitivo): 2
- c2 (Coeficiente de aprendizado social): 2

A Tabela 1 mostra a melhor posição e pontuação alcançada com o algoritmo PSO, estando apenas representados os melhores resultados por nº de partículas, como se pode observar para todas as partículas, o melhor resultado foi alcançado com o nº máximo de iterações testadas, no caso 150.

Tabela 1 Resultados do PSO na função Ackley

	Partículas	Iterações	Melhor Posição	Melhor Pontuação	Tempo (s)
PSO 2D	5	150	[0.00012687, -0.00107261]	1.1665906731482472e-06	0.0099
PSO 2D	15	150	[-9.41863616e-09, -3.20502832e-08]	1.1159313574844114e-15	0.0172
PSO 2D	30	150	[1.40804266e-07, 7.20250222e-08]	2.5013445028717245e-14	0.0270
PSO 2D	**55**	**150**	**[2.53205671e-08, 2.81371809e-09]**	**6.49048125305632e-16**	**0.0562**
PSO 3D	5	150	[-4.23063445e-05, 2.64962548e-04, -5.15660332e-04]	3.379005563526719e-07	0.0107
PSO 3D	15	150	[-0.00128006, -0.00193696, 0.00022682]	5.441821447583096e-06	0.0181
PSO 3D	30	150	[-1.77337299e-06, 1.12094807e-07, -1.64500326e-06]	5.863452734250126e-12	0.0388
PSO 3D	55	150	[2.38340836e-08, 1.02079135e-07, -5.57370794e-08]	1.4094835455532198e-14	0.0723

Os parâmetros do WOA incluem:

- a (Coeficiente de controle para o movimento de exploração): 2
- a2 (Valor para controle da exploração/exploração): 0.1
- r1 e r2 (Variáveis aleatórias): Gerados aleatoriamente dentro do intervalo [0, 1]
- C (Coeficiente de deslocamento): 2

Book: Handbook Of Swarm Intelligence
Article: A Systematic and Meta-Analysis Survey of Whale Optimization Algorithm

Também no algoritmo WOA, quanto maior o número de iterações, 150, maior é a pontuação alcançada, como se observa na Tabela 2.

Tabela 2 Resultados do WOA na função Ackley

	Whales	Iterações	Melhor Posição	Melhor Pontuação	Tempo (s)
WOA 2D	5	150	[-6.01074667e-28, 1.90751240e-27]	3.999894295435154e-54	0.0677
WOA 2D	15	150	[4.65076475e-33, 3.48779537e-33]	3.3794329278693545e-65	0.3205
WOA 2D	30	150	[1.58651443e-43, -7.55393286e-43]	5.957892970865273e-85	0.3124
WOA 2D	**55**	**150**	**[-2.18160518e-48, -4.98542873e-49]**	**5.0079461585627186e-96**	**0.4107**
WOA 3D	5	150	[-3.02290079e-17, -3.78767790e-16, -6.13511328e-16]	5.207749818058027e-31	0.0238
WOA 3D	15	150	[-5.42484767e-23, -8.92416192e-24, 2.75649270e-22]	7.900505766715188e-44	0.0719
WOA 3D	30	150	[-2.75273263e-33, -1.13734598e-34, -1.22690175e-33]	9.095760399788669e-66	0.1674
WOA 3D	55	150	[-3.65722148e-35, 4.43091089e-36, -3.11427070e-35]	2.3270280657301487e-69	0.2792

Como se pode observar nas tabelas, ambos os algoritmos apresentam melhores resultados quando se trata de um espaço 2D. O WOA apresenta valores mais próximos de 0, no entanto requer mais tempo de execução.

A Figura 2 seguinte mostra as pontuações dos 4 experimentos, relativamente ao número de partículas/*whales*. No início do experimento, o PSO 3D e WOA 2D, apresentam valores mais distantes de 0. A partir das 15 partículas, a curva decresce para valores de pontuação próximos de 0, exceto para PSO 3D que só consegue atingir os mesmos valores quando possui 30 partículas.

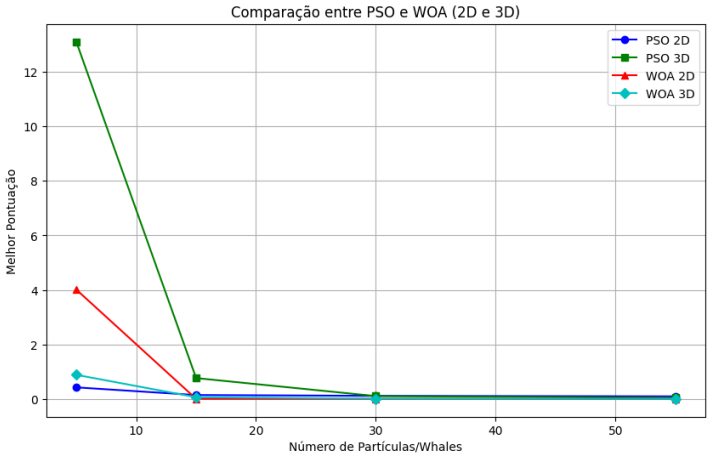

Figura 2 Comparação entre PSO e WOA (2D e 3D)

Book: Handbook Of Swarm Intelligence
Article: A Systematic and Meta-Analysis Survey of Whale Optimization Algorithm

4. Otimização de híper-parâmetros

Para a otimização de Hiper parâmetros, o *Dataset* utilizado para o experimento é o CoMNIST., que contém 34 classes de imagens de letras do alfabeto cirílico, na Figura 3 apresenta-se o nome da classe, e uma imagem *random* da respetiva classe:

Figura 3 Imagem random do Dataset

A figura é referente às seguintes letras:

Tabela 3 Letras do Dataset

А	**В**	**С**	**Е**	**Н**	**І**	**К**
М	**О**	**Р**	**Т**	**Х**	**Б**	**Г**
Д	**Ё**	**Ж**	**З**	**И**	**Й**	**Л**
П	**У**	**Ф**	**Ц**	**Ч**	**Ш**	**Щ**
Ъ	**Ы**	**Ь**	**Э**	**Ю**	**Я**	

Uma arquitetura CNN, de três camadas, como se apresenta na *Figura 4*, com uma divisão de 70% Treino, 10% Validação e 20% Teste, aplicada ao *Dataset*, atinge uma *accuracy* de teste de **54.39%,** para 10 iterações.

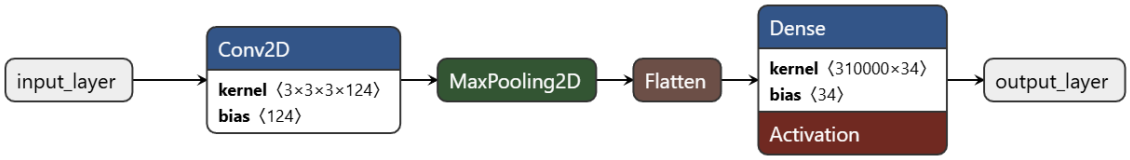

Figura 4 Modelo CNN, em netron

Parâmetros e híper parâmetros foram ajustados dentro de intervalos definidos usando o PSO e WOA para encontrar a configuração ideal que maximiza a *accuracy* do modelo.
Os experimentos foram conduzidos utilizando diferentes combinações de partículas/*whales* e iterações, como se apresenta de seguida:

- Número de Partículas: 1, 2, 4 e 8.
- Número de Iterações: 3, 6, 9, 12 e 15.

Foram selecionados 3 Hiper parâmetros da arquitetura CNN, que variam num intervalo de:

- Número de Filtros na Camada Convolucional (Conv Filters): [32, 128]

- Número de Unidades na Camada Densa (Dense Units): [32, 128]
- Tamanho do Lote (Batch Size): [16, 64]

Para cada iteração, a partícula/*whale* avalia a qualidade da solução com base numa função *fitness*, do conjunto de validação; as partículas ajustam-se às posições que consideram as melhores soluções locais e globais; e processo termina após o número definido de iterações, retornando a melhor combinação encontrada.
Alguns experimentos resultaram em erros de ResourceExhaustedError. Esses erros ocorreram devido à falta de memória disponível no ambiente do Jupyter, uma vez que o experimento lida com imagens de alta dimensão, 100x100 pixels. Inicialmente o experimento foi feito para imagens 32x32 pixels, mas, devido à complexidade do *Dataset*, os valores de *accuracy* eram demasiado baixos.
A Tabela 4 apresenta os resultados obtidos com o PSO:

Tabela 4 Resultados para PSO

Partículas	Iterações	Conv Filters	Dense Units	Batch Size	Accuracy	Tempo (min)
1	3	43	71	22	5,28%	6.03
1	6	115	87	23	68,3%	26.17
1	9	39	128	17	68,7%	21.31
1	12	82	111	55	69,7%	43.40
1	15	69	54	27	63,4%	30.57
2	3	81	128	37	70,1%	22.46
2	6	128	32	52	65,1%	37.33
2	9	110	112	19	69,4%	82.94
2	12	34	109	62	65,7%	59.89
2	15	76	93	30	68,1%	84.22
4	3	93	128	64	70,6%	42.78
4	6	128	128	64	69,6%	102.83
4	9	ResourceExhaustedError				
4	12	ResourceExhaustedError				
4	15	ResourceExhaustedError				
8	3	86	100	28	69,9%	83.94

Nota: Os valores dos Hiper parâmetros foram arredondados inteiro mais próximo.

Aumentar o número de partículas resultou numa melhoria ligeira na *accuracy,* indicando maior diversidade de soluções. Contudo, o custo computacional também aumentou proporcionalmente.
A melhor configuração com o PSO foi de 4 partículas, 3 iterações, 93 Conv Filters, 128 Dense Units, e Batch Size de 64, onde se obteve uma *accuracy* de 70,6%. No entanto, é de ressaltar que este experimento demorou quase o dobro do tempo de execução que o experimento de 2 partículas e 3 iterações, onde se obteve uma *accuracy* de 70,1%, ou seja, uma redução de apenas 0,5%, mas com um tempo de execução consideravelmente mais baixo. Este *trade-off* entre *accuracy* e custo computacional deve ser considerado ao ajustar o número de partículas e iterações, dependendo das restrições de tempo ou dos recursos computacionais disponíveis.
A tabela Tabela 5 apresenta os resultados obtidos com o PSO:

Tabela 5 Resultados para WOA

Whales	Iterações	Conv Filters	Dense Units	Batch Size	Accuracy	Tempo (min)
1	3	76	115	64	67,5%	10.67

Book: Handbook Of Swarm Intelligence
Article: A Systematic and Meta-Analysis Survey of Whale Optimization Algorithm

1	6	125	85	21	68,2%	26.05
1	9	79	116	64	69,8%	32.58
1	12	60	35	53	2,94%	24.96
1	15	119	65	57	67,8%	57.80
2	3	50	106	42	66,9%	17.79
2	6	99	73	43	66,6%	34.88
2	9	128	128	64	68,2%	76.74
2	**12**	**128**	**98**	**42**	**70,7%**	**98.95**
2	15	ResourceExhaustedError				
4	3	80	128	52	67,2%	25.89
4	6	91	98	51	68,7%	47.88
4	9	128	128	64	70,5%	180.30
4	12	ResourceExhaustedError				
4	15	ResourceExhaustedError				
8	3	ResourceExhaustedError				

Nota: Os valores dos Hiper parâmetros foram arredondados inteiro mais próximo.

Para o algoritmo WOA, aumentar o número de *whales* também resultou numa melhoria ligeira na *accuracy*, e o custo computacional também aumentou proporcionalmente.
A melhor configuração com o WOA foi de 2 *whales*, 12 iterações, 128 Conv Filters, 98 Dense Units, e Batch Size de 42, onde se obteve uma *accuracy* de 70,7%.
O PSO apresentou uma vantagem no custo computacional, tendo o valor da *accuracy* do WOA, valores ligeiramente superiores, mas pouco significativos.
Para o conjunto de teste, optei com escolher a melhor configuração obtida com o WOA, nomeadamente, 128 Conv Filters, 98 Dense Units, e Batch Size de 42. Com 10 iterações, o valor da *accuracy* para o conjunto de teste foi de **72.08%**, cuja Matriz de Confusão se apresenta na Figura 5, sendo superior ao Modelo CNN base aplicado anteriormente, cuja *accuracy* era de apenas **54.39%.**

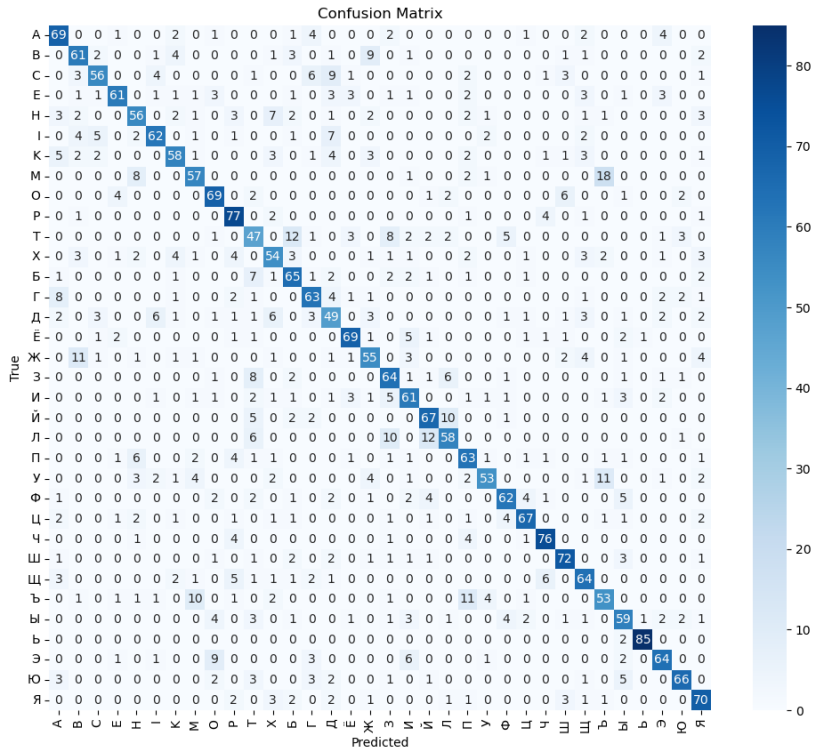

Figura 5 Matriz de Confusão do Modelo CNN com os melhores parâmetros

Book: Handbook Of Swarm Intelligence
Article: A Systematic and Meta-Analysis Survey of Whale Optimization Algorithm

5. Conclusão e Discussão de resultados

O trabalho teve como objetivo explorar e comparar a aplicação dos algoritmos *Particle Swarm Optimization* (PSO) e *Whale Optimization Algorithm* (WOA) para duas finalidades principais: (i) otimização da função de *benchmark Ackley* em espaços multidimensionais e (ii) ajuste de Hiper parâmetros em uma arquitetura CNN.

Os experimentos com a função de Ackley demonstraram a capacidade dos algoritmos PSO e WOA de encontrar soluções próximas do ótimo global, evidenciando:

- Para ambas as dimensões testadas, o WOA demonstrou maior eficiência inicial, convergindo rapidamente para soluções de alta qualidade conforme o número de *whales* e iterações aumentavam;
- PSO 3D, teve o pior desempenho, apenas convergiu para valores próximo de 0, com 30 partículas;
- Ambos os algoritmos apresentaram melhores desempenhos em espaços 2D.

Ao aplicar os algoritmos para a otimização de Hiper parâmetros numa CNN treinada no Dataset selecionado, os resultados mostraram que:

- O PSO e WOA, não têm uma divergência de valores significativa, mas o custo computacional é muito superior no WOA.
- Os experimentos conduzidos não foram suficientes para verificar que o aumento de número de partículas/*whales*, e as respetivas iterações têm benefícios na exploração do espaço, devido ao uso de imagens de grande dimensão.

Além disso, a comparação entre PSO e WOA evidencia suas características complementares: enquanto o WOA oferece maior estabilidade e *accuracy* nas soluções finais, o WOA apresenta vantagens em termos de rapidez na exploração inicial.

A Tabela 6 mostra uma síntese dos resultados obtidos no problema de minimização e maximização.

Tabela 6 Melhores resultados

Problema	Contexto	Algoritmo	Accuracy	Tempo	Observações
Minimização	Função Ackley (2D)	PSO	6.49048125305632e-16	0.0562	Convergência muito rápida
	Função Ackley (3D)	PSO	1.4094835455532198e-14	0.0723	Convergência muito lenta
	Função Ackley (2D)	WOA	5.0079461585627186e-96	0.4107	Convergência lenta, mas melhor solução
	Função Ackley (3D)	WOA	2.3270280657301487e-69	0.2792	Convergência rápida
Maximização	Hiperparametro CNN	PSO	70,6%	42.78	Custo computacional moderado e boa solução
	Hiperparametro CNN	WOA	70,7%	98.95	Custo computacional elevado e melhor solução

Book: Handbook Of Swarm Intelligence
Article: A Systematic and Meta-Analysis Survey of Whale Optimization Algorithm