WWW.IJRPR.COM

**International Journal of Research Publication and Reviews**

Journal homepage: www.ijrpr.com ISSN 2582-7421

# Implementation of Grey Wolf Optimizer in Determining the Best Parameters in K-NN to Improve Model Performance: An Experiment

***Said Al Afghani Edsa*** [a]

[a] *CBI, Jl. Jenderal Sudirman No.Kav. 32, RW.2, Karet Tengsin, Kecamatan Tanah Abang, Kota Jakarta Pusat, Daerah Khusus Ibukota Jakarta 10220*

**A B S T R A C T**

There are several ways to improve the performance of a model, such as determining the best parameters and selecting relevant variables. In this study, a procedure for determining the best model attribute (K) in terms of improving model performance will be given, namely the K-NN model, the K-NN model was chosen because it is relatively easy to understand in general. The best attribute (K) are determined using Grey Wolf Optimizer, we try to combine KNN- Grey Wolf Optimizer to get the best K, in this case study using experimental data we obtained accuracy : 0.98 and auc: 0.97 with parameter K = 11.

**Keywords:** Model parameters, K-NN, Grey Wolf Optimizer, Model Performance

## 1. Introduction

In principle, there are several approaches to improve the performance of a machine learning model, for example adding features as predictors, adding training samples, changing parameters or updating model parameters, improving or updating feature engineering processes, preprocessing data, and so on.

Many researchers have developed various methods to improve the performance of the model [6] [8], for example through the feature selection approach, while feature selection is used using various methods, one of which is using the meta heuristic approach. In this experimental study, a procedure will be built to determine the best attribute K of the K-NN model. It is understood that for the K-NN model itself, in determining K, it will certainly affect the performance of the model itself, apart from the characteristics of the data. If the K selection is done manually, we do not know at which K until K-NN really gives its best performance, that is the motivation for our study, where we combine K-NN and Grey Wolf Optimizer to get the optimal solution so that the K obtained is the optimal K, namely K which can provide the best performance.

## 2. Research Method

This research uses a literature study and experimental approach. Other researchers such as Seyedali Mirjalili in his research [9] developed GWO hybridization for the case of feature selection with a binary approach, inspired by that we try to develop a combination of GWO and K-NN to obtain the optimal K.

The process flow carried out in this experiment is given below:

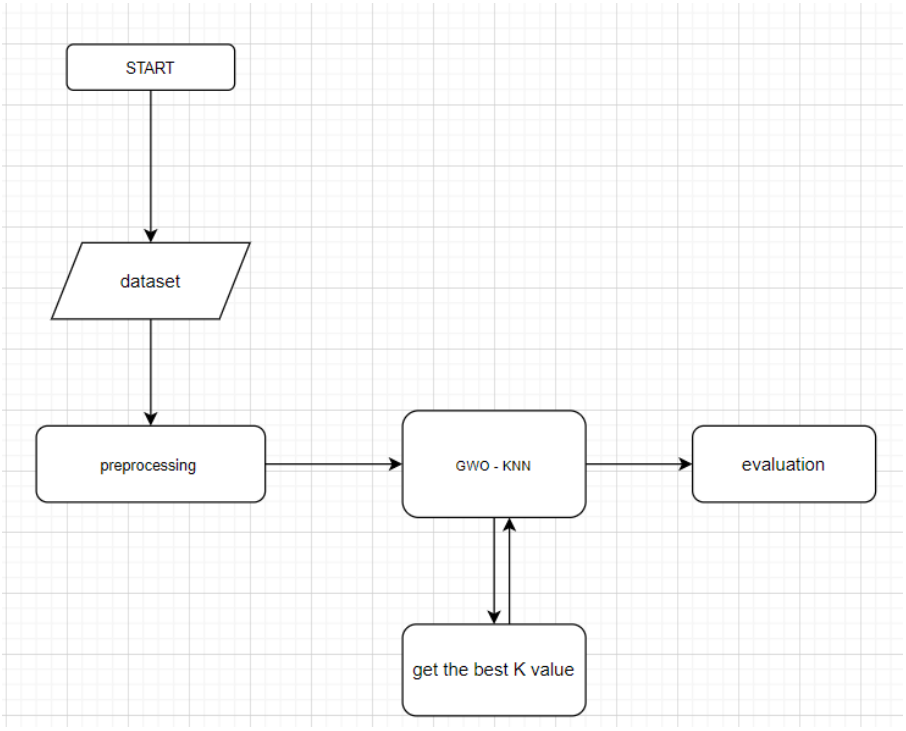


**Fig. 1** – Process flow

**1.** Grey Wolf Optimizer (GWO)

Grey wolf Optimization(GWO) is the swarm intelligence optimization technique which was first introduced in [6]. It is inspired by the leadership hierarchy and hunting process of the grey wolf in nature. The simple mechanism of GWO makes it easy to implement over others. Also, it has fewer decision variables, less storage required, and does not possess any rigorous mathematical equations of the optimization problem. Muro [5] explained the hunting behaviour of wolf into three stages as:

1. Social hierarchy: The social hierarchy of grey wolf has four levels: alpha α, beta β, delta δ and omega ω. The leaders are responsible for decision making and is denoted as the alpha wolf. The second level, called beta wolf works as a helping hand to the alpha for any activity. In the third level, the delta wolf is placed, which plays the role of scapegoat in grey-wolf packing. The rest of the wolves are categorized as omega wolf and is dominated by all other wolves.

2. Encircling the Prey: The encircling process is given by the following mathematical equation :

$$D = CX_{p,t} - X_t \quad (1)$$

$$X_{t+1} = X_{p,t} - A.D \quad (2)$$

where A and C are coefficient vectors, $\boldsymbol{X_{p,t}}$ denotes the position vector of the prey at current iteration t, and $\boldsymbol{X_{t+1}}$ denotes the position vector of a grey wolf at next iteration. The vectors are determined as:

$$A = 2\,a\,r_1 - a \quad (3)$$

$$C = 2r_2 \quad (4)$$

The vector a is a linearly decreasing function from 2 to 0 and $r_1$ and $r_2$ are random vectors in [0,1].

3. Hunting To encircle the position of prey, and the wolf position is approximated

by the average of alpha, beta, and gamma wolf positions, the below equations are used:

$$D_\alpha = C_1 X_\alpha^d - X_t^d \quad (5)$$

$$D_\beta = C_2 X_\beta^d - X_t^d \quad (6)$$

$$D_\delta = C_3 X_\delta^d - X_t^d \quad (7)$$

$$X_1^d = X_\alpha^d - A_1 D_\alpha \quad (8)$$

$$X_2^d = X_\beta^d - A_2 D_\beta \quad (9)$$

$$X_3^d = X_\delta^d - A_3 D_\delta \quad (10)$$

The position of prey is estimated as:

$$X_{t+1} = \frac{X_1 + X_2 + X_3}{3} \quad (11)$$

The pseudocode of the GWO algorithm is presented [6]:

*Initialize the grey wolf population $X_i$ (i = 1, 2, ..., n)*
*Initialize a, A, and C*
*Calculate the fitness of each search agent*
*$X_\alpha$=the best search agent*
*$X_\beta$=the second best search agent*
*$X_\delta$=the third best search agent*
***while** (t < Max number of iterations)*
  ***for** each search agent*
    *Update the position of the current search agent by equation (3.7)*
  ***end for***
  *Update a, A, and C*
  *Calculate the fitness of all search agents*
  *Update $X_\alpha$, $X_\beta$, and $X_\delta$*
  *t=t+1*
***end while***
*return $X_\alpha$*

**Fig. 3** Pseudocode of the GWO algorithm

**2.** K – Nearest Neighbours (K-NN)

K-Nearest Neighbours is a simple classification technique, and belongs to the non-parametric classification group, as there are no parameters in the model. The principle of this model is to classify new data based on the distance (for instance, Euclidean distance) of the new data to several nearest neighbour data, in which case the number of nearest neighbour data is determined by the user, and expressed in K, further voting will be used to determine what class or label the new data is in.

In summary, the K-NN procedure is given as follows:

1. Input: training dataset along with its labels, testing data, k-value

2. For all testing data i, calculate its distance to each training data

3. Determine the k training data whose distance is closest to the testing data i

4. Check the labels of the k data

5. Determine the label with the highest frequency.

6. Put the testing data i into the class with the most frequency

7. Repeat steps (3) to (6) for all other testing data.

The best k is the k that can provide the best performance, and can prevent overfitting/underfitting.

**3.** K-NN and GWO

The fitness function used in GWO-KNN is accuracy, when the highest accuracy is obtained, k is taken as the best k.

The following snippet is implemented in python:

```
class GreyWolfOptimizer:
  def __init__(self):
    self.search_space = None
    self.population = None
    self.fitness = None
```

```
    self.leader_position = None
    self.leader_fitness = None
    self.best_solution = None
def initialize_search_space(self, lower_bound, upper_bound):
    self.search_space = np.vstack((lower_bound, upper_bound))
def initialize_population(self, population_size=10):
    num_features = self.search_space.shape[1]
    self.population = np.random.uniform(low=self.search_space[0], high=self.search_space[1], size=(population_size, num_features))
    self.fitness = np.zeros(population_size)
def calculate_fitness(self, X_train, y_train):
    for i, wolf in enumerate(self.population):
        k = int(round(wolf[0]))
        k = max(1, k)
        knn = KNeighborsClassifier(n_neighbors=k)
        knn.fit(X_train, y_train)
        self.fitness[i] = knn.score(X_train, y_train)
def update_leader_positions(self):
    leader_index = np.argmax(self.fitness)
    self.leader_position = self.population[leader_index]
    self.leader_fitness = self.fitness[leader_index]
    self.best_solution = self.leader_position.copy()
def update_positions(self, iteration, max_iterations):
    a = 2 - iteration * ((2) / max_iterations)
    num_wolves = self.population.shape[0]
    for i in range(num_wolves):
        A1 = 2 * np.random.rand() - 1
        C1 = 2 * np.random.rand()
        D_alpha = np.abs(C1 * self.leader_position - self.population[i])
        X1 = self.leader_position - A1 * D_alpha
        r1 = np.random.rand()
        A2 = 2 * r1 - 1
        C2 = 2 * np.random.rand()
        D_beta = np.abs(C2 * self.leader_position - self.population[i])
        X2 = self.leader_position - A2 * D_beta
        r2 = np.random.rand()
        A3 = 2 * r2 - 1
        C3 = 2 * np.random.rand()
        D_delta = np.abs(C3 * self.population[i] - self.population[i])
        X3 = self.population[i] - A3 * D_delta
```

```
            updated_wolf = (X1 + X2 + X3) / 3
            updated_wolf = np.clip(updated_wolf, self.search_space[0], self.search_space[1])
            self.population[i] = updated_wolf
            # Increment k value
            self.population[i][0] = self.population[i][0] + a
    def get_best_solution(self):
        return self.best_solution
def knn_gwo(X_train, y_train, X_test, y_test, gwo_iterations=10):
    lower_bound = [1, 0.01]
    upper_bound = [10, 0.99]
    gwo = GreyWolfOptimizer()
    gwo.initialize_search_space(lower_bound, upper_bound)
    gwo.initialize_population()
    best_accuracy = 0.0
    best_k = 1
    for iteration in range(gwo_iterations):
        gwo.calculate_fitness(X_train, y_train)
        gwo.update_leader_positions()
        gwo.update_positions(iteration, gwo_iterations)
        best_solution = gwo.get_best_solution()
        k = int(round(best_solution[0]))
        k = max(1, k)
        knn = KNeighborsClassifier(n_neighbors=k)
        knn.fit(X_train, y_train)
        y_pred = knn.predict(X_test)
        accuracy = accuracy_score(y_test, y_pred)
        if accuracy >= best_accuracy:
            best_accuracy = accuracy
            best_k = k
        print ("Iteration:", iteration, "| Accuracy:", accuracy, "| the best K :", best_k)
    return best_k
```

## 3. Result and Discussion

In this experiment, we used breast cancer dataset from *sklearn api*, with 569 rows and 31 columns. In the modelling process it is carried out with general procedures through the separation of training and testing data, and evaluation is carried out by looking at the accuracy metrics and reviewing the susceptibility to overfitting.

**Table 1 – Iteration and K**

| Iteration | (Accuracy) | Best K |
|---|---|---|
| 0 | **0.9298** | **1** |
| 1 | **0.9385** | **4** |

| | | |
|---|---|---|
| 2 | **0.9561** | **7** |
| 3 | **0.9561** | **8** |
| 4 | **0.9561** | **8** |
| 5 | **0.9824** | **11** |
| 6 | **0.9824** | **11** |

From the results given, it is obtained that the accuracy value experiences the highest value at K = 11 (and auc = 0.9767), this will certainly give the advantage that the maximum K in order to get the best results from the GWO procedure leads us to an effective step compared to taking K manually.

## 4. Conclusion

From the results of the study conducted, that GWO-assisted K determination can provide competitive results. From the procedure stated, of course this will help in determining the optimal K, because if done manually we do not know how many K there are, with the help of GWO we can take the optimal solution, namely K by taking the best alpha position (and enrich K-NN itself in the process of improving its performance). Furthermore, this procedure can be further developed for example making the binary principle for GWO combined with K-NN simultaneously to determine the best K and feature selection.

***Acknowledgements***

Acknowledgements and Reference heading should be left justified, bold, with the first letter capitalized but have no numbers. Text below continues as normal.

**References**

[1] Yang, Q., C. Zhang, and S. Zhang, 2003. Data Preparation for Data Mining. Applied Artificial Intelligence 17:375-381

[2] Jiawei Han and Micheline Kamber, “Data Mining:Concepts and Techniques,” Morgan Kaufmann Publishers, August 2000. ISBN 1-55860-489-8.

[3] Mirjalili, S., Mirjalili, S. M., Lewis, A.: Grey wolf optimizer. Advances in engineering software, 69, 46-61, (2014).

[4] Reeti, Deep, K. (2023). A Modified Lévy Flight Grey Wolf Optimizer Feature Selection Approach to Breast Cancer Dataset. In: Thakur, M., Agnihotri, S.

[5] Rajpurohit, B.S., Pant, M., Deep, K., Nagar, A.K. (eds) Soft Computing for Problem Solving. Lecture Notes in Networks and Systems, vol 547. Springer, Singapore.

[6] Muro, C., Escobedo, R., Spector, L., Coppinger, R. P.: Wolf-pack (Canis lupus) hunting strategies emerge from simple rules in computational simulations. Behavioural Processes, 88(3), 192-197, (2011).

[7] Seyedali Mirjalili, Seyed Mohammad Mirjalili, Andrew Lewis, Grey Wolf Optimizer, Advances in Engineering Software, Volume 69, 2014, Pages 46-61, ISSN 0965-9978.

[8] Mirjalili, S. Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput & Applic 27, 1053–1073 (2016). https://doi.org/10.1007/s00521-015-1920-1.

[9] Jiawei Han, Micheline Kamber, nad Jian Pei, Data Mining : Concepts and Techniques, 3rd edition, Morgan Kaufmann, 2011

[10] Al-Tashi, S. J. Abdul Kadir, H. M. Rais, S. Mirjalili and H. Alhussian, "Binary Optimization Using Hybrid Grey Wolf Optimization for Feature Selection," in IEEE Access, vol. 7, pp. 39496-39508, 2019, doi: 10.1109/ACCESS.2019.29067