Software Effort Estimation Using Multilayer Perceptron and Long Short Term Memory

Eduard-Florin PREDESCU, Alexandru ȘTEFAN, Alexis-Valentin ZAHARIA
Bucharest University of Economic Studies, Romania
eduard.florin.predescu@gmail.com, alexstefan610@yahoo.ro, alexis.zaharia@gmail.com

Software effort estimation is a hot topic for study in the last decades. The biggest challenge for project managers is to meet their goals within the given time limit. Machine learning software can take project management software to a whole new level. The objective of this paper is to show the applicability of using neural network algorithms in software effort estimation for project management. To prove the concept we are using two machine learning algorithms: Multilayer Perceptron (MLP) and Long Short-Term Memory (LSTM). To train and test these machine learning algorithms we are using the Desharnais dataset. The dataset consists of 77 sample projects. From our results we have seen that Multilayer Perceptron algorithm has better performance than Long Short-Term Memory, by having a better determination coefficient for software effort estimation. Our success in implementing a machine learning that can estimate the software effort brings real benefits in the field of project management assisted by computer, further enhancing the ability of a manager to organize the tasks within the time limit of the project. Although, we need to take into consideration that we had a limited dataset that we could use so a real advancement would be to implement and test these algorithms using a real life company as a subject of testing.

Keywords: *software effort estimation, multilayer perceptron, long short-term memory, neural network algorithms, machine learning, Desharnais dataset*

1 Introduction

In this paper our aim is to prove the effectiveness of machine learning in the planning and execution of a vast amount of projects. Machine learning, is a process of intense data analysis that concludes with the automation of analytical model building. It is a branch derived from artificial intelligence which consists of the fact that systems can learn from analyzed data, recognize patterns and make calculated decisions with minimal or no human interaction needed.

The challenge that this paper undertakes is to indicate which of the two machine learning algorithms studied MLP or LSTM is more efficient in the field of the machine learning in project management. The direct result of the aforementioned algorithms is to simplify the tasks of managers, to produce more precise predictions and finally and perhaps most importantly to increase sales and efficiency within the target company.

The primary challenge that most project managers face is the achieving of project goals within some different given constraints. Typically, this include but are not limited to, estimating a precise time frame in which the project will be completed which can be one of the trickiest parts of the job. Machine learning algorithms offer a simple solution to that conundrum as they can make project management simpler and more efficient. Different machine learning frameworks offer different benefits based on the particular field that they are being developed for.

Some of the advantages of machine learning are as follows:

- Predict and assign tasks to team members based on their skills;
- Predict and determine when deadlines aren’t going to be met and what can be done to prevent it;
- Correct task time estimates as a means to update the project information and notify the manager that budget and time frames may require extension.

If implemented correctly and trained accordingly a machine learning software can reach a level of automation in which it automatically

DOI: 10.12948/issn14531305/23.2.2019.07

tracks and detects different problems and delays caused by various unexpected factors. It can also automatically detect delays within a project and act according to what the situation demands. This way the industry would see a total shift in the way projects are run and executed and this should lead to a positive impact on overall team performance and efficiency.

Based on the evolutionary trajectory of machine learning algorithms in project management they will fundamentally change the way we as people think about the running and execution of projects, the challenges that arise on the way, in the form of technology and software limitations is something that must be considered very thoroughly. The implications of undertaking the implementation of machine learning algorithms in a large-scale legacy project can prove to be and insurmountable obstacle if we are to consider the following problems:

Hardware and software expenses, the amount of money required to be invested in both software development and hardware to train a software product can reach a staggering amount, due to the fact that machine learning algorithms are developed and trained with a particular objective in mind and quality of the required hardware components. While not necessarily a barring problem for big corporations with huge profit numbers and ability to invest in such a technology, it is still a time consuming operation which if not done correctly can lead to serious problems down the road.

We aim to achieve this goal by implementing and testing two machine-learning algorithms on the Desharnais dataset: Multi Layered Perceptron (MLP) and Long Short Term Memory (LSTM). In the following sections we will cover the aforementioned machine learning algorithms:

The sections that will follow this part of our study aim to provide and analyze real, tangible results. This will be realized though the construction of a machine learning system that will take as input data from a variety of real life projects, and will return the best possible result aimed at improving the planning and execution of similar projects. In order for this to be realized and correctly implemented various objectives must always be considered and treated accordingly, objectives such as:

- Analyzing and identifying how workforce effort and performance is estimated and carried out in large-scale distributed projects;
- Analyzing the accuracy of the aforementioned effort and performance estimation processes in large-scale distributed projects;
- Identifying and investigating the major factors that impact the accuracy of effort and performance estimates in large-scale distributed projects.

2 Literature review

Machine Learning is an application of artificial intelligence that is revolutionizing the way companies can do business. ML algorithms allow programs to analyze big data to increase the prediction power of the business which in return increase the efficiency of the business and its sales.

The biggest challenge for project managers is to meet the goals of their projects within the given time limit. Normally, to predict the time frame of the project can be the hardest part of the job. Machine Learning can take project management software to a whole new level. Many companies are currently testing Machine Learning to predict which actions to take.

There are already efforts of researching the effectiveness of ML algorithms for project management and our search has led us to a few publications [2],[3],[4].

A case study analysed by Usman et al. on a large-scale project concerned with the development and maintenance of a telecommunication software product at Ericsson shows that although traditional estimation methods may deliver reliable and accurate results, underestimation is a frequent trend at both quotation and analysis stages of a software project estimation [5].

Therefore, we can affirm that traditional estimation techniques may be inefficient and impractical as outlined by Pospieszny et al. due to the fact that, often, pressure from clients or

DOI: 10.12948/issn14531305/23.2.2019.07

management or limited knowledge about the domain or risks involved may lead to overoptimistic estimates, thus impacting severely project outcomes [3].

The same Pospieszny et al. suggests that with smart data preparation and implementation of ML algorithms, we can obtain very accurate results compared to other approaches and are suitable for deployment in practice, but for the best results, it is recommended that homogeneous data must be used for greater prediction accuracy [3].

Another context that needs to be addressed when implementing machine learning algorithms is data, specifically data preprocessing being a fundamental stage of a machine learning method that has a large impact on the accuracy of the predictions as presented by Huang et al. [1].

Furthermore, Panda et al. and Rijwani et al. demonstrate that machine learning is a viable approach in software effort estimation, but it is highly dependent on the quality of the input dataset and the types of algorithms used for the actual predictions [2][5].

Outlined below, table 1 presents the approaches undertaken by the researchers in the articles chosen as references for this paper to validate machine learning as a viable software effort estimation technique.

Table 1. Short presentation of literature review

Author	Title	Subject	Argument	Conclusion
Pospieszny et al., 2018	An effective approach for software project effort and duration estimation with machine learning algorithms	Tackles the limitations and narrows the gap between up to date research in machine learning and potential deployment of machine learning algorithms in practice for project effort estimation	Usage of the ISBSG dataset for modelling with three machine learning algorithms for cross-validation validates the practicality of ML algorithms in real world project management	Traditional and parametric estimation techniques may be inefficient and impractical. Proposed ML algorithms are suitable for estimating duration and effort.
Huang et al., 2015	An empirical analysis of data preprocessing for machine learning-based software cost estimation	Studies the effectiveness of data preprocessing techniques on machine learning methods using empirical study to validate the techniques	Data preprocessing of multiple datasets (including ISBSG and Desharnais) has a large impact on the accuracy of machine learning algorithms	More extensive experiments on various datasets are to be considered to generalise the findings of this study
Usman et al., 2018	Effort estimation in large-scale software development: An industrial case	Usage of an two-stage estimation process using traditional methods (to be used as a measure for	Practitioners have produced inaccurate and unreliable effort estimation resulting in budget	A two-stage estimation process can improve project estimation accuracy in large scale agile

DOI: 10.12948/issn14531305/23.2.2019.07

	study	comparison with ML algorithms)	and schedule overruns	development
Panda et al., 2015	Empirical Validation of Neural Network Models for Agile Software Effort Estimation based on Story Points	Usage of multiple types of neural networks to enhance the prediction accuracy of agile software effort estimation using the story point approach (SPA)	Predictability is the primary goal of project management, the study aims to provide a reliable way of doing such predictions	The results are empirically validated, the cascade network has outperformed other network types in SPA prediction
Rijwani & Jain, 2016	Enhanced Software Effort Estimation using Multi Layered Feed Forward Artificial Neural Network Technique	Usage of an artificial neural network to estimate effort in software	Estimating software effort is a crucial responsibility in project management, the industry being challenging in case of effective effort estimation	An artificial neural network is advantageous because it allows any number of relevant inputs to be added to predict the required software effort

Our research of these publications has shown that artificial neural network algorithms are advantageous for software effort estimation. Having that in mind we are proposing the use of artificial neural network algorithms for assistance in project management, our chosen algorithms being Multilayer Perceptrons (MLP) and Long Short-Term Memory (LSTM).

3 Research methodology

3.1 Data

In this study, the Desharnais dataset is used to train and test the system. The dataset consists of 77 sample projects. Through this data we can not only train our machine learning algorithm, but also help prove the real life applications of such a software and the edge that it can provide in the context in which it is applied.

The data is divided by columns, such it goes as follows:

- Project number – representing the number of the project;
- Team Experience – representing the mean of the development's team experience is the respective field, measured in years;
- Manager Experience – representing the amount of experience the manager leading the project possess, measured in years;
- Year end – representing the last two digits of the year corresponding to the date the project was finished;
- Length – representing the number of months during which the project was in development;
- Effort – representing the cumulated number of hours that each person has put in the respective project;
- Transactions – representing the number of basic logical transactions made within the system;
- Entities – representing the number of entities in the system's data model;
- Points Non Adjust – representing the size of the project measured in adjusted function points;
- Adjustment - representing the size of the project measured in adjustment points;

- Points Adjust – representing the size of the project measured in unadjusted points;
- Language – representing the programming language in which the project was written.

Before implementing the algorithms, we must analyze the data to get an overview of the quality of the dataset and if it is homogeneous enough so that the algorithms implemented in this paper will produce sufficiently accurate results. The source of the dataset is Kaggle and the analysis was carried out in Microsoft Office Excel [12].

3.2 Descriptive statistics

As outlined in the tables 2 through 11, the standard deviation for all measures is large meaning that there are some outliers in the data set or the latter may have to be clustered into classes for better accuracy of the models.

Table 2. Team experience descriptive statistics

Parameter name	Value	Conclusion
Mean	2.185185185	
Std Dev	1.415195314	Due to the fact that the standard deviation is greater than half the mean we determine the fact that it is not representative;
Skewness	-0.17421324	Due to the fact that the skewness is negative we can conclude that this is an asymmetrical distribution with a long tail to the left;
Kurtosis	-0.971174567	As the kurtosis is negative and by extension lower than 3 so that makes it Platykurtic;

Table 3. Manager experience descriptive statistics

Parameter name	Value	Conclusion
Mean	2.530864198	
Std Dev	1.643824958	Due to the fact that the standard deviation is greater than half the mean we determine the fact that it is not representative;
Skewness	-0.021332638	Due to the fact that the skewness is negative we can conclude that this is an asymmetrical distribution with a long tail to the left;
Kurtosis	0.055703646	As the kurtosis is negative and by extension lower than 3 so that makes it Platykurtic;

Table 4. Project length descriptive statistics

Parameter name	Value	Conclusion
Mean	11.66666667	
Std Dev	7.424621202	Due to the fact that the standard deviation is greater than half the mean we determine the fact that it is not representative;
Skewness	1.598916732	Due to the fact that the skewness is positive we can conclude that this is an asymmetrical distribution with a long tail to the right;
Kurtosis	3.081156302	As that the kurtosis is positive and greater than 3 so that makes it Leptokurtic;

DOI: 10.12948/issn14531305/23.2.2019.07

Table 5. Effort descriptive statistics

Parameter name	Value	Conclusion
Mean	5046.308642	
Std Dev	4418.767228	Due to the fact that the standard deviation is greater than half the mean we determine the fact that it is not representative;
Skewness	2.006536656	Due to the fact that the skewness is positive we can conclude that this is an asymmetrical distribution with a long tail to the right;
Kurtosis	4.715274686	As the kurtosis is positive and greater than 3 so that makes it Leptokurtic;

Table 6. Transactions descriptive statistics

Parameter name	Value	Conclusion
Mean	182.1234568	
Std Dev	144.0350984	Due to the fact that the standard deviation is greater than half the mean we determine the fact that it is not representative;
Skewness	2.286437001	Due to the fact that the skewness is positive we can conclude that this is an asymmetrical distribution with a long tail to the right;
Kurtosis	7.376622924	As the kurtosis is positive and greater than 3 so that makes it Leptokurtic;

Table 7. Entities descriptive statistics

Parameter name	Value	Conclusion
Mean	122.3333333	
Std Dev	84.88212415	Due to the fact that the standard deviation is greater than half the mean we determine the fact that it is not representative;
Skewness	1.340890535	Due to the fact that the skewness is positive we can conclude that this is an asymmetrical distribution with a long tail to the right;
Kurtosis	1.475513847	As the kurtosis is positive and lower than 3 so that makes it Platykurtic;

Table 8. Points non adjusted descriptive statistics

Parameter name	Value	Conclusion
Mean	304.4567901	
Std Dev	180.2101585	Due to the fact that the standard deviation is greater than half the mean we determine the fact that it is not representative;
Skewness	1.729235265	Due to the fact that the skewness is positive we can conclude that this is an asymmetrical distribution with a long tail to the right;
Kurtosis	4.757254874	As the kurtosis is positive and greater than 3 so that makes it Leptokurtic;

Table 9. Adjustment descriptive statistics

Parameter name	Value	Conclusion
Mean	27.62962963	
Std Dev	10.59179452	The standard deviation is low, so that means the observations are close concentrated around the mean;
Skewness	-0.110489192	Due to the fact that the skewness is positive we can conclude that this is an asymmetrical distribution with a long tail to the right;
Kurtosis	-0.283934607	As the kurtosis is positive and greater than 3 so that makes it Leptokurtic;

Table 10. Points adjust descriptive statistics

Parameter name	Value	Conclusion
Mean	289.2345679	
Std Dev	185.7610879	Due to the fact that the standard deviation is greater than half the mean we determine the fact that it is not representative;
Skewness	1.617378261	Due to the fact that the skewness is positive we can conclude that this is an asymmetrical distribution with a long tail to the right;
Kurtosis	3.98399961	As the kurtosis is positive and greater than 3 so that makes it Leptokurtic;

Table 11. Language descriptive statistics

Parameter name	Value	Conclusion
Mean	1.555555556	
Std Dev	0.707106781	The standard deviation is low, so that means the observations are close concentrated around the mean;
Skewness	0.886121156	Due to the fact that the skewness is positive we can conclude that this is an asymmetrical distribution with a long tail to the right;
Kurtosis	-0.481012658	As the kurtosis is negative and by extension lower than 3 so that makes it Platykurtic;

This fact is proved further by the high values of the skewness/kurtosis for each measure in the data set. We will further analyse this hypothesis after outputting the results from both machine learning algorithms.

3.3 Performance measures

For evaluating the performance of the model, as an accuracy criterion, we used the following metrics:

- Mean squared error (MSE) - measures the average squared difference between the estimated values and what is estimated;
- Mean absolute error (MAE) - measures the average magnitude of the errors in a set of predictions, without considering their direction;
- R^2 score (R2) - measures the proportion of the variance in the dependent variable that is predictable from the independent variable(s).

Additionally, we calculated the Magnitude of Relative Error (MRE) for each predicted value as follows:

$$MRE_i = \frac{|Actual\ Effort_i - Predicted\ Effort_i|}{Actual\ Effort_i},$$

where *i* is a project in the dataset.

3.4 Machine learning algorithms used

As mentioned in the introduction phase our

DOI: 10.12948/issn14531305/23.2.2019.07

study is focused on two machine learning algorithms: Multilayer perceptron (MLP) and Long Short-Term Memory (LSTM).

A multilayer perceptron is a type of feedforward artificial neural network that consists of, at least, three layers of nodes: an input layer to receive the signal, one or more hidden layers that represent the computational engine of the MLP and an output layer that makes a decision or prediction about the input. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. The typical MLP structure can be observed in Figure 1 [7].

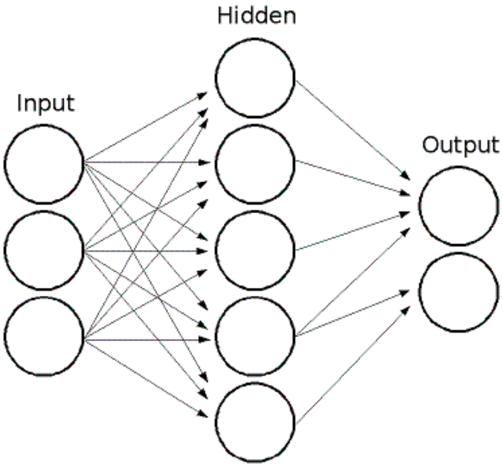

Fig. 1. Multilayer perceptron [8]

Long short-term memory (LSTM) is a type of artificial recurrent neural network (RNN) used in the field of deep learning. LSTM has feedback connections, unlike standard feedforward neural networks. A RNN using LSTM units can be trained in a supervised fashion, on a set of training sequences [10].

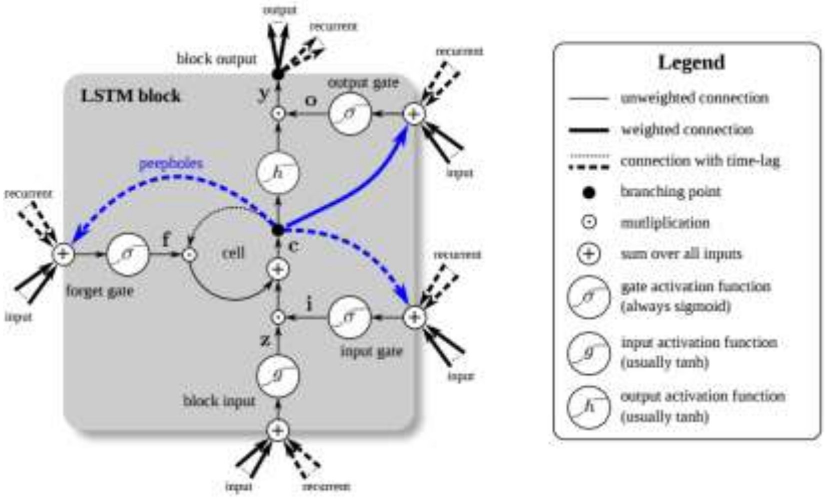

Fig. 2. Long Short-Term Memory [11]

A LSTM unit is composed of a cell (the memory part of the LSTM unit) and three gates, which regulate the flow of information inside the LSTM unit: an input gate, an output gate and a forget gate. The cell is responsible for keeping track of the dependencies between the elements in the input sequence. The input gate controls the flow of values into the cell, the forget gate controls for how long a value remains in the cell and the output gate controls how the value in the cell is used to compute the output activation of the LSTM unit. Figure

DOI: 10.12948/issn14531305/23.2.2019.07

2 outlines the structure of a LSTM unit [10].

4 Implementation

For both algorithms, the 77 projects of the Desharnais dataset are divided into a training set and a test set. The training set consists of 62 projects which amounts to 80% of the dataset, the rest of the projects are used for testing the models.

We aim to predict the effort required for completion of a software project by using the rest of the quantitative variables in the dataset as inputs (excluding irrelevant data such as project number or year end).

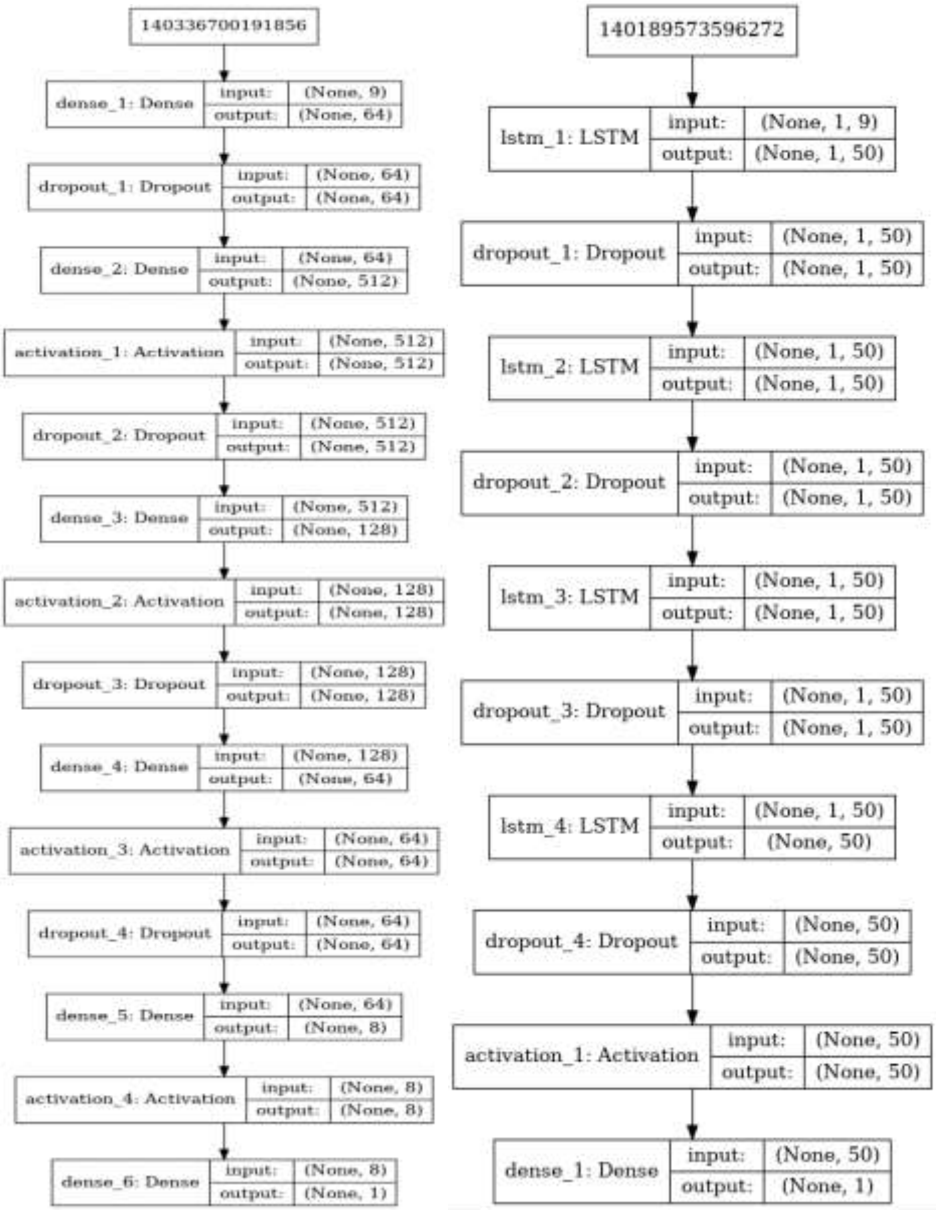
140336700191856
dense_1: Dense | input: (None, 9) | output: (None, 64)
dropout_1: Dropout | input: (None, 64) | output: (None, 64)
dense_2: Dense | input: (None, 64) | output: (None, 512)
activation_1: Activation | input: (None, 512) | output: (None, 512)
dropout_2: Dropout | input: (None, 512) | output: (None, 512)
dense_3: Dense | input: (None, 512) | output: (None, 128)
activation_2: Activation | input: (None, 128) | output: (None, 128)
dropout_3: Dropout | input: (None, 128) | output: (None, 128)
dense_4: Dense | input: (None, 128) | output: (None, 64)
activation_3: Activation | input: (None, 64) | output: (None, 64)
dropout_4: Dropout | input: (None, 64) | output: (None, 64)
dense_5: Dense | input: (None, 64) | output: (None, 8)
activation_4: Activation | input: (None, 8) | output: (None, 8)
dense_6: Dense | input: (None, 8) | output: (None, 1)
140189573596272
lstm_1: LSTM | input: (None, 1, 9) | output: (None, 1, 50)
dropout_1: Dropout | input: (None, 1, 50) | output: (None, 1, 50)
lstm_2: LSTM | input: (None, 1, 50) | output: (None, 1, 50)
dropout_2: Dropout | input: (None, 1, 50) | output: (None, 1, 50)
lstm_3: LSTM | input: (None, 1, 50) | output: (None, 1, 50)
dropout_3: Dropout | input: (None, 1, 50) | output: (None, 1, 50)
lstm_4: LSTM | input: (None, 1, 50) | output: (None, 50)
dropout_4: Dropout | input: (None, 50) | output: (None, 50)
activation_1: Activation | input: (None, 50) | output: (None, 50)
dense_1: Dense | input: (None, 50) | output: (None, 1)

Fig. 3. Layer structure of the MLP and LSTM algorithms source: Python

We have implemented the aforementioned algorithms using Python and several machine learning libraries: Keras, Tensorflow and Sklearn.

The first step of our analysis consists of normalizing the data using a MinMax Scaler provided by Sklearn. The next step is building the network using Keras by training the models on 10000 epochs. Figure 3 shows both networks' structures.

5 Results and discussion

In this study, prediction of software effort estimation was done by using a Multilayer Per-

DOI: 10.12948/issn14531305/23.2.2019.07

ceptron and a Long Short Term Memory neural net. The effort required for 17 projects from the Desharnais dataset were predicted. Predictions and performance measurements are presented in tables 12, 13 and 14.

Table 12. MLP Individual Prediction Accuracy

Actual Effort	**Predicted Effort**	**MRE**
3626	7769	1.142
6783	4100	0.395
11361	12252	0.078
1267	2414	0.905
2548	2362	0.072
1155	3041	1.633
546	3397	5.223
2275	3987	0.752
9100	11290	0.241
595	3369	4.662
3941	4124	0.046
13860	9202	0.336
1400	3754	1.682
2800	5950	1.125
9520	14569	0.530
5880	3134	0.466
23940	19734	0.175

Table 13. LSTM Individual Prediction Accuracy

Actual Effort	**Predicted Effort**	**MRE**
3626	4066	0.121
6783	2316	0.658
11361	12001	0.056
1267	1512	0.193
2548	1867	0.266
1155	1695	0.467
546	1000	0.831
2275	2494	0.096
9100	17089	0.877
595	1044	0.755
3941	3066	0.221
13860	15699	0.132
1400	1264	0.096
2800	5008	0.788
9520	19090	1.005
5880	1499	0.744
23940	19588	0.187

Table 14. Performance comparison between MLP and LSTM

Algorithm/Measure	**MSE**	**MAE**	**R2**
MLP	0.015	0.107	0.766
LSTM	0.0241	0.099	0.629

DOI: 10.12948/issn14531305/23.2.2019.07

The results which have the best performance are those with a lower MRE for individual predictions and in general the ones which have low values for MSE and MAE. The R2 score should be as close to 1 as possible.

We can see that, on average LSTM offers better individual predictions than MLP, but the best performance belongs to MLP with a higher R2 score (0.766) and lower Mean Standard Error(0.015).

Individual prediction values vary between 0.056 and 1.005 for LSTM and 0.046 and 4.662 for MLP suggesting that the dataset should be partitioned into classes for better accuracy of individual results and thus better performance of the models.

Threats to the validity of the model are as follows:

- the Desharnais data set is small in size, having only 77 projects, meaning that we only have 15 projects for testing the validity of the models, thus the model's optimal accuracy and performance cannot be guaranteed;
- prediction values are varied, some being very far off the actual values because of the size of data set and the fact that the data may have outliers.

6. Conclusions

Machine learning technology has the unique ability to process and interpret large amount of data without the need of constant human interference, the existing application are the real life, tangible proof of this fact.

This paper aims to bring an incremental upgrade to the way programmers and scientists understand the diverse benefits and capabilities of machine learning in the management of singular or enterprise projects, to highlight this fact we have brought into foreground the following points of interest:

- While there is a strong correlation between the amount and accuracy of the data involved in the training of the algorithm, the end result may still not be foolproof and may require further chiseling;
- Throughout our analysis it is concluded that data related pitfalls, such as not having enough data, low quality of the data, or biased data, are the most severe in both the training of the machine learning algorithm, as it can lead to a wrongly executed training procedure and also in the formulation of future predictions and analysis;
- It is shown that realizing the value of long-term solutions regarding machine learning in the management of projects can be a very difficult and arduous task, although the risks diminish and the ability increases with experience.

Further improvements to the two machine learning algorithms used in this paper would be:

- applying cauterization on the dataset and applying the algorithms on each cluster;
- testing the two algorithms on other data sets such as ISBSG data set;
- fine tuning the algorithms for each data set for improved accuracy and performance.

The fact that we successfully implemented and developed a machine learning algorithm that brings real benefits in the field of project management constitutes one of the many stepping stones required for future development. A main path of future progress would be the implementation and testing of the algorithm using a real life company as a subject of testing.

References

[1] Huang, J., Li, Y., & Xie, M. (2015). An empirical analysis of data preprocessing for machine learning-based software cost estimation. *Information And Software Technology*, *67*, 108-127. doi: 10.1016/j.infsof.2015.07.004

[2] Panda, A., Satapathy, S., & Rath, S. (2015). Empirical Validation of Neural Network Models for Agile Software Effort Estimation based on Story Points. *Procedia Computer Science*, *57*, 772-781. doi: 10.1016/j.procs.2015.07.474

[3] Pospieszny, P., Czarnacka-Chrobot, B., & Kobylinski, A. (2018). An effective approach for software project effort and duration estimation with machine learning algorithms. *Journal Of Systems And Software*, *137*, 184-196. doi: 10.1016/j.jss.2017.11.066

[4] Rijwani, P., & Jain, S. (2016). Enhanced Software Effort Estimation Using Multi Layered Feed Forward Artificial Neural Network Technique. *Procedia Computer Science*, *89*, 307-312. doi: 10.1016/j.procs.2016.06.073

[5] Usman, M., Britto, R., Damm, L., & Börstler, J. (2018). Effort estimation in large-scale software development: An industrial case study. *Information And Software Technology*, *99*, 21-40. doi: 10.1016/j.infsof.2018.02.009

[6] How can machine learning be used to improve project management software? - Quora - https://www.quora.com/How-can-machine-learning-be-used-to-improve-project-management-software - Accessed 9th March 2019

[7] Multilayer perceptron – Wikipedia - https://en.wikipedia.org/wiki/Multilayer_perceptron - Accessed 24th March 2019

[8] A Beginner's Guide to Multilayer Perceptrons (MLP) | Skymind - https://skymind.ai/wiki/multilayer-perceptron - Accessed 24th March 2019

[9] Multilayer perceptrons - http://users.ics.aalto.fi/ahonkela/dippa/node41.html - Accessed 24th March 2019

[10] Long short-term memory – Wikipedia - https://en.wikipedia.org/wiki/Long_short-term_memory - Accessed 24th March 2019

[11] A Beginner's Guide to LSTMs and Recurrent Neural Networks | Skymind - https://skymind.ai/wiki/lstm - Accessed 24th March 2019

[12] Desharnais dataset | Kaggle - https://www.kaggle.com/toniesteves/desharnais-dataset - Accessed 9th March 2019

Eduard-Florin PREDESCU has graduated the Faculty of Cybernetics, Statistics and Economic Informatics, Cybernetics specialization, in 2017. He holds a Master diploma in the research of Economic Informatics from 2019. He is currently working as a front-end developer in outsourcing. His main interests are AI, UI/UX and front end.

Alexandru ȘTEFAN is a hardware and software enthusiast. He received his license in Economics Informatics in 2017 from the Bucharest University of Economic Studies and his master's degree in 2019 from the same university. He is currently developing retail sales systems in a multinational company. His main research interests are Machine Learning and Games Development.

Alexis-Valentin ZAHARIA is a software developer enthusiast who graduated the Faculty of Cybernetics, Statistics and Economic Informatics, economic informatics specialization, in 2017. He holds a Master diploma in the research of Economic Informatics from 2019. Currently he is working as a Java software developer. His current interests are in the development of programmable machines and data analysis.

DOI: 10.12948/issn14531305/23.2.2019.07