Strojniški vestnik - Journal of Mechanical Engineering 69(2023)1-2, 61-72
DOI:10.5545/sv-jme.2022.238
Original Scientific Paper
Received for review: 2022-06-16
Received revised form: 2022-10-10
Accepted for publication: 2022-11-03

The Application of Neural Networks to Modular Arrangements of Predetermined Time Standards

Emmanuel Basitere[1] – Ilesanmi Daniyan[1,*] – Khumbulani Mpofu[1] – Adefemi Adeodu[2]

[1] Tshwane University of Technology, Department of Industrial Engineering, South Africa
[2] University of South Africa, Department of Mechanical Engineering, South Africa

Modular arrangements of predetermined time standards (MODAPTS) is an effective and efficient method to measure work and the activities associated with it. The time standard is used all over the world in different industries, but the method is old, slow, and difficult for first-time users to work with. This study applied neural networks (NN) to MODAPTS. Primary training data in the form of MODAPTS keywords were employed. The training data were acquired as raw data in the form of MODAPTS time studies. These data however was then broken and processed to extract the keywords for the training of the NN. The NN was also trained with the data collected using the TensorFlow algorithm assisted by the Keras library. This was achieved by first learning the fundamentals of creating a NN. Thereafter, consolidating several tools, such as the Python programming language and the Keras library, were used to implement the artificial neural network (ANN). The results obtained indicated that 94.7 % of successful predictions were made by the NN while only 5.3 % were manually entered codes to correct the ANN chatbot. The mean difference between the two methods is 0.25 minutes; the t-test was calculated at 95 % confidence level (0.05) and produced a P-value of 0.9663. The computed P-value was greater than 0.05, showing that there is no significant difference between the two generated studies. The MODAPTS-ANN technique demonstrated in this study possesses great potential to improve and refine work measurement.

Keywords: artificial neural network, modular arrangement of predetermined time standard, TensorFlow algorithm

Highlights

- *Implementation of the ANN using the combination of Python programming language and the Keras library.*
- *Coupling of the MODAPTS with NN to further extend the use of MODAPTS for work measurement and motion analysis.*
- *Work measurement and motion prediction using the integrated tool of MODAPTS and ANN.*

0 INTRODUCTION

Modular arrangements of predetermined time standards (MODAPTS) is a highly reliable method of conducting timekeeping, process planning, and formulating workplace efficiencies with ease and pace in application [1]. According to Cho and Park [2], MODAPTS was authored by Heyde in 1966, and its core components are still used today to define work and its activities. MODAPTS has been defined as the most popular time standard amongst its closest counterparts [2].

Conducting a MODAPTS analysis usually involves several components that would allow for a successful feasibility study; this includes but is not limited to understanding an activity by video analysis or continuous observation, analysing possible efficiencies from that observation and, most importantly, in-depth knowledge of the MODAPTS time standard and its principles to prove those efficiencies. Mastering the MODAPTS time standard can be somewhat challenging and rather complicated for the engineer, as mentioned by Mallembakam [3]. The total time it takes to conduct a MODAPTS study may be too long due to the nature of the observed activity and converting those activity tasks to MODAPTS code. This may reduce producing reputable studies and may hinder achieving set targets.

MODAPTS is regarded as one of the most popular time standards, widely used in the manufacturing, healthcare, service, and garment industries [4]. However, constructing a reputable MODAPTS time study consumes too much time. According to Mallembakam [3], it takes approximately two hours to accurately conduct, analyse, and present the findings of a designated feasibility study. Furthermore, MODAPTS requires great care and focus with regard to implementation. Therefore, a thorough understanding of the time standard is required to avoid any errors or mistakes and may prove challenging to a novice engineer [5]. The significance of this study is that the time to conduct a MODAPTS study can be significantly reduced. Furthermore, applying MODAPTS principles will be significantly easier, as the MODAPTS code will be automatically generated.

0.1 Artificial Neural Networks (ANN)

According to Graupe [6], artificial neural networks (ANN) are computational networks that attempt to stimulate, in a gross manner, the decision process in networks of nerve cells (neurons) of the biological

*Corr. Author's Address: Tshwane University of Technology, Department of Industrial Engineering, South Africa, afolabiilesanmi@yahoo.com

(human or animal) central nervous system. ANNs mimic the human brain by making use of different algorithms, which allow a computer to be trained and learn by taking new information. One way in which a NN is trained is accomplished by using supervised training. Supervised training or a multi-layered feedforward (MLF) NN is when the NN knows the desired output and the adjusting of weight coefficient is done in such a way that the outputs are as close as possible [7].

The basic processing components of a NN are called artificial neurons or nodes [8]. These nodes are layered into groups of three that construct a MLF NN structure, which include: the input layer, hidden layer(s), and the output layer [7]. The input(s) are the data that are fed into the artificial intelligence (AI) and attached to the hidden layer making use of a random initial weight [8]. The hidden layer is the sum of all weights multiplied by inputs. A bias value is then added to the final equation [8]. The bias is a value that is used to shift the result of an activation function towards the positive or negative side of an output value [8].

0.2 Natural Language Processing (NLP)

MODAPTS activities are described making use of the English language. Applying machine learning to that language requires incorporating a field called natural language processing (NLP) [9]. NLP is a subfield of AI and linguistics devoted to make computers understand statements or words written in human languages [10]. Brownlee [11] simplifies the definition by stating that they are automatic manipulations of natural language, like speech and text by software.

For any NN training to take place for which the inputs consist of text, an encoding must take place to convert the text to digits so that the NN can understand it. There are two ways to encode text in preparation for a NN: one-hot encoding and word embedding [12].

Word embeddings (WE) have been proposed to represent words as dense vectors that are derived by various training methods inspired from neural-network language modelling [13]. WE are a type of word representation that allows words with similar meanings to have a similar word representation [14]. The advantage of this technique is that it allows for words to be represented as vectors in a real vector space [13].

One-hot encoding allows each word to be represented singularly, so that no two words will have the same one hot vector representation. Additionally, a one-hot vector is much simpler and high dimensional than the word embedding technique, which is dense, low dimensional, and complex [12].

0.3 Chatbots

Chatbots are machine agents that serve as natural language user interfaces for data and service providers [15]. Chatbots agents anticipate a given input (voice, text) and attempt to give a favourable response to the intended user. A simpler explanation would be a program that simulates a natural human conversation [16].

According to Lishchynska [16], there are three types of chatbots:

1. Rule-based; this is the simplest mechanism that a chatbot can use as its structure is predefined each time an interaction takes place with a user. This type of approach is favourable for small noncomplex scenarios and often requires longer interactions to give correct and relevant answers to the user.
2. Intellectually independent chatbots make use of Machine Learning (ML) to interact with a user. This is done by the chatbot learning a set of keywords or phrases to generate a response. The chatbot becomes more precise as it learns more keywords and phrases, and through continuous use by the user over time.
3. AI-powered chatbots, make use of the best practices followed by rule-based and intellectual independent approaches in that they understand language and have predefined mechanisms to solve a problem. They make use of NLP and can understand a given context to emulate a conversation as accurate as possible.

Data must be presented in a particular manner such that the bot can easily interpret and process it to create meaningful and interactive conversations [17]. Fig. 1 shows a flow chart structure of a typical ML chatbot setup.

Five phases are involved when raw input is passed into the chatbot [17]. The first phase is called tokenisation and is the process of breaking down text into simple more manageable units [18]. This is usually the first process in any NLP task [19], and it involves breaking down raw text or sentences into individual words for the training of the NN [17].

The second phase is called text processing and is the method of cleansing the tokenized words. This includes removing any punctuation marks, digits (if not needed) and stop words (words that are meaningless to the NN) [17]. The third phase would be to categorise the purified tokens into distinct groups

of classes based on the theme of the chatbot, the method is called named entity recondition [17]. The fourth phase is called intent classification and is the process of understanding what the user wants so that an intent is clearly established; this is usually done by training a NN to classify requests into intents [17]. The final phase is developing a response based on the given intent. The response must suit the theme of the chatbot.

The recurrent neural networks (RNN) are a type of an ANN and are powerful tools for modelling sequences; there are flexibly extensible and can incorporate various types of information including temporal order [20]. RNNs have the ability to remember previous inputs from previous layers as opposed to a normal NN. RNNs are not only dependant on weights and previous outputs but they are also dependant on the context from the previously derived inputs and outputs [20]. This further echoed by Donkers et al. [20] who highlighted that they are capable of incorporating input from past consumption events to derive sequence to sequence mappings.

The context approach will be very useful specifically in remembering MODAPTS activities during the experiment. However, consistent iterations of the NN will accumulate a series of inputs, which will begin to degrade the network [12]. A special RNN technique called long-short term memory (LSTM) can be used to compensate for the memory shortfall. LSTM has complicated dynamics that allow it to easily "memorize" an extended number of time steps. Fig. 1 displays the LSTM NN [21].

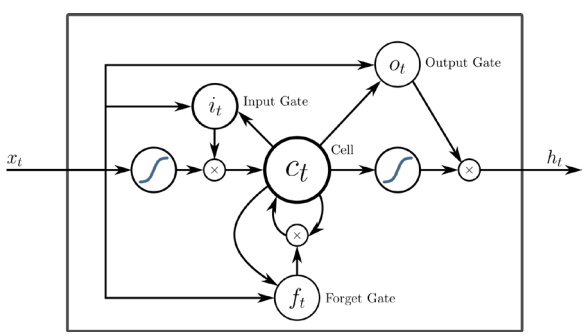

Fig. 1. *Illustration of an LSTM type of RNN* [21]

The MODAPTS is an effective method of conducting work measurement, which is due to its minimalistic coding mechanism and framework compared with other time standards such as Maynar's operations sequence technique and methods time measurement. The time standard has been applied in many industries globally and its methodology has been extended to analyse supporting factors, such as ergonomics and fine/gross motor skills.

Although the MODAPTS is versatile in application, its methodology has become dated and rather complicated, particularly when seeking to help engineers understand full systems in the present day [22]. Producing reputable time studies using MODAPTS requires a significant understanding of the many codes and rules that governs the time standard. A slight deficiency in knowledge of the time standard may lead to human error and unwanted mistakes during the observation process [3].

The time standard can be coupled with the NN to further extend its use and make it practical for an engineer to use and quickly understand for work measurement and motion analysis. This is a significant knowledge gap that has not been sufficiently explored by the existing literature. This study seeks to take advantage of the data constantly getting emitted during each MODAPTS study observation and pair that with the ML concept ANN to make the MODAPTS time standard relatable and accessible.

1 METHODOLOGY

The study employs a quantitative research approach with the overall aim of conducting an experiment to measure the total time it takes to conduct a MODAPTS time study making use of the conventional paper-based method versus the ANN approach. The focus of this methodology was extracting data from the research participants to prove that machine learning could be used to advance MODAPTS. The choice of the machine-learning approach, particularly the NN, stems from the fact that it can iteratively train the data and store information on the entire network to make predictions in a time-effective manner [23] and [24].

The testing the chatbot involved creating a Python file that would access the saved model, submit the user query, and give the prediction. Each prediction was assessed based on the intents file. If an error occurred, the intents file was then adjusted, and the model was re-trained. The chatbot was built by making use of the Python programming language, it used an ML module called Keras to train and implement the NN. The Python code was executed using an integrated development environment called PyCharm. The choice of the Python programming is due to the fact it is simple to implement, compatible, and has many libraries, such as the Keras library employed in this study, with in-built data structures [17] and [25].

The participants who took part in the research were four logistics engineers (LEs) and were selected

based on the material handling knowledge and experience that the LEs had attained. A questionnaire was given to the participants to collect information on their experience and skill level when using the MODAPTS time standard.

Data produced during a MODAPTS study is usually analysed, presented, and left as is [4]. This research method seeks to make meaningful computations out of that data, such that it becomes easier for the next engineer to implement a MODAPTS study. A large part of this research method focuses on how an engineer describes (by writing it) an observed activity. There are usually many words, lines, and sentences involved when describing an activity, and that can mean a lot of data. Introducing ANN allows for that data to be quickly organised and quantified to produce patterns and predictions [26].

ML allows for continuous improvement so that it can take a problem that has been traditionally solved and employ techniques such as genetic algorithms to improve its output [27]. This is true when looking at frameworks used to conduct MODAPTS as they have been tried and tested to produce results. The data collected from those frameworks may be used together with ML to improve the time standard. ML is vast in application and contains many fields and algorithms to solve a given problem. NLP makes it possible for the computer to learn the English language and is pivotal in teaching it MODAPTS [10].

The study used a chatbot to test interactions between the research participant and the MODAPTS NN. The chatbot was programmed to accept user input in the form of text, specifically a MODAPTS activity description. The activity descriptions were then separated into manageable smaller sentences delaminated either using a full stop or a comma.

The delaminated sentence or activity description was then fed into a trained NN to produce a prediction or response in the form of a MODAPTS code. If the prediction was successful, the code was then passed into a MODAPTS function, which converted it into the appropriate number of mods. The user would be prompted to enter the correct MODAPTS code if the prediction was incorrect, the code would also subsequently be passed into the MODAPTS function to extract the total number of modular arrangements.

1.1 Disposing of Empty Box

A material handler collects an empty box from a return chute; the material handler walks five steps to the racking location and bends below the knees to obtain the box. The material handler then walks ten steps to the waste bin and deposits the box into the waste bin. The material handler then walks back to the racking location.

Table 1 highlights the steps involved in collecting and disposing an empty box.

1.2 Line Feeding

A material handler walks five steps to the picking trolley and obtains it; the material handler then proceeds to the auto-call trolley five steps away pushing the picking trolley. The material handler then obtains two boxes (one at a time) from the auto-call trolley and places the boxes on the picking trolley beneath the knees. The material handler then takes the picking trolley to a racking location (10 steps away); he then cuts each box and position them individually to a rack position. The line feeding activities is depicted in Table 2.

1.3 Picking a Pick-to-Light (P2L)

A material handler starts by collecting a pick sheet from the printer, then checks if the sequence number on the picking sheet matches with the sequence on the display board. The material handler then takes five steps to the empty bin trolley and obtains the bin. The material handler then takes another five steps to the start of the P2L cell and positions the bin onto the picking conveyor. The material handler then proceeds to take 10 more steps while picking a total of eight

Table 1. *Disposing of an empty box*

	Collecting empties from plant 1 sequential											
1	Step to the racking location (10 steps)	5	W	5								25
2	Obtain empty box from return chute and bring to self	M	7	G	1	M	7	P	0	0	0	15
3	Step to the waste bin	10	W	5								50
4	Aside box to waste bin	M	2	P	0							2
5	Step back to racking location	10	W	5								50
											TOTAL	142

parts (pressing lights) for that zone. The material handler then returns to the printer walking 20 steps (Table 3).

1.4 Supplying P2L Bins

A material handler obtains six bins from a P2L roller conveyor and asides them onto a supply trolley. The trolley is then taken to the production line ten steps away and the material handler repacks the production trolley with the six bins. The material handler then takes the six empty bins from the return chute and places them on the supply trolley. The trolley is then taken back to the start of the P2L cell fifteen steps away (Table 4).

1.5 Sequencing a Part

A material handler walks ten steps to the printer and obtains picking list, the material handler sorts the

Table 2. *Line feeding*

	Collecting empties from plant 1 sequential												
1	Take 5 steps to the picking trolley	5	W	5									25
2	Obtain trolley	M	2	G	1								3
3	Take 5 steps to the autocall trolley	5	W	5									25
4	Obtain first box, bring to self and aside to trolley below knees	M	2	G	1	M	3	P	0	M	7	P	0	13
5	Obtain second box, bring to self and aside to trolley below knees	M	7	G	1	M	3	P	0	M	7	P	0	18
6	Obtain trolley	M	4	G	1									5
7	Take 10 steps to racking location	10	W	5										50
8	Obtain safety knife and remove from pocket	M	4	G	1	M	4	P	0					9
9	Obtain box from trolley and bring to self	M	7	G	1	M	7	P	0					15
10	Aside knife to box	M	3	P	0									3
11	Cut box on the three sides (total of 5 moves)	M	3	P	0									15
12	Obtain box flap opens it (opens all 4 sides)	M	3	P	0	M	3	P	0					24
13	Remove inner card box layer and aside box to racking location	M	3	G	1	M	3	P	0	M	3	P	2	12
14	Obtain box from trolley and bring to self	M	7	G	1	M	7	P	0					15
15	Aside knife to box	M	3	P	0									3
16	Cut box on the three sides (total of 5 moves)	M	3	P	0									15
17	Obtain box flap opens it (opens all four sides)	M	3	P	0	M	3	P	0					24
18	Remove inner cardboard box layer and aside Box to racking location	M	3	G	1	M	3	P	0	M	3	P	2	12
												TOTAL		286

Table 3. *Pick-to-light picking*

	Collecting empties from plant 1 sequential											
1	Obtain picking list from printer and bring to self	M	3	G	1	M	3	P	0			7
2	Check both digital information screen and pick sheet for sequence number	2	R	3								6
3	Take 5 steps to picking trolley with bins	5	W	5								25
4	Obtain bin and bring bin to self	M	2	G	1	M	4	P	0			7
5	Take 5 steps to conveyor at start of P2L	5	W	5								25
6	Aside bin to conveyor	M	2	P	2							4
7	Position pick sheet inside bin	M	4	P	2							6
8	Obtain and engage first light button	M	3	G	0	M	1	P	0			4
9	Wait time as system starts	T	15.4									15.4
10	Take 10 steps for the total zone	10	W	5								50
11	Obtain and engage 8 light buttons	M	3	G	0	M	1	P	0			32
12	Obtain box simo obtain part	M	4	G	1	M	4	P	0			9
13	Aside part to bin	M	4	P	0							4
14	Obtain bin and prepare to move to next light	M	3	G	1							4
15	Step back 20 steps to the printer	20	W	5								100
											TOTAL	298.4

Table 4. *Supplying pick-to-light bins*

	Collecting empties from plant 1 sequential															
1	Obtain bin from conveyor and aside onto supply trolley (×6)	M	4	G	1	M	4	P	2							66
2	Obtain trolley	M	3	G	1											24
3	Step to the production line	10	W	5												50
4	Obtain bin and pack onto roller conveyor lineside	M	2	G	1	M	4	P	2							9
5	Obtain bin and pack onto roller conveyor lineside (5 remaining picks)	M	2	G	1	M	4	P	2							45
6	Obtain and repack empty grey bins onto the supply trolley(×6)	M	4	G	1	M	4	P	2							66
7	Obtain supply trolley	M	3	G	1											4
8	Step back to the P2L cell	15	W	5												75
															TOTAL	339

Table 5. *Supplying a sequenced part*

	Collecting empties from plant 1 sequential																	
1	Walk 10 steps to the printer	10	W	5														50
2	Obtain sequence sheets from printer and bring to self	M	2	G	1	M	3	P	0									6
3	Sort them (3× sheets)	M	3	P	0	M	3	P	0	M	3	G	1	R	2			36
4	Step back to the trolley	10	W	5														50
5	Obtain tape from pocket and bring to self	M	4	G	1	M	4	P	0									9
6	Cut tape and stick on page	M	3	G	3	M	3	P	0	M	1	P	0	M	3	P	0	13
7	Aside page to trolley	M	3	P	2													5
8	Obtain scanner from pocket and scan barcode to commence picking	M	4	G	1	M	4	P	2	M	1	P	0	M	1	P	0	13
9	Check the part indicated on screen	1	R	2														2
10	Step 5 steps to part and collect it and bring to self	5	W	5														25
11	Scan part barcode	M	4	P	2	M	1	P	0	M	1	P	0					40
12	Step back to the trolley	5	W	5														25
13	Scan slot position	M	2	P	2	M	1	P	0	M	1	P	0					6
14	Aside part into scanned position	M	3	P	2													5
																	TOTAL	285

picking list until the designated list is found. The material handler steps back to the sequence trolley and obtains tape. The material handler tears off a section of the tape and sticks it to the sequence sheet and then pastes the page onto the trolley. The material handler then obtains a scanner and scans the trolley to see the first part to be picked. The material handler then walks five steps to the part and obtains it. The material handler then scans the part and walks five steps back to the trolley. The material handler then scans the trolley and deposits the part onto the slot of the trolley (Table 5).

1.6 Data Preparation for the Chatbot NN

The following code was adapted from the work created by Pykes **[25]** on his implementation of a chatbot. A "training.py" file was created to hold all subroutines needed to train the chatbot.

The first step was to obtain the "intents.js" file and cycle through all intents to get access to each pattern. The patterns were then tokenized, and each token was added to a words list followed by the pattern and tag getting added to their own list. The tokenisation process is presented as follows:

```
# This will get use the stem of the word
Lemmatizer= WordNetLemmatizer ()
Data=json.loads(open('intents.json').read())
# list to hold all of the tokenized words
words = []
# list to hold all of the tags from the json string
classes = []
# associated inputs
doc_x = []
# associated outputs
doc_y = []
for intent in data["intents"]
for pattern in internt["intents"]:
tokens=nltk.word_tokenized(pattern)
words.extend(tokens)
doc_x.append(pattern)
doc_y.append(intent["tag"])
```

The next phase was to lemmatize the tokenised words to find all inflected words for that token. The

words were then converted to lowercase for each word that did not appear on the punctuation list. The words list and tags (classes list variable) were then sorted into alphabetical order ignoring any duplicated values. The code for the lemmatizing process is presented as follows:

```
words = [lemmatizer.lemmatize(word.lower()) for word
    in words if word not in string.punctuation]
words = sorted(set(words))
classes = sorted(set(classes))
```

1.7 Training the Chatbot NN

The NN was expected to be fed numerical values to train the chatbot, hence the word list made use of the "bag of words" technique to achieve that requirement. Once the bag of words was attained then the X and Y (features and target labels) parameters needed to feed the NN were defined. Fig. 2 shows a code snippet to establish the "X" and "Y" parameter.

```
# creating the bag of words model
for idx, doc in enumerate(doc_x):
    bow = []
    text = lemmatizer.lemmatize(doc.lower())
    for word in words:
        bow.append(1) if word in text else bow.append(0)
    # mark the index of class of class that the current pattern is associated
    # to
    output_row = list(out_empty)
    output_row[classes.index(doc_y[idx])] = 1
    # add the one not encoded Bow and associated classes to training
    Training.append([bow, output_row])
# shuffle the data and convert it to an array
random.shuffle(training)
training = np.array(training, dtype=object)
# split the features and target labels
train_x = np.array(list(training[:, 0]))
train_y = np.array(list(training[:, 1]))
```

Fig. 2. *Code snippet of the features and targets generation*

The NN was structured to predict a given response based on the associated tag when feeding features as inputs [25]. The NN model was a sequential model with three primary layers (input-hidden-output) and two dropout layers that randomly adjusted the inputs to 0. The input layer was dense and contained 128 neurons and used an activation function called the rectified linear unifier (ReLU). The next layer was a hidden dense layer with 64 neurons also using the ReLU as its activation function. The last layer was a dense output layer using the SoftMax as its activation function.

The model used an Adam optimiser with a learning rate of 1 % and was trained for 200 cycles (epochs). Fig. 3 shows the dense NN structure of the chatbot.

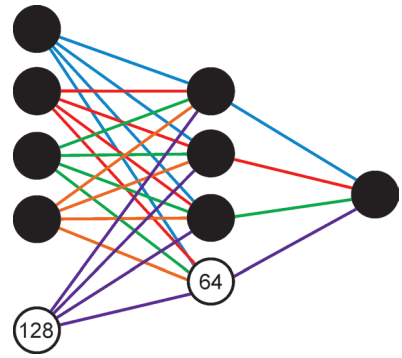

Fig. 3. *Dense structure of the NN*

The model was trained and then saved under the name "chatbot2.h5". Fig. 4 shows a code snippet of the training process.

```
input_shape = (len(train_X[0]),)
output_shape = len(train_y[0])
epochs = 200
# the deep learning model
model = Sequential()
model.add(Dense(128, input_shape=input_shape, activation="relu"))
model.add(Dropout(0.5))
model.add(Dense(64, activation="relu"))
model.add(Dropout(0.3))
model.add(Dense(output_shape, activation="softmax"))
adam = tf.keras.optimizers.Adam(learning_rate=0.01, decay=1e-6)

model.compile(loss='categorical_crossentropy',
              optimizer=adam,
              metrics=["accuracy"])

hist = model.fit(x=train_X, y=train_y, epochs=200, verbose=1)
model.save('chatbot2.h5', hist)
print("Done")
```

Fig. 4. *Code snippet of the NN chatbot training phase*

1.8 Chatbot Application

The chatbot application was a simple webpage that captured user input in the form of MODAPTS activities. An initial signup or login screen was needed to gain access to user data and select the scenario. The selected scenario was represented by a button on the main screen interface. Pressing the scenario button would bring up a dialog box containing scenario data. This action could be accessed anytime during the experiment for scenario reference. The main interface also consisted of a prediction button and an input field that was used by the research participant to submit the query to the chatbot NN. A single input field was used to key in user data and was required to be entered each time the prediction button was pressed. The application would present a dialog box with the predicted MODAPTS code when the prediction button was pressed. The user would then select the code if

the prediction was true or manually input the correct code if the chatbot NN presented a false prediction. Each prediction was summarized at the main screen interface shown as a scrollable list. Each list item showed the MODAPTS code, activity description, MODS for that activity, and the total duration taken for that list item. Each MOD from the list items were collectively added together and shown on the main screen and changes each time an activity was added or removed from the list.

The application made use of several timestamps to track movement done by the user and the performance of the chatbot during the experiment.

The chatbot NN was programmed to expect a structured user query from the research participant. The activity was done first by describing a terminal action, then followed by the movement class and then finally the level of difficulty to perform the action.

2 RESULTS AND DISCUSSION

This section discusses the research results and analysis of the quantitative data obtained from the four research participants. The section is divided into two parts, the first part interpreted results from the questionnaire in relation to all participants while the second part focuses on the results and analysis obtained when conducting a traditional MODAPTS study.

The third part focused on the results and analysis of the Machine Learning approach of conducting the same MODAPTS study. The last part compared the two approaches in relation to the final duration with aid of a t-test analysis.

2.1 Questionnaire Results

The following tables show results obtained from the questionnaire, a total of seven questions were presented to the research participants to understand their work experience and their role in using MODAPTS.

Table 6 shows the current industry in which the research participants are working.

Table 6. *Work industry*

Industries	Score
Automotive	4
Energy, utilities, and mining	0
Government and public sector	0
Technology	0
Other	0

The research participants all work currently for the automotive industry. Their focus is on material handling activities affecting people, the production line, and their customer overall.

Table 7 describes the work experience in years each research participant possesses in the selected work environment.

Table 7. *Research MODAPTS experience*

Experience	Score
Less than 3 years	1
Less than 6 years	2
Greater than 6 years	1

There was only one research participant with less than three years of experience using MODAPTS, the remaining participants were highly qualified with more than three years of experience.

Table 8 describes the work measurement techniques that each participant used at the workplace.

Table 8. *Work measurement techniques*

Work measurement	Score
General estimation	1
Stopwatch analysis	2
Predetermined motion time systems	4

Each research participant made use of predetermined motion time systems (PMTS) to conduct daily time studies. Half the research participants made use of stopwatch analysis to become familiar with the process before applying PMTS. Only one research participant made use of general estimation but that was done as a forethought with regards to planning an activity.

Table 9 shows certification status of each research participant.

Table 9. *Work measurement certification*

Certification	Score
Yes	3
No	1

Three out of the four research participants had been trained and certified to use MODAPTS. The remaining research participant had yet to be trained and learned from stored MODAPTS company user manuals.

Table 10 shows the total duration taken to conduct a MODAPTS time study.

There was a total of two people who conducted MODAPTS studies in four hours; the remaining

participants took a day or more to complete the studies. The extended length in conducting a MODAPTS study was due to the need for accuracy rather than inexperience.

Table 10. *Duration of study*

Duration of time study	Score
2 hours	0
4 hours	2
1 day	1
more than 1 day	1

Table 11 shows the uses of the MODAPTS time standard employed.

Table 11. *MODAPTS study usage*

Time study function	Score
Efficiencies	4
Rebalancing of work	4
Time keeping	2
Other	0

Each research participant uses MODAPTS for efficiencies and to rebalance workload as part of customer requirements and managing resources. Only two research participants used MODAPTS for time keeping.

Table 12 shows for whom the MODAPTS time study is intended.

Table 12. *MODAPTS study designation*

Time study designation	Score
Top management	2
Middle management	4
Shop floor staff	0

Table 13. MODAPTS presentation format

Time study presentation	Score
Raw Data	2
Graphical	3
Key performance indicators	1
Other	1

The majority of the research participants chose middle management to present their findings. Presenting to this level was due to middle management needing to prove validity of resources to the customer or top management. There were only two people that selected the top management. Three out of the four research participants selected graphical presentation as it was easier and quicker to convey the intended message. Raw data were also used, mostly to show variance in a process activity element by element. Table 13 shows the presentation format used by the research participants.

2.2 MODAPTS Chatbot Analytics

A series of measurements were collected from the research participants while using the chatbot. These measurements were used to establish interaction between the research participant and the NN chatbot. Each measurement was captured and sent to a unique database table on the created server. The measurements included: prediction accuracy, progressive rate, activity formulation, error accumulation and total duration.

2.3 Prediction Accuracy

Prediction accuracy measures the performance of the trained NN application, focusing on the number of successful predictions against the total number of incorrect predictions. The application does this by keeping a count of keypresses each time the prediction and manual buttons are pressed.

Table 14. *Predictions table*

Scenario	Manual	Predictions	Total
SC1	1	18	19
SC2		52	52
SC3	4	50	54
SC4		84	84
SC5	9	44	53
Total	14	248	262

Source: Researcher's synthesis

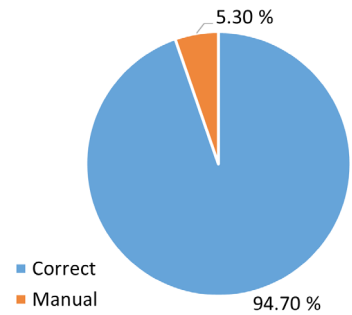

Fig. 5. Prediction accuracy

A total of 262 activities were entered by all four research participants; 248 (94.7 %) were successful predictions made by the NN and only 14 (5.3 %) were manually entered codes to correct the ANN chatbot. Looking at the manual entries, there were a

total of 9 MODAPTS related errors produced by the chatbot relating to the codes "E", "R", and the "finger-movement". The remaining manual entries were due to a dialect miscommunication between the research participant and the chatbot. Table 14 shows the total number of predictions against manual predictions.

Fig. 5 shows the percentage difference between total manual predictions against the correct predictions made by the ANN chatbot.

2.4 Progressive Rate

The progressive rate is a measure that calculates the response time of each research participant moving from one element (buttons, inputs, etc.) to another in the NN application. The measure was applied to establish how quickly a research participant used and understood the NN application.

Table 15 shows the average time (in seconds) that the research participants took to move from one element in the NN application to the next throughout each scenario.

Table 15. *Application element times [s]*

Elements	SC1	SC2	SC3	SC4	SC5	Average
Inactive - Scenario Button Open	6.06	2.88	3.32	3.74	3.66	3.75
Prediction Button Close - Main Input	4.88	4.85	4.81	3.92	6.33	5.02
Prediction Button Open - Prediction Button Confirm	4.22	3.23	2.25	1.81	2.13	2.42
Average	5.07	3.33	2.95	2.34	3.32	

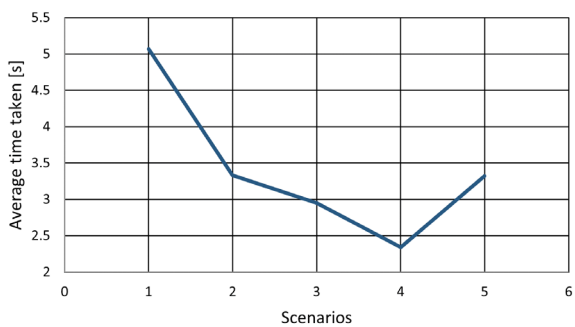

Fig. 6. *Progressive rate trendline*

Scenario 1 (SC1) showed that the research participants spent more time navigating around the template on average at 5.07 s. This was expected as the template was new to each research participant. The time dropped significantly through each proceeding scenario at average of 2.99 s (SC2 to SC5). A percentage decrease was calculated and found a drop rate to be 41.02 % from the initial scenario time. Fig. 6 shows the trendline with regards of the progressive rate.

2.5 Activity Formulation

Activity formulation is a measure that keeps track of the number of entered activities and the time spent observing the scenario description by the research participant. The measure correlates the time spent viewing a scenario description with the number of produced activities by the ANN template from the research participant. Table 16 shows the duration and corresponding activities of each scenario.

A total of 262 activities was entered collectively by all research participants. A total duration of 370 minutes was spent viewing the activity scenario descriptions by all research participants. The number of entered activities directly impacted the total time a scenario activity was viewed.

Table 16. *Scenario description viewing duration*

Elements	SC1	SC2	SC3	SC4	SC5	TTE
Duration	54	85	80	58	93	370
Activities	19	52	54	84	53	262

2.6 Paired T-Test – ANN against Traditional MODAPTS

The following analysis tests for a statistical difference when conducting a MODAPTS study traditionally against the ANN Chatbot alternative. Participant 4 was removed from the final analysis as their final time was an outlier toward the rest of the research participants. The analysis makes use of a paired t-test analysing the total durations of each scenario from the two groups, as shown in Table 17.

Table 17. *Error entities by scenario*

Scenario	Traditional	ANN
SC1	13.30	19
SC2	37.65	27
SC3	30.90	23
SC4	18.82	38
SC5	39.60	32
Mean	28.05	27.80

The following analysis made use of Excel's t-test function to compute the P-value.

The t-test was paired so that the two groups were compared looking at five scenarios, with no reference

from each other. The mean difference between the two methods is 0.25 minutes, the t-test was calculated at 95 % confidence level (0.05) and produced a P-value of 0.9663. The computed P-value was greater than the 0.05 showing that no significant difference between the two generated studies.

3 CONCLUSION AND RECOMMENDATIONS

The objective of the study was to apply NN to MODAPS. This was achieved by gathering training data in the form of MODAPTS keywords. Training data were acquired as raw data in the form of MODAPTS time studies. This data, however, were then broken and processed to extract the keywords for the training of the NN. The NN was also trained with the data collected using the TensorFlow algorithm assisted by the Keras library. This was achieved by first learning the fundamentals of creating a NN. Thereafter, consolidating several tools, such as the Python programming language and the Keras library, were used to implement the ANN. The approach of this ANN made use of a chatbot application, such that the collected keywords would be trained in the NN and act as dialect for the chatbot when in conversation with the research participant.

After conducting a successful study, the following could be concluded:

1. 94.7 % successful predictions were made by the NN while only 5.3 % were manually entered codes to correct the ANN chatbot. The mean difference between the two methods is 0.25 minutes, the t-test was calculated at 95 % confidence level (0.05) and produced a P-value of 0.9663. The computed P-value was greater than the 0.05 showing that there is no significant difference between the two generated studies.
2. The ANN approach could replace tedious tasks of manually conducting MODAPTS using spreadsheets and other laborious methods.
3. Although some degree of MODAPTS knowledge is required, the time for a novice engineer to learn and understand the time standard would have greatly reduced.
4. The chatbot is able keep track of user events, such that more patterns besides the ones mentioned in this research could be developed for the continual benefit of the engineer and study.

The results show that the MODAPTS enabled by machine learning applications such as the NN can replace the tedious MODAPTS analysis on the computer with a mobile alternative that consistently records the user and study analytics to improve the overall analysis. The MODAPTS-ANN template demonstrated in this study possesses great potential to improve and refine work measurement. Furthermore, the template is flexible enough to be moulded into a tool that all engineers can adapt to their different working environments, not only the automotive industry.

It can make the data sharing amongst engineers easier owing to its descriptive database. Hence, there is a potential to add other ML learning techniques to time study methods. The novelty of this study lies in the fact that the coupling of the MODAPTS with NN to further extend the use of MODAPTS and make it practical for an engineer to use and understand it quickly for work measurement and motion analysis has not been sufficiently highlighted by the existing literature. The ANN approach makes it far easier to implement MODAPTS in multiple industries owing to the customisable keywords and dialect recognition of the chatbot.

The study is limited to the MODAPTS time standard in relation to material handling activities conducted by the logistics company, DSV. The process activities were sourced from actual feasibility studies conducted by logistics engineers carried out throughout each study observation with a focus falling on the keywords. The study also explored the use of machine learning in the form of NNs with a view to applying them to chatbots. This included its functionality regarding the building blocks of a simple NN structure, the activity of training the NN through these collected process keywords, and the delivery of the template. This study did not build a NN from scratch but rather to make use of an open-source library called Keras to assist with all NN-training activities. Hence, the combination of the NN and MODAPTS is recommended for conducting work measurement and motion analysis in the industry.

The newly proposed approach was compared with the traditional approach and the mean difference between the two methods was 0.25 minutes. This implies that this approach still has room for improvement. Therefore, future work can consider improving the vocabulary of the chatbot; this is to allow for a more natural flow of communication when conversating with the chatbot. This would require going deeper in research field of NLP to understand and to gain more tools to make this possible.

4 REFERENCES

[1] Golpîra, H. (2013). Estimating duration of projects manual tasks using MODAPTS plus method. *International Journal of Research in Industrial Engineering*, vol. 2, vol. 1, p. 12-19.

[2] Cho, H., Park, J. (2014). Motion-based method for estimating time required to attach self-adhesive insulators. *Computer-Aided Design*, vol. 56, p. 68-87, **DOI:10.1016/j.cad.2014.06.004**.

[3] Mallembakam, V.R. (2020). Incorporating modular arrangement of predetermined time standard with a wearable sensing glove. *Electronic Theses and Dissertations*, 8524, University of Windsor, Windsor.

[4] Kumar, R., Charak, A., Thakur, G. (2020). Productivity improvement of an automotive assembly line using modular arrangement of predetermined time standards (MODAPTS). *i-Manager's Journal on Future Engineering and Technology*, vol. 16, no. 2, p. 32-42, **DOI:10.26634/jfet.16.2.17694**.

[5] Cho, H., Lee, S., Park, J. (2014). Time estimation method for manual assembly using MODAPTS technique in the product design stage. *International Journal of Production Research*, vol. 52, no. 12, p. 3595-3613, **DOI:10.1080/00207543.2013.878480**.

[6] Graupe, D. (2013). *Principles of Artificial Neural Networks*, University of Illinois, Chicago, World Scientific, **DOI:10.1142/8868**.

[7] Svozil, D., Kvasnicka, V., Pospichal, J.Í. (1997). Introduction to multi-layer feed-forward neural networks. *Chemometrics and Intelligent Laboratory Systems*, vol. 39, no. 1, p. 43-62. **DOI:10.1016/S0169-7439(97)00061-0**.

[8] Abraham, A. (2005). *Artificial Neural Networks. Handbook of Measuring System Design*. John Wiley & Sons, London, **DOI:10.1002/0471497398.mm421**.

[9] Landset, S., Khoshgoftaar, T.M., Richter, A.N., Hasanin, T. (2015). A survey of open source tools for machine learning with big data in the Hadoop ecosystem. *Journal of Big Data*, vol. 2, no. 24, **DOI:10.1186/s40537-015-0032-1**.

[10] Chopra, A., Prashar, A., Sain, C. (2013). Natural language processing. *International Journal of Technology Enhancements and Emerging Engineering Research*, vol. 1, p. 131-134.

[11] Brownlee, J. (2017). Deep learning for natural language processing: develop deep learning models for your natural language problems. *Machine Learning Mastery*. Edition v1.1, p. 1-414.

[12] Nelson, D. (2021). Text Generation with Python and Tensor/Flow. Available from: *https://stackabuse.com/text-generation-with-python-and-tensorflow-keras*, accessed on 2021-07-30.

[13] Levy, O., Goldberg, Y. (2014). Dependency-based word embeddings. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, vol. 2: Short Papers, p. 302-308, **DOI:10.3115/v1/P14-2050**.

[14] Agrawal, S. (2019). What the heck is word embedding, from: *https://towardsdatascience.com/what-the-heck-is-word-embedding-b30f67f01c81*, accessed on 2020-08-01.

[15] Brandtzaeg, P.B., Følstad, A. (2017). Why people use chatbots. Internet Science. *Lecture Notes in Computer Science*, vol. 10673, p. 377-392, Springer, Cham, **DOI:10.1007/978-3-319-70284-1_30**.

[16] Lishchynska, D. (2017). What are bots, how do chat bots work? from: *https://botscrew.com/blog/what-are-bots/#:~:text=Chatbot%20or%20bot%20%E2%80%93%20is%20a,an%20instant%20pre%2Dset%20answer*, accessed on 2021-02-01.

[17] Manaswi, N.K. (2018). Developing chatbots. Deep Learning with Applications Using Python. *APress*, Berkeley, **DOI:10.1007/978-1-4842-3516-4_11**.

[18] Reese, R.M., Bhatia, A. (2018). *Natural Language Processing with Java: Techniques For Building Machine Learning and Neural Network Models for NLP*, Packt Publishing Ltd., Birmingham.

[19] Park, K., Lee, J., Jang, S., Jung, D. (2020). An empirical study of tokenization strategies for various Korean NLP tasks. *arXiv* preprint arXiv:2010.02534, **DOI:10.48550/arXiv.2010.02534**.

[20] Donkers, T., Loepp, B., Ziegler, J. (2017). Sequential user-based recurrent neural network recommendations. *Proceedings of the 11th ACM Conference on Recommender Systems*, p. 152-160, **DOI:10.1145/3109859.3109877**.

[21] Zaremba, W., Sutskever, I., Vinyals, O. (2014). Recurrent neural network regularization. *arXiv*, preprint arXiv:1409.2329, from *http://arxiv.org/abs/1409.2329*.

[22] Wu, S., Wang, Y., Bolabola, J.Z., Qin, H., Ding, W., Wen, W., Niu, J. (2016). Incorporating motion analysis technology into modular arrangement of predetermined time standard (MODAPTS). *International Journal of Industrial Ergonomics*, vol. 53, p. 291-298, **DOI:10.1016/j.ergon.2016.03.001**.

[23] Daniyan, I.A., Tlhabadira, I., Mpofu, K., Adeodu, A.O. (2020). Development of numerical models for the prediction of temperature and surface roughness during the machining operation of titanium alloy (Ti6Al4V). *Acta Polytechnica Journal*, vol. 60, no. 5, p. 369-390, **DOI:10.14311/AP.2020.60.0369**.

[24] Daniyan, I.A., Mpofu, K., Tlhabadira, I. Ramatsetse, B.I. (2021). Process design for milling operation of titanium alloy (Ti6Al4V) using artificial neural network. *International Journal of Mechanical Engineering and Robotics Research*, vol. 10, no. 11, p. 601-611, **DOI:10.18178/ijmerr.10.11.601-611**.

[25] Pykes, K. (2021). Build A Simple Chatbot In Python with Deep Learning. available from: *https://towardsdatascience.com/a-simple-chatbot-in-python-with-deep-learning-3e8669997758*, accessed on 2022-03-31].

[26] Alpaydin, E. (2021). *Machine Learning*, MIT Press, Massachusetts, **DOI:10.7551/mitpress/13811.001.0001**.

[27] Aissani, N., Beldjilali, B., Trentesaux, D. (2008). Use of machine learning for continuous improvement of the real time heterarchical manufacturing control system performances. *International Journal of Industrial and Systems Engineering*, vol. 3, no. 4, p. 474-497, **DOI:10.1504/IJISE.2008.017555**.