ABSTRACT

An intelligent pricing system for used mobiles is developed that can be utilized in the market place by the customers to do a real-time search on sale items, such as mobile phones. Certain parameters about the phone are provided by the seller to the intelligent system such as usage condition, years of usage of the item, item category and item specifications in order for the intelligent system to find the suitable price for the given mobile phone for sale. Using an artificial neural network, a price for an item or a product is deduced by training the system using above constraints. The application implements an intelligent system that finds the suitable selling price for used mobile phones based on the previous selling history stored in the database using an artificial neural network.
# TABLE OF CONTENTS

ABSTRACT ................................................................................................................................................... ii

TABLE OF CONTENTS ................................................................................................................................ iii

LIST OF FIGURES ........................................................................................................................................ v

LIST OF TABLES ........................................................................................................................................ vii

1. BACKGROUND AND RATIONALE ........................................................................................................ 1

   1.1 INTRODUCTION ................................................................................................................................. 1

       1.1.1 ARTIFICIAL NEURAL NETWORK ............................................................................................. 2

       1.1.2 FEED FORWARD NEURAL NETWORK ..................................................................................... 2

       1.1.3 MULTI LAYER PERCEPTRON ................................................................................................. 4

           1.1.3.1 ACTIVATION FUNCTION ............................................................................................... 4

           1.1.3.2 BACK PROPAGATION ..................................................................................................... 5

       1.1.4 SPRING FRAMEWORK .............................................................................................................. 8

           1.1.4.1 MODEL VIEW CONTROLLER .................................................................................... 8

       1.1.5 BOOTSTRAP FRAMEWORK ................................................................................................. 9

   1.2 LITERATURE SURVEY .................................................................................................................... 9

   1.3 EXISTING SYSTEM .......................................................................................................................... 12

       1.3.1 DIGICIRCLE .............................................................................................................................. 12

       1.3.2 SWAPPA .................................................................................................................................. 12

       1.3.3 GLYDE .................................................................................................................................... 12

   1.4 PROPOSED SYSTEM ........................................................................................................................ 13

2. NARRATIVE ............................................................................................................................................. 14

   2.1 PROBLEM STATEMENT .................................................................................................................... 14

   2.2 MOTIVATION ..................................................................................................................................... 14

   2.3 SYSTEM REQUIREMENTS .............................................................................................................. 14
3. PROPOSED SYSTEM DESIGN ........................................................................ 15
   3.1 SYSTEM ARCHITECTURE ............................................................... 15
   3.2 APPLICATION DESIGN AND ARCHITECTURE ........................... 16
      3.2.1 AUTHORIZATION ................................................................. 17
      3.2.2 RESTFUL SERVICES ......................................................... 17
      3.2.3 HIBERNATE ORM FRAMEWORK ...................................... 17
   3.3 DESIGN FLOW ............................................................................ 18
   3.4 SCHEMA DIAGRAM ................................................................. 19
   3.5 GAIN AND RATE OF DEPRECIATION ........................................ 21
   3.6 CLIENT SERVER COMMUNICATION ......................................... 22
      3.6.1 CLIENT SIDE ................................................................. 23
      3.6.2 SERVER SIDE ............................................................... 23
4. IMPLEMENTATION ............................................................................. 24
   4.1 LOGIN ....................................................................................... 24
   4.2 CALCULATING BEST PRICE .................................................... 28
   4.3 SEARCHING USED PHONES .................................................... 29
   4.4 TRAINING DATA ......................................................................... 30
5. RESULTS .......................................................................................... 41
   5.1 SELLER’S PERSPECTIVE .......................................................... 41
   5.2 PRODUCT TABLE DESCRIPTION ............................................. 43
   5.3 BUYER’S PERSPECTIVE .......................................................... 44
   5.4 PRICE COMPARISON ............................................................... 45
6. CONCLUSION AND FUTURE WORK .................................................. 49
8. REFERENCES .................................................................................... 50
**LIST OF FIGURES**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Layout of Feed Forward Neural Network</td>
<td>3</td>
</tr>
<tr>
<td>1.2</td>
<td>Sigmoid Activation Function</td>
<td>5</td>
</tr>
<tr>
<td>1.3</td>
<td>Pseudo Code of Backpropagation Algorithm</td>
<td>7</td>
</tr>
<tr>
<td>1.4</td>
<td>MVC Components</td>
<td>9</td>
</tr>
<tr>
<td>3.1</td>
<td>System Architecture</td>
<td>15</td>
</tr>
<tr>
<td>3.2</td>
<td>Application Architecture</td>
<td>16</td>
</tr>
<tr>
<td>3.3</td>
<td>Design Flow</td>
<td>18</td>
</tr>
<tr>
<td>3.4</td>
<td>Schema Diagram</td>
<td>20</td>
</tr>
<tr>
<td>4.1</td>
<td>Seller Login Page</td>
<td>24</td>
</tr>
<tr>
<td>4.2</td>
<td>Seller Page</td>
<td>25</td>
</tr>
<tr>
<td>4.3</td>
<td>Buyer Login Page</td>
<td>26</td>
</tr>
<tr>
<td>4.4</td>
<td>Buyer Page</td>
<td>27</td>
</tr>
<tr>
<td>4.5</td>
<td>Used Phone Details Given by Seller and the Best Price by ANN</td>
<td>28</td>
</tr>
<tr>
<td>4.6</td>
<td>Buyer Giving Price and Retrieving the List of Used Phones</td>
<td>29</td>
</tr>
<tr>
<td>4.7</td>
<td>Training Data Before Normalization</td>
<td>30</td>
</tr>
<tr>
<td>4.8</td>
<td>Training Data After Normalization</td>
<td>31</td>
</tr>
<tr>
<td>4.9</td>
<td>Training Neural Network</td>
<td>32</td>
</tr>
<tr>
<td>4.10</td>
<td>Training Data Storage and MLP Object Creation</td>
<td>33</td>
</tr>
<tr>
<td>4.11</td>
<td>ANN Learning</td>
<td>35</td>
</tr>
<tr>
<td>4.12</td>
<td>ANN Training Using Backpropagation Method</td>
<td>36</td>
</tr>
<tr>
<td>4.13</td>
<td>Computation of Weights</td>
<td>37</td>
</tr>
<tr>
<td>4.14</td>
<td>Computing Gradient Descent</td>
<td>38</td>
</tr>
<tr>
<td>4.15</td>
<td>Computing Delta Weights</td>
<td>39</td>
</tr>
<tr>
<td>4.16</td>
<td>Updating Weights</td>
<td>40</td>
</tr>
</tbody>
</table>
Figure 5.1: Inputs Given By the Seller and the Output from ANN………………………………41
Figure 5.2: Showing the Results on Seller Side……………………………………………………41
Figure 5.3: Inputs Changed By the Seller and the Output from ANN…………………………42
Figure 5.4: Showing the Results on Seller Side……………………………………………………42
Figure 5.5: Inputs Given By the Seller to Observe the Rate of Depreciation…………………..43
Figure 5.6: Description of Product Table …………………………………………………………44
Figure 5.7: List of Used Phone for a Given Price………………………………………………45
LIST OF TABLES

TABLE 5.1: List of Used Phone Compared with Other Websites………………………….46

TABLE 5.2: User Rating……………………………………………………………………48
1. BACKGROUND AND RATIONALE

1.1 Introduction

Information technology is rapidly used in the development of online marketing. Various trading platforms are used in buying/selling of products over the internet that lowered the commissions over the internet and also improved the speed of transactions. The trading over the internet can be called as Electronic trading or e-trading [1]. It was during the late 20th century when transactions over remote locations became much preferred, compared to transactions over physical locations. Many trading platforms were designed for direct customer marketing.

It has become a universal that as everything is touched and affected by modern technology [2]. There are certain important things that should be concerned before purchasing used phones. This includes brief description of the device, its parts, performance, features, years of usage and most importantly the price of it.

The major point of buying a used phone is to get the product at a good price [3]. Used phones are described in various levels of conditions, so there is a need to know whether the condition meets what the user is looking for. Once such a product is found look whether the price specified is worth for the conditions of the phone specified. The price of a new phone with similar specification is considered while making a decision about the used mobile to decide whether the deal is worth or not.
1.1.1 Artificial Neural Network (ANN)

Artificial neural network is a computational model inspired by natural neurons in biological neural networks. It is interconnected with neurons also known as units to transmit messages between each other and the connection between each unit is called weight. [4] ANNs combine artificial neurons in order to process information. ANNs can be used for:

- Capturing connections within a set of patterns.
- Where the capacity and size of the data is huge.
- The associations between the data are uncertain.
- The associations are difficult to describe with traditional approaches.

It consists of three layers input, hidden and output layer. In [5] a neural network where input units are used to feed the data into network which eventually triggers the layers of hidden inputs to perform various functions and finally arrive at the output layers. Its main purpose in the application is to recommend best selling price of the used products to both seller and buyer in the website during trading. There are different types of Artificial Neural Networks such as feedforward, recurrent, modular, physical etc. that use variety of topologies and algorithms to learn the network.

1.1.2 Feed Forward Neural Network

It is an artificial neural network in which its neurons or units are connected in such a way that they do not form a directed cycle [6], [7]. They move in only one direction from the input layer to the output layer. It has many layers or sub-groups of neurons or units or processing elements, every layer makes its own independent computations and sends it forward to the next layer. Each unit computes based on the weighted sum of its inputs. Figure 1.1 shows the layered model of the proposed application. All the nine factors represent the nine inputs in the input layer and the
neurons in hidden and output layers; $w_1$ is the output (weighted sum) of the hidden layer where $o_1$ is the final output in the feed forward phase. The nine inputs are fed into nine individual neurons along with the bias neuron which increases the performance of the ANN that computes an activation function for each neuron in the input layer and forwards it to the hidden layer. The output of the input layer is input to the hidden layer it also contains a bias neuron and computes the activation function for the neuron in the hidden layer and forwards the result to the output layer. The neuron in the output layer uses the output of hidden layer which is the input of output layer and computes the activation function to obtain the final output.

![Figure 1.1 Layout of Feed Forward Neural Network](image-url)
1.1.3 Multi-Layer Perceptrons (MLPs)

To learn about binary classifiers in feedforward neural network, an algorithm is used in machine learning called the perceptron [8], [9]. Its major function is to check whether an input belongs to one class or another class. There are two types of perceptrons:

1) Single layer perceptron - that deals with linear decisions which classify the prediction linearly using linear prediction function. It then combines the set of weights of each element with the feature vector and processes the elements in a training set one at a time. They are capable of learning only linearly separable patterns.

2) Multi-layer perceptron - In order to train and recognize many classes of patterns a new field is developed in the neural network called Multi-layer perceptron (MLP) that can deal with non-linear decisions. It was identified as a feedforward neural network with two or more layers that has a greater processing power when compared to single layer perceptron. It is most commonly used type of neural network that consist of input layer, output layer and one or more hidden layers. Each layer consists of neurons that are linked to the neurons from the previous and the next layer with the help of directions. To train the network, MLP uses a learning technique called “Back-Propagation”. It is a form of learning where the function provides continuous feedback on the quality of solutions obtained so far.

1.1.3.1 Activation Function

All neurons in a multi-layer perceptron have linear activation function [9] which maps the weighted inputs to the output of each neuron. This can be easily shown by linear algebra that it can be deduced to an input-output layer model. The activation function used in this application is a sigmoid as shown in Figure 1.2. This function is finite as it significantly presents weights within
a range (-1, 1) and consists of results which lies in both positive and negative area. Based on different gradient training methods it can be optimized into gradient-descent and gradient-ascent methods. A sigmoid is a mathematical function having an “S” shape referring to a special case of the logistic function and is defined by the formula

\[ S(t) = \frac{2}{1 + e^{-2t}} - 1 \]

Activation = sum of product of inputs and its respective weight at each node

![Sigmoid Activation Function](image)

**Figure 1.2 Sigmoid Activation Function**

### 1.1.3.2 Back-Propagation (BP)

It is a supervised learning technique, every piece of data is processed and then the connection weights are changed for learning to take place. It works by approximating the non-linear relationship between the input and output by changing the weights. It can be divided into two steps:

  - Feedforward and Backpropagation [10].
In the first step, an input pattern is applied to the first layer that is the input layer and the weight from that layer is given as input to the neurons in the next layer (hidden layer) and so on until we get the result in the last layer that is output layer. It finds the outputs at the output layer after presenting the input at input layer. Then it computes the weighted sum (activation) and the value “f (activation)” which is the output of each neuron in each layer. Activation is the sum of product of inputs and its respective weight at each node.

In the second step, an error signal is calculated by comparing the expected output with the output value, all the nodes in the previous layers have contributed some part for this error so this error signals are transferred backwards from the output layer to the hidden layers to the input layer and all the neurons/nodes calculate their relative error signal, the errors are used to update the weight at each neuron/node. This algorithm using a delta rule or gradient-descent technique keeps looking for the minimum error function value which is then considered as a solution to the learning. In the delta rule, gradient descent of each node is computed starting from the output neurons in output layer towards the input neuron in the input layer. The delta weights are computed for all the neurons in hidden and output layers based on previous layer’s output and finally update weights. After updating the weights, it calculates the error for all the training data and checks if the error exceeds certain limit. If the error is above the given limit it continues the back-propagation algorithm until the value is maintained below the limit and stops the learning. Figure 1.3 shows the pseudo code that describes the working of the back propagation algorithm:
Assign all inputs and output the weights

Initialize all weights with small random numbers

repeat

    for every pattern in the training set

        Display the pattern to the network

        //promote the input using forward pass through the network:

        for each layer in the network

            for every node in the layer

                1. Compute the weight sum of inputs

                2. Add threshold to sum

                3. Compute the activation function

            end

        end

        //promote the errors using backward pass through the network

        for every node in the output layer

            Compute the error signal

        end

        for all hidden layers

            for every node in the layer

                1. Compute the node's signal error

                2. Update each node's weight in the network

            end

        end

        // Compute Global Error

        Compute the Error Function

    end

end

Figure 1.3 Pseudo Code of Backpropagation Algorithm
1.1.4 Spring Framework

The proposed application uses the spring framework [11] for the JAVA platform. It is an application framework providing several services. The application uses the Model-View-Controller (MVC) module of the spring framework.

1.1.4.1 Model-View-Controller (MVC)

MVC is a software architecture [11] that implements user interface. It separates the software application into three components model, view and controller as shown in Figure 1.4. These three components are interconnected.

- Model is a part that implements logic for the application’s data. They store and retrieve commands from the controller and displayed in the view.
- View is a part that display the application’s user interface (UI) based on changes in the model.
- Controller is the part that handles user interaction. It passes the commands to the model to update the model’s state and passes commands to view to modify its presentation based on model.
1.1.5 Bootstrap Framework

It is an open source front end framework, used to create web applications. It contains several design templates for HTML, CSS and Javascript extensions.

1.2 Literature Survey

Most of the artificial neural network projects are developed using back-propagation (BP) algorithm. A model is implemented which predicts rainfall using artificial neural network [6] using feedforward method which computes the weighted sum of inputs using the three layers.

To build this model input and output data are selected for learning using the supervised BP. Input and output are of different units which need to be normalized in order to get a correlation between the input and output values. A normalized data is obtained using the Matlab Back-propagation algorithm. The learning cycle in the back-propagation algorithm is usually done in two phases:
1) Input patterns are propagated through the network.

2) Output is adapted by changing the weights in the neural network. Random training sample is selected to train the data; every time we train the data the set of data is not fixed and finally testing is conducted in order to keep the error rate below tolerance level. Finally a comparison is made between the actual and predicted output to compute the accuracy of the model.

The BP (Back Propagation) artificial neural network [12] algorithm’s prominent design feature is to adjust the weight of the network continuously by calculating the quadratic sum of the error between output weight of the network model and the consistency sample output to achieve the desired weight. The main goal of the algorithm is to adjust the weight to the smallest total error. Its learning method consists of forward and back propagation. Through repetitive iterations of these propagations, neuron weights and thresholds are changed continually to reduce the error function such that there will be no longer reducing of error function. The objective of this learning process in this algorithm is mainly to minimize the value of the error function, which trains repeatedly. Eventually the value of the error function converges to the minimum point while training, then the details of the weights and bias values are saved in order to process the input samples, thereby computing the output of the sample that is predicted.

The trading platform utilized in the market place by the customers to do a real-time search on sale items. The decision support system [13] coordinates with the hybrid neighborhood search algorithm to find the cost of the sale item when it is transacted over the Internet. Certain constraints are provided by the seller to the intelligent system such as usage condition, years of usage of the item and item category in order to produce a real-time search on suggested items in the market
along with their recommended price. The data mining tools are used in the database to efficiently process and identify multiple groups of data. With the help of an artificial neural network items or products are recommended to the buyer using the previous purchase history from the databases. Items can be suggested to the customers based on recently purchased products or the customer can express his own desires. An intelligent, simple-to-use and user-friendly trading platform that uses direct customer-to-customer marketing in the website for an amiable trading experience is developed.

Various experiments are performed to predict results using ANN [14] - one such research which predicts the reservoir sensitivity by diagnosing damage formation by conducting many experiments. Two methodologies are studied to predict the reservoir sensitivity 1) Back-propagation used to evaluate the reservoir sensitivity and 2) Adaptive Resonance to diagnose the formation damage in it. In order to handle inaccurate and uncertain data artificial neural network uses back-propagation algorithm to evaluate the reservoir by giving sensitivity-influence factors as input in the input layer neurons and output is retrieved using output layer neurons which consists of the parameters of the reservoir sensitivity. The hidden layer uses forty-nine neurons to determine the structure of the network to evaluate the sensitivity of formation. Based on the different features of the diagnosis of formation damage adaptive resonance is selected which is a real-time and online learning ability. This methodology can handle pivotal features based on the ratio of the certain features to all the features. The resultant output after the evaluation of formation damage is reasonable and dependable.
1.3 Existing System

1.3.1 Digicircle

Digicircle is a website [15] that provides customer service via internet through which phones are sold and bought worldwide. Individuals or businesses can use second-hand items, such as mobile phones and other electronic gadgets. It conducts wholesale, retail, online event trading and other services. Trading items online can reduce global carbon emission.

1.3.2 Swappa

It is a website [16] where the ads are free or paid depending on the product classification and geographical market. It’s a simple marketplace for buying and selling mobile phones over the internet. The phones are bought for cheaper price even without a detailed description of their condition.

1.3.3 Glyde

It is a website [17] that provides quality used products without paying the full retail price. It offers a simple selling experience to the seller and ensures a complete satisfaction to the buyer. Sitting at home, the seller can sell an item from an item list in seconds.
1.4 Proposed System

The proposed approach provides an intelligent pricing system for the seller who wants to sell his/her used phone in any marketplace. To develop this intelligent system, it uses the artificial neural network technique. The application is developed for estimating a suitable price for a product which compares certain features of the product and uses an intelligent method to derive the rate of depreciation of the product. By this approach, the online sale on products is organized and completed by the seller. The major goal of this project is to provide an intelligent pricing system for the sellers and buyers to get the best mobiles for what they can afford for.
2. NARRATIVE

2.1 Problem Statement

As the technology is increasing rapidly, there are multiple new mobiles that are entering market day-by-day. Users are always fascinated to the new look and feel of the new mobiles that are emerging; often wish to sell their mobiles to buy new ones. Trading platforms are used for buying used mobiles for extended use with cheaper price and good condition. The seller doesn’t know exactly for how much he/she can sell his/her used phone, he/she just sets the price by relying on his/her intuition most of the times.

2.2 Motivation

There are many applications of artificial neural networks in the field of technology. One such application is calculating the price for used mobile phone by the intelligent system in a way that it is beneficial to both the sellers and buyers. This calculation involves considering many factors like rate of depreciation of the product. System gets trained on the previous data (historical data) to produce apt selling price for the mobile phone that was used earlier. It provides an intelligent pricing platform for both sellers and buyers which is simple and user friendly.

2.3 System Requirements

To develop this application, the following systems and tools are used:

- Operating System: Windows
- Development Environment: Eclipse
- Software: Java SE 8
- Framework: Spring Framework, Hibernate
- Application Server: Apache Tomcat 8
3. PROPOSED SYSTEM DESIGN

3.1 System Architecture

Figure 3.1 shows the architecture of the system that employs different methods of information technology implemented in it. The intelligent system is designed with the user-friendly interface. A suitable solution is proposed in determining a best price for the items (used phones) in the marketplace. Since the application aims at selling used phones, there would be depreciation for a product that depends on the selling trend of the product. A pricing engine is used which works on the principle of artificial neural networks that trains the data from the past trading data (historical data) in the website. A database is needed in retrieving the data of related items.

Figure 3.1 System Architecture
To determine a best price for the used products a price engine is used which is based on an artificial neural network, and the conditions provided are price, number of years of product usage, product description, and some product specifications etc.

### 3.2 Application Design and Architecture

The architecture design of the proposed application is shown in Figure 3.2.

![Figure 3.2 Application Architecture](image-url)
3.2.1 Authorization

The users of the application should be authenticated before they can access the application using a browser. The login mechanism is provided only to registered users, which is secured using spring security. The user interface (UI) is designed using HTML/JSP, Javascript or JQuery in the presentation layer.

3.2.2 RESTFUL Services

The domain/business layer uses the Spring framework, and REST web service endpoints to communicate between the presentation and business layer. These services are built especially to work best on web. [8] The RESTful (Representational State Transfer) architecture is a typical HTTP and is designed to use in a stateless communication protocol. The resources are exchanged between the clients and servers using a standardized protocol and interface.

3.2.3 Hibernate ORM Framework

The persistence layer is designed using the Hibernate ORM framework [18] which maps Java classes to database tables in order to query and retrieve the data. It is a framework through which traditional relational database is mapped to an object oriented model in Java language and MYSQL database is used as a datasource. To retrieve the original price of the items a pricing engine is used based on an artificial neural network to suggest a suitable price on the regular and used products for trading in website.
3.3 Design Flow

Figure 3.3 Design Flow

Figure 3.3 shows the basic design flow of how the trading is accomplished by the seller. At first the user logs in into the application and only a registered user can find the best price for the product to sell in any website. Various factors such as the suitable price for the product, condition of the product, and a brief description of the product are mentioned by the buyer. With the help of these factors, the seller computes the depreciated cost price and gain of the product. Both of these constraints are inversely proportional to each other. The artificial neural network trains the data based on past trading information from the website and derives a suitable solution by minimizing the rate of depreciation and keeping the product gain high to produce a best price of the product.
3.4 Schema Diagram

Figure 3.4 depicts a schema diagram of the application. The user table which contains the user login information is enabled by default. It directly leads to the authority table which differentiates between the seller (admin) and buyer (user) based on their user id. The customer table consists of the buyer profile along with an order number of the particular product. The order table bundles the number of orders by the particular buyer and increments it based on the number of orders. The payment table takes care of the payment for a particular order.

The stock keeping units (SKU) table stores the information of different products with varying prices from different sellers. The category table keeps the information regarding the type of product, whether it’s a mobile, laptop, desktop, television set or any other items. The product table stores the various features of the products, conditions of the product and to which category this product belongs. In all the tables except the user and authorities table it contains an attribute called metadata which stores the metadata of the product in bytes (binary format) and helpful in serialization of the product from one format to binary. When the data is transmitted in the network between the clients and servers it’s easy for them if the data is in binary format and also for training the data using artificial neural network it’s useful. Even if a new feature is added for a particular product if that feature is in metadata it can be directly added into the tables without modifying the tables which will complicate while training the data.
Figure 3.4 Schema Diagram

The training table consists of the important features based on which the best price is computed for a product is specified. Training the data is performed by the server and the resultant results are viewed in the client side of the network. The best price is calculated for a product by checking the historical sales record data of the product. The mobile data is given by the client.
along with its different features then the sever trains the weights based on the features and rate of depreciation is computed for the particular product which should be less when compared to the gain of the product inorder to provide a suitable and best price for the product. Only the table’s category, users, authority, product and training_data are used in the present implemented intelligent pricing system. The other tables are used for the trading platform that is implemented in future.

3.5 Gain and Rate of Depreciation

To sell the used items in online marketing through internet the following formula is used in the proposed solution: To compute the selling price

\[
\text{Selling Price (S.P)} = \text{Cost Price (C.P)} + \text{GAIN.}
\]

Since the website sells used products, there would be depreciation for that particular product which is computed using

\[
\text{Depreciated Cost Price} = \left(1 - \frac{r}{100}\right)^n \times \text{C.P.}
\]

Where r is the rate of depreciation, n is number of years used and C.P is Cost Price. Here the rate of depreciation is unknown as it is not a constant value and depends on the selling trend of the product. The Artificial neural network is used in the proposed solution to learn the past trading information from the website and derives a rate of depreciation such that the gain is high by training the data from the database.

So now,
\[
S.P = \left(1 - \frac{r}{100}\right)^n \times C.P + \text{GAIN}
\]

Thus,

\[
\text{GAIN} = S.P - \left(1 - \frac{r}{100}\right)^n \times C.P, \text{ where GAIN is inversely proportional to rate of depreciation.}
\]

Seller provides his phone specs along with accessories included, phone condition (good, ok, bad), number of years used. As the network is trained, it calculates output at the output layer by feedforwarding the input pattern at the input layer. Now, it calculates the rate of depreciation value using the following formulas:

\[
F = (\text{delta factor} - \text{accessories included})^{\text{phonecondition}}
\]

\[
C = F < (1\text{-output})? [(1\text{-output}) + F]: (1\text{-output})
\]

\[
\text{Rate} = \text{Rate} \times C^{\text{number of years}}
\]

\[
\text{Output} = \text{output computed from the ANN}
\]

### 3.6 Client Server Communication

The RESTful endpoints are exposed in the server side and the front end AngularJS communicates through the web services.

The RESTful web services that are consumed, then communicates with MySQL database to persist/read data, “Hibernate” is used as an ORM framework to perform the CRUD operations in the database. Business logic, such as ANN (training, learning the neural network) is handled in the Java end.
3.6.1 Client Side

The client side is built using HTML with bootstrap CSS design. The Javascript framework used is AngularJS. According to the seller’s perspective, system takes the phone specifications to calculate the suitable selling price where as in buyer’s perspective, system asks the buyer to enter the price within his budget to buy a used phone. In this case, system displays all the available mobile phones from the database to the buyer. The AngularJS is used to collect the data and sends it to server. It displays the best selling price for used phones for the seller and details of used phones list for the buyer after getting response from the server.

3.6.2 Server Side

The server side is built using Java- Spring Rest framework. The database used is MySQL, and “hibernate” is used as an ORM framework. There are three layers in the server side: outermost layer is controller, middle layer is service layer and the innermost layer is DAO (data access object).

The data from the seller/buyer is collected at the controller layer in the server-side. The controller request services from the service layer and the service layer accomplishes it by accessing the data through DAO layer.
4. IMPLEMENTATION

4.1 LOGIN

Separate login information is created for both the seller and the buyer as shown in Figure 4.1. Buyer credentials will be hidden when a seller logs in and vice versa.

Seller logging in:

![Figure 4.1 Seller Login Page](image-url)
After seller logs in:

![Seller Page](image)

**Fig 4.2 Seller Page**

Figure 4.2 is exhibited after the seller logs in with his/her used phone specifications such as phone model, platform, RAM, battery etc., number of years the phone has been used, and phone condition.
Buyer logging in:

The login page of the buyer looks like as shown in Figure 4.3.

Fig 4.3 Buyer Login Page
After buyer logs in:

![Figure 4.4 Buyer Page](image)

The screen shown in Figure 4.4 is exhibited after the buyer logs in with his credentials asking for price that he/she wants to pay for the used phone.
### 4.2 Calculating Best Price

The seller gives all the factors of his/her used phone as shown in Figure 4.5. When the seller clicks the “Calculate Price” button it computes the best price for the required product and displays the result.

**Figure 4.5 Used Phone Details Given by Seller and the Best Price by ANN.**
4.3 Searching Used Phones

Figure 4.6 Buyer Giving Price and Retrieving the List of Used Phones.

When the buyer gives a price and clicks the “Search Phones” button a list of used phones whose price is appropriate to the given price is displayed as shown in Figure 4.6.
4.4 Training Data

The data for used phones is collected from various customer-to-customer trading websites. Figure 4.7 shows the training data before normalization:

The data is normalized before it is trained. Normalization is done by dividing all the values with the maximum value with in the respective column and adjusts the values. Thus, it scales all the values in the range (0, 1). The formula to normalize the data is given below:

\[
X(\text{new}) = \frac{X - X(\text{min})}{X(\text{max}) - X(\text{min})}
\]
In our application, normalization is used to adjust the weights of different inputs such as the RAM, Front camera, Battery, Screen resolution etc. Normalization is done to achieve the same range of values for each of the inputs to the ANN model; otherwise the network will be ill-conditioned. It guarantees stable convergence of weights and biases. The data after normalization would be like in Figure 4.8.

![Figure 4.8 Training Data After Normalization](image-url)
The code in Figure 4.9 explains how the Neural Network is trained:

```java
@Override
@SuppressWarnings("unchecked")
@Transactional
public List<TrainingData> getTrainingData() {
    Session session = this.sessionFactory.getCurrentSession();
    Criteria trainingDataCriteria = session.createCriteria(TrainingData.class);
    List<TrainingData> trainingList = trainingDataCriteria.list();
    return trainingList;
}
```

**Figure 4.9 Training Neural Networks**

It fetches the training data from the database and sends it as a list to train the neural network.

First, it gets the current session into session object by “getCurrentSession()” function, then it creates a “TrainingData” class object by creating a criterion object “trainingDataCriteria”. Then, the training data is fetched from the database by “trainingDataCriteria.list()” function and stored into a list “trainingList”.
List<TrainingData> trainingDataList = dataAccessService.getTrainingData();
for (int i = 0; i < trainingDataList.size(); ++i) {
    ex.add(new float[9]);
    out.add(new float[1]);
}
for (int i = 0; i < trainingDataList.size(); i++) {
    ex.get(i)[0] = trainingDataList.get(i).getFactor1();
    ex.get(i)[1] = trainingDataList.get(i).getFactor2();
    ex.get(i)[2] = trainingDataList.get(i).getFactor3();
    ex.get(i)[3] = trainingDataList.get(i).getFactor4();
    ex.get(i)[4] = trainingDataList.get(i).getFactor5();
    ex.get(i)[5] = trainingDataList.get(i).getFactor6();
    ex.get(i)[6] = trainingDataList.get(i).getFactor7();
    ex.get(i)[7] = trainingDataList.get(i).getFactor8();
    ex.get(i)[8] = trainingDataList.get(i).getFactor9();
    out.get(i)[0] = trainingDataList.get(i).getResult();
}
int nn_neurons[] = { ex.get(0).length, 1, 1 };
Mlp mlp = new Mlp(nn_neurons, ex, out);
mlp.createMlpForMobile();

**Figure 4.10 Training Data Storage and MLP Object Creation**

The code in Figure 4.10 gets the training data having 9 factors and 1 output and trains the neural network by creating a multi-layer-perceptron object. “nn_neurons” is an array that sets the length of input, output and hidden layer (9, 1, and 1) respectively here.
It gets the training data through the function “getTrainingData()” and stores it into the “trainingDataList”. “ex” and “out” are two arrays, “ex” has 9 float variables each containing factors of the used phone and out contains 1 float variable for best price, “nn_neurons” is an array which has 9, 1, 1 as its values. Then it creates an MLP object and sends the “nn_neurons”, “ex” and “out” as the attributes to the MLP object, “createMlpForMobile()” function trains the neural network.
public void createMlpForMobile() {

for (int i = 0; i < 5000; ++i) {

    learn(ex, out, 0.3f);

    float error = evaluateQuadraticError(ex, out);

}

//learn function

public void learn(ArrayList<float[]> examples, ArrayList<float[]> results, float learning_rate) {

    float e = Float.POSITIVE_INFINITY;

    while (e > 15f) {

        batchBackPropagation(examples, results, learning_rate);

        e = evaluateQuadraticError(examples, results);

    }

}

Figure 4.11 ANN Learning
In Figure 4.11 the function “CreateMlpForMobile()”, a learn method is called to learn the ANN iteratively (here 5000 times) and the error rate is set to 0.3. And the learn method is to learn (train) the ANN. Inside the learn function, the data is trained through “batchBackPropagation” method.

```
private void batchBackPropagation(ArrayList<float[]> examples,
                                 ArrayList<float[]> results,
                                 float learning_rate)
{
    resetWeightsDelta();
    for (int l = 0; l < examples.size(); ++l) {
        evaluate(examples.get(l));
        evaluateGradients(results.get(l));
        evaluateWeightsDelta();
    }
    updateWeights(learning_rate);
}
```

**Figure 4.12 ANN Training Using Backpropagation Method**

The code in Figure 4.12, trains ANN using Backpropagation method. Its goal is to optimize the weights so that the neural network can learn how to correctly map arbitrary inputs to outputs. The function “evaluate(examples.get(l))” evaluates the output (weighted sum) for all the neurons. For
the input layer, the weighted sum is calculated and forwarded to the next layer and so on till the last layer. The sigmoid activation function is used in this algorithm to decide whether to fire or not the neuron.

In the function evaluate, it calls one more function, evaluate for each layer which in turn calls function activate for each neuron in a layer is given below in Figure 4.13 to show how the weights are calculated:

```java
public float activate(float inputs[]) {
    _activation = 0.0f;
    assert(inputs.length == _synapticWeights.length);
    for (int i = 0; i < inputs.length; ++i) // dot product (product scalar)
        _activation += inputs[i] * _synapticWeights[i];
    // phi(_activation), our activation function (tanh(x))
    return 2.0f / (1.0f + (float) Math.exp(-_activation) * lambda)) - 1.0f;
}
```

**Figure 4.13 Computations of Weights**

As shown in Figure 4.14, the function “evaluateGradients(results.get(l))” produces an array named as results[] containing output variables of the neurons in a layer. This function finds the error signal at each node in each layer by multiplying with the activationDerivative. Activation derivative is the derivative of the activation function used (sigmoid).
In the `evaluateGradients` function, error signal is calculated as shown in the following code, 
“`getActivationDerivative`” is derivative of the activation function used (sigmoid).

```java
private void evaluateGradients(float[] results)
{
    // for each neuron in each layer
    for (int c = _layers.size() - 1; c >= 0; --c) {
        for (int i = 0; i < _layers.get(c).size(); ++i) {
            // if it's output layer neuron
            if (c == _layers.size() - 1) {
                _grad_ex.get(c)[i] =
                2 * (_layers.get(c).getOutput(i) - results[0]) * 
                _layers.get(c).getActivationDerivative(i);
            }
            else { // if it's neuron of the previous layers
                float sum = 0;
                for (int k = 1; k < _layers.get(c+1).size(); ++k)
                    sum += _layers.get(c+1).getWeight(k, i) * 
                    _grad_ex.get(c+1)[k];
                _grad_ex.get(c)[i] =
                _layers.get(c).getActivationDerivative(i)*sum;
            }
        }
    }
}
```

**Figure 4.14 Computing Gradient Descent**
As shown in Figure 4.15, the function “evaluateWeightsDelta()” calculates the delta value for each weight for each neuron/node. Delta values will be in the delta_w array.

```java
private void evaluateWeightsDelta() {
    // evaluate delta values for each weight
    for (int c = 1; c < _layers.size(); ++c) {
        for (int i = 0; i < _layers.get(c).size(); ++i) {
            float weights[] = _layers.get(c).getWeights(i);
            for (int j = 0; j < weights.length; ++j)
                delta_w.get(c)[i][j] += _grad_ex.get(c)[i] * 
                _layers.get(c-1).getOutput(j);
        }
    }
}
```

Figure 4.15 Computing Delta Weights

And finally as shown in Figure 4.16, the function “updateWeights(learning_rate)” updates the weights of each neuron in every layer.
private void updateWeights(float learning_rate) {
    for (int c = 0; c < _layers.size(); ++c) {
        for (int i = 0; i < _layers.get(c).size(); ++i) {
            float weights[] = _layers.get(c).getWeights(i);
            for (int j = 0; j < weights.length; ++j)
                _layers.get(c).setWeight(i, j, _layers.get(c).getWeight(i, j) - (learning_rate * _delta_w.get(c)[i][j]));
        }
    }
}
5. RESULTS

5.1 Seller’s Perspective

The seller gives his/her used phone specifications and clicks the calculate price button, and then the price will be suggested on the right hand side of the webpage as shown in Figure 5.2.

Input given and the corresponding output are shown in Figure 5.1.

<table>
<thead>
<tr>
<th>Phone Model</th>
<th>platform</th>
<th>screen res</th>
<th>Accessories</th>
<th>weight</th>
<th>ram</th>
<th>front cam</th>
<th>battery</th>
<th>memory</th>
<th>multi-sim</th>
<th>condition</th>
<th>years used</th>
<th>buyin price</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple iphone 6</td>
<td>iphone</td>
<td>4.7</td>
<td>yes</td>
<td>129</td>
<td>1</td>
<td>1.2</td>
<td>1810</td>
<td>128</td>
<td>no</td>
<td>good</td>
<td>1</td>
<td>800</td>
<td>570</td>
</tr>
</tbody>
</table>

Figure 5.1 Inputs Given by the Seller and the Output from ANN

Figure 5.2 Showing the Results on Seller Side

Now, the factors like accessories and condition of the used phone have been changed as shown in Figure 5.3, and correspondingly Figure 5.4 shows the output.
Figure 5.3 Inputs Changed by the Seller and the Output from ANN

To show that the non-numerical data of the used phone are also considered in training the data, for a multi-sim, two options yes or no are converted into 0.8 and 0.2 respectively. For conditions of the phone like good, bad and ok, the rated depreciates 90%, 30% and 50% respectively.

Figure 5.4 Showing the Results on Seller side

The data like phone condition and accessories are changed, but any other factors such as memory, RAM, front camera have not been changed because the normalized data has already been given in
the database, so a random phone details cannot be given i.e., the artificial neural network should know the used phone (from its historical data) to suggest a best price for it.

5.2 Product Table Description

The best price calculated for a list of used phones with different phone models and their respective factors are exhibited in Figure 5.5. The output is stored in the product table. The list of used phones that has been queried are shown in Figure 5.5. The depreciation rate is observed by changing the non-numerical factors such as accessories and condition for the same phone.

**Figure 5.5 Inputs Given by the Seller to Observe the Rate of Depreciation**

The description of the product table is shown in Figure 5.6 to demonstrate that the above list has actually been queried and executed.
5.3 Buyer’s Perspective

A buyer can give the price that he/she wants to buy a used phone for and clicks the “search phones” button, then a list of used phones will be suggested as shown in Figure 5.7.
Figure 5.7 List of Used Phone for a Given Price.

Figure 5.7 shows the list of used phones within the range $(x-51, x+51)$ for a given price of $x$, this list is received from the product table.

5.4 Price Comparison

The selling price which was computed using the ANN is compared with other ecommerce websites for the same phones. The price of the phone is checked based on the condition and the number of years used with the similar phones from different websites. From the Table 5.1 shows the comparison of the selling price of the used mobile phones generated by intelligent pricing system and the other websites.
Table 5.1 List of Used Phone Compared with Other Websites

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of years used</th>
<th>Price computed using ANN ($)</th>
<th>Prices in other websites ($)</th>
<th>Condition</th>
<th>Difference in price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Samsung Galaxy Alpha</td>
<td>1</td>
<td>213</td>
<td>250 (swappa)</td>
<td>Ok</td>
<td>17.37</td>
</tr>
<tr>
<td>Samsung Galaxy S4</td>
<td>2</td>
<td>70</td>
<td>99.8 (ebay)</td>
<td>Good</td>
<td>42.57</td>
</tr>
<tr>
<td>Sony Xperia Z3+</td>
<td>3</td>
<td>199</td>
<td>274.99 (ebay)</td>
<td>Bad</td>
<td>38.18</td>
</tr>
<tr>
<td>Nokia Lumia 1520</td>
<td>4</td>
<td>151</td>
<td>200 (ebay)</td>
<td>Ok</td>
<td>32.45</td>
</tr>
<tr>
<td>Samsung Galaxy Note 5</td>
<td>1</td>
<td>359</td>
<td>575 (ebay)</td>
<td>Ok</td>
<td>60.16</td>
</tr>
<tr>
<td>Apple Iphone 4S</td>
<td>1</td>
<td>122</td>
<td>142 (ebay)</td>
<td>Good</td>
<td>16.39</td>
</tr>
<tr>
<td>BlackBerry Porsche Design P9983</td>
<td>1</td>
<td>783</td>
<td>999 (ebay)</td>
<td>Ok</td>
<td>27.58</td>
</tr>
<tr>
<td>Mobile</td>
<td>Rating</td>
<td>Price</td>
<td>Platform</td>
<td>Condition</td>
<td>Rating</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------</td>
<td>--------</td>
<td>-----------</td>
<td>-----------</td>
<td>--------</td>
</tr>
<tr>
<td>Nokia Lumia 1020</td>
<td>2</td>
<td>110</td>
<td>159.5 (bonanza)</td>
<td>Bad</td>
<td>45</td>
</tr>
<tr>
<td>Samsung Galaxy Note II</td>
<td>2</td>
<td>134</td>
<td>151.81 (ebay)</td>
<td>Ok</td>
<td>13.29</td>
</tr>
<tr>
<td>Nokia Lumia 830</td>
<td>3</td>
<td>141</td>
<td>160 (ebay)</td>
<td>Good</td>
<td>13.47</td>
</tr>
</tbody>
</table>

The prices in the websites such as ebay, swappa and bonanza shown in the table 5.1 are the values given by the sellers with the intension of gaining profits. ANN generates the price by considering factors of the mobile by keeping buyer in perspective. Seller can take this price as suggestion to claim the mobile for higher price (for profit).

**Reviews**

This application is given to different users to record feedback. Actual names of the reviewers have been replaced by fictitious names for the sake of anonymity. Overall rating is recorded as 3.95. Table 5.2 shows the rating of few users for the application. Since the price suggested by ANN is less, users who gave low rating have used “seller” feature and who gave higher rating have used “buyer” perspective.
Table 5.2 User Rating

<table>
<thead>
<tr>
<th>Users</th>
<th>Rating (5 max)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>4.0</td>
</tr>
<tr>
<td>Tom</td>
<td>3.5</td>
</tr>
<tr>
<td>Jennifer</td>
<td>4.1</td>
</tr>
<tr>
<td>Bob</td>
<td>4.0</td>
</tr>
<tr>
<td>Julian</td>
<td>3.8</td>
</tr>
</tbody>
</table>
6. CONCLUSION AND FUTURE WORK

Artificial neural network model is used in the development of an intelligent pricing system which is based on the historical data of its databases that can suggest a price for the seller to sell his/her used phones and can search a list of used mobiles from its database for the buyer according to the price he/she prefers. The major goal is to provide a seller with the best-selling price to sell his/her used mobiles using artificial neural network. The application suggests an intelligent pricing method for used phones using ANN. In this system the seller/buyer cannot sell/buy used phones which can be developed in future. If large dataset is used for training, the result will be accurate. The proposed system has been implemented only for used mobile phones to suggest a best-selling price, but it can be applied to many other products such as laptops, tablets, etc. An e-trading website can be developed for used phones where a seller/buyer can sell/buy used phones through it. So, in future the system can be used to train with data for different categories and put all the different categories of products together and develop a trading website which is genuine for both the buyer and seller.
7. REFERENCES


