

PARETO OPTIMIZATION OF PRODUCT MATERIAL SELECTION

Submitted in partial fulfillment of the requirements
for the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

CH SRIYA(38110561)
MOUNIKA CHATHARAJUPALLI (38110328)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119**

MAY– 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with “A” grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **CH.SRIYA (38110561)** and **MOUNIKA CHATHARAJUPALLI (38110328)** who carried out the project entitled “**Pareto Optimization Of Product Material Selection**” under my supervision from November 2021 to March 2022.

Internal Guide

Dr Veena K, ME, PhD, Assistant Professor

Head of the Department

Dr. S. Vigneshwari M.E., Ph.D., and Dr. L. Lakshmanan M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **CH.SRIYA (38110561)** and **MOUNIKA CHATHARAJUPALLI (38110328)** hereby declare that the Project Report entitled “**Pareto Optimization Of Product Material Selection** ” done by us under the guidance of **Dr Veena K, ME, PhD, Assistant Professor** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in 2018-2022.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE:

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E.,Ph.D., Dean**, School of Computing **Dr.S.Vigneshwari M.E., Ph.D.** and **Dr.L.Lakshmanan M.E., Ph.D.** , Heads of the Department of Computer Science and Engineering for providing us necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and a deep sense of gratitude to my Project Guide **Dr Veena K, ME, PhD, Assistant Professor** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

One of the biggest problems humanity faces today is that our adverse effects on the environment are much greater than the steps taken to mend them. One such step or process is waste management. Waste management begins with reducing the production of consumer waste; the waste caused by product materials. Consumer waste can be reduced by choosing sustainable product materials.

To choose sustainable product materials, the product designer must take into account not only environmental factors but mechanical and economic factors as well to ensure the maximum efficiency of the chosen material with respect to the product requirements. It is often difficult to accurately weigh the various properties of the materials and derive the most optimal material that adheres to all the objectives of the product.

A cohesive machine learning solution of differential evolution (DE) and back propagation neural networks (BPNs) is proposed to optimize the Pareto product material selection. This approach has been validated by an application that can select the most optimal material given a list of potential materials for a product. The application has been created such that it will assist product designers during the planning stages of product designing.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO
	ABSTRACT	i
	LIST OF FIGURES	v
		v
		i
1.	INTRODUCTION	1
1.1	Overview	1
1.2	Background and Motivation	1
1.3	Literature Survey	2
1.4	Problem Statement	3
1.5	Goal and Vision	3
1.6	Application Validation	4
2.	REQUIREMENT ANALYSIS	4
2.1	Requirement Specification	4
2.1.1	User Characteristics	4

2.1.2	Functional Requirements	5
2.1.3	Dependencies	5
2.1.4	Performance Requirements	6
2.1.5	Hardware Requirements	6
2.1.6	Constraints and Assumptions	6
2.2	Feasibility Study	6
2.2.1	Economic Feasibility	6
2.2.2	Technical Feasibility	7
2.2.3	Operational Feasibility	7
2.2.4	Behavioural Feasibility	7
3.	DESIGN	7
3.1	Data Design	7
3.2	Model Design	10
3.2.1	Differential Evolution	10
3.2.2	Backpropagation Neural Networks	12
3.2.3	Integration of DE and BPN for Pareto Optimization	14
3.3	Function Oriented Design	17
3.3.1	Pareto Objective Function	17

3.3.2	Fitness Function	19
3.4	GUI Design	20
4.	IMPLEMENTATION	21
4.1	Code	21
4.2	Sample Output and Input	29
5.	TESTING	33
5.1	Test Plan	33
5.2	Test Report	35
5.2.1	Performance	35
5.2.2	Drawbacks	36
6.	FUTURE WORKS	36
7.	CONCLUSION	37
8.	APPENDICES	38
8.1	Appendix 1: The Importance of Having a Different Fitness Function	38
8.2	Appendix 2: Differential Evolution Algorithm for FeedForward Neural Networks	40
9.	REFERENCE	41

LIST OF FIGURES

1. **Fig.1** Flowchart of the model with DE &BPN
2. **Fig.2** The feedforward neural network
3. **Fig.3** The GUI of Application
4. **Fig.4** Sample Output Console displaying the database
5. **Fig.5** Sample GUI
6. **Fig.6** Sample GUI with materials selected by user
7. **Fig.7** Sample Output Console with fitness values of materials
8. **Fig.8** Sample GUI with result
9. **Fig.9** Sample Output Console with material removed
10. **Fig.10** Sample GUI with material
11. **Fig.11** Sample Output Console with warning

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

This report discusses the result of the work done in development of "Pareto Optimization of Product Material Selection: Artificial Neural Networks & Differential Evolution Approach" using machine learning and python. It aims at the development of an application that assists product designers choose a sustainable product material.

1.2 BACKGROUND AND MOTIVATION

India is one of the places in the world where the most garbage is disposed of. India's urban population of 429 million citizens produce 62 million tonnes of garbage annually. Studies have shown that higher income levels and increased urbanization mean more waste is generated. Of all the waste, the total packaging waste is 32 million tonnes per annum (predominantly single use plastics). This shows a definite link between consumerism and waste. Unless something is done about this consumer waste, WEF Report estimates that by 2050 there will be more plastic in the ocean than fish. The field that deals with waste is known as waste management.

The most troublesome objective of waste management is reducing the production of waste.

1.3 LITERATURE SURVEY

Various scientific journals were analyzed to further investigate and understand the domains of waste management, material selection, and machine learning. Many papers offered an insight that was parallel to the motivation of this project, and existing working systems were understood.

In 2009, a system was created to select sustainable materials for bottled beverages. It used Genetic Algorithm and Artificial Neural Networks to select a material based on multiple objectives like minimizing cost and adverse environmental effects. Environmental impacts were calculated by LCA method (Life Cycle Assessment Method). Material selection was explored and multicriteria decision making methods were studied to gather background knowledge.

Material selection using Linear Assignment method helped formulate a basic understanding of the process needed to select a product material. Based on the drawbacks of the existing system, papers were read to find an improvement to the Genetic Algorithm. In this process many algorithms were touched upon till Differential Evolution was found to have superior command over the decision space.

Then, the working of Artificial Neural Networks, Differential Evolution and Genetic Algorithm were separately studied in various papers. Insight like the general activation function for a neural network being a sigmoid function was gathered. It was also learned that differential evolution's extent of application to neural networks is still being studied.

1.4 Problem Statement

The materials a product is made of can not just be determined based on environmental factors. The product designer will have to consider mechanical and economic factors as well. There will be specific objectives based on what the product is built to do its purpose. This type of multiobjective optimization of product material selection been done previously using genetic algorithms and artificial neural networks. However, this solution is not the most optimal.

1.5 Goal and Vision

The goal is to develop an application for product designers to determine the best product materials based on various objectives. This application should give the highest accuracy possible and provide a better insight to the product designer. The product designer can select a list of potential materials he or she is considering from a friendly GUI and get an output of the most optimal material.

The vision is to use differential evolution (a better evolutionary algorithm for Pareto optimization in comparison to genetic algorithms) with back propagation neural networks to select the optimal material.

1.6 Application Validation

The validity of the application will be tested by applying it to the selection of materials for the product, kitchen cutlery specifically knives, forks, and spoons.

CHAPTER 2

REQUIREMENT ANALYSIS

This phase of the project was carried out to understand the current available solutions to product materials selection, their drawbacks, the requirements of products material determination, how materials are selected, understanding various properties of materials, research into parameters, the product designing process, data needs, algorithm requirements, etc.

2.1 REQUIREMENT SPECIFICATION

2.1.1 User Characteristics

The user that interacts with the application is a product designer. A product designer understands the relationship between art, science and technology. They will spend their time planning, designing and modeling products, producing prototypes and conducting rigorous testing. During the planning stage itself, the designer will brainstorm a list of potential materials that can be used. This is where the application can be utilized by the designer.

The designer will interact with a GUI that presents a form with a list of materials. He or she can select the candidate materials they are considering. On submitting the form, one optimal material from the list will be displayed as the output. It is assumed that the designer has no specific knowledge about the materials. This means that toxic materials or otherwise unusable materials may be entered by the designer as input.

2.1.2 Functional Requirements

Actor: User

Input: Selecting Checkboxes as Input

Description: User provides input that gets processed in the background and instant result will be returned.

2.1.3 Dependencies

Modern modules of Python 3.6 have been used to develop the frontend and backend of the application. The modules Numpy and Pandas are used to develop the backend, and PyQt5 is used for the frontend. PyInstaller is used to combine the necessary modules and run the application on a click by generating an .exe file.

2.1.4 Performance Requirements

The performance of the model was tested using the manually calculated data of few materials. The LCA parameters were fetched and processed to compare the output of the model and to determine the performance.

2.1.5 Hardware Requirements

This application can be run on any laptop or PC with a fairly decent CPU like an Intel Core i3, 2GB RAM, Windows or Ubuntu Operating System, and Python should be installed.

2.1.6 Constraints and Assumptions

Application assumes that user knows the “English” language and is designing a product that is made of only one material.

2.2 FEASIBILITY STUDY

The feasibility study is a process executed to determine if the actualization of the application is possible with the available resources.

2.2.1 Economic Feasibility

The economic analysis is to determine the benefits and savings that are expected from the application and compare them with cost. The system is economically feasible, as most designers possesses the hardware and software resource required for the functioning of the application. Any additional resources, if required, can also be easily acquired.

2.2.2 Technical Feasibility

The application's backend centres around data, and the type of data required is attainable from public sources. The model built for processing is technically applicable as well.

2.2.3 Operational Feasibility

The user will not see the machine learning part of the application. The application just requires a user friendly interface, which it has.

2.2.4 Behavioral Feasibility

In today's world, computers are an inevitable entity, therefore the application requires no special efforts thus it will be widely accepted.

CHAPTER 3

DESIGN

The detailed implementation of the application for kitchen cutlery material selection based on the feasibility study is split into 3 modules; the dataset, the model (integrated algorithm), and the GUI.

3.1 DATA DESIGN

The dataset utilized by the application is a collection of public numeric data about the properties split into three main divisions, and under them are different parameters as follows:

Mechanical Properties

- Young's Modulus
- Shear Modulus
- Poisson's Ratio
- Elastic Modulus
- Tensile Strength
- Compressive Strength
- Hardness
- Density
- Economical Properties
- Price of Raw Material
- Price of Recycle
- Environmental Properties
- Embodied Energy
- Embodied Carbon
- Average Recycle Content
- Acidification Potential
- Global Warming Potential
- Toxicity

The materials have been selected based on research about kitchen cutlery. Materials in the dataset are as follows:

- Alloy Steel
- Aluminum
- Bamboo
- Black Iron
- Carbon Steel
- Cast Iron
- Ceramic Clay
- Cobalt
- Copper
- Glass
- Gold
- Lead
- lista Ferritic stainless steel(grade 409)
- lista Martensitic stainless steel(grade 410)
- Obsidian
- Plastics(Polyethylene Terephthalate)
- Porcelain
- Silicone
- Silver
- Stainless steel
- Teflon
- Titanium Alloys
- Tool steel
- Wood

3.2 MODEL DESIGN

The application's key element is the machine learning model. The model is an integration of differential evolution and backpropagation neural networks. The following is a flowchart explaining how the model is incorporated into the application:

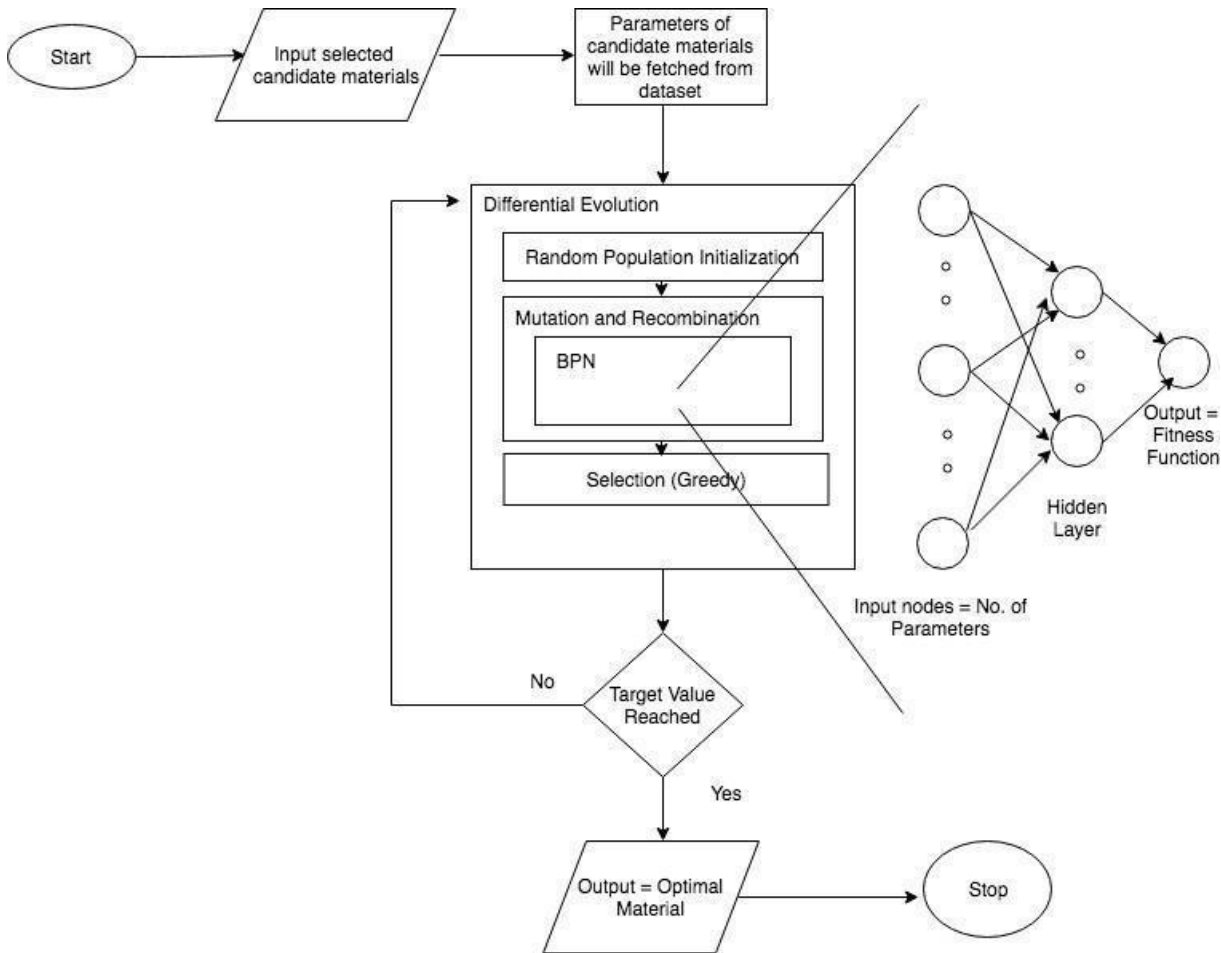


Fig.1 Flowchart of the model with DE &BPN

3.2.1 Differential Evolution

Step 1: Start.

Step 2: A population of children with values between 0 and 1 is randomly generated initially.

Step 3: The fitness of each individual of the initial population is calculated using an objective function. It is considered the general fitness. These fitness values are stored into an array.

Step 4: The individual in the population with the highest fitness is selected. When there are multiple parameters involved, the mean is taken to determine the total fitness.

Step 5: Every individual is mutated using a fitness function, and fitness values are calculated.

Step 6: The highest fitness value from step 4 will be recombined with the maximum fitness value from step 5.

Step 7: Repeat from step 1 till stopping criteria met; meaning either a set number of iterations crossed or a standard fitness value reached.

Step 8: All fitness values compared and best one selected and given as output.

Step 9: Stop.

3.2.2 Backpropagation Neural Networks

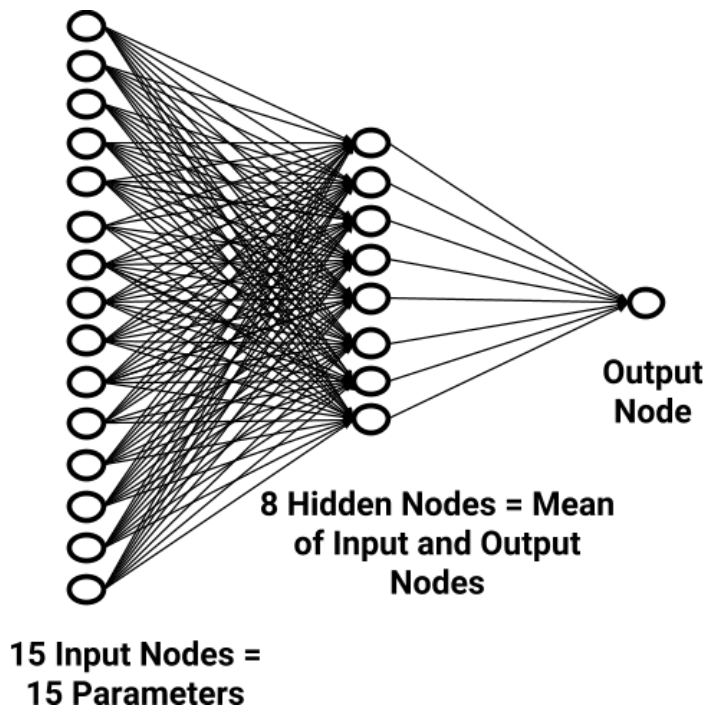


Fig.2 The feed forward neural network

Backpropagation neural network is a type of artificial neural network. The backpropagation neural network works in the following manner:

Step 1: Start.

Step 2: The initial set of weights for each layer is initialized. The feedforward neural network propagates from one layer to another.

Step 3: After propagation through one layer, it is activated by the sigmoid function. The sigmoid function acts as the activation function for this model.

Step 4: Once the final activated value is obtained from the output layer that is a single node, the error is calculated by subtracting it from the target output, the original objective function value.

Step 5: The delta value is calculated by applying the derivative of the sigmoid function to the error. This is carried out in both the layers of the neural net.

Step 6: The neural network is back propagated by adjusting the weights of each layer. The

delta values obtained from the error are used to adjust the weights.

Step 7: Repeat Steps 2 to 6 until an iteration limit is reached and the neural network returns the maximum fitness value.

Step 8: Returned fitness value is the output of the neural network.

Step 9: Stop.

3.2.3 Integration of DE and BPN for Pareto Optimization

The model works in the following manner (refer **Fig.1**):

Step 1: Start.

Step 2: Input from the user is obtained. It will be a list of candidate materials that the user believes are suitable for the product. In this case, cutlery.

Step 3: The Pareto objective function is declared which reduces the cost and adverse environmental effects while maximizing the mechanical and physical properties of the material.

Step 4: The fitness value is calculated for each material such that each of them are passed through the model iteratively.

Step 5: Initially, the objective function is passed with the values of the parameters of the candidate material. This calculates the fitness of the candidate material directly.

Step 6: To optimize it, the Differential Evolution algorithm generates the initial set of population within the bounds [0,1]. The population size is 10 times the number of parameters passed into the function.

Step 7: Every individual in the randomly generated population is passed through the objective function to calculate their general fitnesses. The obtained fitness of each individual is stored in an array.

Step 8: The best fitness in this population is stored in a variable.

Step 9: The DE proceeds with the iteration of its process. The number of iterations is predefined by the user. Each individual from the population is selected.

Step 10: The target individual is the above selected individual. Apart from this, three random individuals are selected from the population for mutation. The mutation rate is predefined by the user, which is set at 0.8 (which works best for this model).

Step 11: The difference of two individuals is multiplied with the third one to mutate. Now the mutant is crossed over with the existing population.

Step 12: The crossover point is determined by a preset value by the user. The crossover rate is also set which determines the crossover point in the existing population.

Step 13: The crossover trial value is obtained. The trial is the new child which is fed into the

neural network. The neural network has the target (original fitness) value of the material.

Step 14: The trail is back propagated through the neural network which results in another value.

Step 15: The fitness values is compared with the best which was determined directly by searching for the minimum in the array. If the newly obtained fitness value of trial is less than the original, then the original individual is replaced with the new individual. (see Appendix B)

Step 16: The fitness values of each material are now stored and compared. The maximum fitness value of the materials are found and the material with the highest fitness value is declared as the Pareto optimized material.

Step 17: Stop.

3.3 FUNCTION ORIENTED DESIGN

The integrated model of Differential Evolution and Backpropagation Neural Networks is used to optimize two functions at its most basic, the Pareto objective function and the fitness

3.3.1 Pareto Objective Function

The Pareto objective function is the key function that the model is to minimize. The Pareto objective function will be as follows:

$$\text{fobj}(X) = \min(1/B_m + B_e + B_c)$$

This function will determine the lost optimal product material.

3.3.2 Fitness Function

The fitness function is the measure of fitness of an individual with respect to the objective function. In most cases, the objective function is considered the fitness function, however in this case there are multiple objectives like minimizing the cost of the material while maximizing the mechanical properties of the material.

This leads to the usage of another function for measuring the fitness. The population varies from [0,1] and therefore the random parameters are passed to the neural net. Within the neural net, backpropagation obtains a better fitness value than it would with a normal objective function (see Appendix A).

3.4 GUI DESIGN

For the ease of the user, the background processing is kept hidden. A graphical user interface (GUI) takes input and gives out output. The GUI has been designed such that a list of materials are given on a form and the user can select their candidate materials using checkboxes. After the form is submitted using an “optimize” button, the most optimal material is displayed at the top of the form.

The PyQt5 module was used to design the GUI. The PyQt5 module is a package that is available for many programming languages. The Python distributed version of PyQt5 was used to design the basic GUI.

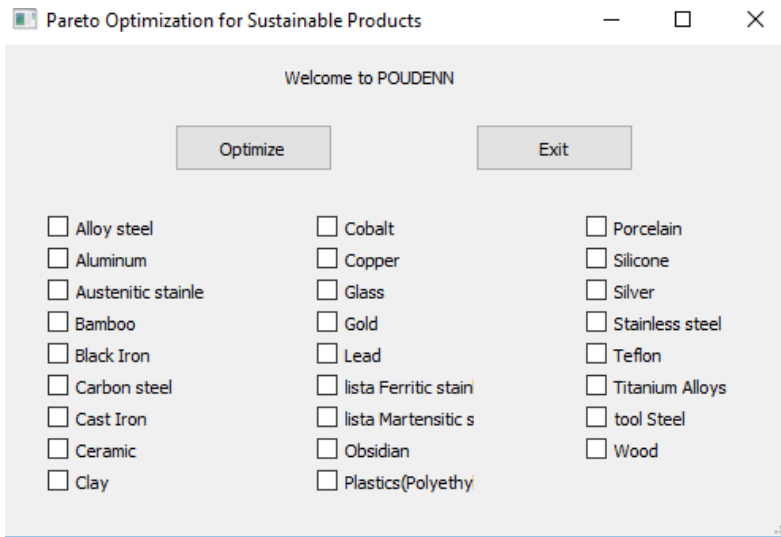


Fig.3 The GUI of Application

The above picture shows the GUI layout , and its design.

The layout that was implemented was the boxed layout that comes along with the package of PyQt. The checkboxes, labels and button features were used to pack this GUI simply with the necessary functionalities.

CHAPTER 4

IMPLEMENTATION

4.1 CODE

```
import pandas as pd
import numpy as np
import time
import sys

from PyQt5.QtWidgets import QWidget, QDesktopWidget, QApplication, QAction, QMainWindow, QGridLayout,
QLabel, QLineEdit, QTextEdit, QPushButton, QCheckBox
from PyQt5.QtCore import pyqtSignal, QObject, Qt
from PyQt5.QtGui import QFont
import PyQt5.QtGui

#product designer enters selected candidate materials
#candidates = input("Welcome, enter your candidate materials:")
#print ("Your candidate materials: " + candidates )
#candidates_list = candidates.split(",")
main_mats = []

class Neural_Network(object):
    def __init__(self):
        #parameters
        self.inputSize = 15
        self.outputSize = 1
        self.hiddenSize = 8

        #weights
        self.W1 = np.random.randn(self.inputSize, self.hiddenSize) # (15x8) weight matrix from input to hidden layer
        self.W2 = np.random.randn(self.hiddenSize, self.outputSize) # (8x1) weight matrix from hidden to output layer

    def forward(self, X):
        #forward propagation through our network
        self.z = np.dot(X, self.W1) # dot product of X (input) and first set of 15x8 weights = 1x8
        #print(self.z)
        self.z2 = self.sigmoid(self.z) # activation function 1x8
        #print(self.z2)
        self.z3 = np.dot(self.z2, self.W2) # dot product of hidden layer (z2) and second set of 8x1 weights 1x1
        #print(self.z3)
        o = self.sigmoid(self.z3) # final activation function
        return o

    def sigmoid(self, s):
        # activation function
        return 1/(1+np.exp(-s))

    def sigmoidPrime(self, s):
        #derivative of sigmoid
        return s * (1 - s)

    def backward(self, X, y, o):
        # backward propgate through the network
        self.o_error = y - o # error in output
```

```
self.o_delta = self.o_error*self.sigmoidPrime(o) # applying derivative of sigmoid to error
```

```
self.z2_error = self.o_delta.dot(self.W2.T) # z2 error: how much our hidden layer weights contributed to output error  
self.z2_delta = self.z2_error*self.sigmoidPrime(self.z2) # applying derivative of sigmoid to z2 error
```

```
self.W1 += X.T.dot(self.z2_delta) # adjusting first set (input --> hidden) weights  
self.W2 += self.z2.T.dot(self.o_delta) # adjusting second set (hidden --> output) weights
```

```
def train (self, X, y):  
    o = self.forward(X)  
    self.backward(X, y, o)
```

```
def fobj(x):  
    #best mechanical property function  
    phy_prop = np.array(x[:8],dtype=float)  
  
    #weights of importance of properties  
    phy_weights = np.array([0.1,0.1,0.1,0.1,0.1,0.1,0.2,0.2,0.1])  
    totpy = phy_prop * phy_weights  
    #print(totphy)  
    bphy = sum(totphy)  
    bmech = 1/bphy  
  
    #design_quality  
    dq = 0.5  
  
    #purchase cost  
    pur_cost = np.array(x[9],dtype=float)  
    #recycle cost  
    rec_cost = np.array(x[10],dtype=float)  
  
    eco_fac = (pur_cost+rec_cost)*dq  
  
    #best environmental  
    env_prop = np.array(x[10:15],dtype=float)  
  
    env_fac = sum (env_prop) *dq  
  
    return np.array([bmech,env_fac,eco_fac])
```

```
def fitnessOf(X,y):  
    NN = Neural_Network()  
    for i in range(1000): # trains the NN 1,000 times  
        print ("Input: \n" + str(X))  
        print ("Actual Output: \n" + str(y))  
        print ("Predicted Output: \n" + str(NN.forward(X)))  
        print ("Loss: \n" + str(np.mean(np.square(y - NN.forward(X))))) # mean sum squared loss  
        print ("\n")  
        NN.train(X, y)
```

```
#preprocessing of dataset
```

```

df = pd.read_csv("newset.csv")

df.columns = ['Materials','Youngs_modulus','Shear_modulus','Poissons_ratio',
              'Elastic_limit','Tensile_strength','Compressive_strength','Hardness',
              'Density','Price_of_raw_material','Price_of_recycle','Embodied_energy',
              'Embodied_carbon','Avg_recycled_content','Acidification_potential','Global_warming_potential','Toxicity']
df.index=['Alloy steel', 'Aluminum', 'Austenitic stainless steel','Bamboo','Black Iron','Carbon steel','Cast Iron',
          'Ceramic', 'Clay','Cobalt','Copper','Glass','Gold','Lead','lista Ferritic stainless steel(grade 409)',
          'lista Martensitic stainless steel(grade 410)','Obsidian','Plastics(Polyethylene Terephthalate)','Porcelain',
          'Silicone','Silver','Stainless steel','Teflon','Titanium Alloys','Tool steel','Wood']

df.replace('?',0, inplace=True)
material_data=df.drop(['Materials'],1)

print(material_data)

#print(candidate_mats)

#differential evolution
def de(fobj, mat_prop, bounds=15, mut=0.8, crossp=0.7, popsize=150, its=1000):

    print("Algorithm Running")

    #the dimensions of the code
    dimensions = bounds

    #generating random population between [0,1]
    pop = np.random.rand(popsize, dimensions)

    #calculate the initial fitness of the existing population
    fitness = np.asarray([sum(fobj(ind)) for ind in pop])

    original_fitness = sum(fobj(mat_prop))

    #find the index of fittest element
    mmin = np.argmin(fitness)

    #The best element so far
    best = pop[mmin]

    for i in range(its):
        for j in range(0,popsize):

            target = pop[j]

            #store the set of ids of the population
            idxs = [idx for idx in range(popsize) if idx!=j]

            #select three random individuals from the population
            selected = np.random.choice(idxs, 3, replace=False)

            a,b,c = pop[selected]

```

```

#mutation
mutant = a + mut * (b-c)

mutant = np.clip(mutant,0,1)

#crossover points
cross_points = np.random.rand(dimensions) < crossp

```

```

#X = np.array((target), dtype=float)
#X = np.array([2,9,3,4,5,6,7,8,9,0,1,1,3,4,5]), dtype=float)
# = np.array([2,9,3,4,5,6,7,8,9,0,1,1,3,4,5]), dtype=float)
#y = np.array([92]), dtype=float)
#X = X/np.amax(X, axis=0) # maximum of X array
#y = y/100 # max test score is 100
#print(X)
#y = np.array([92]), dtype=float)
#print(y)

```

```

#trail from the population
trial = np.where(cross_points,mutant,pop[j])

```

```

#fitness of trial
f = sum(fobj(trial))
#print(f)
if f < fitness[j]:
    fitness[j] = f
    pop[j] = trial
    if f < fitness[mmin]:
        mmin = j
        best = trial

```

```

return original_fitness

```

```

class pareto(QMainWindow):

```

```

    def __init__(self):
        super().__init__()

        self.initUI()

```

```

    def initUI(self):

```

```

        material_buttons = ['Alloy steel', 'Aluminum', 'Austenitic stainless steel','Bamboo','Black Iron','Carbon steel','Cast
Iron',
        'Ceramic', 'Clay','Cobalt','Copper','Glass','Gold','Lead','lista Ferritic stainless steel(grade 409)',

```

```
'lista Martensitic stainless steel(grade 410)','Obsidian','Plastics(Polyethylene Terephthalate)','Porcelain',  
'Silicone','Silver','Stainless steel','Teflon','Titanium Alloys','Tool steel','Wood']
```

```
btn1 = QPushButton("Optimize", self)  
btn1.move(110, 50)
```

```
btn2 = QPushButton("Exit", self)  
btn2.move(300, 50)
```

```
grid = QGridLayout()  
self.setLayout(grid)
```

```
cb = QCheckBox('Alloy steel', self)  
cb.move(30, 100)  
cb.stateChanged.connect(self.changeTitle)
```

```
cb1 = QCheckBox('Aluminum', self)  
cb1.move(30, 120)  
cb1.stateChanged.connect(self.changeTitle)
```

```
cb2 = QCheckBox('Austenitic stainless steel', self)  
cb2.move(30, 140)  
cb2.stateChanged.connect(self.changeTitle)
```

```
cb3= QCheckBox('Bamboo', self)  
cb3.move(30, 160)  
cb3.stateChanged.connect(self.changeTitle)
```

```
cb4= QCheckBox('Black Iron', self)  
cb4.move(30, 180)  
cb4.stateChanged.connect(self.changeTitle)
```

```
cb5= QCheckBox('Carbon steel', self)  
cb5.move(30, 200)  
cb5.stateChanged.connect(self.changeTitle)
```

```
cb6= QCheckBox('Cast Iron', self)  
cb6.move(30, 220)  
cb6.stateChanged.connect(self.changeTitle)
```

```
cb7= QCheckBox('Ceramic', self)  
cb7.move(30, 240)  
cb7.stateChanged.connect(self.changeTitle)
```

```
cb8= QCheckBox('Clay', self)  
cb8.move(30, 260)  
cb8.stateChanged.connect(self.changeTitle)
```

```
ccb = QCheckBox('Cobalt', self)  
ccb.move(200, 100)  
ccb.stateChanged.connect(self.changeTitle)
```

```
ccb1 = QCheckBox('Copper', self)
ccb1.move(200, 120)
ccb1.stateChanged.connect(self.changeTitle)

ccb2 = QCheckBox('Glass', self)
ccb2.move(200, 140)
ccb2.stateChanged.connect(self.changeTitle)

ccb3= QCheckBox('Gold', self)
ccb3.move(200, 160)
ccb3.stateChanged.connect(self.changeTitle)

ccb4= QCheckBox('Lead', self)
ccb4.move(200, 180)
ccb4.stateChanged.connect(self.changeTitle)

ccb5= QCheckBox('lista Ferritic stainless steel(grade 409)', self)
ccb5.move(200, 200)
ccb5.stateChanged.connect(self.changeTitle)

ccb6= QCheckBox('lista Martensitic stainless steel(grade 410)', self)
ccb6.move(200, 220)
ccb6.stateChanged.connect(self.changeTitle)

ccb7= QCheckBox('Obsidian', self)
ccb7.move(200, 240)
ccb7.stateChanged.connect(self.changeTitle)

ccb8= QCheckBox('Plastics(Polyethylene Terephthalate)', self)
ccb8.move(200, 260)
ccb8.stateChanged.connect(self.changeTitle)


cccb = QCheckBox('Porcelain', self)
cccb.move(370, 100)
cccb.stateChanged.connect(self.changeTitle)

cccb1 = QCheckBox('Silicone', self)
cccb1.move(370, 120)
cccb1.stateChanged.connect(self.changeTitle)

cccb2 = QCheckBox('Silver', self)
cccb2.move(370, 140)
cccb2.stateChanged.connect(self.changeTitle)

cccb3= QCheckBox('Stainless steel', self)
cccb3.move(370, 160)
cccb3.stateChanged.connect(self.changeTitle)

cccb4= QCheckBox('Teflon', self)
cccb4.move(370, 180)
cccb4.stateChanged.connect(self.changeTitle)
```

```

cccb5= QCheckBox('Titanium Alloys', self)
cccb5.move(370, 200)
cccb5.stateChanged.connect(self.changeTitle)

cccb6= QCheckBox('tool Steel', self)
cccb6.move(370, 220)
cccb6.stateChanged.connect(self.changeTitle)

cccb7= QCheckBox('Wood', self)
cccb7.move(370, 240)
cccb7.stateChanged.connect(self.changeTitle)

btn1.clicked.connect(self.buttonClicked)
btn2.clicked.connect(QApplication.instance().quit)

self.statusBar()

self.lbl = QLabel("Welcome to POUDENN", self)
self.lbl.move(180,10)
self.lbl.resize(300, 20)

self.setGeometry(300, 300, 500, 310)
self.setWindowTitle('Pareto Optimization for Sustainable Products')
self.show()

```

```

def changeTitle(self, state):

```

```

    sender = self.sender()

    if state == Qt.Checked:
        main_mats.append(sender.text())
        self.statusBar().showMessage(sender.text() + ' was selected')

    else:
        main_mats.remove(sender.text())
        print(sender.text() + " was removed from candidate materials!")

```

```

def buttonClicked(self):

```

```

    sender = self.sender()

    self.statusBar().showMessage(sender.text() + ' was pressed')

```

```

#extract needed materials data
candidate_data = material_data.loc[main_mats]

```

```

fitness = 0
fittest_name = "None"

```

```

#make it into an np array
material = candidate_data.values

```

```

candidate_mats = np.array(material,dtype=float)

#print(candidate_mats)
#print("Executing Differential Evolution")
k=0
print(k)
for mat in candidate_mats:
    print(main_mats[k])
    if(mat[15]==1):
        print(main_mats[k] + " is a toxic material.")

        self.lbl.setText(main_mats[k] + " is a toxic material.")

    k=k+1
else:

    new_fit = de(fobj,mat)
    print(main_mats[k] + " fitness is" + str(new_fit))
    if(fitness < new_fit):
        fitness = new_fit
        fittest_name = main_mats[k]
    k=k+1
self.lbl.setText(fittest_name + " is the most optimized product material.")

```

```

def center(self):

```

```

    qr = self.frameGeometry()
    cp = QDesktopWidget().availableGeometry().center()
    qr.moveCenter(cp)
    self.move(qr.topLeft())

```

```

if __name__ == '__main__':

```

```

    app = QApplication(sys.argv)
    ex = pareto()
    sys.exit(app.exec_())

```

4.2 SAMPLE OUTPUT AND INPUT

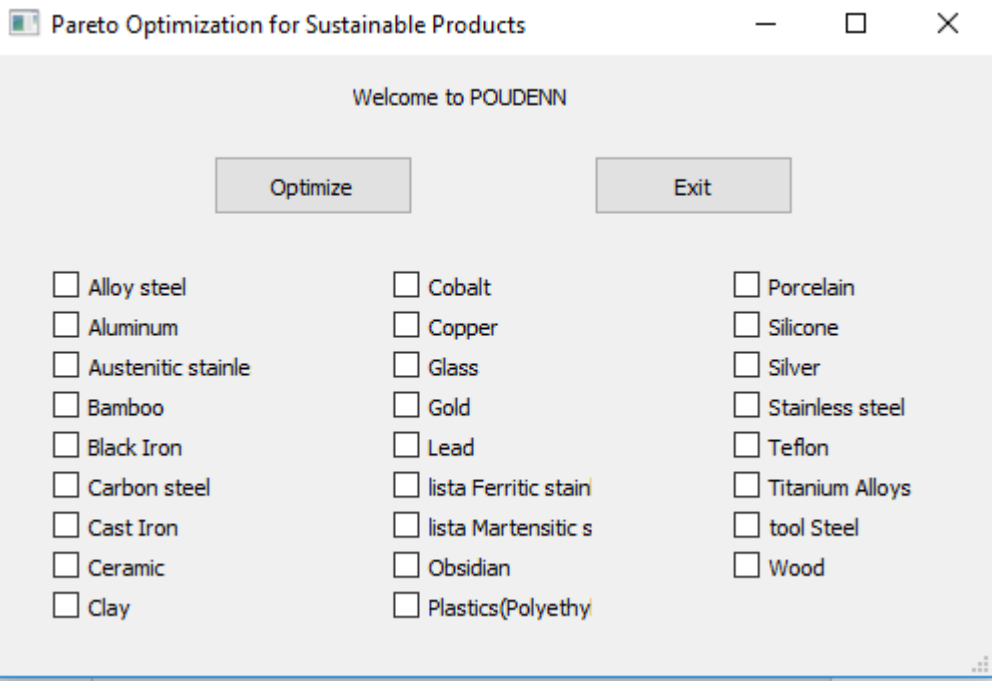


Fig.4 Sample Output Console displaying the database

```
G:\Programming\ML Project>py main.py
```

	Youngs_modulus	Shear_modulus	...	Global_warming_potential	Toxicity
Alloy steel	200.00	80.00	...	1.50	0.0
Aluminum	210.00	24.00	...	10.00	0.0
Austenitic stainless steel	203.00	81.00	...	5.20	0.0
Bamboo	9.60	0.02	...	1.62	0.0
Black Iron	190.00	41.00	...	7.20	0.0
Carbon steel	200.00	80.00	...	5.20	0.0
Cast Iron	120.00	41.00	...	6.80	0.0
Ceramic	15.00	18.00	...	0.80	0.0
Clay	0	0	...	0.20	0.0
Cobalt	199.00	82.60	...	1.00	0.0
Copper	133.00	44.00	...	1.90	0.0
Glass	85.00	36.00	...	1.50	0.0
Gold	81.00	30.00	...	0	0.0
Lead	15.00	6.00	...	1.64	1.0
lista Ferritic stainless steel(grade 409)	205.00	81.00	...	1.70	0.0
lista Martensitic stainless steel(grade 410)	210.00	83.00	...	1.80	0.0
Obsidian	12.80	15.00	...	0	0.0
Plastics(Polyethylene Terephthalate)	11.00	38.50	...	3.20	0.0
Porcelain	15.00	27.00	...	0.25	0.0
Silicone	0.05	0.02	...	1.90	0.0
Silver	74.00	28.00	...	0	0.0
Stainless steel	190.00	86.00	...	1.70	0.0
Teflon	0.50	0.01	...	1.00	0.0
Titanium Alloys	105.00	39.00	...	1.30	0.0
Tool steel	208.00	88.00	...	1.20	0.0
Wood	11.00	12.00	...	0.20	0.0

Fig.5 Sample GUI

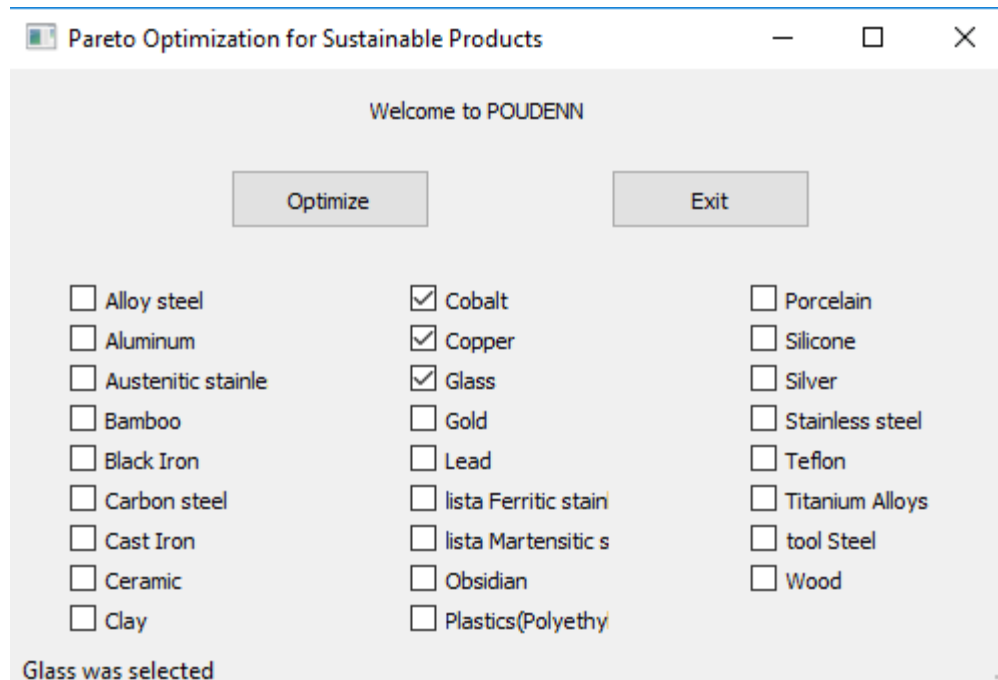


Fig.6 Sample GUI with materials selected by user

```
C:\WINDOWS\system32\cmd.exe - py main.py

Copper      133.00      44.00      ...      1.90      0.0
Glass       85.00      36.00      ...      1.50      0.0
Gold        81.00      30.00      ...      0         0.0
Lead        15.00      6.00      ...      1.64      1.0
lista Ferritic stainless steel(grade 409)  205.00      81.00      ...      1.70      0.0
lista Martensitic stainless steel(grade 410)  210.00      83.00      ...      1.80      0.0
Obsidian    12.80      15.00      ...      0         0.0
Plastics(Polyethylene Terephthalate)  11.00      38.50      ...      3.20      0.0
Porcelain   15.00      27.00      ...      0.25      0.0
Silicone    0.05       0.02      ...      1.90      0.0
Silver      74.00      28.00      ...      0         0.0
Stainless steel  190.00      86.00      ...      1.70      0.0
Teflon      0.50       0.01      ...      1.00      0.0
Titanium Alloys  105.00      39.00      ...      1.30      0.0
Tool steel  208.00      88.00      ...      1.20      0.0
Wood        11.00      12.00      ...      0.20      0.0

[26 rows x 16 columns]
QWidget::setLayout: Attempting to set QLayout "" on pareto "", which already has a layout
0
Cobalt
Algorithm Running
Cobalt fitness is154.66091642900608
Copper
Algorithm Running
Copper fitness is66.88588359907575
Glass
Algorithm Running
Glass fitness is31.655337311017288
```

Fig.7 Sample Output Console with fitness values of materials

```
C:\WINDOWS\system32\cmd.exe - py main.py
Obsidian          12.80      15.00    ...          0          0.0
Plastics(Polyethylene Terephthalate)  11.00      38.50    ...          3.20      0.0
Porcelain         15.00      27.00    ...          0.25      0.0
Silicone          0.05       0.02    ...          1.90      0.0
Silver            74.00      28.00    ...          0          0.0
Stainless steel   190.00     86.00    ...          1.70      0.0
Teflon            0.50       0.01    ...          1.00      0.0
Titanium Alloys   105.00     39.00    ...          1.30      0.0
Tool steel        208.00     88.00    ...          1.20      0.0
Wood              11.00      12.00    ...          0.20      0.0

[26 rows x 16 columns]
QWidget::setLayout: Attempting to set QLayout "" on pareto "", which already has a layout
0
Cobalt
Algorithm Running
Cobalt fitness is154.66091642900608
Copper
Algorithm Running
Copper fitness is66.88588359907575
Glass
Algorithm Running
Glass fitness is31.655337311017288
Cobalt was removed from candidate materials!
Copper was removed from candidate materials!
Glass was removed from candidate materials!
0
Lead
Lead is a toxic material.
```

Fig.8 Sample GUI with result

```
C:\WINDOWS\system32\cmd.exe - py main.py
Lead              15.00       6.00    ...          1.64      1.0
lista Ferritic stainless steel(grade 409)  205.00     81.00    ...          1.70      0.0
lista Martensitic stainless steel(grade 410)  210.00     83.00    ...          1.80      0.0
Obsidian          12.80      15.00    ...          0          0.0
Plastics(Polyethylene Terephthalate)  11.00      38.50    ...          3.20      0.0
Porcelain         15.00      27.00    ...          0.25      0.0
Silicone          0.05       0.02    ...          1.90      0.0
Silver            74.00      28.00    ...          0          0.0
Stainless steel   190.00     86.00    ...          1.70      0.0
Teflon            0.50       0.01    ...          1.00      0.0
Titanium Alloys   105.00     39.00    ...          1.30      0.0
Tool steel        208.00     88.00    ...          1.20      0.0
Wood              11.00      12.00    ...          0.20      0.0

[26 rows x 16 columns]
QWidget::setLayout: Attempting to set QLayout "" on pareto "", which already has a layout
0
Cobalt
Algorithm Running
Cobalt fitness is154.66091642900608
Copper
Algorithm Running
Copper fitness is66.88588359907575
Glass
Algorithm Running
Glass fitness is31.655337311017288
Cobalt was removed from candidate materials!
Copper was removed from candidate materials!
Glass was removed from candidate materials!
```

Fig.9 Sample Output Console with material removed

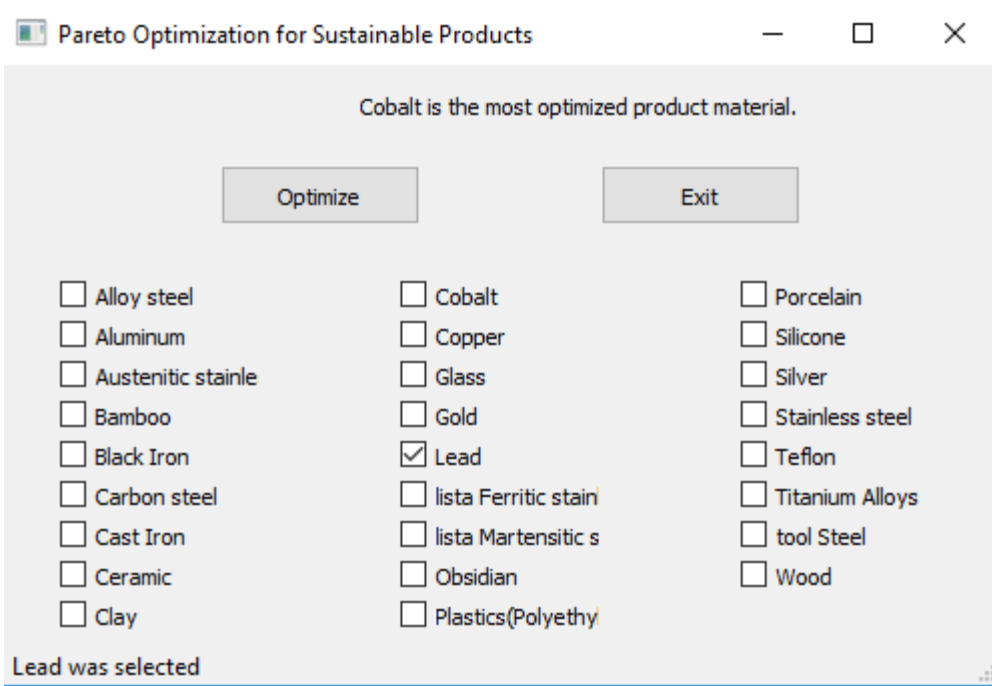


Fig.10 Sample GUI with material

```

C:\WINDOWS\system32\cmd.exe - py main.py
Obsidian          12.80      15.00      ...          0          0.0
Plastics(Polyethylene Terephthalate)  11.00      38.50      ...          3.20      0.0
Porcelain         15.00      27.00      ...          0.25      0.0
Silicone          0.05       0.02      ...          1.90      0.0
Silver            74.00      28.00      ...          0          0.0
Stainless steel   190.00     86.00      ...          1.70      0.0
Teflon            0.50       0.01      ...          1.00      0.0
Titanium Alloys   105.00     39.00      ...          1.30      0.0
Tool steel        208.00     88.00      ...          1.20      0.0
Wood              11.00      12.00      ...          0.20      0.0

[26 rows x 16 columns]
QWidget::setLayout: Attempting to set QLayout "" on pareto "", which already has a layout
0
Cobalt
Algorithm Running
Cobalt fitness is154.66091642900608
Copper
Algorithm Running
Copper fitness is66.88588359907575
Glass
Algorithm Running
Glass fitness is31.655337311017288
Cobalt was removed from candidate materials!
Copper was removed from candidate materials!
Glass was removed from candidate materials!
0
Lead
Lead is a toxic material.

```

Fig.11 Sample Output Console with warning

CHAPTER 5

TESTING

Testing is the process of executing a program with the goal to find errors in the result. This process allows us to determine the performance and accuracy of the algorithm.combination of input values that are tested in a systematic way to detect the errors.

5.1 TEST PLAN

Three testing methods were undertaken: Regression Testing, Usability Testing, and Software Performance Testing. Regression Testing was done on the model to verify the functioning of the neural network which is the fitness function of the algorithm. The neural network was fed with sample data discretely to check the efficiency of the model.

INPUT	Flow name	Quantity	Amount	Unit
	<i>Materials</i>			
	Coke (C carrier)	Mass	373.77	kg
	Cathode carbon	Mass	1074.35	kg
	Pitch	Mass	85.45	kg
	Refractory	Mass	10.82	kg
	Steel sheet part (St)	Mass	4.97	kg
	Cooling water	Mass	4436.30	kg
	<i>Energy and Fuels</i>			
	Electricity	Energy	213.00	MJ
OUTPUT	Thermal energy	Energy	1128.23	MJ
	Natural gas	Mass	138.52	kg
	Diesel	Mass	0.40	kg
	Oil (unspecified)	Mass	14.19	kg
	Flow name	Quantity	Amount	Unit
	<i>Products</i>			
	Anode (C carrier)	Mass	437.23	kg
	<i>Waste for Recovery</i>			
	Steel scrap (St)	Mass	3.10	kg
	Refractory	Mass	1.46	kg
	<i>Emission to Air</i>			
	Benzo{a}pyrene	Mass	0.00	kg
	Carbon dioxide	Mass	239.17	kg
	Dust (PM10)	Mass	0.05	kg
	Hydrogen fluoride	Mass	0.00	kg
	Nitrogen oxides	Mass	0.28	kg
	Sulphur dioxides	Mass	2.86	kg
	Polycyclic aromatic hydrocarbons	Mass	3.20E-02	kg

Usability Testing was carried out after the initial GUI was designed. The flexibility of choosing the materials were tested and the checkboxes were added. The minimal buttons to execute the model and a label to view the result were tested.

Software Performance Testing was done by collecting the fitness value results of each material multiple times, recursively. Then comparing the model's output with the actual value,

the LCA values of certain materials.

Life Cycle Assessment is the traditional way product designers determine the sustainability of a material. The LCA values were calculated along with other parameters manually and were tested

5.2 TEST REPORT

The neural network discretely performed with a great accuracy of 93%. The regression testing gave good performance results for the neural network. The usability testing result was that the GUI is very user friendly.

The GUI has been equipped to display the necessary details to the user and produce the result with proper abstraction of data. The report of the Software Performance Testing is the result of the comparison between the output of the model and the manually calculated data using the LCA values.

5.2.1 Performance

The performance determines the successful scoring capability of the model. The Mean Absolute Error is used as a parameter to determine the performance of the same. Performance testing results in an accuracy of 87%. The accuracy of the algorithm is calculated by determining how close the output of the model is to the original calculated output that is used for testing.

CHAPTER 6

Future Work

Future work is to improve the model by expanding the dataset and increasing the parameters to calculate the sustainability of a product material. The increase in parameters and a larger dataset will lead to a better accuracy of the model. The GUI can also be improved and developed into a software application, which could facilitate the product designer to choose the sustainable product material with a few clicks. The LCA values could be incorporated into the dataset as a parameter which could improve the performance.

The EcoIndicator 99 method can be used to calculate the environmental factors that offers a wider domain to process the data. New eco friendly materials will also be incorporated into the system like Coconut Waste, Sugarcane Waste, etc. Large corporations, who are manufacturing such new eco friendly products, can be approached to obtain the data. This larger option of materials will envision the product designer with newly upcoming sustainable product materials.

CONCLUSION

To reduce consumer waste, an application for product designers to determine the best product materials based on various objectives (using Differential Evolution and Backpropagation Neural Networks) was successfully created. It has been validated with the example of selecting materials for kitchen cutlery, and has an accuracy of 87%.

CHAPTER 7

APPENDICES

7.1 APPENDIX 1

The Importance of Having a Different Fitness Function.

The Objective function is the actual objective of the problem such as in generation we need to minimize the generation cost so the objective function is the generation cost.

The fitness function is one that measures the fitness of the solution to the objective function. In some applications the fitness function is same as the objective function and in others they are not. Such as in cost minimization there are some constraints. For example, minimize the cost but satisfy the load and work within limits. In such constrained problems, fitness function contains two parts the object function and penalty function .

Penalty function is the function which is added in fitness function when constraints are violated to make the solutions feasible and discard the unfeasible solutions , Scaling function scales the fitness function to be suitable for your algorithm For example, if the fitness function is varying from 1 to 1 and your algorithm needs the fitness to be positive we can make scaling to be in range (0 to 2) by adding 1 to our original function. OR from (0 5) by multiply by 5 and add 5.

7.2 APPENDIX 2

Differential Evolution Algorithm for FeedForward Neural Networks -

An evolutionary optimization method over continuous search spaces , differential evolution, has recently been successfully applied to real world and artificial optimization problems and proposed also for neural network training. However, differential evolution has not been comprehensively studied in the context of training neural network weights, i.e., how useful is differential evolution in finding the global optimum for expense of convergence speed. In this study, differential evolution has been analyzed as a candidate global optimization method for feedforward neural networks.

In comparison to gradient based methods, differential evolution seems not to provide any distinct advantage in terms of learning rate or solution quality. Differential evolution can rather be used in validation of reached optima and in the development of regularization terms and nonconventional transfer functions that do not necessarily provide gradient information.

REFERENCES

- 1) Michela Dalle Mura¹ · Francesco Pistolesi² · Gino Dini¹ · Beatrice Lazzerini.. End-of-life product disassembly with priority-based extraction of dangerous parts. Springer(2020)
- 2) Multiresource-constrained selective disassembly with maximal profit and minimal energy consumption Xiwang Guo, MengChu Zhou, Shixin Liu, Liang Qi, IEEE Transactions on Automation Science and Engineering 18 (2), 804-816, 2020
- 3) Is it a strategic move to subsidize consumers instead of the manufacturer? Subrata Saha, Sani Majumder, Izabela Ewa Nielsen, IEEE access 7, 169807-169824, 2019
- 4) Selection of cost-effective investment and output subsidies for eco-friendly products Xiao Su, Bing Sun, Yijiang Liu Journal of Cleaner Production 286, 124985, 2021
- 5) Design and analysis of novel hybrid multi-objective optimization approach for data-driven sustainable delivery systems H Giray Resat IEEE Access 8, 90280-90293, 2020
- 6) A benchmark TEAM problem for multi-objective Pareto optimization in magnetics: The time-harmonic regime Paolo Di Barba, Fabrizio Dughiero, Michele Forzan, David A Lowther, Maria Evelina Mognaschi, Elisabetta Sieni, Jan K Sykulski IEEE Transactions on Magnetics 56 (1), 1-4, 2019
- 7) Variable-length Pareto optimization via decomposition-based evolutionary multiobjective algorithm Hui Li, Kalyanmoy Deb, Qingfu Zhang IEEE Transactions on Evolutionary Computation 23 (6), 987-999, 2019
- 8) A multi-objective genetic algorithm based on fitting and interpolation Chuang Han, Ling Wang, Zhaolin Zhang, Jian Xie, Zijian Xing IEEE Access 6, 22920-22929, 2018
- 9) Multi-island genetic algorithm and Kriging model-based design of vehicle product comprising multi-material Fangwu Ma, Lu Han, Yang Zhou, Shixian Chen, Yongfeng Pu IEEE Access 6, 53397-53408, 2018
- 10) Dalle Mura, M., & Dini, G. (2017). A multi-objective software tool for manual assembly line balancing using a genetic algorithm. CIRP Journal of Manufacturing Science and Technology, 19, 72–83.
- 11) Mahajan, R., & Kaur, G. (2013). Neural Networks using Genetic Algorithms. International Journal of Computer Applications, 77(14), 611. doi:10.5120/135491153
- 12) Mezura Montes, E., & Coello, C. A. (n.d.). Use of Multiobjective Optimization Concepts to Handle Constraints in Genetic Algorithms. Advanced Information and Knowledge Processing Evolutionary Multiobjective Optimization, 229254. doi:10.1007/1846281377_10
- 13) Pratimsarangi, P., Sahu, A., & Panda, M. (2013). A Hybrid Differential Evolution and BackPropagation Algorithm for Feedforward Neural Network Training. International Journal of Computer Applications, 84(14), 19. doi:10.5120/146412943
- 14) Slowik, A., & Bialko, M. (2008). Training of artificial neural networks using differential evolution algorithm. 2008 Conference on Human System Interactions. doi:10.1109/hsi.2008.4581409

- 15) Jauhar, S. K., Pant, M., & Abraham, A. (2014). A Novel Approach for Sustainable Supplier Selection Using Differential Evolution: A Case on Pulp and Paper Industry. *Advances in Intelligent Systems and Computing Intelligent Data Analysis and Its Applications, Volume II*, 105117. doi:10.1007/9783319077734_11
- 16) Jahan, A., Ismail, M. Y., Mustapha, F., & Sapuan, S. M. (2010). Material selection based on ordinal data. *Materials & Design*, 31(7), 31803187. doi:10.1016/j.matdes.2010.02.024
- 17) Jauhar, S. K., Pant, M., & Abraham, A. (2014). A Novel Approach for Sustainable Supplier Selection Using Differential Evolution: A Case on Pulp and Paper Industry. *Advances in Intelligent Systems and Computing Intelligent Data Analysis and Its Applications, Volume II*, 105117. doi:10.1007/9783319077734_11
- 18) Kanthababu, M. (n.d.). MultiObjective Optimization of Manufacturing Processes Using Evolutionary Algorithms. *Industrial Engineering*, 352376. doi:10.4018/9781466619456.ch021
- 19) Livani, E., Nguyen, R., Denzinger, J., Ruhe, G., & Banack, S. (2013). A Hybrid Machine Learning Method and Its Application in Municipal Waste Prediction. *Advances in Data Mining. Applications and Theoretical Aspects Lecture Notes in Computer Science*, 166180. doi:10.1007/9783642397363_13
- 20) Tušar, T., & Filipič, B. (n.d.). Differential Evolution versus Genetic Algorithms in Multiobjective Optimization. *Lecture Notes in Computer Science Evolutionary MultiCriterion Optimization*, 257271. doi:10.1007/9783540709282_22

PLAGIARISM REPORT



Document Information

Analyzed document	Sriya,mounika paper (1).docx (D123742541)
Submitted	2021-12-28T05:36:00.0000000
Submitted by	
Submitter email	veena_k7@rediffmail.com
Similarity	2%
Analysis address	veena_k7.veltec@analysis.orkund.com

Sources included in the report

W	URL: http://www.iaea.org/inis/collection/NCLCollectionStore/_Public/47/127/47127064.pdf Fetched: 2021-12-28T05:36:26.2800000		1
W	URL: https://www.ijcaonline.org/archives/volume84/number14/14641-2943 Fetched: 2021-12-14T07:34:27.3470000		1

JOURNAL PAPER

Pareto Optimization of Product Material Selection: Artificial Neural Networks & Differential Evolution Approach

Ch.Sriya , Mounika Chatharajupalli and Veena K

sriyach07@gmail.com, mounikach2611@gmail.com, veenakanagaraj07@gmail.com

Department of Computer Science and Engineering,

Sathyabama Institute of Science and Technology, Chennai, Tamil Nadu, India

Abstract

Rectifying our adverse effects on the environment, Waste Management, is one of the most significant problems humanity faces and it begins with reducing the production of consumer waste; the waste caused by-product materials. Consumer waste can be reduced by choosing sustainable product materials. To choose sustainable product materials, the product designer must take into account not only environmental factors but mechanical and economic factors as well. This ensures the maximum efficiency of the chosen material concerning the product requirements. It is often difficult to accurately weigh the various properties of the materials and derive the most optimal material that adheres to all the objectives of the product. A cohesive machine learning solution of differential evolution (DE) and back propagation neural networks (BPNs) is proposed to optimize the Pareto product material selection. This approach has been

validated by a model that can select the most optimal material given a list of potential materials for a product. The model has been created such that it will assist product designers during the planning stages of product designing to reduce consumer waste.

Keywords---Material Selection, Pareto Optimization, Differential Evolution, Waste Management

INTRODUCTION

India is one of the places in the world where most garbage is disposed of. India's urban population of 429 million citizens produces 62 million tonnes of garbage annually. Studies have shown that higher income levels and increased urbanization mean more waste is generated. Of all the waste, the total packaging waste is 32 million tonnes per annum (predominantly single-use plastics). This shows a definite link between consumerism and waste. Unless something is done about this consumer waste,

WEF Report estimates that by 2050 there will be more plastic in the ocean than fish. The field that deals with waste is known as waste management. The most troublesome objective of waste management is reducing the production of waste. This objective is one of the hardest to attain because industries would rather focus on financial woes than environmental ones. Instead of dealing with an issue after its creation, prevention of the issue itself is always better. For the issue at hand, consumer waste, it can be prevented by using sustainable materials to create products. This project aims to create a model to allow the product designer to do so.

In 2009, a system was created to select sustainable materials for bottled beverages. [1] It used Genetic Algorithm and Artificial Neural Networks (Back Propagation Neural Networks) to select a material based on multiple objectives like minimizing cost, maximizing mechanical properties and minimizing adverse environmental effects. Environmental impacts were calculated by the LCA method (Life Cycle Assessment Method). [2] The system faced drawbacks of not having data that represented all operations involved to produce the materials, not all environmental factors being considered. Hence it was concluded that it could be made to be more accurate, comprehensive and useful.

To create a better system, the general process of material selection needs to be understood. Material selection [3] using the Linear Assignment method helped formulate a basic

understanding of the process needed to select a product material. [4] Various stages in the process require meticulous precision and thought. An improvement to the Genetic Algorithm is Differential Evolution [5] because it has a superior command over the decision space when multiple objectives need to be optimized. [6] It assists in analyzing more variations of the given options when numerical optimization is required (Differential Evolution is limited to the vector representation of solutions).

Further, the extent of Differential Evolution's application to neural networks is still being studied; where the general activation function [7] for a neural network may be a sigmoid function. [8]

LITERATURE SURVEY

Various scientific journals were analyzed to further investigate and understand the domains of waste management, material selection, and machine learning. Many papers offered an insight that was parallel to the motivation of this project, and existing working systems were understood. In 2020, a system was created to reduce this massive impact on the environment is to disassemble these devices with the aim of reusing and recycling as many parts as possible, they have used the algorithm a multi-objective formulation of the Disassembly Line Balancing Problem (DLBP) and Genetic algorithm which promotes efficiency and includes a new objective that increases the level of safety. There was another paper which studied Evolutionary algorithms and their

applications to engineering problems using Genetic algorithm Genetic programming Differential evolution . simiallry long back in 2009 In 2009, a system was created to select sustainable materials for bottled beverages.[15] It used Genetic Algorithm and Artificial Neural Networks to select a material based on multiple objectives like minimizing cost and adverse environmental effects. Environmental impacts were calculated by LCA method (Life Cycle Assessment Method).

Based on the drawbacks of the existing system, papers were read to find an improvement to the Genetic Algorithm. In this process many algorithms were touched upon till Differential Evolution was found to have superior command over the decision space.[14] Papers with different applications of Genetic Algorithm and Artificial Neural Networks were studied to further understand its working.[1]

Then, the working of Artificial Neural Networks, Differential Evolution and Genetic Algorithm were separately studied in various papers. Insight like the general activation function for a neural network being a sigmoid function was gathered.[3] It was also learned that differential evolution's extent of application to neural networks is still being studied.[2]

PROPOSED WORK AND DESIGN

The materials a product is made of cannot just be determined based on environmental factors. The product designer will have to consider mechanical and economic factors as well. There will be specific objectives based on what the product is built to do - its purpose. This type of

multi-objective optimization of product material selection has been done previously using genetic algorithms and artificial neural networks. [1] However, this solution is not the most optimal. A more accurate solution that manipulates the decision space more efficiently is required. The vision is to use differential evolution (a better evolutionary algorithm for Pareto optimization in comparison to genetic algorithms) with backpropagation neural networks to select the optimal material.

The validity of the model is tested by applying it to the selection of materials for the product, kitchen cutlery - specifically knives, forks, and spoons.

The detailed implementation of the model for kitchen cutlery material selection based on the feasibility study is split into three modules; the dataset, the model (integrated algorithm), and the GUI.

A. Data Design

The dataset utilized by the model is a collection of public numeric data about various properties of materials. The properties are split into three main divisions, and under them are different parameters as follows:

1. Mechanical Properties
2. Economical Properties
3. Environmental Properties

B. Model Design

The model's key element is the machine learning model. The model is an integration of differential evolution and backpropagation neural networks.

Mechanical Properties	Economical Properties	Environmental Properties
Young's Modulus	Price of Raw Material	Embodied Energy
Shear Modulus	Price of Recycle	Embodied Carbon
Poisson's Ratio	-	Average Recycle Content
Elastic Modulus	--	Acidification Potential
Tensile Strength		Global Warming Potential
Compressive Strength		Toxicity
Hardness	-	-
Density	-	-

Table 1. Parameters Chosen for Material Selection

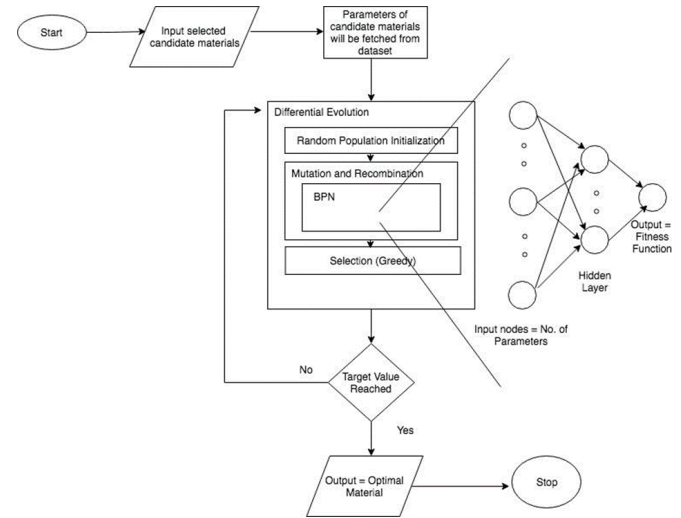


Fig. 1 Flowchart of the Model

METHODOLOGY

The integrated model of Differential Evolution and Backpropagation Neural Networks are used to optimize two functions at its most basic, the Pareto objective function and the fitness function.

The Pareto objective function is the key function that the model is to minimize. The function has three objectives to satisfy pertaining to these properties: mechanical, economic, and environmental.

The function aims to maximize the mechanical properties like hardness and strength while on the other hand, it aims to minimize the economic properties and adverse environmental properties.

(B_m) Best of Mechanical Properties =

$$i=1nMiwi$$

Where $w = 1$ and M_i is the i th mechanical property. Since we are going to maximize this, it'll be $\min(1/B_m)$

RESULTS AND DISCUSSIONS

The model was trained and tested with the selected class of materials for cutlery. As in Fig 2., the list of materials was listed for the user to compare the best fit or choose the optimal material to use for manufacturing the cutlery. The model then processes the selected materials and gives a score that offers the best solution. As in Fig. 3, the output on the console provides the score of each material and also warns the user if a particular element is toxic.

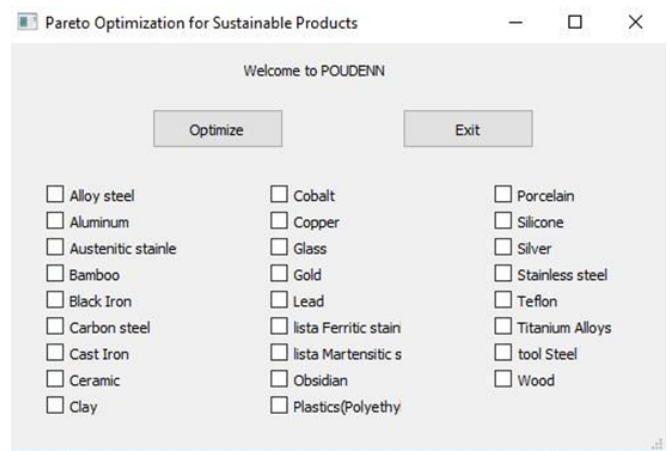


Fig. 2. UI of the Model

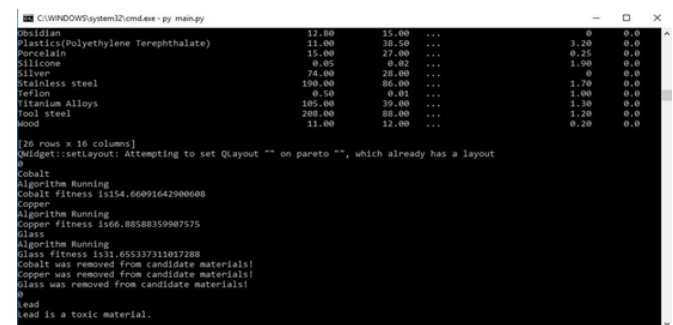


Fig. 3. The output of the Model

A. Performace

The performance determines the successful scoring capability of the model. The Mean Absolute Error is used as a parameter to determine the performance of the same.

CONCLUSION

(Bc) Best of Economical Properties = $\sum_{i=1}^n C_i w_i$

Where $w = 1$ and C_i is the i th property that affects cost. We will be minimizing this thus, $\min(B_c)$

(Be) Best of Environmental Properties = $\sum_{i=1}^n E_i w_i$

Where $w = 1$ and E_i is the i th environmental property. We will be minimizing this thus, $\min(B_e)$

Combining the above-mentioned individual objectives and functions, the Pareto objective function will be as follows:

$$fobj(X) = \min(1/B_m + B_e + B_c)$$

This function will determine the most optimal product material.

The fitness function is the measure of the fitness of an individual with respect to the objective function. In most cases, the objective function is considered the fitness function; however, in this case, there are multiple objectives like minimizing the cost of the material while maximizing the mechanical properties of the material.

This leads to the usage of another function for measuring fitness. The population varies from $[0,1]$, and therefore the random parameters are passed to the neural net. Within the neural net, backpropagation obtains a better fitness value than it would with a normal objective function

To reduce consumer waste, an application for product designers to determine the best product materials based on various objectives (using Differential Evolution and Backpropagation Neural Networks) was successfully created. It has been validated with the example of selecting materials for kitchen cutlery.

FUTURE WORK

The GUI can also be improved and developed into a software application, which could facilitate the product designer to choose the sustainable product material with a few clicks. The LCA values could be incorporated into the dataset as a parameter which could improve the performance. The Eco Indicator 99 method can be used to calculate the environmental factors that offer a wider domain to process the data. New eco-friendly materials will also be incorporated into the system like Coconut Waste, Sugarcane Waste, etc. Large corporations, who are manufacturing such new eco-friendly products, can be approached to obtain the data. This larger option of materials will envision the product designer with newly upcoming sustainable product materials.

REFERENCES

- 1) Michela Dalle Mura¹ · Francesco Pistolesi² · Gino Dini¹ · Beatrice Lazzerini¹. End-of-life product disassembly with priority-based extraction of dangerous parts. Springer(2020)
- 2)Multi resource-constrained selective disassembly with maximal profit and minimal energy consumptionXiwang Guo, MengChu Zhou, Shixin Liu, Liang Qi,IEEE Transactions on Automation Science and Engineering 18 (2), 804-816, 2020
- 3) Is it a strategic move to subsidized consumers instead of the manufacturer?Subrata Saha, Sani Majumder, Izabela Ewa Nielsen ,IEEE access 7, 169807-169824, 2019
- 4) Selection of cost-effective investment and output subsidies for eco-friendly productsXiao Su, Bing Sun, Yijiang LiuJournal of Cleaner Production 286, 124985, 2021
- 5) Design and analysis of novel hybrid multi-objective optimization approach for data-driven sustainable delivery systemsH Giray ResatIEEE Access 8, 90280-90293, 2020
- 6) A benchmark TEAM problem for multi-objective Pareto optimization in magnetics: The time-harmonic regimePaolo Di Barba, Fabrizio Dughiero, Michele Forzan, David A Lowther, Maria Evelina Mognaschi, Elisabetta Sieni, Jan K SykulskiIEEE Transactions on Magnetics 56 (1), 1-4, 2019
- 7)Variable-length Pareto optimization via decomposition-based evolutionary multi-objective algorithmHui Li, Kalyanmoy Deb, Qingfu ZhangIEEE Transactions on Evolutionary Computation 23 (6), 987-999, 2019
- 8) A multi-objective genetic algorithm based on fitting and interpolationChuang Han, Ling Wang, Zhaolin Zhang, Jian Xie, Zijian XingIEEE Access 6, 22920-22929, 2018
- 9) Multi-island genetic algorithm and Kriging model-based design of vehicle product

comprising multi-material Fangwu Ma, Lu Han, Yang Zhou, Shixian Chen, Yongfeng Pu IEEE Access 6, 53397-53408, 2018

10) Dalle Mura, M., & Dini, G. (2017). A multi-objective software tool for manual assembly line balancing using a genetic algorithm. *CIRP Journal of Manufacturing Science and Technology*, 19, 72–83.

11) Mahajan, R., & Kaur, G. (2013). Neural Networks using Genetic Algorithms. *International Journal of Computer Applications*, 77(14), 611. doi:10.5120/135491153

12) MezuraMontes, E., & Coello, C. A. (n.d.). Use of Multiobjective Optimization Concepts to Handle Constraints in Genetic Algorithms. *Advanced Information and Knowledge Processing Evolutionary Multiobjective Optimization*, 229254. doi:10.1007/1846281377_10

13) Pratimsarangi, P., Sahu, A., & Panda, M. (2013). A Hybrid Differential Evolution and BackPropagation Algorithm for Feedforward Neural Network Training. *International Journal of Computer Applications*, 84(14), 19. doi:10.5120/146412943

14) Slowik, A., & Bialko, M. (2008). Training of artificial neural networks using differential evolution algorithm. *2008 Conference on Human System Interactions*. doi:10.1109/hsi.2008.4581409

15) Jauhar, S. K., Pant, M., & Abraham, A. (2014). A Novel Approach for Sustainable Supplier Selection Using Differential Evolution: A Case on Pulp and Paper Industry. *Advances in*

Intelligent Systems and Computing Intelligent Data Analysis and Its Applications, Volume II, 105117. doi:10.1007/9783319077734_11

16) Jahan, A., Ismail, M. Y., Mustapha, F., & Sapuan, S. M. (2010). Material selection based on ordinal data. *Materials & Design*, 31(7), 31803187. doi:10.1016/j.matdes.2010.02.024

17) Jauhar, S. K., Pant, M., & Abraham, A. (2014). A Novel Approach for Sustainable Supplier Selection Using Differential Evolution: A Case on Pulp and Paper Industry. *Advances in Intelligent Systems and Computing Intelligent Data Analysis and Its Applications*, Volume II, 105117. doi:10.1007/9783319077734_11

18) Kanthababu, M. (n.d.). MultiObjective Optimization of Manufacturing Processes Using Evolutionary Algorithms. *Industrial Engineering*, 352376. doi:10.4018/9781466619456.ch021

19) Livani, E., Nguyen, R., Denzinger, J., Ruhe, G., & Banack, S. (2013). A Hybrid Machine Learning Method and Its Application in Municipal Waste Prediction. *Advances in Data Mining. Applications and Theoretical Aspects Lecture Notes in Computer Science*, 166180. doi:10.1007/9783642397363_13

20) Tušar, T., & Filipič, B. (n.d.). Differential Evolution versus Genetic Algorithms in Multiobjective Optimization. *Lecture Notes in Computer Science Evolutionary MultiCriterion Optimization*, 257271. doi:10.1007/9783540709282

