Ri-STEM-2021 Proceedings

STUDENTSKI ZBOR
SVEUČILIŠTA
U RIJECI

RITEH

RITEH
AI AND ROBOTICS GROUP

STEM

Ri-STEM-2021

Proceedings

June 2021

International Scientific Student Conference

RI-STEM-2021

Rijeka, Croatia

Proceedings

Editors

Ivan Lorencin
Sandi Baressi Šegota
Zlatan Car

RI-STEM-2021

Proceedings of the International Scientific Student Conference

Organizational Committee

- str. surad. Sandi Baressi Šegota, mag. ing. comp. Croatia
- asist. Ivan Lorencin, mag. ing. el. Croatia
- str. surad. Daniel Štifanić, mag. ing. el. Croatia
- str. surad. Jelena Musulin, mag. ing. el. Croatia
- Tijana Šušteršič, MSc mech. eng. Serbia
- Anđela Blagojević, MSc el. eng. comp. scien. Serbia
- Klara Smolić, dr. med. Croatia
- Ana Zulijani, dr. med. dent. Croatia
- asist. Luka Sevšek, MSc Slovenia
- stud. Ariana Rabac, bacc. obs. Croatia
- stud. Denis Mijolović, univ. bacc. ing. el. Croatia
- stud. Marina Banov, univ. bacc. ing. comp. Croatia
- stud. Tina Mlinarić Croatia
- stud. Arian Čarapina Croatia

Scientific Committee

- red. prof. dr. sc. Zlatan Car Croatia
- prof. dr. sc. Nenad Filipović Serbia
- prof. dr. sc. Zoran Marković Serbia
- izv. prof. dr. sc. Dubravko Franković Croatia
- doc. dr. sc. Igor Poljak Croatia
- asist. dr. sc. Nikola Anđelić Croatia
- Ing. Jan Kudláček, Ph.D. Czechia

ISBN 978-953-8246-22-7

SPONSORS & SUPPORTERS

University of Rijeka Student Council

https://sz.uniri.hr/

Faculty of Engineering – University of Rijeka

http://www.riteh.uniri.hr

Association of Technical Science Students “Hertz”

Riteh AI and Robotics Group

https://www.facebook.com/RitehAIandRobot/

The Conference Website can be found below:

https://sites.google.com/view/ri-stem-2021

Foreword

Respected colleagues,

We thank you for your interest in the first RI-STEM international student scientific conference.

Last year, when we began our work with students, we realized that many students have a profound interest in science – but lack avenues which would allow them an accessible manner of learning, performing scientific research, compiling it in a manuscript and publishing their papers. It was our goal to provide students with such a possibility.

Thanks to the support from University of Rijeka's Student Council, Faculty of Engineering Rijeka, Organization of Technical Science Students Hertz and RiTEH AI and Robotics Group we have managed to achieve our goals this year. With over 20 students, overseen by committee members from 4 different countries publishing their work and presenting it on our conference.

We hope that this conference will continue in the following years and providing students with an easy first step into the world of scientific research and publications.

Kindest regards,

SZSUR Project Lead

Ivan Lorencin

Organizational Board President

Sandi Baressi Šegota

Table of Contents

ISBN: 978-953-8246-22-7

Use of Regressive Artificial Intelligence and Machine Learning Methods in Modelling of COVID-19 Spread (COVIDAi): Project Review

Anđela Blagojević[1,2], Tijana Šušteršič[1,2], Ivan Lorencin[3*], Nenad Filipović[1,2]

[1] Faculty of Engineering, University of Kragujevac, Sestre Janjić, 34000 Kragujevac, Serbia; andjela.blagojevic@kg.ac.rs, tijanas@kg.ac.rs, fica@kg.ac.rs
[2] Bioengineering Research and Development Centre (BioIRC), Prvoslava Stojanovića 6, 34000 Kragujevac,Serbia
[3] Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia; ilorencin@riteh.hr
* Correspodence: ilorencin@riteh.hr

Abstract: In this paper, a review of the project Use of Regressive Artificial Intelligence and Machine Learning Methods in Modelling of COVID-19 Spread (COVIDAi) is presented. The main goal of the project is to design two main AI-based models: epidemiological and personalized. After the introduction, a brief description of project partners and activities is provided. Furthermore, a brief description of the two main project activities is provided. After the description of the aforementioned project activities, a review of scientific papers published during project execution is presented.

Keywords: Artificial intelligence, COVID-19, Epidemiological models, Machine learning, Personalized models

1. Introduction

COVID-19, colloquially coronavirus, is a severe respiratory disease caused by SARS-CoV2 virus [1]. In March 2020, The World Health Organization (WHO) announced the pandemic [2]. Ever since, the world is, in some form, under the restrictive measures used to interrupt the spread of COVID-19 infection. In the early days of the pandemic, the main goal was to predict the spread of the infection in order to minimize the number of infected individuals, and consequently the number of deceased patients [3]. Along with the outbreak of the pandemic, especially in the second wave in late 2020, the need to develop more effective methods for the treatment and care of COVID-19 patients increased [4]. Such a condition is a direct consequence of the far-reaching spread of the infection and its growing impact on the healthcare system [5].

Artificial intelligence (AI) and machine learning (ML) are today widely integrated into the medical profession, with a wide range of applications [6, 7]. Following the described trends, a team composed of scientists from the University of Kragujevac (Serbia) and the University of Rijeka (Croatia) launched the COVIDAi project, with the aim of applying AI and ML methods in the fight against the COVID-19 pandemic. This conference paper aims to briefly describe the activities carried out within the COVIDAi project and give a brief overview of the results and findings.

2. A brief description of project activities and partners

COVIDAi project is a project supported by Central European Initiative (CEI) under the grant number 305.6019-20. The project consortium consists of University of Kragujevac, University of Rijeka, Faculty of Engineering, and Clinical Hospital Center Rijeka. The main researchers of the project are Professor Nenad Filipović (PI) from the University of Kragujevac and Professor Zlatan Car (CO-PI) from University of Rijeka, Faculty of Engineering. All project activities are performed in order of

creating a tool that will be used in medical practice. Main activities performed within the COVIDAI project include the development of two models:

- Personalized AI model for COVID-19 prediction (monitoring of patient's condition and prediction of disease progression in time)
- An epidemiological model for COVID-19 (monitoring of number of people susceptible/exposed/infected/dead/recovered from COVID-19)

COVIDAi tool would help medical experts to decide whether the patient will be subjected to further analysis and prescribe adequate therapy. Predictive models based on machine learning can provide useful data in terms of prediction of epidemiological events, which can save time for the timely and optimal response of both the health system and the society.

3. Epidemiological models for COVID-19

COVIDAi uses a compartmental epidemiological model, based on the partial differential equations to describe the spread and clinical progression of COVID-19. The basic model structure is inspired by several studies on the natural clinical progression of COVID-19 infection. Alongside models based on differential equations, models based on the utilization of AI and ML algorithms were also developed. The most prominent results were achieved by using a multilayer perceptron (MLP) and genetic programming (GP). Prediction of COVID-19 spread with COVIDAi tool is performed by using data provided by institutions such as WHO, Johns Hopkins University or Institutes of Public Health, as presented in Figure 1.

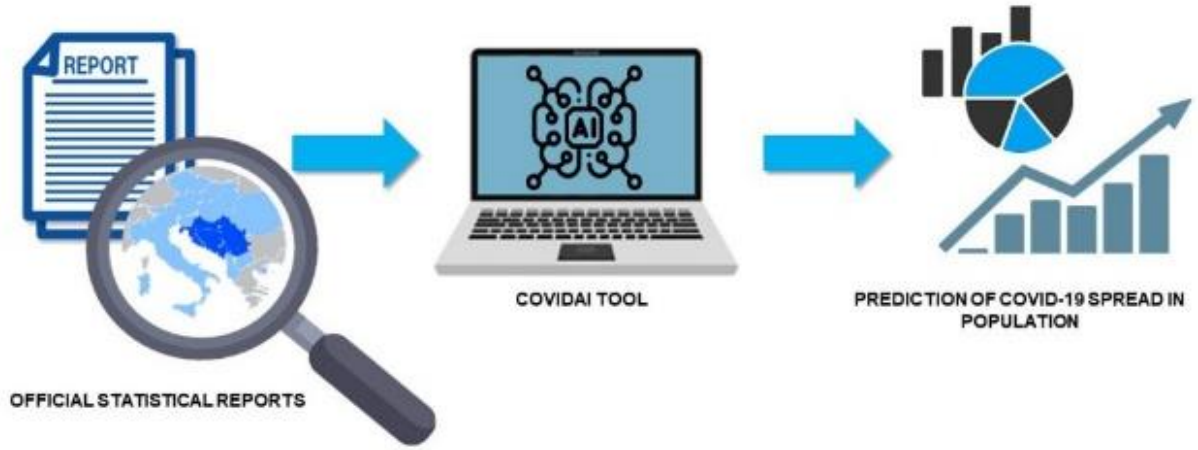


Figure 1. *A flowchart of COVID-19 spread prediction using COVIDAi tool*

Alongside the epidemiological aspect [8], modeling of COVID-19 spread can also be used in forecasting the stock market dynamics [9].

4. Personalized AI model for COVID-19 prediction

The developed disease progression tool uses machine learning methods to mine heterogeneous patient data provided by Clinical Centers in Rijeka, Croatia, and Kragujevac, Serbia. The main aim of this tool is to assess the disease progression of the patient infected with COVID-19 in the next couple of days. In order to estimate the disease progression, the input dataset consisting of:

- demographic data,
- clinical image,
- blood test data and

- imaging data

is used. The result of the model is a prediction of the category risk of mortality.

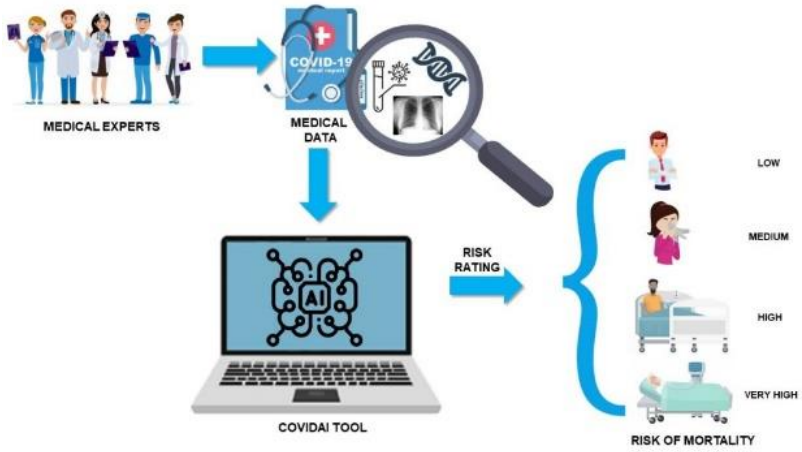


Figure 2. *A Flowchart for mortality risk estimation using personalized COVIDAi tool*

A unique feature of COVID-19 interstitial pneumonia is an abrupt progression to respiratory failure. Patient-specific lung models developed during COVIDAi project are focusing on the spread of virus-laden to many regions of the lungs from the initial site of infection. An example of such a model is presented in Figure 3.

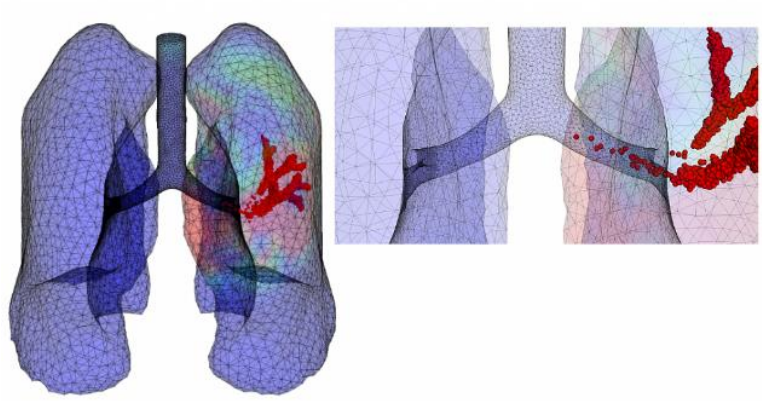


Figure 3. *An example of patient-specific lung model*

5. Achieved results

The results of the project have been published in several scientific articles in eminent journals. The first research published during the project is an article by Anđelić et. al. [10]. The aim of the research presented in this article was to implement genetic programming algorithm (GP) in order to model the spread of COVID-19 in China, Italy, Spain, and the USA. The modeling process is based on the estimation of epidemic curves derived from the number of infected, deceased and recovered patients. Alongside individual country models, the global model was developed as well. From the results, it can be seen that models for the number of infected and deceased cases achieved the R^2 scores of 0.999, while the models for the estimation of the number of recovered patients achieved the R2 scores of 0.998.

Another similar research was also published during the project. The research presented in [11] was based on the utilization of GP for estimation of COVID-19 epidemic curves for the USA, for each state

individually. On the described way, R^2 scores in ranges 0.9406–0.9992, 0.9404–0.9998 and 0.9797–0.99955 were achieved for the estimations of infected, deceased and recovered patients respectively.

An overview of AI-based epidemiological models was presented in [12], where a detailed systematic review was given.

Alongside epidemiological models, another goal of the project was to develop automatic, personalized, models that will be used in the treatment of COVID-19 patients. The aforementioned models are used in order to develop an automatic decision support system that will be used in clinical practice as help to medical professionals. In the research published in [13] a system for the automatic evaluation of the lung condition of COVID-19 patients was presented. The aim of this research was to estimate the clinical picture of the patient from the x-ray images of the lungs. During the research, multiple convolutional neural networks (CNN) were used, and these are:

- AlexNet,
- VGG-16,
- ResNet50,
- ResNet101 and
- ResNet152.

Results show that the best classification performances can be achieved with ResNet152. By using this CNN, micro-AUC and macro-AUC values up to 0.94 were achieved.

Another research, based on the prognosis of disease development, is presented in [14]. In this research, a personalized model for COVID-19 disease prognosis is presented. The model is based on the combination of machine learning and finite element simulation. By using the presented approach, prediction accuracy up to 90% was achieved.

6. Conclusion

In the presented project review main activities conducted during the project duration were presented. For each activity and model developed a brief description was provided. In addition to the description of the activities, a brief description of the publications published as part of the project is given.

Acknowledgments

This research has been (partly) supported by the CEEPUS network Ciii-HR-0108, European Regional Development Fund under the grant K.01.1.1.01.0009 (DATACROSS), project CEKOM under the grant KK.01.2.2.03.0004, CEI project “COViDAi” (305.6019-20), project Metalska jezgra Čakovec (KK.01.1.1.02.0023) and University of Rijeka scientific grant uniri-tehnic-18-275-1447.

References

[1] Velavan, Thirumalaisamy P., and Christian G. Meyer. "The COVID-19 epidemic." Tropical medicine & international health 25.3 (2020): 278.
[2] Cucinotta, Domenico, and Maurizio Vanelli. "WHO declares COVID-19 a pandemic." Acta Bio Medica: Atenei Parmensis 91.1 (2020): 157.

[3] Wieczorek, Michał, Jakub Siłka, and Marcin Woźniak. "Neural network powered COVID-19 spread forecasting model." Chaos, Solitons & Fractals 140 (2020): 110203.
[4] Felsenstein, Susanna, et al. "COVID-19: Immunology and treatment options." Clinical Immunology (2020): 108448.
[5] Rahmatizadeh, Shahabedin, Saeideh Valizadeh-Haghi, and Ali Dabbagh. "The role of artificial intelligence in management of critical COVID-19 patients." Journal of Cellular & Molecular Anesthesia 5.1 (2020): 16-22.
[6] Musulin, Jelena, et al. "An Enhanced Histopathology Analysis: An AI-Based System for Multiclass Grading of Oral Squamous Cell Carcinoma and Segmenting of Epithelial and Stromal Tissue." Cancers 13.8 (2021): 1784.
[7] Lorencin, Ivan, et al. "On Urinary Bladder Cancer Diagnosis: Utilization of Deep Convolutional Generative Adversarial Networks for Data Augmentation." *Biology* 10.3 (2021): 175. https://doi.org/10.3390/biology10030175
[8] Car, Zlatan, et al. "Modeling the spread of COVID-19 infection using a multilayer perceptron." *Computational and mathematical methods in medicine* 2020 (2020).. https://doi.org/10.1155/2020/5714714
[9] Štifanić, Daniel, et al. "Impact of covid-19 on forecasting stock prices: an integration of stationary wavelet transform and bidirectional long short-term memory." Complexity 2020 (2020).
[10] Anđelić, Nikola, et al. "Estimation of COVID-19 epidemic curves using genetic programming algorithm." Health Informatics Journal 27.1 (2021): 1460458220976728.
[11] Anđelić, Nikola, et al. "Estimation of covid-19 epidemiology curve of the united states using genetic programming algorithm." International Journal of Environmental Research and Public Health 18.3 (2021): 959.
[12] Musulin, J., Baressi Šegota, S., Štifanić, D., Lorencin, I., Anđelić, N., Šušteršič, T., ... & Markova-Car, E. (2021). Application of Artificial Intelligence-Based Regression Methods in the Problem of COVID-19 Spread Prediction: A Systematic Review. International Journal of Environmental Research and Public Health, 18(8), 4287.
[13] Lorencin, Ivan, et al. "Automatic Evaluation of the Lung Condition of COVID-19 Patients Using X-ray Images and Convolutional Neural Networks." Journal of Personalized Medicine 11.1 (2021): 28.
[14] Blagojević, Anđela, et al. "Combined machine learning and finite element simulation approach towards personalized model for prognosis of COVID-19 disease development in patients." EAI Endorsed Transactions on Bioengineering and Bioinformatics 1.2 (2021): e6.

ISBN: 978-953-8246-22-7

Multiclass Classification of Oral Squamous Cell Carcinoma

Jelena Musulin [1*], Daniel Štifanić[1], Ana Zulijani[2] and Zlatan Car [1]

[1] Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia; dstifanic@riteh.hr (D.Š.), car@riteh.hr (Z. C.)
[2] Department of Oral Surgery, Clinical Hospital Center Rijeka, Krešimirova ul. 40, 51000 Rijeka, Croatia; ana.zulijani@sz.uniri.hr (A.Z.)
[*] Correspondence: jmusulin@riteh.hr (J.M.)

Abstract: Oral cancer (OC) is a type of head and neck cancer in which malignant cells appear on the lips or in the oral cavity (in the mouth). Early detection of OC may increase the chances of survival in individuals, but new technologies may be expensive and time-consuming. Lately, the possibility of automated medical diagnosis with the aid of Artificial Intelligence (AI) tools has been receiving much attention. In this research, the integration of preprocessing techniques along with Xception algorithm is proposed for oral squamous cell carcinoma classification. The dataset was obtained from the Clinical Hospital Center in Rijeka and consists of 257 histopathology images. The proposed system achieved satisfactory results in terms of multiclass classification.

Keywords: Artificial Intelligence, Histopathology images, Oral Squamous Cell Carcinoma, Xception

1. Introduction

Oral cancer belongs to the group of head and neck cancers and most commonly affects the tissues and mucous membranes of the mouth and throat [1]. It is one of the most common neoplasms occurring in both sexes, where oral squamous cell carcinoma (OSCC) leads this group of malignancies [2]. Consumption of tobacco and alcohol are major risk factors for the development of the OC, and their synergistic effect increases the risk of developing OC by up to 15 times than in the population that does not consume these products. The standard oral cancer diagnostic procedure is based on histopathologic examination, however, the major issue in this type of procedure is tumor heterogeneity. Clinician subjective component of the examination could have a direct impact on the patient-specific treatment intervention. For this reason, image processing techniques and Artificial Intelligence (AI) algorithms can be used to enhance objectivity and reproducibility in order to improve survival rates and treatment [4]. Recently, numerous AI algorithms have proven to be successful in the field of medicine as well as other various fields [5-13].

The main aim of this research is the multiclass classification of OSCC which could assist the clinician in reducing the time needed for histopathological examination as well as reducing inter- and intra-observer variability. The overview of the proposed methodology is given in Figure 1.

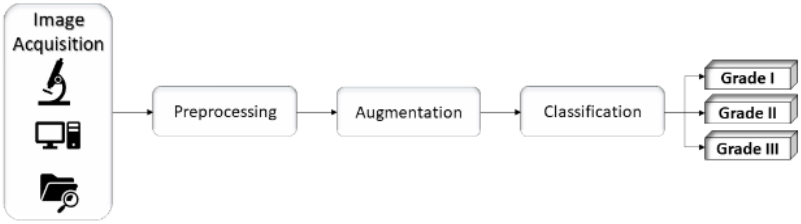


Figure 1. *Block diagram of the proposed framework.*

2. Methodology

The formalin-fixed, paraffine-embedded oral mucosa tissue blocks of histopathologically reported cases of oral lichen planus (OLP), oral leukoplakia (OL) and oral squamous cell carcinoma (OSCC) were retrieved from the

archives of Clinical Department of Pathology and Cytology, Clinical Hospital Center in Rijeka. Sections were graded into multiple pathologic categories (inflammation, mild OED, moderate OED, severe OED architectural and cytological changes in neoplastic epithelium) according to the World Health Organization criteria for architectural and cytological changes in epithelium. The OSCC group includes well-differentiated OSCC, moderately and poorly differentiated OSCC (grade I, grade II, grade III) shown in Figure 2.

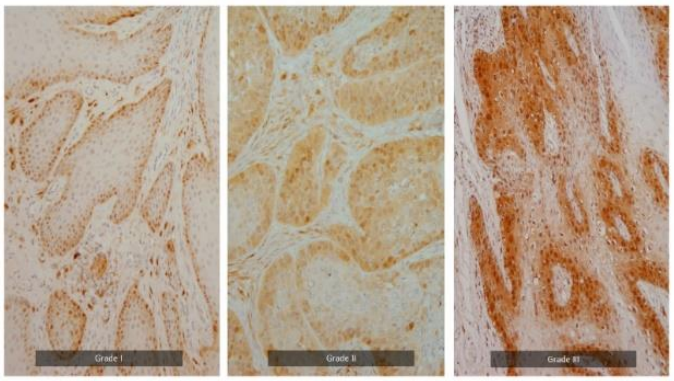


Figure 2. OSCC group of well-differentiated OSCC, moderately differentiated OSCC and poorly differentiated OSCC (grade I, II, and III respectively)

Images were captured using the light microscope (Olympus BX51, Olympus, Japan) equipped with a digital camera and transmitted to a computer by CellF software. Histopathology images of OSCC are used as input data for AI algorithm to create a diagnostic system for multiclass classification of OSCC. When it comes to image data for classification, the data availability can be problematic. To address concerns like time-consuming collecting or the small number of images, data augmentation was performed. By utilizing augmentation techniques, a new set of 1799 images has been created, which gives a total of 2056 images.

In order to achieve robust classification, a feature extraction algorithm, Stationary Wavelet Transform (SWT) is used. Such pre-processing approach can reduce the computational complexity of the AI algorithms which will result in shorter training time and potentially better performance. To enhance important features of an image, SWT coefficient mapping function is utilized. Such function is proposed and mathematically described in [14]. Classification algorithm takes a set of digital images obtained by experts and returns a class. Xception is convolutional neural network architecture used in this research for diagnosis and outcome prediction of oral cancer. It consists of 36 convolutional layers structured in 14 modules [15].

3. Results and Discussion

This section demonstrates the obtained experimental results achieved with Xception architecture, which is pretrained on ImageNet. To determine the quality of the model statistical measures such as Micro- and Macro- Area Under the Curve (AUC) are adopted.

The first experimental results achieved with Xception trained with three optimizers: Stochastic Gradient Descent (SGD), Adam, and RMSprop are shown in Figure 3.

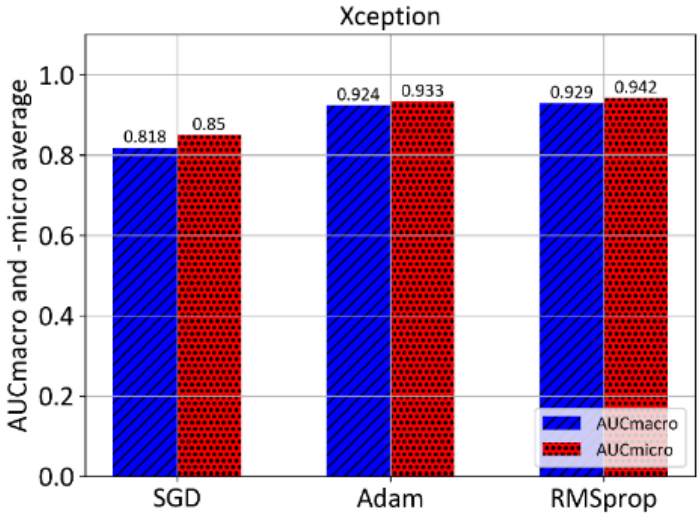


Figure 3. Comparison of mean AUCmacro and -micro values of three optimizers on pre-trained Xception

According to the 5-fold cross-validation results, in the case of multiclass classification with no preprocessing, it can be concluded that high values of 0.929 AUCmacro and 0.942 AUCmicro are achieved in a combination with the RMSprop optimizer.

With the help of SWT, the images were decomposed on the LL, LH, HL, and HH subbands which allowed coefficient weighting. LL represents approximation coefficients while LH, HL, and HH represent detail coefficients of an image. After the decomposition process, detail coefficients are weighted using the mapping function, which resulted in new, modified LH', HL', HH' subbands. Modified subbands along with unmodified LL subband were used for SWT reconstruction in order to obtain input image for AI algorithm. Using Bayesian optimization, the aim was to find optimal values of wavelet mapping function constants, by which the maximum value of the performance measure is reached. The most effective constant configuration is shown in Table 1.

Table 1. *Constants of coefficient mapping function along with wavelet function and the corresponding value of performance measure.*

Parameters					Xception + SWT	
a	b	c	d	wavelet	$AUC_{macro} \pm \sigma$	$AUC_{micro} \pm \sigma$
0.0084	0.0713	0.0599	0.0566	sym2	0.956 ± 0.054	0.964 ± 0.040
0.0091	**0.0301**	**0.0086**	**0.3444**	**db2**	**0.963 ± 0.042**	**0.966 ± 0.027**
0.0063	0.0021	0.0771	0.3007	db2	0.947 ± 0.092	0.954 ± 0.069

If the performances, in the case of multiclass classification with preprocessing, are compared, it can be noticed that the highest AUCmacro value of 0.963 and AUCmicro value of 0.966 with a standard deviation of 0.042 and 0.027, respectively, are achieved with the optimal selection of wavelet coefficient mapping function constants and wavelet function. Moreover, when all results are summed up, it can be concluded that the highest values of performance measure are achieved using the Xception in combination with SWT as a preprocessing technique.

The training-validation-testing process during the research was performed on Z4 server. It is a GPU based High Performance Computing (HPC) server. The server consists of two Intel Xeon Gold CPUs (24C/48T, at 2.4Ghz), 768 GB of ECC DDR4 RAM, and five Nvidia Quadro RTX 6000 GPUs, with 24 GB of RAM, 4,608 CUDA and 576 Tensor cores.

4. Conclusion

This research highlights the huge potential of the application of AI algorithms and preprocessing techniques in order to achieve an effective prognosis of OSCC. From obtained results, it can be concluded that integration of Xception and SWT resulted in the highest performance value of 0.963 AUCmacro and 0.966 AUCmicro. Future work should use a dataset with more histopathological images, in order to achieve a more robust classification system.

Acknowledgments
This research has been (partly) supported by the CEEPUS network CIII-HR-0108, European Regional Development Fund under the grant KK.01.1.1.01.0009 (DATACROSS), project CEKOM under the grant KK.01.2.2.03.0004, CEI project “COVIDAi” (305.6019-20) and University of Rijeka scientific grant uniri-tehnic-18-275-1447.

Reference

[1] Torre, Lindsey A., et al. "Global cancer incidence and mortality rates and trends—an update." Cancer Epidemiology and Prevention Biomarkers 25.1 (2016): 16-27. DOI: https://doi.org/10.1158/1055-9965.epi-15-0578
[2] Marur, Shanthi, and Arlene A. Forastiere. "Head and neck cancer: changing epidemiology, diagnosis, and treatment." Mayo Clinic Proceedings. Vol. 83. No. 4. Elsevier, 2008. DOI: https://doi.org/10.4065/83.4.489
[3] Bagan, Jose, Gracia Sarrion, and Yolanda Jimenez. "Oral cancer: clinical features." Oral oncology 46.6 (2010): 414-417.DOI:https://doi.org/10.1016/j.oraloncology.2010.03.009
[4] Musulin, Jelena, et al. "An Enhanced Histopathology Analysis: An AI-Based System for Multiclass Grading of Oral Squamous Cell Carcinoma and Segmenting of Epithelial and Stromal Tissue." Cancers 13.8 (2021): 1784. DOI: https://doi.org/10.3390/cancers13081784
[5] Lorencin, Ivan, et al. "Using multi-layer perceptron with Laplacian edge detector for bladder cancer diagnosis." Artificial Intelligence in Medicine 102 (2020): 101746. DOI: https://doi.org/10.1016/j.artmed.2019.101746
[6] Musulin, Jelena, et al. "Bladder cancer detection: Integration of feature extraction algorithms and MLP." DOI: none
[7] Watanabe, Alyssa T., et al. "Improved cancer detection using artificial intelligence: a retrospective evaluation of missed cancers on mammography." Journal of digital imaging 32.4 (2019): 625-637. DOI: https://doi.org/10.1007/s10278-019-00192-5
[8] Musulin, Jelena, et al. "COMPARISON OF THREE ARTIFICIAL INTELLIGENCE ALGORITHMS FOR SEPSIS PREDICTION." World of Health (2020). DOI: none
[9] Lorencin, Ivan, et al. "Edge Detector-Based Hybrid Artificial Neural Network Models for Urinary Bladder Cancer Diagnosis." Enabling AI Applications in Data Science. Springer, Cham, 2020. 225-245. DOI: https://doi.org/10.1007/978-3-030-52067-0_10
[10] Štifanić, Daniel, et al. "Use of Convolutional Neural Network for Fish Species Classification." Pomorski zbornik 59.1 (2020): 131-142. DOI: https://doi.org/10.18048/2020.59.08.
[11] Elkatatny, Salaheldin, et al. "New insights into the prediction of heterogeneous carbonate reservoir permeability from well logs using artificial intelligence network." Neural Computing and Applications 30.9 (2018): 2673-2683. DOI: https://doi.org/10.1007/s00521-017-2850-x

[12] Anđelić, Nikola, et al. "Estimation of COVID-19 epidemic curves using genetic programming algorithm." Health Informatics Journal 27.1 (2021): 1460458220976728. DOI: https://doi.org/10.1177/1460458220976728

[13] Baressi Šegota, Sandi, et al. "Frigate Speed Estimation Using CODLAG Propulsion System Parameters and Multilayer Perceptron." NAŠE MORE: znanstveni časopis za more i pomorstvo 67.2 (2020): 117-125. DOI: https://doi.org/10.17818/nm/2020/2.4

[14] Štifanić, Daniel, et al. "Impact of covid-19 on forecasting stock prices: an integration of stationary wavelet transform and bidirectional long short-term memory." Complexity 2020 (2020). DOI: https://doi.org/10.1155/2020/1846926

[15] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1251-1258). DOI: https://doi.org/10.1109/cvpr.2017.195

ISBN: 978-953-8246-22-7

Gait speed prediction based on walking parameters using MLPRegressor

Dejan Radulović [1*], Dino Negovanović [2]

[1*] Faculty of Engineering RITEH University of Rijeka, Vukovarska 58, 51000 Rijeka, Hrvatska, dradulovic@riteh.hr.

[2] Faculty of Engineering RITEH University of Rijeka, Vukovarska 58, 51000 Rijeka, Hrvatska, dino.negovanovic1@gmail.com.

Abstract: The aim of this paper is to regress the walking speed from all other parameters in given dataset. Given dataset consists of the parameters that are divided into four groups: Basic Parameters, Temporary Parameters, Spatial Parameters and Height Parameters. Based on the given data, the goal is to train the neural network to predict gait speed from other features. Many Python libraries have been used, and in addition to pandas and statistics, the most important are sklearn.neural_network and sklearn.model_selection. The neural network architecture used is a multilayer perceptron (MLP), and the regression quality is evaluated using R^2 and RMSE metrics. A GridSearchCV was implemented, which was used to optimize ie. tune hyperparameters and the results are presented in tables that contain true, predicted values, mean squaed errors and standard deviations. The main conclusion is that by increasing the number of hidden layers that define the structure of the artificial neural network, the training takes longer but also reduces the error, ie. the difference between the true and predicted values.

Keywords: Artificial intelligence, Machine Learning, MLPRegressor, Multilayer Perceptron

1. Introduction

The problem to be solved is the prediction of gait speed from other gait parameters using a multilayer perceptron. Joo et al. (2014) [1] applied artificial neural networks and machine learning algorithms with the goal of gait speed prediction from plantar pressure. By applying the methods of artificial intelligence, more precisely the multilayer perceptron, it is possible to regress the gait speed value from the dataset. By correctly adjusting the parameters of the multilayer perceptron, more accurate results are obtained. Gait Classification Data Set was created by calculating the walking parameters of a total 16 different volunteers, 7 female and 9 male. The volunteers of 16 volunteers ranged between 20 and 34 years old, and their weight ranged from 53 to 95. In order to calculate each walking parameter, people were asked to walk the 30-meter long course for three rounds [2]. As mentioned earlier, Gait Data consists of the following parameters; Basic Parameters (Speed, Variability, Symmetry), Temporary Parameters (Heel Press Time, Cycle Time, Cadence, Posture, Oscillation, Loading, Foot Press, Thrust, Double Support), Spatial Parameters (Maximum Heel Height, Maximum Finger Tip Height, Minimum Finger Tip Height) and Height Parametars (Step Length, Step Speed, Peak Angle Speed, etc.). After analysis of the data set it was necessary to regress the first (Speed) from all other parameters using a multilayer perceptron in Python programming language.

2. Methodology

The shared dataset file (Gait Classification Data Set) contains X and y data where X represents gait data and y represents person information for the relevant sample. First of all, it was necessary to convert .MAT to .csv format in order to be able to open it in Microsoft Office® Excel for easier feature analysis. The conversion was done using the pandas library. In the given dataset there is a total of 48 rows representing 16 volunteers where each of them did the test 3 times. There are 24 columns representing the gait features as previously described. To solve the regression problem Multilayer Perceptron (MLP) has been applied by the authors. Multilayer perceptron is a type of a feed-forward neural network

characterized by a single output, multiple inputs and one or more hidden layers [3]. Figure 1. explains the architecture in more detail: Multilayer perceptron is a neural network learning algorithm with a teacher (supervised learning) that learns a function $f(\cdot): R^m \rightarrow R^o$ on a given set of data where m represents the number of input dimensions, and o represents the number of output dimensions. For a given set of input data: $X = x_1, x_2, \ldots, x_m$ and a given set of output data y, a neural network can learn an approximate nonlinear function for classification or regression.

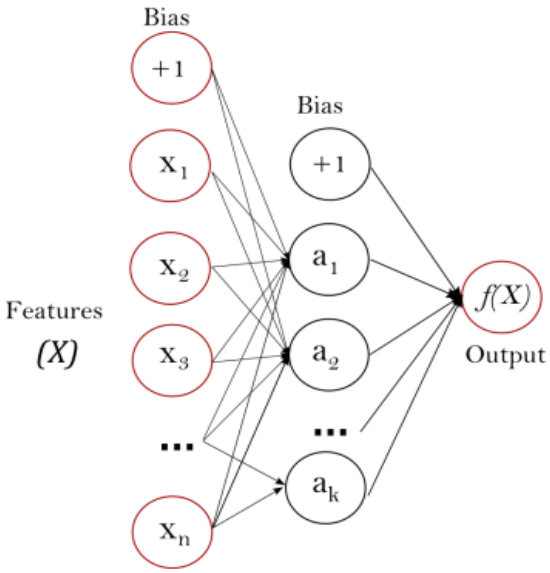


Figure 1. *MLP – Multi Layer Perceptron* [4]

The first layer represents the input layer consisting of a set of neurons, and they represent the input characteristics. Each neuron in the hidden layer transforms the sum of input values and weight factors $w_1x_1 + w_2x_2 + \cdots + w_mx_m$, by applying an activation function (in this case relu, tanh and logistic activation functions were used). The output layer receives values from the hidden layer and transforms them into output values. The module contains two available characteristics, named coefs_ and intercepts_. Coefs_ represents a matrix of weight factors in which the index i represents the weight factor between layer i and layer i + 1. Intercepts_ is a list of bias vectors, where a vector with index i represents the bias value added to layer i + 1. After conversion to .csv format, using the same library (pandas) a data frame is provided whose values are assigned to a unique variable called data. Furthermore, variable X contains only dataset features (Variability, Symmetry, Heel Press Time, etc.), without tags (Speed). On the contrary, variable y contains only tags, without features. Additionally, 80% of the dataset is randomly selected for the training, and the remaining 20% of the data is used to test the model. One important factor in the performance of the MLPRegressor model are its hyperparameters, once the appropriate values for these hyperparameters are set, the performance of a model can improve significantly [5]. Since it is difficult to guess the best parameters for the regressor, tuning of hyperparameters is performed using GridSearchCV and Cross Validation. GridSearchCV (Figure 2.) is the process of performing hyperparameter tuning in order to determine the optimal values for a given model [6]. Aforementioned, the performance of a model significantly depends on the value of hyperparameters. Because there is no way to know in advance the best values for hyperparameters so ideally, all possible values must be tried to determine the optimal values. This function helps to loop through predefined hyperparameters and fit the estimator (model) on particular training set. So, in the end, the selection of the best parameters from the listed hyperparameters is performed. As above-stated, predefined values for hyperparameters are passed to the GridSearchCV function. This is done by defining a dictionary in which we mention a particular hyperparameter along with the values it can take.

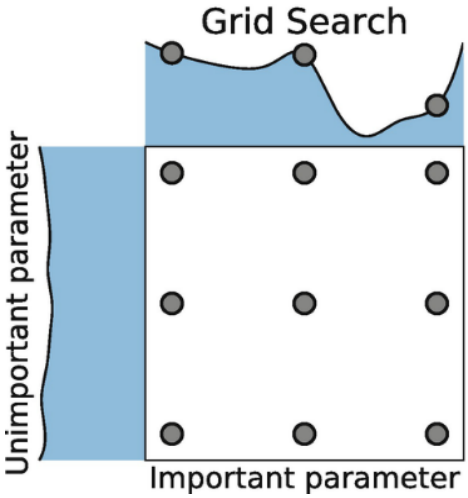


Figure 2. *Grid Search [7]*

In general, Cross-Validation (Figure 3.) is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called k or cv that refers to the number of groups that a given data sample is to be split into [8]. As such, the procedure is often called k-fold cross-validation. In this particular case, given data samples were split into 5 groups (k=cv=5). After optimization, along with the best regressor model, prediction results, mean error, and standard deviations are displayed using a dictionary.

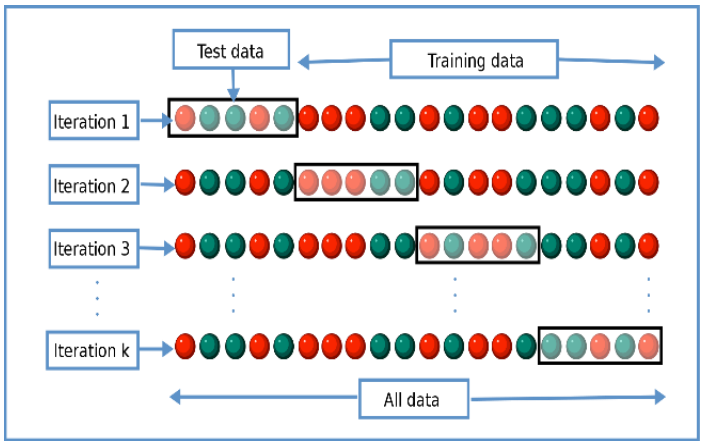


Figure 3. *Cross-Validation [9]*

3. Results and Discussion

The tables below were used to show how the performance of the regressor model changes. The performance of the regressor model depends mostly on the number of hidden layers of the neural network which define its structure. Therefore, different values of the number of hidden layers, total variance explained by model/total variance (R^2 metrics), mean squared error values (RMSE metrics), and standard deviations are given in the tables as well as the predicted and the true values for the particular number of hidden layers.

Table 1. *Hidden layer sizes = (50, 100, 50)*

R^2	R^2 stdev	RMSE	RMSE stdev
0.82080	0.08364	0.00173	0.00086

Predicted values: [1.25, 1.26, 1.35, 1.26, 1.27, 1.29, 1.27, 1.26, 1.26, 1.35]

True values: [1.31, 1.28, 1.26, 1.27, 1.25, 1.21, 1.25, 1.33, 1.28, 1.18]

Despite the fact that the training lasts longer, for this case the smallest values of errors and the smallest deviations of the predicted walking speeds from the true values are obtained.

Table 2. *Hidden layer sizes = (30, 30, 30)*

R^2	R^2 stdev	RMSE	RMSE stdev
0.57567	0.19618	0.00383	0.00201

Predicted values: [1.30, 1.28, 1.38, 1.28, 1.28, 1.29, 1.28, 1.28, 1.40, 1.32]

True values: [1.31, 1.28, 1.26, 1.27, 1.25, 1.21, 1.25, 1.33, 1.28, 1.18]

As expected, the training lasted shorter than in the previous case, but the deviation between the predicted and exact values also increased. In the following case the size of the model is further reduced and less time is expected to be required for training, as even greater deviations than in this case.

Table 3. *Hidden layer sizes = (20, 20, 20)*

R^2	R^2 stdev	RMSE	RMSE stdev
0.25434	0.06417	0.00694	0.00173

Predicted values: [1.27, 1.29, 1.33, 1.32, 1.32, 1.31, 1.31, 1.28, 1.30, 1.29]

True values: [1.31, 1.28, 1.26, 1.27, 1.25, 1.21, 1.25, 1.33, 1.28, 1.18]

Table 4. *Hidden layer sizes = (20, 10, 2)*

R^2	R^2 stdev	RMSE	RMSE stdev
0.26591	0.37596	0.00715	0.00400

Training lasts significantly shorter than in previous cases, it is also logical to expect larger deviations from previous cases, which is easy to conclude from the given predictions. In this case, the inability of

the model to predict speeds well due to the significantly smaller number of hidden layers, ie. the cramped model, is also observed.

Predicted values: [1.33, 1.33, 1.34, 1.33, 1.33, 1.33, 1.33, 1.33, 1.33, 1.34]

True values: [1.31, 1.28, 1.26, 1.27, 1.25, 1.21, 1.25, 1.33, 1.28, 1.18]

Table 5. *Hidden layer sizes = (8, 4, 2)*

R^2	R^2 stdev	RMSE	RMSE stdev
0.01612	0.02242	0.00913	0.00203

Predicted values: [1.33, 1.33, 1.33, 1.33, 1.33, 1.33, 1.33, 1.33, 1.33, 1.33]

True values: [1.31, 1.28, 1.26, 1.27, 1.25, 1.21, 1.25, 1.33, 1.28, 1.18]

In the latter case, training takes the least amount of time compared to other cases. Also, the largest deviations and the inability of the model to correctly predict the mark, ie. the gait speed due to the given limits in the number of hidden layers, are also observed.

4. Conclusion

From the results of numerical experiments, it is concluded that the multilayer perceptron, more precisely the MLP Regressor, better predicts the values of gait speed with a larger number of hidden neurons in the layer. This is easily seen from the tables if we look at the results of the metrics (R^2 and RMSE where R^2 changes in the range from 0 to 1 and a value closer to 1 means better prediction or less error) [10]. Likewise, a greater number of hidden neurons results in a longer neural network training time and a reduced error in predicting gait speed values. The model has been shown to best regress the value of the label (Speed) when the size of the hidden layer is (50, 100, 50). The solver that fits the best model is 'lbfgs' and the activation function is 'logistic'. Potential points for future work would be additional optimization of hyperparameters and the possibility of predicting walking speed for a specific volunteer.

Acknowledgments

We thank assistants Sandi Baressi Šegota and Nikola Anđelić for their mentorship during the work on this project and article.

References

[1] Joo, S. B., Oh, S. E., Sim, T., Kim, H., Choi, C. H., Koo, H., & Mun, J. H. (2014). Prediction of gait speed from plantar pressure using artificial neural networks. *Expert Systems with Applications*, *41*(16), 7398-7405. https://doi.org/10.1016/j.eswa.2014.06.002

[2] UCI Machine Learning Repository: Gait Classification Data Set, accessed: 15.04.2021; from: http://archive.ics.uci.edu/ml/datasets/Gait+Classification

[3] Murtagh, F. (1991). Multilayer perceptrons for classification and regression. *Neurocomputing*, *2*(5-6), 183-197. https://doi.org/10.1016/0925-2312(91)90023-5

[4] Walczak, S. (2019). Artificial neural networks. In *Advanced Methodologies and Technologies in Artificial Intelligence, Computer Simulation, and Human-Computer Interaction* (pp. 40-53). IGI Global. https://doi.org/10.4018/978-1-5225-2255-3.ch011

[5] Feurer, M., & Hutter, F. (2019). Hyperparameter optimization. In *Automated Machine Learning* (pp. 3-33). Springer, Cham. https://doi.org/10.1007/978-3-030-05318-5_1

[6] Lerman, P. M. (1980). Fitting segmented regression models by grid search. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, *29*(1), 77-84. https://doi.org/10.2307/2346413

[7] Jiménez, Á. B., Lázaro, J. L., & Dorronsoro, J. R. (2007). Finding optimal model parameters by discrete grid search. In *Innovations Hybrid Intelligent Systems* (pp. 120-127). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-74972-1_17

[8] Picard, R. R., & Cook, R. D. (1984). Cross-validation of regression models. *Journal of the American Statistical Association*, *79*(387), 575-583. https://doi.org/10.1080/01621459.1984.10478083

[9] Hjorth, J. U. (1993). *Computer intensive statistical methods: Validation, model selection, and bootstrap*. CRC Press. https://doi.org/10.1201/9781315140056-3

[10] Miles, J. (2014). R squared, adjusted R squared. *Wiley StatsRef: Statistics Reference Online*. https://doi.org/10.1002/9781118445112.stat06627

ISBN: 978-953-8246-22-7

A Brief Note on the Influence of Storage Choices on Machine Learning Algorithm Training Times

Sandi Baressi Šegota[1*], Daniel Štifanić[2], Jelena Musulin[3], Zlatan Car[4]

[1] Faculty of Engineering – University of Rijeka, Vukovarska 58, 5100 Rijeka, Croatia, sbaressisegota@riteh.hr
[2] Faculty of Engineering – University of Rijeka, Vukovarska 58, 5100 Rijeka, Croatia, dstifanic@riteh.hr
[3] Faculty of Engineering – University of Rijeka, Vukovarska 58, 5100 Rijeka, Croatia, jmusulin@riteh.hr
[4] Faculty of Engineering – University of Rijeka, Vukovarska 58, 5100 Rijeka, Croatia, car@riteh.hr

Abstract: Training times of ML algorithms is one of their biggest pitfalls, with a large number of researchers and developers working on ways to speed up the process. This paper attempts to determine the influence of used storage, within a realistic environment – foregoing bloated datasets and models, on the training times. The research utilizes two models, on two different architectures, across four different storage options: Solid State Drive (SSD), Hard Disk Drive (HDD), RAMDISK and network accessed storage. The results show that there is not a large difference in training times, when observing realistic cases and suggests that a significant influence on training times is not exhibited by storage.

Keywords: Artificial Intelligence, Execution Timing, Machine Learning, Performance Profiling

1. introduction

Artificial intelligence algorithms have a growing range of applications, including medicine [1, 2], propulsion systems [3], robotics [4], economy [5] and others. AI has many branches, and one of the most commonly used is Machine Learning (ML). ML algorithms consist of training and testing stages, with the training stage of the algorithms being extremely computationally complex [6]. Many efforts have been made in the past in order to lower the training times, such as using pretrained models in the so-called transfer learning, exhibited by as shown by Sai et al. (2020) [7], Pathak et al. (2020) [8], Pesciullesi et al. (2020) [9] and others, with more examples being shown in a survey performed by Zhuang et al. (2020) [10]. Another way of speeding up the training procedure is using different computer architectures for training such as Graphic Processing Units (GPU) as shown Ridnik et al. (2021) [11] by or Tensor Processing Units (TPU), as shown by Wongpanich (2020) [12].
Another important feature of ML algorithms is that they develop data driven models. In the training stage, the algorithms will read and utilize the existing data in order to adjust the internal parameters of the models. This data needs to be loaded from the persistent memory into memory from which it will be read. For this reason, a question arises – is it possible to speed up the training process of ML algorithms, by using a higher speed storage? This paper will time the execution time of training processes of two distinct ML algorithms on different sets of data, in a realistic environment (using realistic datasets and algorithms) to determine the influence of data storage on the training times. Furthermore, two different computer architectures will be used for training – Central Processing Unit (CPU) and GPU, details of which are given in the “Methodology” section.

2. Methodology

In this section the overview of methodology used in the research will be given. First, the overview of used architectures and datasets is provided, followed by the description of used hardware – both the training devices and storage options, with the timing approach being described as the final part of this section.

2.1. Used Architectures

Two architectures have been used – Multilayer Perceptron (MLP) and AlexNet.
Both algorithms are implemented in Python 3.7 using Tensorflow 2.3 and built-in keras library [13].

2.1.1. MLP

MLP is a feed-forward neural network consisting of an input, output and one or more hidden layers. Each layer consists of neurons which act as sumators of the output values of the neurons of the previous layer, all of which are connected to it [14]. The architecture used in the presented research consists of four hidden layers with four neurons each, with the architecture shown in Figure 1.
The hyperparameters of the MLP are set as follows: SGD is used as solver, with learning rate being set to 1, with batch size of 32. The model is trained for a single epoch. Loss function used is cross entropy.

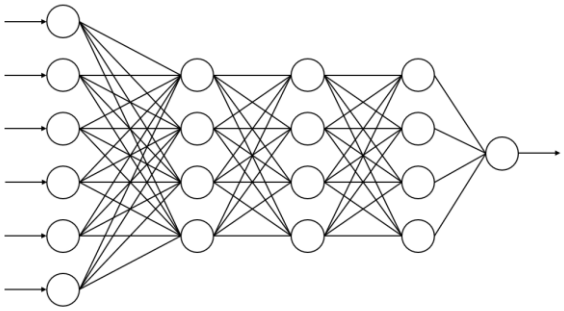


Figure 1. *Visualization of the used MLP architecture*

2.1.2. AlexNet

Alexnet is a convolutional neural network consisting of multiple 2-dimensional convolution layers through which the data passes sequentially – with a densely connected layer at the end, connecting to the output neuron. [15] The architecture of the AlexNet is given in Figure 2. The solver used is Adam, with batch size of 32 and the used model being trained for a total of 5 epochs each execution. Loss function used is, again, cross entropy.

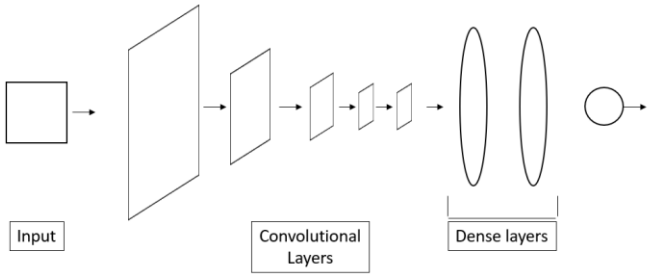


Figure 2. *Visualization of the used AlexNet architecture*

2.1.3. Datasets

Two datasets have been used. First consists of 1000 grayscale images, sized 224*225 pixels, and stored in NPY tensor format with the shape (1000, 224, 224, 1). The second is a numerical dataset consisting of 1000 data points, with 6 input values and a single output value. As the ultimate performance of the algorithms does not factor into the conclusion of this paper both datasets are randomly generated for the purposes of timing the used algorithms. The numerical dataset is used for MLP training, while the image dataset is used for AlexNet training.

2.2 Training Hardware

As previously mentioned the models were trained using GPU and CPU. The CPU used is Intel Xeon Gold 6240, with 24 Cores and 48 Threads [16]. The clock of the CPU has been set to a fixed value of 2.4 GHz. The GPU used for training is Nvidia RTX 6000, with 576 Tensor Cores , 4608 CUDA cores and 24GB of GDDR5 RAM, with nominal clock speeds [17]. Both of the training devices are implemented within the same server node with 768 GB of Registered ECC DDR4 RAM; implemented on the SuperMicro X11DPG-OT-CPU motherboard.

2.3. Used Storage

Four storage variants have been used. First is the SSD used as the boot drive of the server consisting of two drives connected in RAID 1, in M.2 format, connected using NVME protocol.. Second is HDD, consisting of sic 6 TB drives in RAID 5, connected via 6 Gbps SATA connection. Both HDD and SSD storage use BTRFS file system. Third is the Ramdisk, with the size of 10 GB, mounted within the ECC RAM of the node and created using ramdisk Linux utility. Finally, the network storage. In this case, the data is stored on a separate node connected with a 1 Gbps Ethernet connection, and internally mounted using CIFS utility. Due to the speed bottleneck being the Ethernet connection, no matter which media is used for remote storage, multiple variants of remote storage have not been tested.

2.4. Timing

The timing is performed using time built-in Linux system command [18]. This command is used by specifying it before the execution command, for example:

time python mlp.py

The time command will return three values:

- T_R – real time, the time passed between the execution and the return of the command. Sometimes referred to as “Wall time”, in a reference to this time being equal to user timing the execution using a clock.
- T_U – user time, time spent on command execution outside of the kernel, in User mode.
- T_S – system time. Time spent on command execution inside the kernel, in Kernel mode.

Real time is of no real consequence in timing the execution of the processes, as the training process can be interrupted by various system calls of higher priority, extending the real execution time [19]. To obtain the actual time the program has spent on the execution, the time spent on execution on commands in User and Kernel mode needs to be added [20]:

$$T_E = T_U + T_S, \tag{1}$$

where T_E is the actual execution processing time. In the Results and Discussion section, only this times will be presented.

3. Results and Discussion

The execution times recorded using the described methodology are given in Table 1. for GPU and Table 2. for CPU.

Table 1. Average *GPU Performance*

Model	**Storage**	$\overline{T_E}$ **(**σ**) [mm:ss.000] (N=10)**
MLP	SSD	00:48.563 (00:01.752)
	HDD	00:48.083 (00:03.170)
	Ramdisk	00:48.528 (00:01.140)
	Remote	00:46.733 (00:06.113)
AlexNet	SSD	00:39.973 (00:01.081)
	HDD	00:40.049 (00:01.150)
	Ramdisk	00:41.048 (00:01.135)
	Remote	00:40.783 (00:00.869)

Table 1 shows the average execution times over ten runs, with standard deviation noted for CPU. It can be easily noticed that execution time for AlexNet is significantly higher on the CPU.

Table 2 shows the execution times for both architectures on the GPU. It can be seen that the execution times are relatively similar; with AlexNet execution times being somewhat lower. This is caused by the fact that the operations in AlexNet (Convolutions) are more optimal to perform on the GPU architecture then the ones in MLP.

Table 2. Average *CPU Performance*

Device	**Storage**	$\overline{T_E}$ **(**σ**) [mm:ss.000] (N=10)**
MLP	SSD	00:29.432 (00:01.210)
	HDD	00:29.852 (00:00.919)
	Ramdisk	00:30.378 (00:01.301)
	Remote	00:29.506 (00:00.906)
AlexNet	SSD	14:10.753 (00:14.413)
	HDD	14:07.912 (00:14.228)
	Ramdisk	14:15.180 (00:12.434)
	Remote	14:14.639 (00:09.880)

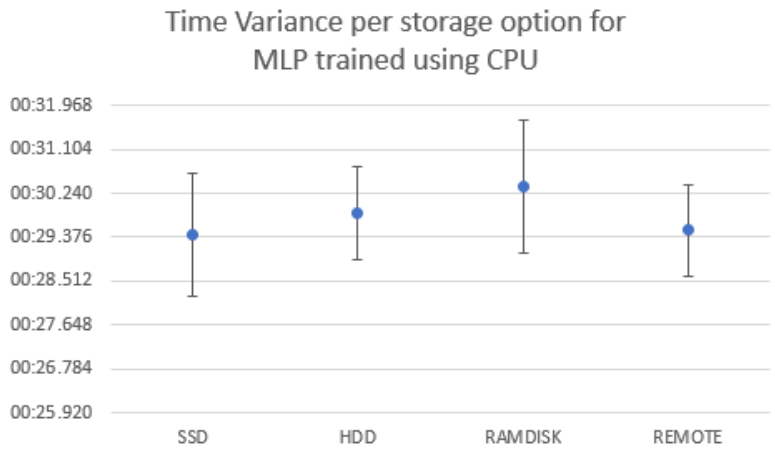


Figure 3. *Training time per storage variant for MLP on CPU*

Figure 3 shows that all the measured times fall within the margin of error of individual measurements, meaning that the influence of storage option is negligent in the observed case.

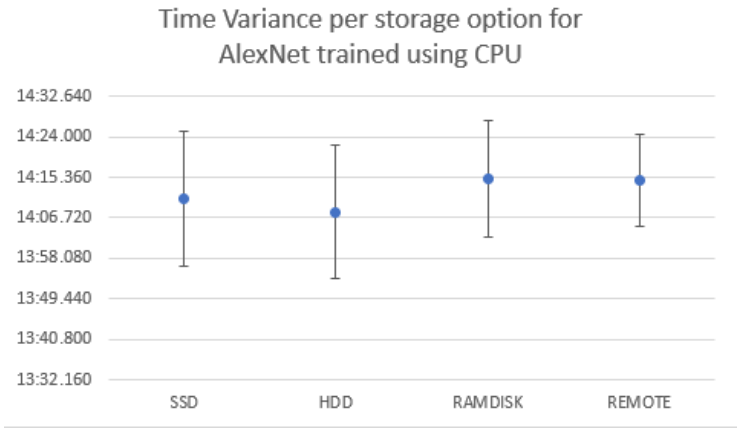


Figure 4. *Training time per storage variant for AlexNet on CPU*

Figure 4 shows the equivalent case as the previous one with measurements falling within the margin of error, determined using standard deviation of individual measurements.

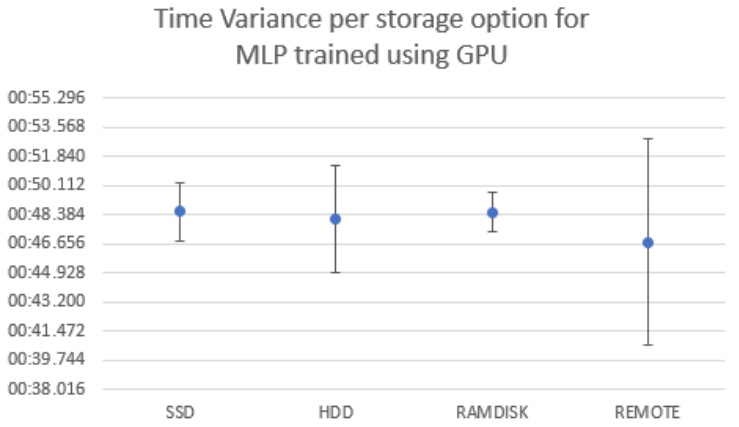


Figure 5. *Training time per storage variant for MLP on GPU*

Figure 5 shows the similar case, with only the remotely accessed data showing a statistically significant slowdown in comparison to the use of Ramdisk in this case. It should also be noted that this case shows a slowdown compared to the CPU times for MLP. The reason for that is easily explainable – the training times for CPU and GPU are similar, due to a small architecture being used, but loading times for GPU are slower as the data needs to be loaded into the RAM (except in the case of the Ramdisk) before being transferred to GPU onboard memory.

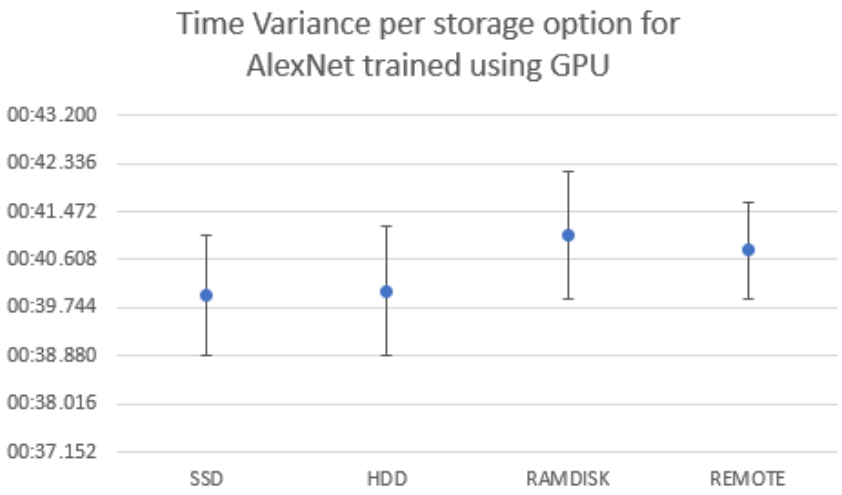


Figure 6. *Training time per storage variant for AlexNet on GPU*

Figure 6 shows the similar situation in regards to the training times. What should be noted is a significantly faster training times exhibited by the GPU in comparison to the same model on the CPU.

4. Conclusion

The observed data shows that storage choice has a negligent influence on the training times, and in realistic cases the data may be stored in whichever storage is readily available to the user. Much more significant difference in the training times are exhibited in the use of appropriate training architectures, which can be easily noted when comparing execution times of AlexNet on CPU and GPU.

Acknowledgments

This research has been (partly) supported by the CEEPUS network CIII-HR-0108, European Regional Development Fund under the grant K.01.1.1.01.0009 (DATACROSS), project CEKOM under the grant KK.01.2.2.03.0004, CEI project “COViDAi” (305.6019-20), and University of Rijeka scientific grant uniri-tehnic-18-275-1447.

References

[1] Lorencin, Ivan, et al. "Automatic Evaluation of the Lung Condition of COVID-19 Patients Using X-ray Images and Convolutional Neural Networks." Journal of Personalized Medicine 11.1 (2021): 28.

[2] Lorencin, Ivan, et al. "On Urinary Bladder Cancer Diagnosis: Utilization of Deep Convolutional Generative Adversarial Networks for Data Augmentation." Biology 10.3 (2021): 175.

[3] Baressi Šegota, Sandi, et al. "Improvement of Marine Steam Turbine Conventional Exergy Analysis by Neural Network Application." Journal of Marine Science and Engineering 8.11 (2020): 884.

[4] Baressi Šegota, Sandi, et al. "Path planning optimization of six-degree-of-freedom robotic manipulators using evolutionary algorithms." International Journal of Advanced Robotic Systems 17.2 (2020): 1729881420908076.
[5] Štifanić, Daniel, et al. "Impact of covid-19 on forecasting stock prices: an integration of stationary wavelet transform and bidirectional long short-term memory." Complexity 2020 (2020).
[6] Car, Zlatan, et al. "Modeling the spread of COVID-19 infection using a multilayer perceptron." Computational and mathematical methods in medicine 2020 (2020).
[7] Cai, Chenjing, et al. "Transfer learning for drug discovery." Journal of Medicinal Chemistry 63.16 (2020): 8683-8694.
[8] Pathak, Yadunath, et al. "Deep transfer learning based classification model for COVID-19 disease." Irbm (2020).
[9] Pesciullesi, Giorgio, et al. "Transfer learning enables the molecular transformer to predict regio-and stereoselective reactions on carbohydrates." Nature communications 11.1 (2020): 1-8.
[10] Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., ... & He, Q. (2020). A comprehensive survey on transfer learning. Proceedings of the IEEE, 109(1), 43-76.
[11] Ridnik, Tal, et al. "Tresnet: High performance gpu-dedicated architecture." Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2021.
[12] Wongpanich, Arissa. "Efficient Parallel Computing for Machine Learning at Scale." (2020).
[13] Abadi, Martín, et al. "Tensorflow: A system for large-scale machine learning." 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16). 2016.
[14] Lorencin, Ivan, et al. "Using multi-layer perceptron with Laplacian edge detector for bladder cancer diagnosis." Artificial Intelligence in Medicine 102 (2020): 101746.
[15] Yu, Wei, et al. "Visualizing and comparing AlexNet and VGG using deconvolutional layers." Proceedings of the 33 rd International Conference on Machine Learning. 2016.
[16] Intel Xeon Gold 6240R Processor https://ark.intel.com/content/www/us/en/ark/products/199343/intel-xeon-gold-6240r-processor-35-75m-cache-2-40-ghz.html, accessed on 15th April 2021
[17] NVIDIA Quadro RTX 6000 https://www.nvidia.com/en-us/design-visualization/quadro/rtx-6000/, accessed on 15th April 2021
[18] Time man page, https://man7.org/linux/man-pages/man1/time.1.html, accessed on 19th of April
[19] Hallinan, Christopher. Embedded Linux primer: a practical real-world approach. Pearson Education India, 2011.

ISBN: 978-953-8246-22-7

Medical data annotation and json to dataset conversion using LabelMe and Python

Karlo Severinski [1], Tajana Cvija [2]

[1] Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia, kseverinski@riteh.hr, tcvija@riteh.hr.
[*] Correspondence: kseverinski@riteh.hr;

Abstract: This paper will discuss the use of the LabelMe tool for annotation of radiological data. Annotations will be made of the prostate and the bladder that
will thus be prepared for the use of U-net semantic segmentation. The goal of this paper is to show how to prepare annotated data for processing using U-net and their conversion from jpeg format to json data and the conversion of json data into an annotation mask for later use in semantic segmentation. The annotation tool used is the already mentioned LabelMe as part of the anaconda package and the programming language Python as support for method implementation. The obtained results are presented in detail in the paper. The results provide the necessary data for later use in semantic segmentation.

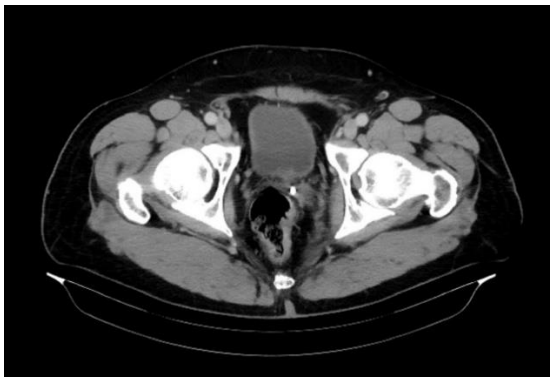
Keywords: annotation, artificial intelligence, dataset, json, LabelMe, Python, semantic segmentation, U-net

1. introduction

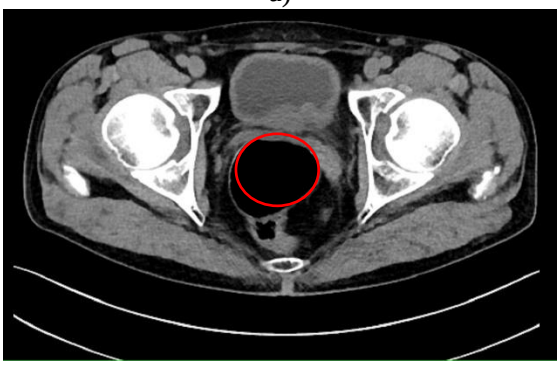
Since its first use in medical purpose in the 1960s, the concept of artificial intelligence has been especially appealing to health care, particularly radiology. With the development of ever more powerful computers from the 1990s to the present, various forms of artificial intelligence have found their way into different medical specialties [1]. Today, the area of computer vision is used mainly in the field of recognizing various cancers such as bladder cancer [2] or in the diagnosis of other diseases such as COVID-19 that may include a CT scan [3]. This is important because using such tools can help medical doctors with faster and more reliable diagnosis. Before processing such data, it is necessary to annotate the data on radiological images, such as CT images of the bladder. In this paper, LabelMe annotation tool was used for the annotation of the bladder and the prostate on the CT scan images. LabelMe is not only a software annotation tool that can be used online or offline on a local computer but also an online database for all kinds of data with already made labels for faster use [4, 5]. The LabelMe tool can be used both for picture and video annotation [6]. The custom dataset consists of 235 frontal CT scan images, 124 horizontal CT scan images and 54 sagittal plane CT scans. In this paper, instead of cancer, the bladder and the prostate are annotated to demonstrate the use of LabelMe and the conversion of this data into an annotation mask. After the annotation, the picture is first transferred to a JSON file format, which is generated with the LabelMe software [7]. JSON is designed to be a data exchange language which is human readable and easy for computers to parse and use [8]. JSON holds the data that is needed to reconstruct the label, the annotation mask, of the labelled body part or organ, in this case the bladder and the prostate [9]. The reconstruction is done by the Python command labelme_json_to_dataset which can be used as a batch command for a large number of annotated data. By the end of this process, ready-made data is obtained for use with U-net semantic segmentation [10, 11] and later with CNN [12].

2. Methodology

As mentioned in the introduction, CT scan data were used for this study, in the proportions given in the introduction. At the beginning, the data were sorted by person, i.e. the data in the first folder were a reference to person 1, in the second folder to person 2 and so on. The given CT data represent a mix of healthy people and people with different types of tumours. The types of tumours in the data are papillary tumours and bladder wall formation and bladder wall thickening. Tumour types are not so important because the paper refers only to the organs and not their deformities. An example of a horizontal CT scan, a healthy bladder compared to a bladder with wall thickening is shown in Figure 1.



a)



b)

Figure 1. a) Healthy bladder, b) Bladder thickening (red circle)

So, because of the assumption that the requirement for a
computer to later recognize both a healthy bladder and a healthy prostate and a bladder and a prostate with cancer, this paper will not touch on the recognition of tumours but organs as a whole, completely generalized (whether healthy or unhealthy).
The next step is to use the LabelMe software package for the organ annotation. LabelMe allows easy loading of the .jpeg format and its annotation using polygons. An example of such annotations is shown in Figure 2. All 413 CT scans were annotated in a similar manner.

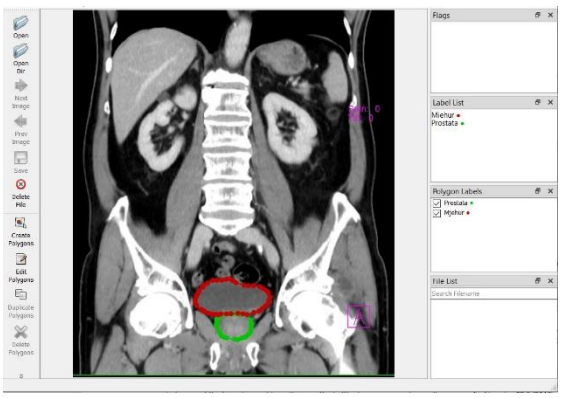


Figure 2. Annotation example using polygons tool, the red annotation represents the bladder and the green annotation represents the prostate.

As already mentioned, LabelMe saves annotations as a .json file and saves the data needed for conversion to a .png label in there. Converting data from .json to .png is possible, and in this paper, it is executed using the Pyton programming language. The command used for the conversion is labelme_json_to_dataset and, to speed up the conversion, it is executed as a batch command. The syntax is displayed in Figure 3.

```
for /1 %a in (0,1,range) do

    labelme_json_to_dataset /path/%a.json
```

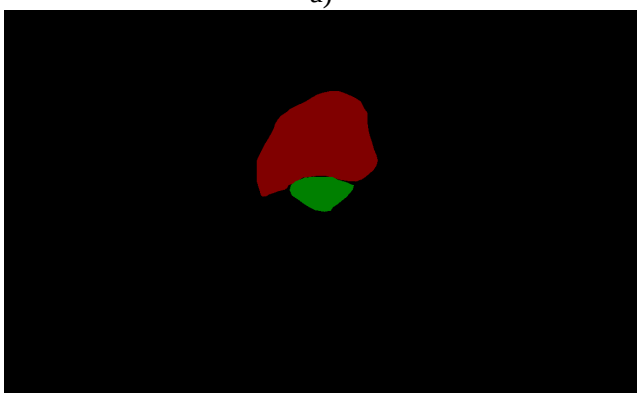
Figure 3. Batch conversion code

Where the "range" variable represents the number of .json files in the folder, the data is named with numbers from 1 to n, where then n = range. The variable "a" represents the current number of iterations of the loop. With this, the annotation of data and their conversion into labels is completed, and their preparation for further use in U-net semantic segmentation and CNN is complete.

3. Results and Discussion

The results obtained after the conversion are shown in Figure 4. for the horizontal plane, Figure 5. for the frontal plane and Figure 6. for the sagittal plane. The final output of the conversion is the original CT scan, then the label names in the text document, the png label and the label display on the original data.

a)

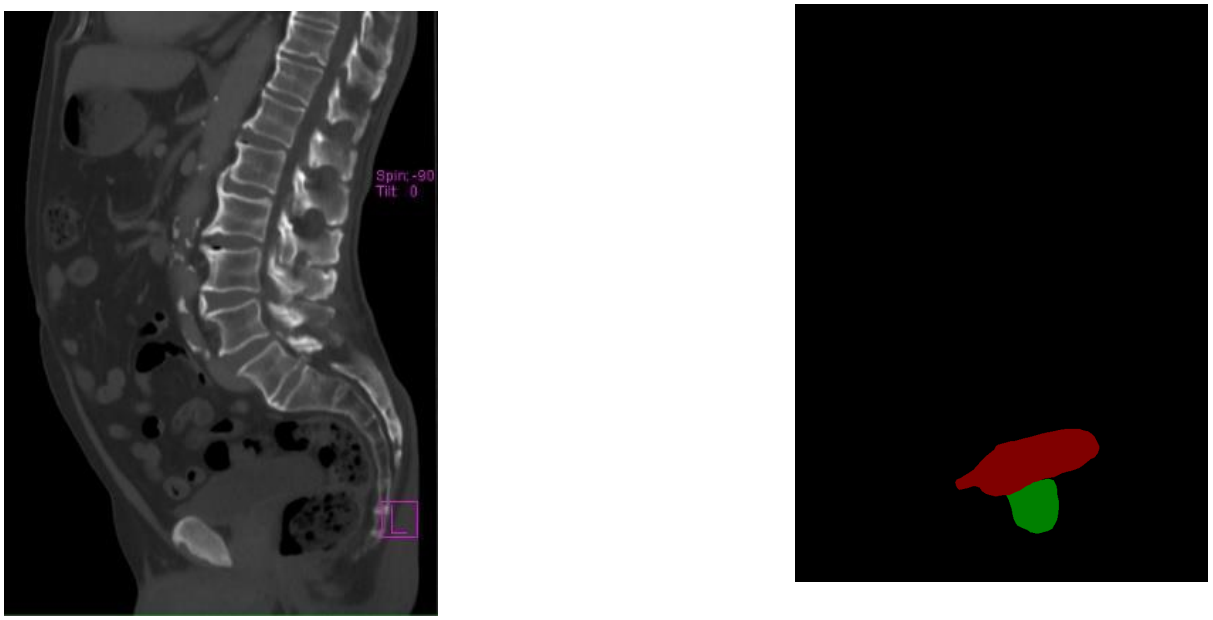


b)

Figure 4. a) Original CT scan image in the horizontal plane, b) The annotation mask (label) of the bladder (red) and the prostate (green) for the given image.

a) b)

Figure 5. a) Original CT scan image in the frontal plane, b) The annotation mask (label) of the bladder (red) and the prostate (green) for the given image



a) b)

Figure 6. a) Original CT scan image in the sagittal plane, b) The annotation mask (label) of the bladder (red) and the prostate (green) for the given image

A CT scan showing the annotation on the original image is also given, which is, as stated, also one of the output data. The view is given for the horizontal plane in the Figure 7.

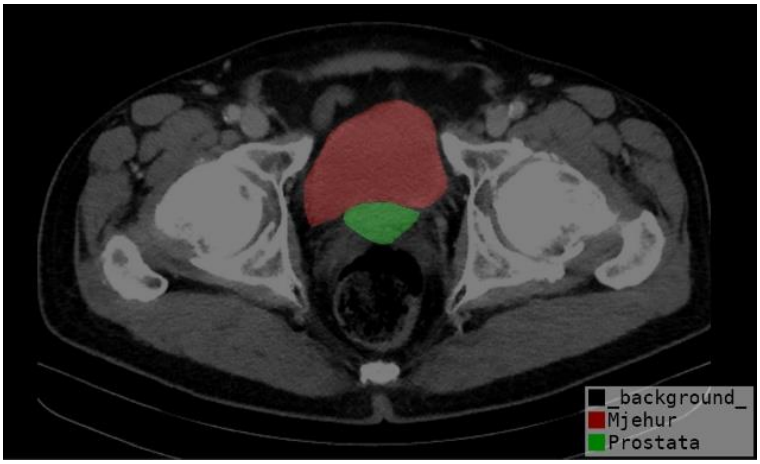


Figure 7. Original image with the annotation

From the given figures it can be seen that, by using the LabelMe software package and later Python, we can get a visually good annotation of the desired organs. All other annotations are similar. The smaller the difference between the lines of annotations and the actual representation of the organ in the original image, the
the better and more accurately will the computer determine it through artificial intelligence algorithms, that is, learn where which organ is. The responsibility for determining the annotation lies, of course, with the person, that is, the person who annotates, so the errors of that nature relate to the human factor. That is why when working with such data it is necessary to seek the opinion of a specialist from the related field in which the work is done, since they greatly help in the annotation.

4. Conclusion

Concluding on the converted data. As already stated and as can be seen this approach to data preparation for semantic segmentation is successful. The preparation process obtained satisfactory output data where the annotation of an individual organ is clearly recognized. The annotations shown on the original CTs were also used to check the annotation. The next step in using this data is U-net [13] semantic segmentation. It is now necessary to sort all input and output data of the patient into one independently, i.e. 6 folders, then load them and do the segmentation using Python code. Further, this data will be used with MobileNetV2 U-net segmentation method.

Acknowledgments
We want to thank the assistant Ivan Lorencin, mag. ing. el. who helped us tremendously in the making of this paper and was our mentor. We hope there will be a lot of joint future work and projects. We would also like to thank Professor Zlatan Car and the urology clinic of the Clinical Hospital Center of Rijeka, headed by Assoc. dr. sc. Josip Španjol for the provided data. We would also like to thank all those who organized this conference and enabled us students to participate with our work.

Reference

[1] Filipovic-Grcic, L., and F. Derke. "Artificial Intelligence in Radiology." RAD CASA-Medical Sciences 537 (2019): 46-47.
[2] Lorencin, I., Baressi Šegota, S., Anđelić, N., Mrzljak, V., Ćabov, T., Španjol, J., & Car, Z. (2021). On Urinary Bladder Cancer Diagnosis: Utilization of Deep Convolutional Generative Adversarial Networks for Data Augmentation. Biology, 10(3), 175.

[3] Lorencin, Ivan, et al. "Automatic Evaluation of the Lung Condition of COVID-19 Patients Using X-ray Images and Convolutional Neural Networks." Journal of Personalized Medicine 11.1 (2021): 28.
[4] Russell, Bryan C., et al. "LabelMe: a database and web-based tool for image annotation." International journal of computer vision 77.1-3 (2008): 157-173.
[5] Torralba, Antonio, Bryan C. Russell, and Jenny Yuen. "Labelme: Online image annotation and applications." Proceedings of the IEEE 98.8 (2010): 1467-1484.
[6] Yuen, Jenny, et al. "Labelme video: Building a video database with human annotations." 2009 IEEE 12th International Conference on Computer Vision. IEEE, 2009.
[7] Liu, Yun-Xia, et al. "Intelligent monitoring of indoor surveillance video based on deep learning." 2019 21st International Conference on Advanced Communication Technology (ICACT). IEEE, 2019.
[8] NURSEITOV, Nurzhan, et al. Comparison of JSON and XML data interchange formats: a case study. Caine, 2009, 9: 157-162.
[9] Simpson, Amber L., et al. "A large annotated medical image dataset for the development and evaluation of segmentation algorithms." arXiv preprint arXiv:1902.09063 (2019).
[10] Zhang, Ziang, et al. "DENSE-INception U-net for medical image segmentation." Computer methods and programs in biomedicine 192 (2020): 105395.
[11] Qin, Xiaofei, et al. "Match Feature U-Net: Dynamic Receptive Field Networks for Biomedical Image Segmentation." Symmetry 12.8 (2020): 1230.
[12] Shin, Hoo-Chang, et al. "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning." IEEE transactions on medical imaging 35.5 (2016): 1285-1298.
[13] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, 2015.

ISBN: 978-953-8246-22-7

Multiple medical images extraction from DICOM and conversion to JPG using Python

Tajana Cvija [1*], Karlo Severinski [2]

[1] Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia, tcvija@riteh.hr
[2] Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia, kseverinski@riteh.hr
[*] Correspodence: tcvija@riteh.hr

Abstract: The Digital Imaging and Communications in Medicine (DICOM) is the standard method for the transmission and storage of medical images and their associated information. However, it is not readable by every software. The goal of this study was to extract medical images from multiple .dcm files at once and convert them into the more popular .jpg format. The extraction and conversion were done by using programming language Python. Multiple files were successfully converted and stored in .jpg format of the satisfying quality and the output data could easily be displayed by any image viewing software. The output data can be henceforward used with the annotation tools and artificial intelligence to diagnose lung cancer or other diseases.

Keywords: DICOM, JPG, Python

1. introduction

In the last few decades the rapid develompent of digital technology has made it enter almost every aspect of people's lives so, naturally, it has become more and more important in medicine as well. Computers are used to read, display, store and process medical data obtained from various diagnostic devices as well as to help in diagnosing certain medical conditions (Computer-Aided Diagnosis) [1]. Because of many directions in development of medical imaging equipment, it was very important to make a standard for connection and information exchange between medical appliances [2]. That is why the American College of Radiology (ACR) and the National Electrical Manufacturers Association (NEMA) formed a joint committee that created a standard method for the transmission of medical images and their associated information [3]. The Digital Imaging and Communications in Medicine (DICOM) Standard specifies a non-proprietary data interchange protocol, digital image format, and file structure for biomedical images and image-related information [4] such as patient's name, age, sex etc. Although the advantages of images being stored as parts of DICOM files are obvious, they are not readable by every software unlike more popular formats such as JPG or PNG, especially for non-professionals. In this paper the conversion from DICOM to JPG file format was done in Python by using Pydicom and Pillow packages. Pydicom is an open source package for reading and writing DICOM files using the Python programming language [5] and Pillow (Python Imaging Library) is a library that adds image processing capabilities to Python interpreter. It is designed for fast access to data stored in a few basic pixel formats [6]. The used .dcm files were CT scans of lungs retrieved from The Cancer Imaging Archive webpage by using with the NBIA Data Receiver. Since converting files one by one can be time consuming, the emphasis was on creating a program that was going to convert multiple .dcm files to .jpg files automatically and save them. It is possible to extract much more information than just the images from DICOM files [7], but that subject is not going to be discussed in this paper.

2. Methodology

As previously mentioned, CT scans of lungs were used in this study. The scans were downloaded from The Cancer Imaging Archive webpage by using with the NBIA Data Receiver. The downloaded files were already sorted into separate folders with each folder referring to a different case. The .dcm files in folders contained the axial plane CT scans of lungs.
The next step was to create a program that was going to convert all the .dcm files inside one folder to .jpg files.
The two previosly mentioned libraries had to be installed prior to that, which was done by typing and executing pip install pydicom and pip install pillow commands in the Command Prompt window. The os and numpy libraries were already installed.

```
import os
import numpy as np
import pydicom
from PIL import Image
```

a)

```
def get_names(directory):

    names = []

    for root, dirnames, filenames in os.walk(directory):
        for filename in filenames:
            _, ext = os.path.splitext(filename)
            if ext in [".dcm"]:
                names.append(filename)

    return names
```

b)

```
names = get_names('Dicoms3')
```

c)

```
def convert(directory):
    im = pydicom.dcmread('Dicoms3/'+directory)
    im = im.pixel_array.astype(float)

    rescaled_image = (np.maximum(im,0)/im.max())*255
    final_image = np.uint8(rescaled_image)

    final_image = Image.fromarray(final_image)
    return final_image
```

d)

```
for name in names:
    image = convert(name)
    image.save(name+'.jpg')
```

e)

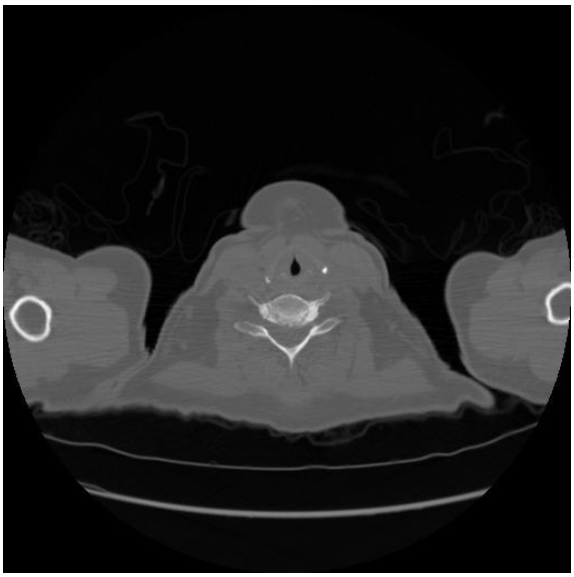
Figure 1. *Multiple files automatic .dcm to .jpg conversion code. a) importing the libraries, b) .dcm files listing function definition, c) applying b) on the specified directory, d) .dcm to .jpg conversion function definition, e) multiple files conversion loop*

The importing of the named packages is done as shown in the Figure 1. a). After that, two functions were defined.
The first function scans all the files in one specified directory and stores their names if they are .dcm type. Its definition is given in Figure 1. b). It is then applied on a specific folder that contains multiple .dcm files as shown in the Figure 1. c).
The definition of the second function is given in the Figure 1. d). The function reads a specified .dcm file inside a specified folder by using pydicom.dcmread command. Then pixels' information of the medical image are stored in an array of float type to avoid overflow or underflow losses [8] by using pixel_array.astype(float) command. Since .dcm files typically contain much more grayscale information than .jpg files do [9], the rescaling had to be done for the images to display the information correctly. The rescaled array was then converted to image by Image.fromarray command.

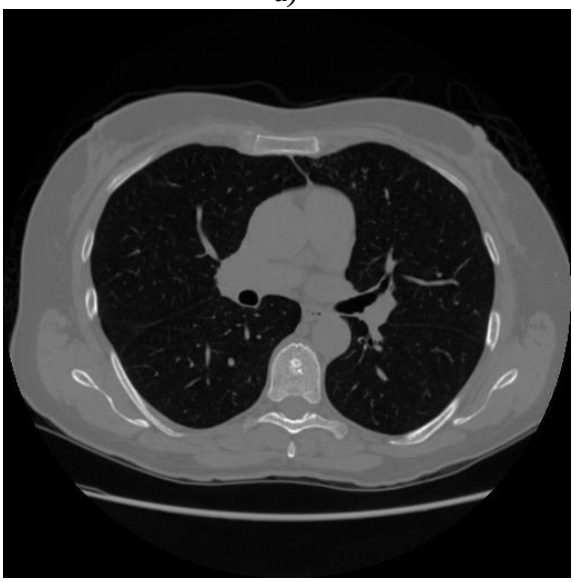
Lastly, the loop that applies the conversion on multiple files and stores them in .jpg format is given in the Figure 1. e).
After executing the whole code, multiple .dcm files inside one folder were automatically converted and stored as .jpg images.

3. Results and Discussion

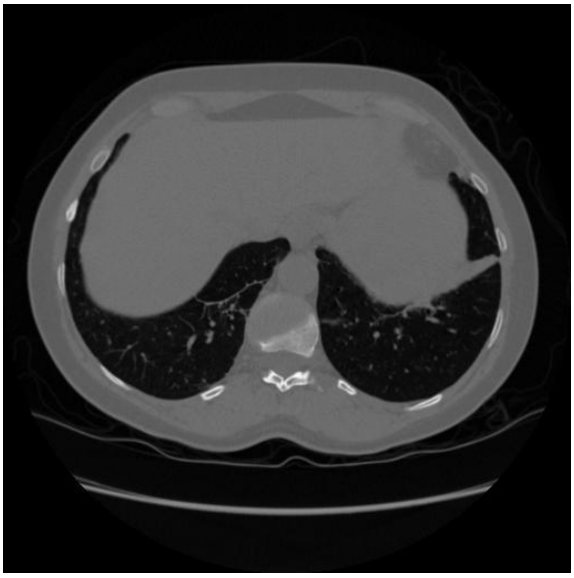
The results obtained from conversion are shown in the Figure 2. The images show different slices of the same CT scan. Figure 2. a) shows the CT scan of the beginning of lungs (upper part). Figure 2. b) shows the CT scan of the middle part of the lungs and Figure 2. c) shows the CT scan of the end of the lungs (lower part) [10].
From the given figures we can see that the Python programming language can be used for medical image data extraction from .dcm files and their conversion to more popular format such as .jpg. The big advantage of such format is that it is easily read and displayed by almost every image viewing software. However, for medical study purposes it cannot replace .dcm files, which store many useful information since they are much more than image data.



a)



b)



c)

Figure 2. *CT scan of lungs converted to .jpg format a) upper part of the lungs, b) middle part of the lungs, c) lower part of the lungs*

4. Conclusion

Concluding on the converted data. The extraction of medical image data from multiple .dcm files and their conversion to .jpg format by using Python has been successful. The execution of the written code obtained satisfactory output data which can be henceforward used with the annotation tools and artificial intelligence to diagnose lung cancer or other diseases.

Acknowledgments

We would firstly like to thank the assistant Ivan Lorencin, mag. ing. el., who was our mentor during the making of this paper. We hope there will be a lot of joint work and projects in the time to come. We also thank Professor Zlatan Car as well as all those who made this conference and our participation in it possible.

Reference

[1] F J Gilbert, H Lemke, "Computer-aided diagnosis ", *The British Journal of Radiology Volume 78, Issue suppl_1*, January 2014, Pages S1–S2 https://doi.org/10.1259/bjr/23717382

[2] M. Mustra, K. Delac and M. Grgic, "Overview of the DICOM standard," *2008 50th International Symposium ELMAR*, 2008, pp. 39-44. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4747434&isnumber=4747417

[3] Digital Imaging and Communications in Medicine (DICOM), NEMA Publications, "DICOM strategic document", 2016, http://medical.nema.org/dicom/geninfo/Strategy.pdf

[4] W. Dean Bidgood, Jr, MD, MS, Steven C. Horii, MD, Fred W. Prior, PhD, Donald E. Van Syckle, "Understanding and Using DICOM, the Data Interchange Standard for Biomedical Imaging", *Journal of the American Medical Informatics Association*, Volume 4, Issue 3, May 1997, pp. 199–212, https://doi.org/10.1136/jamia.1997.0040199

[5] D. Mason, "SU-E-T-33: Pydicom: An Open Source DICOM Library", *The international Journal of Medical Physics Research and Practice*, Volume 38, Issue 6, Part 10, June 2011, pp. 3493-3493 https://doi.org/10.1118/1.3611983

[6] A Clark, "Pillow (PIL Fork) Documentation", 2015 https://www.realmoon.net/wordpress/wp-content/uploads/2019/07/pillow.pdf

[7] L. Stanescu, D. D. Burdescu and C. Stoica-Spahiu, "Application For Complex Querying Of The Databases Obtained By Extracting Data From DICOM Files," 2006 2nd International Conference on Information & Communication Technologies, 2006, pp. 1008-1013, https://doi.org/10.1109/ICTTA.2006.1684512

[8] J. K. Mandal Debashis Das, "Steganography Using Adaptive Pixel Value Differencing (APVD) of Gray Images Through Exclusion of Overflow/Underflow", The Second International Conference on Computer Science, engineering and Applications (CCSEA-2012), Delhi, India. May 2012 https://arxiv.org/ftp/arxiv/papers/1205/1205.6775.pdf

[9] Peijiang Chen, "Study on Medical Image Processing Technologies Based on DICOM", School of Automobile, Linyi University, Linyi, Shandong, China, *Journal of Computers*, Volume 7, Issue 10, October 2012, Pages 2354-2361 http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.249.1518&rep=rep1&type=pdf#page=8

[10] Nisar Ahmed Memon, Anwar Majid Mirza, S.A.M. Gilani, "Segmentation of Lungs from CT Scan Images for Early Diagnosis of Lung Cancer ", Proceeding of World Academy of Science, Endineering and Technology, Volume 14, August 2006, Pages 228-233 https://www.researchgate.net/publication/228981181_Segmentation_of_lungs_from_CT_scan_images_for_early_diagnosis_of_lung_cancer

ISBN: 978-953-8246-22-7

Energy and exergy analysis of a nuclear power plant

Davor Poljančić, Vedran Mrzljak

Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia
Email: dpoljancic@riteh.hr, vmrzljak@riteh.hr

Abstract: This paper presents a calculation of energy and exergy efficiency with energy and exergy losses of the entire nuclear power plant and all of its constituent components. The main idea of the performed analysis is to be a baseline for further improvements and optimizations of the entire nuclear power plant or any of its components. Assumed energy efficiency of the nuclear reactor equal to 90% gives as a result that nuclear reactor generated heat of 3335,73 MW, while the cumulative power of two turbines (high pressure turbine – HPT and low pressure turbine – LPT) was 874,27 MW. Using these data, it is possible to calculate the energy and exergy efficiency of the whole power plant, both of them are around 27%, which is acceptable. However, the performed analysis show which components can be improved – the future work will be based firstly on these components.

Keywords: Efficiency, Energy, Exergy, Nuclear power plant, Steam turbine

1. Introduction

A nuclear power plant is a facility whose primary function is to generate electric energy from the energy contained in the nuclear fuel [1, 2]. The schematic representation of the nuclear power plant is shown in Figure 1, showing all components of the plant with the fluid operating states (operating points) before and after each component [3]. Entry data were given in the form of pressure, temperature, and specific enthalpy for every state of the fluid. The main difference between a nuclear power plant and a classic thermal power plant is in different fuel used for generating heat [4, 5]. Nuclear plants use different isotopes of uranium, which then go through a process of fission, where the atom is broken into two or more pieces which generate an immense amount of energy. These processes take place in the reactor and are not thoroughly described in this work. The steam generator is using produced energy from uranium to generate steam, which is then used to generate useful work in the turbines [6]. After the first turbine (high pressure turbine – HPT), the water droplets in steam are separated in the moisture separator (M.S.), and before the steam enters the second turbine (low pressure turbine – LPT), it's reheated in reheater (R.H.), Figure 1. After the second turbine (LPT), the remaining steam goes in the main steam condenser where the condensation occurs [7]. Water from the main steam condenser is then again heated in a series of heat exchangers (regenerative heaters) [8, 9] by the steam separated from the turbines, moisture separator, and the reheater. Heat exchangers are connected with a series of pumps, which are used to return water to the steam generator [10]. Oxygen and other gasses dissolved in the water are removed in the deaerator (DEA), Figure 1.
In this research is performed energy and exergy analysis of each component (from Figure 1), as well as of the entire nuclear power plant.

2. Energy and exergy equations

Energy is defined by the first law of thermodynamics and it is independent of ambient conditions, while the exergy is defined by the second law of thermodynamics and is heavily influenced by ambient conditions [11, 12].

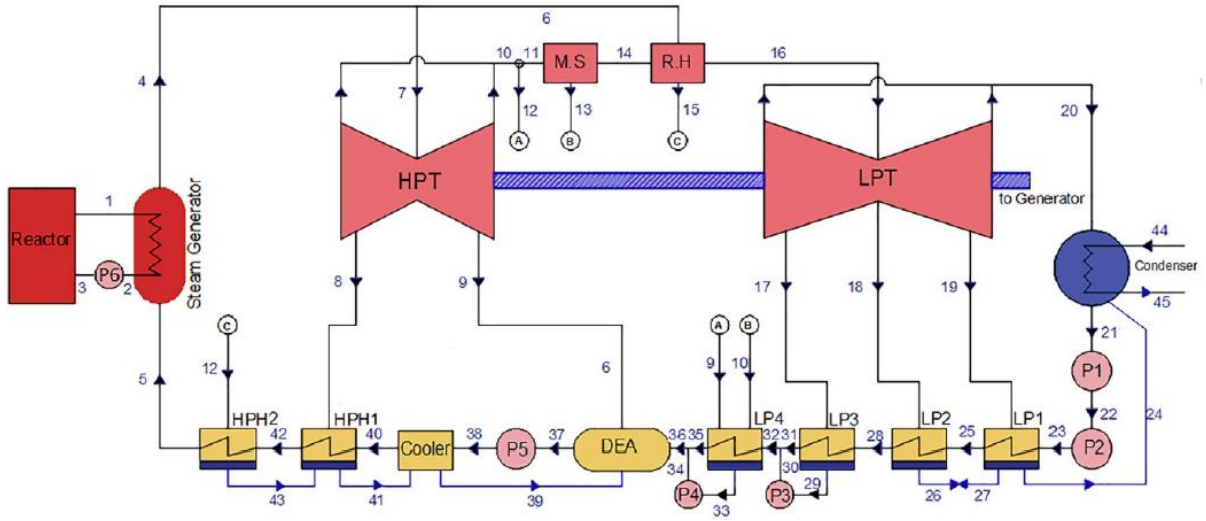


Figure 1. *Schematic representation of the nuclear power plant [3]*

Energy analysis

The mass flow rate before and after each control volume must be the same, assuming no leakage throughout control volume (Eq. 1). Energy balance is shown in Eq. 2, with disregarded potential and kinetic energy [13, 14]:

$$\sum \dot{m}_{\text{in}} = \sum \dot{m}_{\text{out}}, \tag{1}$$

$$\dot{Q} - P = \sum \dot{m}_{\text{out}} \cdot h_{\text{out}} - \sum \dot{m}_{\text{in}} \cdot h_{\text{in}}. \tag{2}$$

In the above equations, $\dot{m}$ is mass flow rate in (kg/s), $\dot{Q}$ is heat energy flux in (W), P is power in (W) and h is specific enthalpy in (J/kg). Indexes in and out denotes input and output streams. The energy flow of any fluid stream [15] can be calculated as shown in Eq. 3:

$$\dot{E}_{\text{eng}} = \dot{m} \cdot h. \tag{3}$$

The overall formula for energy efficiency is displayed by Eq. 4:

$$\eta_{\text{eng}} = \frac{\text{Energy output}}{\text{Energy input}}. \tag{4}$$

Exergy analysis

Exergy balance is defined by Eq. 5 with disregarded potential and kinetic energy [16]:

$$\dot{X}_{\text{heat}} - P = \sum \dot{m}_{\text{out}} \cdot \varepsilon_{\text{out}} - \sum \dot{m}_{\text{in}} \cdot \varepsilon_{\text{in}}. \tag{5}$$

Two elements of Eq. 5 should be additionally defined. Specific exergy (ε) is defined as a universal measure of work potential or quality of different forms of energy in relation to a given environment [17]. The formula for specific exergy is defined by Eq. 6, where h_0, s_0 and T_0 are specific enthalpy, specific entropy in (kJ/kg·K), and temperature in (K) of the ambient [18]:

$$\varepsilon = (h - h_0) - T_0 \cdot (s - s_0). \tag{6}$$

The second element of Eq. 5 ($\dot{X}_{heat}$) is the net exergy transfer by heat at the temperature (T) [19] and its formula is defined by Eq. 7:

$$\dot{X}_{\text{heat}} = \sum(1 - \frac{T_0}{T}) \cdot \dot{Q}. \quad (7)$$

The exergy flow of any fluid stream can be defined by Eq. 8:

$$\dot{E}_{\text{exg}} = \dot{m} \cdot \varepsilon. \quad (8)$$

The general formula for exergy efficiency is shown in Eq. 9:

$$\eta_{\text{exg}} = \frac{\text{Exergy output}}{\text{Exergy input}}. \quad (9)$$

There were some exceptions where variations of formulas for efficiency were used. Since it is unknown how much energy is produced in the reactor, energy efficiency of the steam generator was assumed to be 90% and Eq. 10 was used to calculate heat flow from the reactor:

$$\dot{Q} = \frac{\dot{m}_1 \cdot h_1 - \dot{m}_3 \cdot h_3}{\eta_{\text{eng}}}. \quad (10)$$

Similarly, energy efficiencies for pumps were assumed from 75% to 95% with a 5% step in between for the pump used power calculation.

Turbine energy efficiency was calculated using real (polytropic) and ideal (isentropic) expansion points and steam extraction at a certain pressure points. Both turbines have two cylinders, and the mass flow rate which enters into each cylinder is separated into two equal flows. Similarly, polytropic expansion points were given in the entry data for every pressure point of fluid separation, while the isentropic points were read from the specific enthalpy-specific entropy (*h–s*) diagram with the same specific entropy as at the point on the turbine entrance. Schematic presentation of left part from the low pressure turbine cylinder (LPT) along with operating points required for the analysis is shown in the left part of Figure 2. In the right part of Figure 2 are presented ideal (isentropic) and real (polytropic) steam expansion processes through left LPT part. The same procedure in the energy analysis as for the LPT left part is used for the HPT cylinder parts as well as for the LPT right part. In comparison to energy analysis of each turbine cylinder, exergy analysis requires only real (polytropic) expansion process. Equations for the entire analysis are taken from the available literature [20-22].

All the data required for the analysis are found in [3]. The ambient temperature and pressure required for the exergy analysis are 25 °C and 1 bar. Water and steam specific enthalpies, specific entropies and specific exergies are calculated by using NIST-REFPROP software [23].

3. Results and Discussion

3.1. Turbines

Since turbines are the most important components of a power plant, the turbine powers with its energy and exergy losses are shown in Figure 3. Power is calculated for each part of each cylinder with its losses and then summed up for both turbines. The ideal (isentropic) power of the low pressure turbine (LPT) is equal to 670,65 MW, while the real (polytropic) power is equal to 482,24 MW. The difference between the two powers is equal to energy losses. The same is valid for the high pressure turbine (HPT). High pressure turbine has a real (polytropic) power of 392,03 MW, and the total turbine power is 874,27 MW, which is used to calculate the efficiency of the whole process.

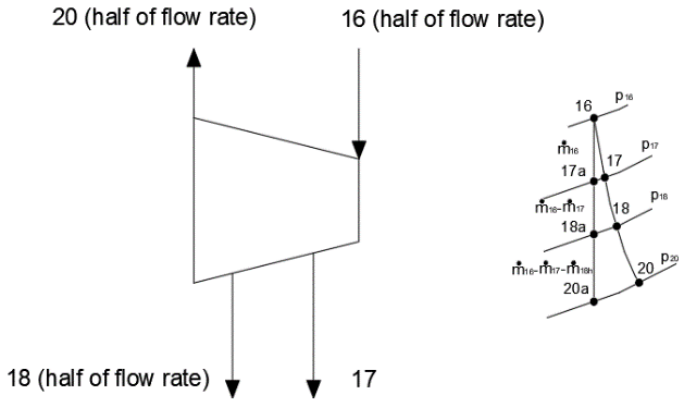


Figure 2. *Mass flows and expansion processes of the low pressure turbine (LPT) - left half*

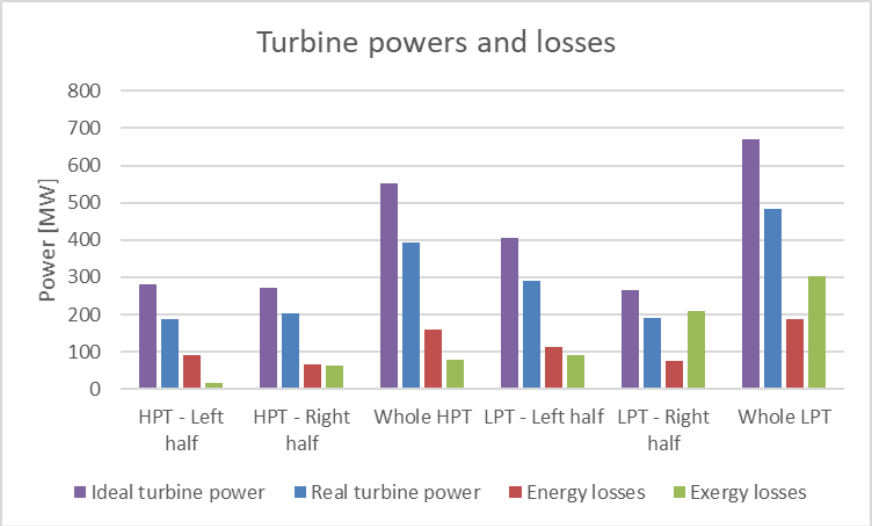


Figure 3. *Turbine powers, energy and exergy losses*

The energy efficiency of the whole high pressure turbine (HPT) is equal to 71,10%, while the exergy efficiency is 84,45%. The whole low pressure turbine (LPT) has an energy efficiency of 71,9% and the exergy efficiency of 61,8% as it can be seen in Figure 4. Along with the mentioned, in Figure 4 can also be seen energy and exergy efficiencies of each part (each half) of each turbine cylinder.

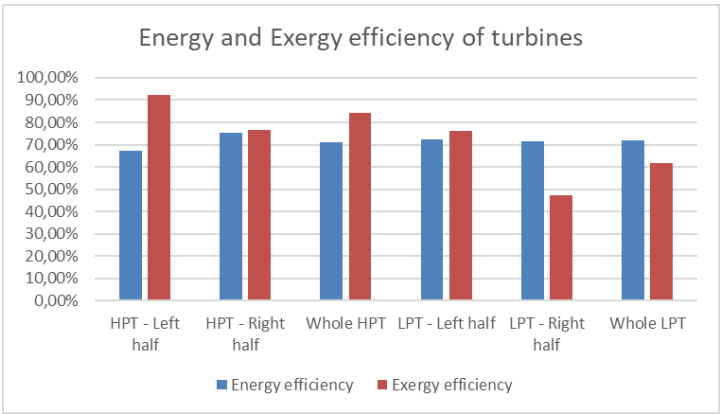


Figure 4. *Turbine energy and exergy efficiency*

3.2. Pumps

The numeration of pumps is performed according to Figure 1. Also, it was mentioned earlier how the calculation of losses and exergy efficiency for each pump was made. The plant has a total of 6 pumps with unknown powers. Figure 5 shows how exergy efficiency for different pumps is changing with different energy efficiencies. Pump 5 had the highest calculated power with different energy efficiencies, around 20 MW, while pump 3 had the lowest power, around 90 kW. Also, the same pumps had the highest and lowest energy losses. Pump 2 had the lowest exergy efficiency, around 40%, and pump 1 had the highest exergy efficiency, around 85%.

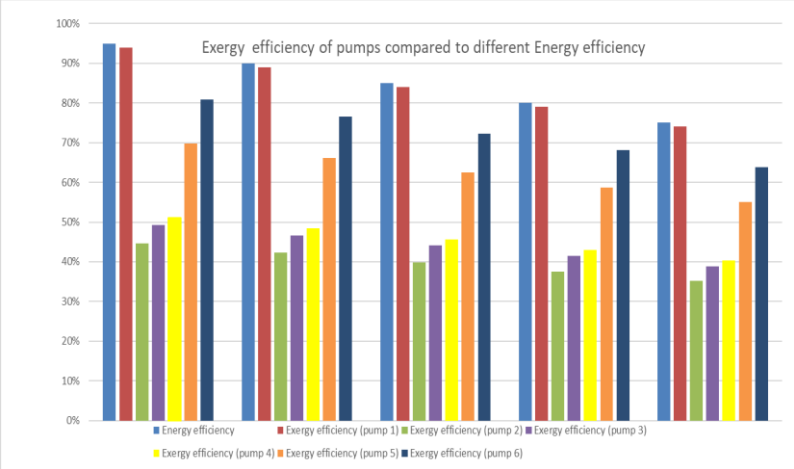


Figure 5. *Exergy efficiency of pumps compared to different energy efficiency*

3.3. Heat exchangers (regenerative heaters)

In the observed nuclear power plant, heat exchangers are used for heating the water from the condenser up to the steam generator. In the power plant, there are 6 closed heat exchangers – four low pressure ones (LP) and two high pressure heaters (HP), Figure 6. Heat exchangers are using steam to heat water (markings are in relation to Figure 1). Figure 6 shows the energy and exergy efficiency of all heat exchangers. As it can be seen, HP1 (the first high pressure heat exchanger) has an energy efficiency of almost 100% so there are no energy losses. Other heat exchangers have energy efficiencies from 84% to 97%. Exergy efficiency varies from 60% to 90% for different heat exchangers. HP2 has the highest energy losses due to the lowest energy efficiency. HP2 also has the biggest exergy losses, while HP1 has the lowest exergy losses.

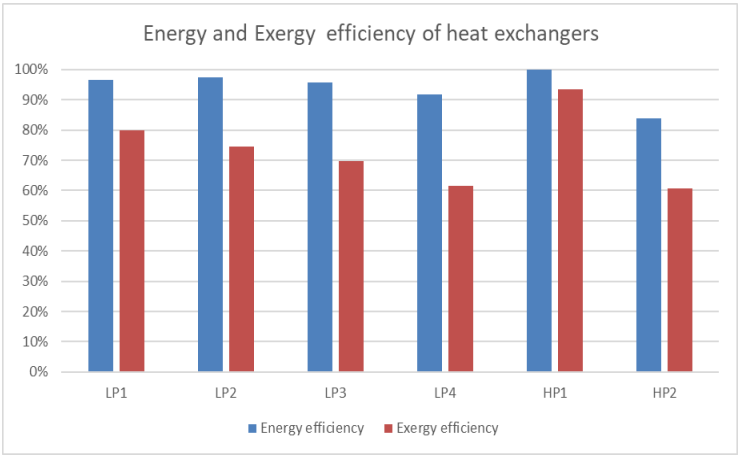


Figure 6. Energy and *exergy efficiency of heat exchangers*

3.4. Other components

Another important component of the power plant is the condenser. The condenser is a component that condenses the remaining steam from the turbines into the water. The condenser has a relatively high energy efficiency of 94%, but its energy losses are very high and equal to 128,76 MW. High energy losses in condenser are usual for any power plant, because of heat loss from turning the steam into water. Its exergy efficiency is equal to 78% and exergy losses are around 22 MW.
Deaerator, reheater, moisture separator and steam generator are components where energy efficiency is almost 100%, so the energy losses are minimal. Exergy efficiency on the other hand is not equal to 100%, it varies from 78% to 98% percent for the specified components.

3.5. The whole process

As mentioned earlier, generated heat in the reactor is equal to 3335,73 MW while the cumulative power of the turbines is 874,27 MW. Using the formula for energy efficiency which says that the efficiency of the process is equal to cumulative produced mechanical power over generated heat, energy efficiency is equal to 26,2%. The formula for exergy efficiency is similar, but generated heat in the formula is multiplied by a factor of 0,94 (0,94 is exergy factor for heat transfer). The exergy efficiency of the process is 27,9%. Energy losses of the whole plant are equal to 2461,46 MW while exergy losses of the whole plant are 2261,32 MW.

4. Conclusion

Nuclear power plants were often a taboo topic, mostly because of human errors in the past. Those errors caused catastrophic events around the globe. But, is nuclear power as scary and dangerous as people often think it is? The answer is no, nuclear power plants are one of the safest facilities that men have designed because of strict safety rules and over-engineering its most dangerous components. The main difference between a nuclear and a thermal plant can be found in the used fuel, which has its advantages and disadvantages (uranium over conventional fuels). The main con of uranium fuel is its storage, as it is still radioactive after use in the plant. But the fuel is not carbon-based and it is not burning, so it has no effect on global warming, which is the biggest problem of thermal power plants. Nuclear power plants around the globe are being shut down, but instead of converting to thermal power plants only, it is necessary to build “green” renewable energy power plants.
The idea of this work was to calculate the efficiency of all components of the power plant using energy and exergy analyses and the results are satisfying. Energy and exergy efficiencies for all components are in order and within reasonable borders. The efficiencies of the whole process are within normal borders too. The idea of this work was to have a better look at all components of the power plant, and an insight into which component has the greatest losses. Future work will be based on the improving the plant efficiency and minimizing plant losses.

Reference

[1] Naserbegi, A., et al. "A novel exergy optimization of Bushehr nuclear power plant by gravitational search algorithm (GSA)." *Energy* 148 (2018): 373-385. https://doi.org/10.1016/j.energy.2018.01.119
[2] Talebi, Saeed, and Nima Norouzi. "Entropy and exergy analysis and optimization of the VVER nuclear power plant with a capacity of 1000 MW using the firefly optimization algorithm." *Nuclear Engineering and Technology* 52.12 (2020): 2928-2938. https://doi.org/10.1016/j.net.2020.05.011
[3] Ebrahimgol, H., et al. "A novel approach in exergy optimization of a WWER1000 nuclear power plant using whale optimization algorithm." *Annals of Nuclear Energy* 145 (2020): 107540. https://doi.org/10.1016/j.anucene.2020.107540

[4] Seyyedi, S. M., M. Hashemi-Tilehnoee, and M. A. Rosen. "Exergy and exergoeconomic analyses of a novel integration of a 1000 MW pressurized water reactor power plant and a gas turbine cycle through a superheater." *Annals of Nuclear Energy* 115 (2018): 161-172. https://doi.org/10.1016/j.anucene.2018.01.028

[5] Elhelw, Mohamed, and Kareem Saad Al Dahma. "Utilizing exergy analysis in studying the performance of steam power plant at two different operation mode." *Applied Thermal Engineering* 150 (2019): 285-293. https://doi.org/10.1016/j.applthermaleng.2019.01.003

[6] Baressi Šegota, Sandi, et al. "Improvement of Marine Steam Turbine Conventional Exergy Analysis by Neural Network Application." *Journal of Marine Science and Engineering* 8.11 (2020): 884. https://doi.org/10.3390/jmse8110884

[7] Škopac, Lana, Vedran Medica-Viola, and Vedran Mrzljak. "Selection Maps of Explicit Colebrook Approximations according to Calculation Time and Precision." *Heat Transfer Engineering* (2020): 1-15. https://doi.org/10.1080/01457632.2020.1744248

[8] Nandini, Manne, Yendaluru Raja Sekhar, and Ganumukkala Subramanyam. "Energy analysis and water conservation measures by water audit at thermal power stations." *Sustainable Water Resources Management* 7.1 (2021): 1-24. https://doi.org/10.1007/s40899-020-00487-4

[9] Mrzljak, Vedran, et al. "Thermodynamic Analysis of a Condensate Heating System from a Marine Steam Propulsion Plant with Steam Reheating." *Journal of Marine Science and Application* (2021): 1-11. https://doi.org/10.1007/s11804-021-00191-5

[10] Chahartaghi, Mahmood, and Amirhossein Nikzad. "Exergy, environmental, and performance evaluations of a solar water pump system." *Sustainable Energy Technologies and Assessments* 43 (2021): 100933. https://doi.org/10.1016/j.seta.2020.100933

[11] Ezzat, M. F., and I. Dincer. "Energy and exergy analyses of a novel ammonia combined power plant operating with gas turbine and solid oxide fuel cell systems." *Energy* 194 (2020): 116750. https://doi.org/10.1016/j.energy.2019.116750

[12] Mrzljak, Vedran, et al. "Energy and exergy analyses of forced draft fan for marine steam propulsion system during load change." Journal of Marine Science and Engineering 7.11 (2019): 381. https://doi.org/10.3390/jmse7110381

[13] Aljundi, Isam H. "Energy and exergy analysis of a steam power plant in Jordan." Applied thermal engineering 29.2-3 (2009): 324-328. https://doi.org/10.1016/j.applthermaleng.2008.02.029

[14] Anđelić, Nikola, et al. "Comparison of Exergy and Various Energy Analysis Methods for a Main Marine Steam Turbine at Different Loads." Pomorski zbornik 59.1 (2020): 9-34. https://doi.org/18048/2020.59.01.

[15] Mrzljak, Vedran, Igor Poljak, and Vedran Medica-Viola. "Dual fuel consumption and efficiency of marine steam generators for the propulsion of LNG carrier." Applied Thermal Engineering 119 (2017): 331-346. https://doi.org/10.1016/j.applthermaleng.2017.03.078

[16] Lorencin, Ivan, et al. "Exergy analysis of marine steam turbine labyrinth (gland) seals." Pomorstvo 33.1 (2019): 76-83. https://doi.org/10.31217/p.33.1.8

[17] Kanoğlu, Mehmet, Yunus A. Çengel, and İbrahim Dinçer. Efficiency evaluation of energy systems. Springer Science & Business Media, 2012.

[18] Tan, Hongbo, et al. "A new boil-off gas re-liquefaction system for LNG carriers based on dual mixed refrigerant cycle." Cryogenics 92 (2018): 84-92.

https://doi.org/10.1016/j.cryogenics.2018.04.009

[19] Mrzljak, Vedran, Igor Poljak, and Jasna Prpić-Oršić. "Exergy analysis of the main propulsion steam turbine from marine propulsion plant." Brodogradnja: Teorija i praksa brodogradnje i pomorske tehnike 70.1 (2019): 59-77. https://doi.org/10.21278/brod70105

[20] Adibhatla, Sairam, and S. C. Kaushik. "Energy and exergy analysis of a super critical thermal power plant at various load conditions under constant and pure sliding pressure operation." Applied thermal engineering 73.1 (2014): 51-65. https://doi.org/10.1016/j.applthermaleng.2014.07.030

[21] Erdem, Hasan Huseyin, et al. "Comparative energetic and exergetic performance analyses for coal-fired thermal power plants in Turkey." International Journal of Thermal Sciences 48.11 (2009): 2179-2186. https://doi.org/10.1016/j.ijthermalsci.2009.03.007

[22] Medica-Viola, Vedran, et al. "Comparison of conventional and heat balance based energy analyses of steam turbine." Pomorstvo 34.1 (2020): 74-85. https://doi.org/10.31217/p.34.1.9

[23] Lemmon, E. W., M. L. Huber, and M. O. McLinden. "NIST standard reference database 23, NIST reference fluid thermodynamic and transport properties, REFPROP, version 9.0." Standard Reference Data Program (2010).

ISBN: 978-953-8246-22-7

Cardiotocography and ultrasound in obstetrical practice

Ariana Rabac [1,2,*]

[1] Clinical Hospital Centre Rijeka, ariana.rabac@gmail.com
[2] University of Rijeka, Faculty of Healthcare

Abstract: Monitoring the fetus during pregnancy and childbirth is a great challenge for the obstetric team. Proper monitoring of the fetus during pregnancy and childbirth is very important in order to detect deviations and complications in a timely manner and thus for a better perinatal outcome for the newborn and the mother. In this paper, a brief description of cardiotocography and ultrasound, as two main methods for fetus monitoring is provided.

Keywords: Cardiotocography, Delivery, Fetus, Pregnancy, Ultrasound

1. Introduction

Monitoring the fetus during pregnancy and childbirth is a great challenge for the obstetric team. Proper monitoring of the fetus during pregnancy and childbirth is very important in order to detect deviations and complications in a timely manner and thus for a better perinatal outcome for the newborn and the mother. Monitoring the fetus in pregnancy and childbirth is still a very current topic today. Modern perinatology aims to enable the birth of a healthy newborn.
Today, obstetrics uses modern and sophisticated devices that monitor the fetus during pregnancy and childbirth. We use cardiotocography and ultrasound to monitor the fetus during pregnancy and childbirth. In this article, a brief description of both procedures will be provided.

2. Cardiotocography

Cardiotocography is a method of monitoring the fetus during pregnancy and childbirth. Cardiotocography is most commonly used around the 30th week of pregnancy and is used until the end of labor. Cardiotocography is considered the gold standard in modern obstetrics. Cardiotocography monitors the heartbeat and contractions of the uterus. It is used routinely, during every gynecological examination after the 30th week of pregnancy and in every birth. Normal, physiological fetal heart rate is 110 / min to 160 / min. If they are below 110 / min then fetal bradycardia occurs and if they are above 160 / min then fetal tachycardia occurs which are pathological conditions where the obstetric team must respond in a timely manner [1].
The principle of recording the fetal heart rate is based on ultrasound Doppler technology. When recording, the cardiotocography machine converts the signals from the probe into an image that we monitor on the screen. The cardiotocograph registers the ultrasonic waves in the form of a curve record on paper. The recording is performed in the supine position of the pregnant woman, who usually lies on her side. A gel is applied to the pregnant woman's abdomen and two round flat probes are attached, which are connected by a cable to a CTG device that registers the fetal heartbeat and transmits them in the form of a curve on paper. The device is equipped with a speaker that gives us the ability to acoustically monitor the heart rate of the fetus. A probe leaning against a pregnant woman's abdomen usually registers the baby's heart rate below the navel (Cardiogram), while another probe located upwards registers the occurrence of uterine contractions and their strength (Tocogram) in a unit of time. CTG is recorded for an average of 15-30 minutes. In case of contractions, CTG recording can be extended to 60 minutes. In childbirth, it can be used continuously or intermittently depending on the course of labor [2].

The cardiotocography device consists of a screen, and a body, and stands on wheels. Below the screen is a strip on which there is a paper with which the record goes. It has two calottes for registering the fetal heartbeat and one for registering the activity of the uterus. In principle, one heart is used in childbirth, but each cardiotocographic device has two in case the pregnancy is a twin pregnancy. Before recording, it is necessary to apply gel to the calottes as a medium through which beats are registered. It is not necessary to apply the gel on the uterus to register the activity of the uterus because it registers the activity of the uterus every time the pregnant woman's abdomen is toned (during labor). The screen of the cardiotocographic device is on the touchscreen. The screen shows the parameters we monitor: the activities of the uterus, the heartbeat and the pulse of the mother, as well as the movements of the fetus. The date and time of the recording were also shown, as well as the name and surname of the pregnant woman, which we enter by hand. When recording, we have options that we can type, such as interventions or conditions that can occur during recording, which can lead to changes in heart rate and uterine activity, such as changes in position (back, left, right side). , vaginal examination, placement of a urinary catheter, placement of epidural analgesia, placement of oxytocin, amniotomy, spontaneous rupture of the fetal membranes, etc., or labor frequencies. If we have a twin pregnancy then we use two calottes to register the heartbeat and it is necessary to separate the curves shown on the paper when recording the record. This option is available on the screen and we must select it before recording. The CTG device used at the Clinic for Gynaecology and Obstetrics of Clinical Hospital Centre Rijeka is presented in Figure 1.

Figure 1. *The CTG device used at the Clinic for Gynaecology and Obstetrics of Clinical Hospital Centre Rijeka*

3. Ultrasound

Ultrasound is an important component in modern obstetrics. It is part of a complete gynecological examination of every pregnant woman. Ultrasound is simple, safe, and clinically important in the prevention and diagnosis of certain conditions that can affect the outcome of pregnancy and childbirth. No significant adverse effects of ultrasound on the fetus in the mother have been identified, which makes it a safe method of monitoring pregnancy and childbirth. The principle of ultrasound is based on the use of high frequencies of sound waves that create an image of organs, in this case) and fetuses, fetuses of water and placenta. There are three types of ultrasound: transvaginal (through the vagina), transabdominal (through the abdomen) and transrectal (through the rectum). Transvaginal and transabdominal ultrasounds are used in obstetrics. Each pregnant woman should have a minimum of three examinations during pregnancy. The first ultrasound examination should be done around the 7th week of pregnancy. The second should be done in the 20th week of pregnancy and the third in the 35th

week of pregnancy. The time and frequency of ultrasound examinations in pregnancy depend on the indication for performing ultrasound examinations. In the first trimester, we assess variability, gestational age, fetal number, and anomalies. In the second trimester, we assess anomalies, fetal morphology, and placental placentation. In the third trimester, the growth and condition of the fetus, as well as the appearance of the placenta and the amount of amniotic fluid are monitored [3].

For the last thirty years, modern gynecology and obstetrics have been unthinkable without a modern and sophisticated ultrasound device, so increasingly modern ultrasound devices are used in monitoring pregnancy and childbirth. In practice, we use two-dimensional (2D), three-dimensional (3D) and four-dimensional (4D) ultrasound. The gold standard in gynecology and obstetrics is two-dimensional (2D) ultrasound, which complements every gynecological examination. The additional use of colored Doppler and pulsating Doppler enables the display of the pelvic organs and the diagnosis of early pregnancy, as well as the display of the fetus, placenta, and amniotic fluid. Three-dimensional (3D) ultrasound was introduced in the 1990s. It allows us to display a spatial, volumetric, and cross-sectional image. In three-dimensional ultrasound, a two-dimensional view is converted into a three-dimensional view using a computer. It enables better diagnosis and monitoring of possible deviations that may have clinical significance in the outcomes of pregnancy and childbirth. Gives accurate and beautiful images of the fetus. It provides a better view of the fetal skeletal system. Four-dimensional ultrasound (4D) is the most modern method of ultrasound dosing in obstetrics. It gives the most accurate and best image of the fetus and the most accurate analysis of structures such as the face, limbs, and spine. It clearly shows the child's movements, facial features and limbs [4,5].

The ultrasonic device consists of a probe, a transmitting pulse generator, a compensation amplifier, a focusing control unit, a digital processor, and a display system. The device works by activating a pulse generator according to the digital computer program, whose electrical impulses are transmitted to the inverter in the probe via the control unit (for directing and focusing). Echoes are received by the same probe, amplified in a compensation amplifier, where at the same time the attenuation of ultrasound in tissues is compensated, and these signals are stored in memory and displayed on a display system (usually a TV monitor). The device operator must adjust the compensation amplifier himself to compensate for the attenuation of the ultrasound in the area of the body being searched [6]. The Ultrasound device used at the Clinic for Gynaecology and Obstetrics of Clinical Hospital Centre Rijeka is presented in Figure 2.

Figure 2. The Ultrasound device used at the Clinic for Gynaecology and Obstetrics of Clinical Hospital Centre Rijeka

4. Conclusion

Proper and timely diagnosis of certain conditions and deviations is a great challenge for modern obstetrics where the main goal is a healthy mother and newborn. In order to make a timely diagnosis and spot deviations in time, it is necessary to have modern and sophisticated devices for monitoring the fetus. Cardiotocography and ultrasound represent the gold standard in monitoring pregnancy and childbirth.

References

[1] Pattison, Neil, and Lesley McCowan. "Cardiotocography for antepartum fetal assessment." Cochrane Database of Systematic Reviews 1 (1999).
[2] Alfirevic, Zarko, et al. "Continuous cardiotocography (CTG) as a form of electronic fetal monitoring (EFM) for fetal assessment during labour." Cochrane database of systematic reviews 2 (2017).
[3] Derchi, Lorenzo E., et al. "Ultrasound in gynecology." European radiology 11.11 (2001): 2137-2155.
[4] Dietz, H. P., et al. "Two-dimensional and three-dimensional ultrasound imaging of suburethral slings." Ultrasound in Obstetrics and Gynecology: The Official Journal of the International Society of Ultrasound in Obstetrics and Gynecology 26.2 (2005): 175-179.
[5] Kavajin, Bruno, et al. "Doprinos četvero-dimenzionalnog ultrazvuka u izučavanju fetalnog ponašanja." Gynaecologia et perinatologia: journal for gynaecology, perinatology, reproductive medicine and ultrasonic diagnostics 13.3 (2004): 106-112.
[6] Chan, Vincent, and Anahi Perlas. "Basics of ultrasound imaging." Atlas of ultrasound-guided procedures in interventional pain management. Springer, New York, NY, 2011. 13-19.

ISBN: 978-953-8246-22-7

Z4 HPC Cluster

Sandi Baressi Šegota[1*], Nikola Anđelić[2], Ivan Lorencin[3], Daniel Štifanić[4], Jelena Musulin[5], Zlatan Car[6]

[1*] Faculty of Engineering – University of Rijeka, Vukovarska 58, 5100 Rijeka, Croatia, sbaressisegota@riteh.hr
[2] Faculty of Engineering – University of Rijeka, Vukovarska 58, 5100 Rijeka, Croatia, nandelic@riteh.hr
[3] Faculty of Engineering – University of Rijeka, Vukovarska 58, 5100 Rijeka, Croatia, ilorencin@riteh.hr
[4] Faculty of Engineering – University of Rijeka, Vukovarska 58, 5100 Rijeka, Croatia, dstifanic@riteh.hr
[5] Faculty of Engineering – University of Rijeka, Vukovarska 58, 5100 Rijeka, Croatia, jmusulin@riteh.hr
[6] Faculty of Engineering – University of Rijeka, Vukovarska 58, 5100 Rijeka, Croatia, car@riteh.hr

Abstract: The computational complexity of research tasks is ever growing, which is something that is extremely apparent in the field of Artificial Intelligence. These computational tasks require High Performance Computers (HPC), which may either be rented, per Infrastructure as a Service (IaaS) paradigm, or purchased in entirety and installed locally. One local cluster is Z4 HPC Cluster installed at the Department of Automation and Electronics, Faculty of Engineering – University of Rijeka. The overview of the hardware and software of the mentioned cluster is given in this paper.

Keywords: High Performance Computing, Computer Clustering, Data Science

1. introduction

With an ever-growing need for more computational power due to the increasing complexity of computational simulations and calculations needed for computer-based modelling the needs of researchers for high-performing computer workstations are getting larger every passing day. Many researchers utilize remote supercomputers available for rent, or cloud-based execution paradigms in order to address this issue – but these approaches have certain issues. First is service unavailability, due to system errors, and second is an ever-present cost which is paid periodically and can be hard to predict or prepare funds for in an academical environment. Due to this, the use of local clusters is common.
This paper will describe the Z4 HPC cluster installed at the Faculty of Engineering, University of Rijeka in Rijeka Croatia on the department of Automation and Electronics, at its' current state in May 2021. The paper will provide an overview of the cluster setup, followed by hardware descriptions of individual nodes, the description of software packages available on the cluster along with the clustering software used, with the final part of the overview being given to the supporting hardware (networking, rack, climatization, and others).

2 Cluster setup

At the time the cluster consists of four nodes – a GPU node and 3 CPU nodes, with the GPU node acting as the storage and control nodes in addition to executing GPU based simulations. The overview of the cluster setup is given in Figure 1. The nodes within the cluster are connected using 1Gb ethernet connection. GPU node also serves as the login node which is used for users to connect to and execute software using the built-in batch system (noted in section 3.3). The details of each nodes' hardware are

given in sections below. The view of the cluster rack is given in Figure 2. Abbreviations used in the rest of the document are given at the end of the paper.

Figure 1. *HPC Cluster appearance inside the rack*

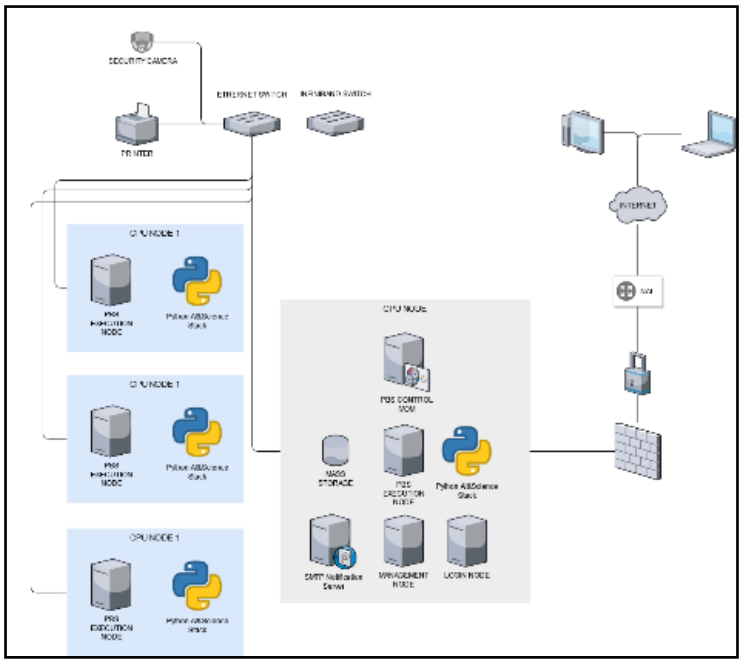


Figure 2. *Cluster setup network map*

3 Hardware information

The hardware specification of the individual cluster nodes are given below. Table 1 shows the specification of GPU node, and Table 2 shows the individual CPU nodes.

Table 1. *GPU Node Specification*

Device	Quantity	Specification
CPU	2	Intel Xeon Gold 6240R [1]
RAM	24	32GB DDR4-2665 ECC REG
SSD	2	Intel D3-S4510 240GB
GPU	5	NVIDIA Quadro RTX 6000 [2]
MBO	1	SuperMicro X11DPG-OT-CPU [3]
PLATFORM	1	6049GP-TRTKPL

The GPU node consists of two Xeon Gold 6240R CPUs, with 24 cores and 48 threads. The CPUs are clocked at 2.4 GHz, with the turbo frequency at 4.0 GHz, and 35.75 MB of L3 Cache. The node is equipped with the total of 768 GB of RAM and 240 GB of storage, consisting of two 240 GB SSD drives in RAID 1. The additional storage is setup with six 6 TB drives in RAID 5. The GPU node has two 1Gb network cards installed, one of which is in use.

Table 2. *CPU Node Specification*

Device	Quantity	Specification
CPU	2	AMD EPYC Rome 7532 [4]
RAM	4	32GB DDR4-32000 ECC REG
SSD	1	Micron 5300 240GB
GPU	-	-
MBO	1	H12SST-PS
PLATFORM	1	2014TP-HTR [5]

Each of the CPU nodes have a total of 128 GB RAM, with AMD EPYC Rome 7532 CPU, consisting of 128 MB L3 Cache, 24 cores and 48 threads, clocked at 2.3 GHz, with 240GB SSD boot drive. No onboard GPU is present. Each of the nodes have two 10 Gbps Ethernet cards installed, with a single one being in use on each node.

4 Software information

The software installed on the cluster is described in this section – starting with details on the operating system, followed by the list of significant software packages available on the cluster, and finally, clustering software.

4.1 Operating System

Nodes are based on GNU/Linux. The used operating systems on all the nodes is Ubuntu 18.04.5., with the Linux kernel revision 4.15.0-140-generic. The OS was selected due to it being supported by needed packages, described in the following section. Between all the needed packages, the support was available for two OSes – the selected Ubuntu version 18 and Red Hat Enterprise Linux (RHEL) version

8; with priority being given to Ubuntu due to this OS being freely available.

4.2 Notable installed packages

The notable packages can be separated in two sections: the GPU stack and the Python scientific stack. The GPU stack includes the drivers and other software needed to enable the use of the installed NVIDIA GPUs. The NVIDIA driver version installed on the GPU node is 418.29, with CUDA version 10.1 [6]. In addition, the NVIDIA CUDA Deep Neural Network (cuDNN) library containing GPU-accelerated primitives for deep neural network is installed [7]. The Python scientific stack is based on Python version 3.8.5 [8], with python modules commonly used in Data Science added to the installation. The modules are installed using Pip Python module manager [9]. The following packages are available for use (versions given in brackets):

- tensorflow-gpu 2.3.0 [10],
- scikit-Learn 0.23.2 [11],
- numpy 1.18.5 [12],
- scipy 1.4.1 [13],
- mpmath 1.2.1 [14],
- gplearn 0.4.1 [15],
- pandas 1.1.4 [16], and
- matplotlib 3.3.1 [17].

The addition of customised Python virtual environments is enabled to users who need different packages. Virtualization and dockerization is also enabled in case custom environments are necessary. In addition to Python, additional scientific simulation/computation software is installed such as LAMMPS [18], GNU Octave [19], and FreeCAD [20]. These packages have been selected based on the current users' needs, with the possibility of additional packages being installed through the contact with the system administrator.

4.3 Clustering

Clustering software used is OpenPBS [21], which is an integrated batch system. The details on the system are available in the provided documentation. The software control node is setup on the GPU node, along with the Login and management nodes. The other hardware nodes are setup as execution nodes. This allows the users to connect to the main node and execute software directly on it, which will then get run on any node which has available resources. In addition to the above, OpenPBS allows queueing of commands for execution, with the setup of multiple queues available. OpenPBS also has the capability of sending e-mail notifications by using the installed SMTP server, to send a notification e-mail at the beginning of the execution of the queued command, the abort of the queued command or the finish.

4. Supporting Hardware

The following additional hardware and devices are installed within the cluster rack or cluster room:

- HP 1Gbps Intelligent Network Switch
- Mellanox Infiniband 24-Port Switch
- UPS
- Patch Panel
- Climate control Unit
- TP-Link KC115 Security Camera
- LN646806 power Delivery Unit
- EVO42U8010DU Rack

5 Current work on cluster

The cluster has been applied in multiple scientific researches and projects. The work has mostly focused on the application of AI in biomedicine – either for cancer diagnosis [22, 23], infective disease spread modelling [24, 25], patient diagnosis [26], complex engineering system modelling [27], and robotics [28-30].

6 Conclusions

The Z4 HPC cluster has been described in the paper. This document is meant to be informative and the information within it is subject to change without prior notice. For up to date information, please check for newer publications of this type or contact the system administrator. Future updates are planned to the cluster – namely the addition of the at least one CPU node, and at least one more GPU node is planned; with upgrades to the networking equipment to enable faster communication speeds between the cluster nodes.

Abbreviations

CPU	Central Processing Unit
GPU	Graphical Processing Unit
MBO	Motherboard
RAM	Random Access Memory
HDD	Hard Disk Drive
SSD	Solid State Drive
ECC	Error Correction
REG	Registered
PBS	Portable Batch System
OS	Operating System
GB	Gigabyte
Gb	Gigabit
GHz	Gigahertz
TB	Terabyte
MB	Megabyte
UPS	Uninterruptible Power Supply

Acknowledgments

This research has been (partly) supported by the CEEPUS network Ciii-HR-0108, European Regional Development Fund under the grant K.01.1.1.01.0009 (DATACROSS), project CEKOM under the grant KK.01.2.2.03.0004, CEI project “COViDAi” (305.6019-20), and University of Rijeka scientific grant uniri-tehnic-18-275-1447.

References

[1]Intel Xeon Gold 6240R Processor https://ark.intel.com/content/www/us/en/ark/products/199343/intel-xeon-gold-6240r-processor-35-75m-cache-2-40-ghz.html, accessed on 15th April 2021
[2] NVIDIA Quadro RTX 6000 https://www.nvidia.com/en-us/design-visualization/quadro/rtx-6000/, accessed on 15th April 2021
[3] Supermicro 6049, https://www.supermicro.com/products/system/4U/6049/SYS-6049GP-TRT.cfm, accessed on 15th April 2021
[4] AMD Epyc Rome 7532 https://www.amd.com/en/products/cpu/amd-epyc-7532, accessed on 15th April 2021

[5] Supermicro 2014 TP https://www.supermicro.com/en/Aplus/system/2U/2014/AS-2014TP-HTR.cfm, accessed on 15th April 2021
[6] Kirk, David, and Barcelona Supercomputing Center. "NVIDIA CUDA software and GPU parallel computing architecture." (2008).
[7] Chetlur, Sharan, et al. "cudnn: Efficient primitives for deep learning." arXiv preprint arXiv:1410.0759 (2014).
[8] Van Rossum, Guido. "Python." (1991).
[9] Package installer for Python https://pypi.org/project/pip/, accessed on 15th April 2021
[10] Abadi, Martín, et al. "Tensorflow: A system for large-scale machine learning." 12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16). 2016.
[11] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." the Journal of machine Learning research 12 (2011): 2825-2830.
[12] NumPy https://numpy.org/, accessed on 15th April 2021
[13] Virtanen, Pauli, et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python." Nature methods 17.3 (2020): 261-272.
[14] Johansson, Fredrik. "mpmath: A Python library for arbitrary-precision floating-point arithmetic (version 0.14)." (2010).
[15] GPLearn https://gplearn.readthedocs.io/en/stable/, accessed on 15th April 2021
[16] Pandas https://pandas.pydata.org/, accessed on 15th April 2021
[17] Hunter, John D. "Matplotlib: A 2D graphics environment." IEEE Annals of the History of Computing 9.03 (2007): 90-95.
[18] Plimpton, Steve. "Fast parallel algorithms for short-range molecular dynamics." Journal of computational physics 117.1 (1995): 1-19.
[19] Eaton, John Wesley, David Bateman, and Søren Hauberg. Gnu octave. London: Network thoery, 1997.
[20] Riegel, Jürgen, Werner Mayer, and Yorik van Havre. "FreeCAD." (2016).
[21] OpenPBS https://www.openpbs.org/, accessed on 15th April 2021
[22] Lorencin, Ivan, et al. "On Urinary Bladder Cancer Diagnosis: Utilization of Deep Convolutional Generative Adversarial Networks for Data Augmentation." Biology 10.3 (2021): 175.
[23] Musulin, Jelena, et al. "An Enhanced Histopathology Analysis: An AI-Based System for Multiclass Grading of Oral Squamous Cell Carcinoma and Segmenting of Epithelial and Stromal Tissue." Cancers 13.8 (2021): 1784.
[24] Anđelić, Nikola, et al. "Estimation of COVID-19 Epidemiology Curve of the United States Using Genetic Programming Algorithm." International Journal of Environmental Research and Public Health 18.3 (2021): 959.
[25] Anđelić, Nikola, et al. "Estimation of COVID-19 epidemic curves using genetic programming algorithm." Health Informatics Journal 27.1 (2021): 1460458220976728.
[26] Lorencin, Ivan, et al. "Automatic Evaluation of the Lung Condition of COVID-19 Patients Using X-ray Images and Convolutional Neural Networks." Journal of Personalized Medicine 11.1 (2021): 28.
[27] Baressi Šegota, Sandi, et al. "Improvement of Marine Steam Turbine Conventional Exergy Analysis by Neural Network Application." Journal of Marine Science and Engineering 8.11 (2020): 884.
[28] Baressi Šegota, Sandi, et al. “Prediction of Robot Grasp Robustness Using Artificial Intelligence Algorithms” Technical Gazette (Forthcoming – Accepted for Publication)
[29] Baressi Šegota, Sandi, et al. “Neural Network-Based Model for Classification of Faults During Operation of a Robotic Manipulator” Technical Gazette (Forthcoming – Accepted for Publication)
[30] Lorencin, Ivan, et al. "Automatic Evaluation of the Lung Condition of COVID-19 Patients Using X-ray Images and Convolutional Neural Networks." Journal of Personalized Medicine 11.1 (2021): 28.

ISBN: 978-953-8246-22-7

Steel plate faults classification using MLP classifier

Lea Mrakovčić [1], Zvonimir Žugčić [1*]

[1] Faculty of Engineering RITEH University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia: lmrakovcic@riteh.hr; zzugcic@riteh.hr;

Abstract: This article tested MLP Classifier applied on database which describes steel plate faults, given in .csv file, including 27 input features and 7 output classes. Goal was to find optimal parameters in classification process using MLP classifier, hyperparameter tuning was automated. Correlation of input conditions is described and distribution of errors is visualized. Test and train data were separated, then MLP Classifier is applied and fitted. Classification report and accuracy are evaluated using F1 and AUC metrics.

Keywords: Artificial intelligence, Machine Learning, MLP classifier, F1 score, AUC metrics.

1. Introduction

Machine learning methods are designed to perform tasks of inspection through classification and categorization of images, shapes, symbols and other types of data. Artificial Neural Networks (ANN) have applications in many fields like industry and medicine. Classification can be done in different ways, and many libraries and frameworks are dedicated to simplifying the creation of neural networks [1]. Neural network software packages are both plug-in and stand-alone Python packages that provide supervised classification methods for multi-band passive optical remote sensing data.
MLP method has been successfully applied on classification and regression problems in the medical area [2,3], energetics [4,5,6], and COVID-19 infection spread model [7].
After the given dataset is imported, steel plate features are separated from faults, then their values are normalized. Features correlation in shown via heatmap. Multi-Layer Perceptron (MLP) Classifier is implemented using Python scikit library and optimal parameters are found automatically using GridSearchCV. Precision and accuracy scores of different MLP hyperparameters are compared and one with the best score is chosen. Accuracy is compared using F1 and AUC metrics. Confusion matrix is used to summarize the performance of a classification algorithm.

2. Methodology

Faulty steel plates dataset comes from research by Semeion, Research Center of Sciences of Communication. The original aim of the research was to correctly classify the type of surface defects in stainless steel plates, with six types of possible defects (plus "other"). The Input vector was made up of 27 indicators that approximately describe the geometric shape of the defect and its outline. According to the research paper, Semeion was commissioned by the Centro Sviluppo Materiali (Italy) for this task and therefore it is not possible to provide details on the nature of the 27 indicators used as Input vectors or the types of the 6 classes of defects.
The dataset includes 1941 observations and 27 features. The data is already labeled, and there are 7 types of steel plate faults that are added to the dataset as 7 fields representing the one-hot-encoding of the label. If there is a defect for certain class 1 will be shown.
Numpy and Pandas libraries are used for computations and handling the dataset. Pandas is required to load the data file into the environment. After the dataset is imported features and faults are separated. Feature dataset is scaled to make sure input data is normally distributed. There are no missing data in

the set and all values are numeric. Data correlation is shown with a heatmap in Figure 1. Heat map is a graphical representation of data where values contained in a matrix are shown as colors. It can be seen that many independent features are strongly correlated.

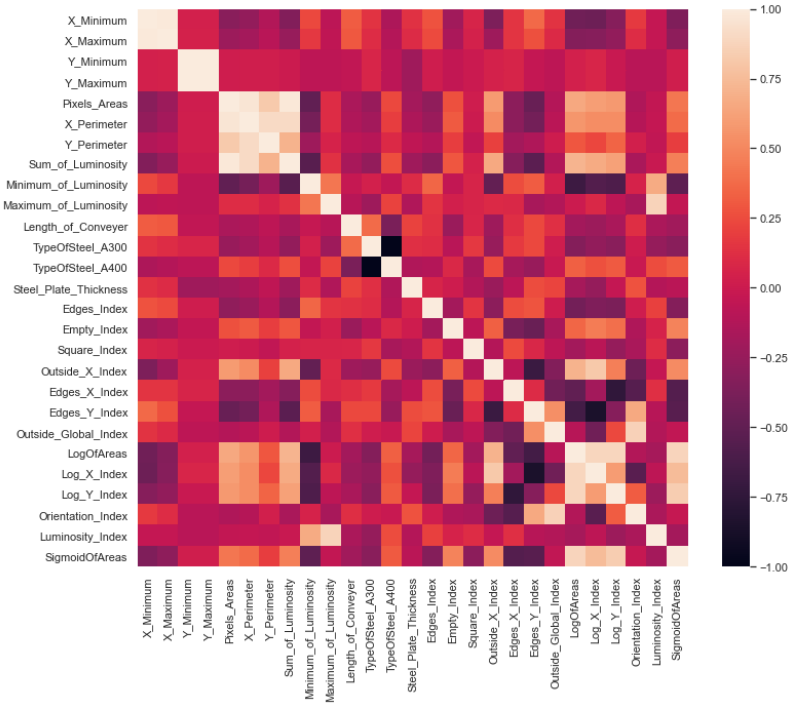


Figure 1. *Features correlation heatmap*

Feature scaling is an essential step in the analysis and preparation of data. Normalization is one of the feature scaling techniques where data features of different scales are converted to a common scale. Using the bellow formula, values are transformed to a range between 0 and 1.

$$x_n = \frac{x - x_{min}}{x_{max} - x_{min}} \qquad (1)$$

Output features dataset is not balanced, and errors vary extremely, with “Other_Faults” category at 673 times (34.7%) and the least amount of outputs in “Dirtiness” category, with 55 times. An uneven class distribution is shown in Figure 2. When separating data into train and test data, “stratify” function is used to evenly distribute classes.

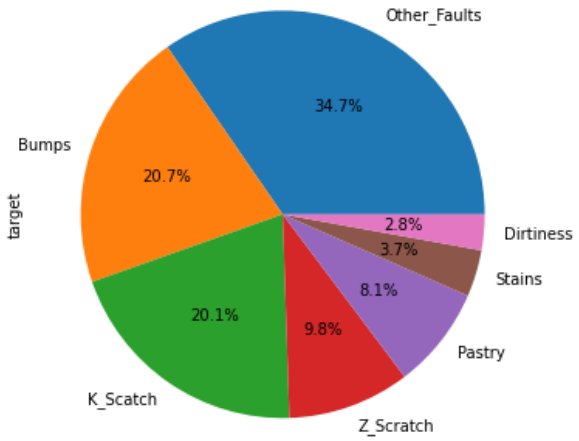


Figure 2. *Steel plate faults*

By splitting the dataset into train and test data, 20% is reserved for testing the accuracy of the trained model. There is no correlation between faults.
Multi-Layer Perceptron (MLP) Classifier optimizes the log-loss function using LBFGS or stochastic gradient descent. MLP Classifier trains iteratively, with each step partial derivates of the loss function with respect to the model parameters are computed to update the parameters. Overfitting can be prevented by adding a regularization term to the loss function that shrinks model parameters. This implementation works with data represented as dense numpy arrays or sparse scipy arrays of floating-point values.
The main parameters to set when building MLP Classifier are the number of layers and sizes, maximal number of iterations, activation function, solver and random state. Other parameters as alpha regularization term, learning rate and batch size can further improve or limit classifier performance.
After initializing, train data is given to fit the model and the trained network can be used to predict. With many possible parameters, it can be hard to determine optimal values for the model. Manually trying all possible values would be time-consuming, thus GridSearchCV was used to automate the tuning of hyperparameters. This function loops through predefined hyperparameters and at the end list the best parameters. Both ReLu and Tanh activation functions performed well, depending on the hidden layer sizes, the learning rate was set to adaptive, and with 'sgd' and 'adam' used as solvers, models were fitted with 2000 set as maximal number of iterations.

Table 1. *Used classifier*

MLP	Classifier
sizes	(64, 32, 16)
activation	tanh
rate	adaptive
solver	adam
rate_init	0.001
alpha	0.01
iterations	1500

Prediction accuracy is calculated using F1 and AUC metrics. The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worse at 0.

The formula for F1 score is shown below:

$$F_1 = 2 * \frac{precision * recall}{precision + recall} \qquad (2)$$

It gives equal importance to precision and recall, which in practice is not always the case.
AUC stands for Area Under the ROC Curve (receiver operating characteristic curve) and is a precision-recall curve which measures the entire two-dimensional area under the curve. Its value ranges from 0 to 1, a model whose predictions are 100% correct has an AUC of 1. It is a graphical representation of model performance. Since ROC curve is restricted to the binary classification, AUC score is calculated instead.

3. Results and Discussion

Different combinations of parameters in modeling MLP classifiers were tested. Adding more hidden layers to the classifier increases the number of iterations. A model with layer sizes (64, 32, 16) gave satisfying accuracy scores shown in table 2:

Table 2. *Accuracy scores*

Accuracy	0.73
F1	0.76
AUC	0.84

Both relu and tanh activation functions had very similar accuracy scores, although relu required far fewer computing iterations, tanh was more accurate.
Confusion matrix in Figure 3. shows output classes:
'Pastry', 'Z_Scratch', 'K_Scatch', 'Stains', 'Dirtiness', 'Bumps', 'Other_Faults'

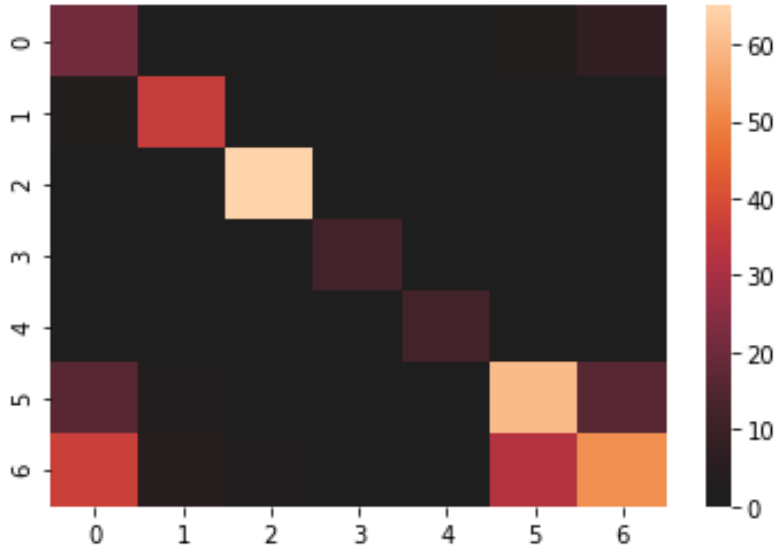


Figure 3. *Confusion matrix*

4. Conclusion

For classification of steel plate faults provided numerical dataset was analyzed, pre-processed and fitted using MLP classifier which shows good prediction accuracy if model is constructed properly. F1 and AUC metric scores varied when trying different parameters. Optimal results were accomplished by classifier with 3 hidden layers.

Acknowledgments
"We thank all assistants and professor for their mentorship during the work on this article".

Reference

[1] Buscema, Massimo, Dr. "MetaNet: The Theory of Independent Judges." *Semeion Research Center of Sciences of Communication (1998)*
[2] Lorencin, I., Anđelić, N., Španjol, J., & Car, Z. (2020). Using multi-layer perceptron with Laplacian edge detector for bladder cancer diagnosis. Artificial Intelligence in Medicine, 102, 101746.

[3] Lorencin, I., Anđelić, N., Baressi Šegota, S., Štifanić, D., Musulin, J., Mrzljak, V., Markova-Car, E., & Car, Z. (2020). Dataset Size-based Approach in Design of Artificial Neural Network for Breast Cancer Diagnosis. World of Health, 3, 13, 19.

[4] Lorencin, I., Anđelić, N., Mrzljak, V., & Car, Z. (2019). Genetic algorithm approach to design of multi-layer perceptron for combined cycle power plant electrical power output estimation. Energies, 12(22), 4352.

[5] Lorencin, I., Anđelić, N., Mrzljak, V., & Car, Z. (2019). Multilayer perceptron approach to condition-based maintenance of marine CODLAG propulsion system components. Pomorstvo, 33(2), 181-190.

[6] Baressi Šegota, S., Lorencin, I., Anđelić, N., Mrzljak, V., & Car, Z. (2020). Improvement of Marine Steam Turbine Conventional Exergy Analysis by Neural Network Application. Journal of Marine Science and Engineering, 8(11), 884.

[7] Car, Z., Baressi Šegota, S., Anđelić, N., Lorencin, I., & Mrzljak, V. (2020). Modeling the spread of COVID-19 infection using a multilayer perceptron. Computational and mathematical methods in medicine, 2020.

ISBN: 978-953-8246-22-7

Efficiencies and losses comparison of various turbofan engines for aircraft propulsion

Lovro Kadi, Vedran Mrzljak

Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia (lkadi@riteh.hr, vmrzljak@riteh.hr)

Abstract: This paper presents efficiencies and losses comparison of 5 different turbofan engines for aircraft propulsion. With some exceptions these turbofan engines were generally comprised of the following components: fan, low-pressure compressor, high-pressure compressor, combustion chamber, high-pressure turbine, and low-pressure turbine. The comparison of engine components and whole engines was done using energy and exergy analysis method. The analysis showed that "*Engine 1"* performed with the highest energy and exergy efficiencies, while "*Engine 5"* performed with the lowest energy and exergy efficiencies. Engine component with the highest energy and exergy loss was the combustion chamber in all of the engines, while engine component with the highest energy and exergy efficiency proved to be the high-pressure turbine in most of the cases.

Keywords: Turbofan engine, energy analysis, exergy analysis, aircraft propulsion, jet engine

1. Introduction

Jet engines are used as a form of aircraft propulsion with turbofan engines being widely represented in that role. A turbofan engine usually consists of a fan, low-pressure compressor, high-pressure compressor, combustion chamber, high-pressure turbine, and low-pressure turbine.
Turbofan engine forces pressurized air around and through the engine. The bypass ratio is the ratio of air passing through the fan (thus bypassing the engine core) to the air that passes through the engine. The higher the ratio of bypassed air to air passing through the engine, the greater is fuel efficiency of the engine [1]. The main energy source for turbofan engines comes from fuel, usually kerosene. Energy analysis has its roots in the first law of thermodynamics, while exergy analysis is based on the second law of thermodynamics. Engines can be analyzed in such a way to improve certain engine components regarding their efficiency and performance. Many studies have been published about this topic. According to Tai et al. (2014) [2], exergy is the maximum theoretical work obtainable during a process that brings the system into equilibrium with its environment. Tuzcu et al. (2020) [3] researched enviroeconomic impact of turbofan engines and applied some of the energy analysis methods on a turbofan engine used in aviation industry. In another study performed by Turgut et al. (2007) [4] exergy analysis was performed on a turbofan engine with an afterburner both at sea level and at altitude of 11000 m. There are many ways turbofan engine performance can be affected. According to Turgut et al. (2009) [5], effects on exergy efficiencies and losses are investigated by modifying the isentropic efficiencies of turbomachinery components. They found that generally the components with the most exergy destruction in a turbofan engine are the fan, engine exhaust and combustion chamber. Hence, in their research fan had an exergy efficiency of only 12,9%.
In this paper five different turbofan engines are analyzed and compared on the basis of their energy and exergy efficiency, including losses.

2. Methodology

Operating parameters of each observed turbofan engine - Temperatures T (K), pressures p (bar) and mass flow rates m (kg/s) were found in the literature [3-7]. Each of the engines used kerosene, with a lower fuel heating value (LHV) equal to 46000 kJ/kg, and fuel mass flow rate m_f. Specific entropy s (kJ/kgK), specific enthalpy h (kJ/kg), and specific exergy ε (kJ/kg) were afterwards calculated for each point of each engine using *NIST-REFPROP* software [8]. Scheme of the first analyzed turbofan engine (Engine 1) is presented in Figure 1.

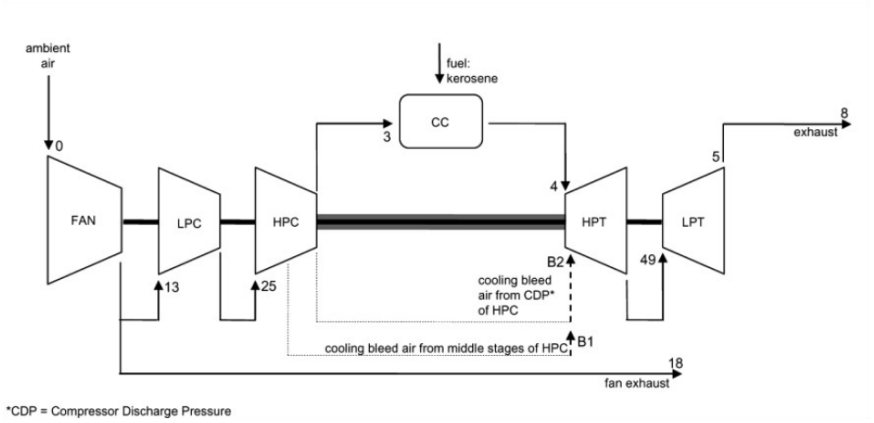


Figure 1. *"Engine 1" and its components [5].*

Each of five observed turbofan engines consists of various components. The list of components for each engine is presented below from a) to e).

a) "Engine 1" consists of: fan, low-pressure compressor (LPC), high-pressure compressor (HPC), combustion chamber (CC), high-pressure turbine (HPT) and low-pressure turbine (LPT) [5].

b) "Engine 2" consists of: fan, low-pressure compressor (LPC), combustion chamber (CC), high-pressure turbine (HPC) and is the only engine with an afterburner [4].

c) "Engine 3" consists of: fan, low-pressure compressor (LPC), high-pressure compressor (HPC), combustion chamber (CC), high-pressure turbine (HPT) and low-pressure turbine (LPT) [6].

d) "Engine 4" consists of: fan, high-pressure compressor (HPC), combustion chamber (CC), high-pressure turbine (HPT) and low-pressure turbine (LPT) [7].

e) "Engine 5" consists of: fan, low-pressure compressor (LPC), combustion chamber (CC), high-pressure turbine (HPT) and low-pressure turbine (LPT) [3].

Each component of each engine was analyzed separately. The equations for the energy and exergy analysis of all the components of each observed turbofan engine were found in the available literature [9, 10]. These equations are not presented in this paper.

Here are presented an energy and exergy equations for the evaluation of whole engines, because they slightly differ in comparison to the equations from a literature. In the equations for evaluation of the whole engines, *inlet* and *outlet* stand for engine inlet and engine outlet. Engine inlet consists of air which enters the engine through the fan while engine outlet consists of two parts: gas which leaves the engine through the main exhaust (after the turbine) and air which leaves the engine through the fan exhaust. Therefore, m_{outlet} consists of those two combined streams. Energy and exergy analysis of the whole engine (valid for each observed engine) were done as follows:

Energy analysis:

- Inlet energy (kW):

$$E_{en,in} = m_{inlet} \cdot h_{inlet} + m_f \cdot LHV, \tag{1}$$

- Outlet energy (kW):

$$E_{en,out} = m_{outlet} \cdot h_{outlet}, \tag{2}$$

- Energy loss (kW):

$$E_{en,l} = E_{en,in} - E_{en,out}, \tag{3}$$

- Energy efficiency:

$$\eta_{en} = \frac{E_{en,out}}{E_{en,in}} \cdot 100\%, \tag{4}$$

Exergy analysis:
- Inlet exergy (kW):

$$E_{ex,in} = m_{inlet} \cdot \varepsilon_{inlet} + m_f \cdot \varepsilon_f, \tag{5}$$

- Outlet exergy (kW):

$$E_{ex,out} = m_{outlet} \cdot \varepsilon_{outlet}, \tag{6}$$

- Exergy loss (kW):

$$E_{ex,l} = E_{ex,in} - E_{ex,out}, \tag{7}$$

- Exergy efficiency:

$$\eta_{ex} = \frac{E_{ex,out}}{E_{ex,in}} \cdot 100\%, \tag{8}$$

3. Results and Discussion

In order to compare the engine parameters, inlet air properties are given in Table 1. “Engine 1” performed with the highest inlet air temperature of 306,5 K (33,35 ºC), while “Engine 3” performed with the lowest inlet air temperature of 288 K (14,85 ºC). Furthermore, “Engine 1” performed with the highest air mass flow rate of 679,184 kg/s and “Engine 5” performed with the lowest air mass flow rate of 70,330 kg/s. Fuel economy is presented in Table 2, and it shows that “Engine 2” was using the most fuel (6,879 kg/s), while “Engine 4” was using the least fuel (1,050 kg/s).

Table 1. *Inlet air properties.*

Inlet air properties		
	Temperature (K)	**Mass flow rate (kg/s)**
Engine 1	306,5	679,184
Engine 2	298,0	101,696
Engine 3	288,0	361,000
Engine 4	288,2	142,700
Engine 5	288,2	70,330

Table 2. *Engine fuel economy.*

	Fuel economy (kg/s)
Engine 1	2,200
Engine 2	6,879
Engine 3	1,284
Engine 4	1,050
Engine 5	2,970

3.1. Energy losses and efficiencies

The fan for which energy losses are the highest and equal to 2815,96 kW can be found in “Engine 2”, while the fan for which energy losses are the lowest can be found in “Engine 5”. Low-pressure

compressor (LPC) for which energy losses are the highest can be found in “Engine 2”, while the low-pressure compressor for which energy losses are the lowest and equal to 896,52 kW can be found in “Engine 1”. “Engine 4” does not have a low-pressure compressor. High-pressure compressor (HPC) for which energy losses are the highest can be found in “Engine 1”, while the high-pressure compressor for which energy losses are the lowest and equal to 2982,33 kW can be found in “Engine 4”. “Engine 2” and “Engine 5” do not have a high-pressure compressor, Figure 2. Combustion chamber (CC) which lost the most energy can be found in “Engine 5”, while the combustion chamber which lost the least energy (3595,93 kW) can be found in “Engine 3”. High-pressure turbine (HPT) for which energy losses are the highest and equal to 4381,81 kW can be found in “Engine 1”, while the high-pressure turbine for which energy losses are the lowest and equal to 43,98 kW can be found in “Engine 5”. Low-pressure turbine (LPT) which lost the most energy can be found in “Engine 1”, while the low-pressure turbine which lost the least energy (124,61 kW) can be found in “Engine 5”. “Engine 2” does not have an LPT, Figure 2. As stated earlier, “Engine 2” is the only engine of the five observed with an afterburner which lost 76961,44 kW of energy. When comparing whole engines, “Engine 2” lost the most energy (162998,51 kW), while “Engine 3” lost the least energy (5036,84 kW), Figure 3.

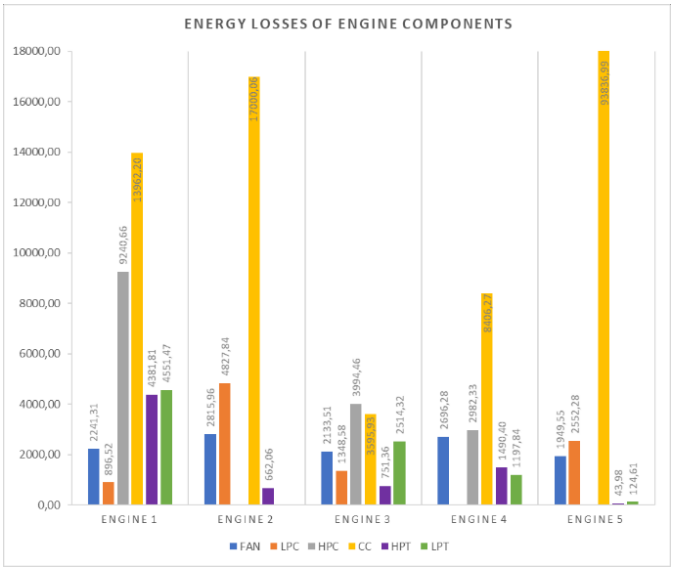


Figure 2. *Energy losses of specific engine components in kW.*

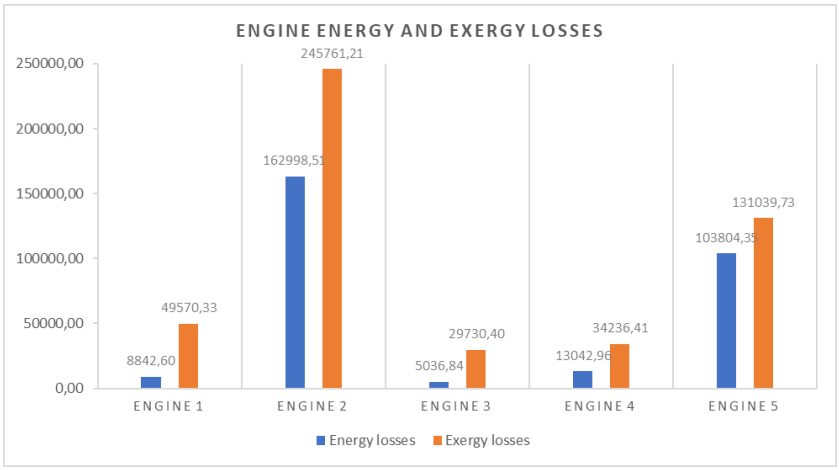


Figure 3. *Whole engine energy and exergy losses in kW.*

The most energy efficient fan (93,87%) is in “Engine 1”, while the least energy efficient fan (77,54%) operates in “Engine 2”. The most energy efficient low-pressure compressor is in “Engine 5” (86,02%),

while the least energy efficient low-pressure compressor operates in “Engine 1” (79,20%), Figure 4. The most energy efficient high-pressure compressor operates in “Engine 3” (85,08%) while the least energy efficient high-pressure compressor operates in “Engine 4” (83,05%). The most energy efficient combustion chamber is found in “Engine 3” (93,91%). Simultaneously, the least energy efficient combustion chamber is found in “Engine 5” (31,32%).

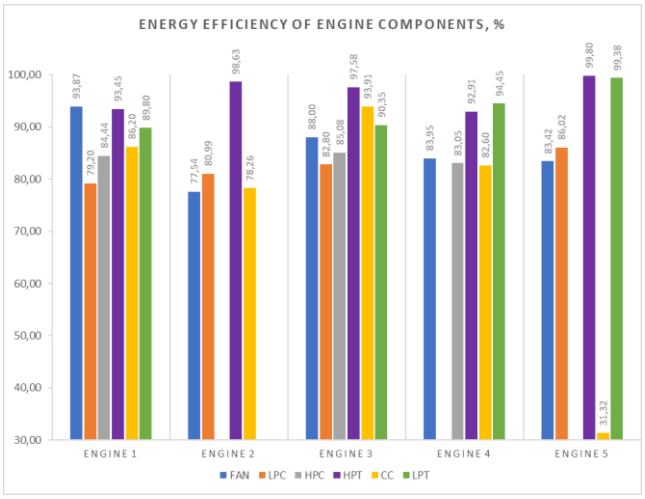


Figure 4. *Energy efficiency of specific engine components.*

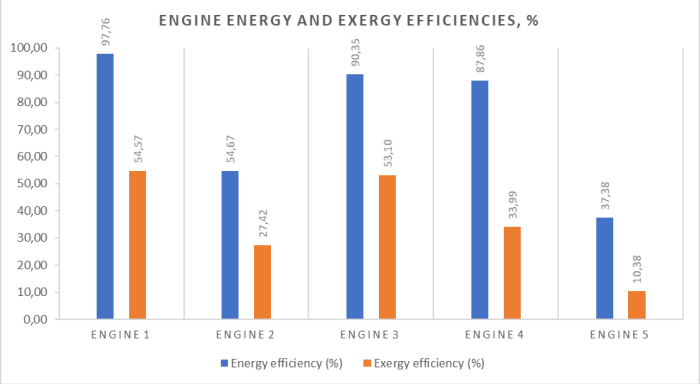


Figure 5. *Whole engine energy and exergy efficiencies.*

The most energy efficient high-pressure turbine operates in “Engine 5” (99,80%), while the least energy efficient high-pressure turbine operates in “Engine 4” (92,91%). The most energy efficient low-pressure turbine is found in “Engine 5” (99,38%), while the least energy efficient low-pressure turbine operates in “Engine 1” (89,80%), Figure 4. “Engine 2” is the only engine of the five observed with an afterburner which runs with energy efficiency of 67,70%.
When comparing whole engines, “Engine 1” has the highest overall energy efficiency of 97,76%, while “Engine 5” has the poorest energy efficiency of 37,38%, Figure 5.

3.2. Exergy losses and efficiencies

The fan which lost the most exergy can be found in “Engine 2”, while the fan which lost the least exergy (1322,12 kW) can be found in “Engine 5”. Low-pressure compressor which lost the most exergy can be found in “Engine 2”, while the low-pressure compressor which lost the least exergy (681,62 kW) can be found in “Engine 1”, Figure 6. High-pressure compressor which lost the most exergy can be found in “Engine 1”, while the high-pressure compressor which lost the least exergy (1310,08 kW) can be found in “Engine 4”. Combustion chamber which lost the most exergy (116073,50 kW) can be found

in “Engine 5”, while the combustion chamber which lost the least exergy (20600,95 kW) can be found in “Engine 3”. High-pressure turbine with the highest exergy losses (1233,99 kW) can be found in “Engine 1”, while the high-pressure turbine with the lowest exergy losses can be found in “Engine 5”. Low-pressure turbine which lost the most exergy (1762,49 kW) operates in “Engine 1”, while the low-pressure turbine which lost the least exergy operates in “Engine 5”, Figure 6. “Engine 2” is the only engine of the five observed with an afterburner which lost 129306,30 kW of exergy. When comparing whole engines, “Engine 2” lost the most exergy (245761,21 kW), while “Engine 3” lost the least exergy (29730,40 kW), Figure 3.

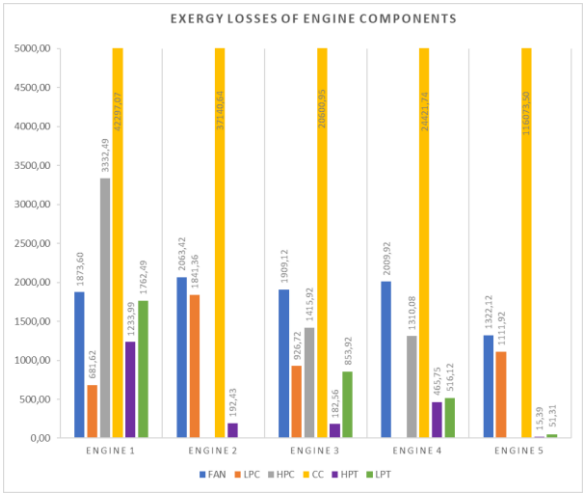


Figure 6. *Exergy losses of specific engine components in kW.*

The most exergy efficient fan operates in “Engine 1” (94,88%), while the least exergy efficient fan operates in “Engine 2” (83,55%). The most exergy efficient low-pressure compressor is found in “Engine 5” (93,91%), while the least exergy efficient low-pressure compressor is found in “Engine 1” (84,19%). The most exergy efficient high-pressure compressor operates in “Engine 3” (94,71%), while the least exergy efficient high-pressure compressor operates in “Engine 4” (92,56%), Figure 7. The most exergy efficient combustion chamber operates in “Engine 3” (87,94%). Simultaneously, the least exergy efficient combustion chamber operates in “Engine 5” (49,86%). The most exergy efficient high-pressure turbine can be seen in “Engine 5” (99,93%), while the least exergy efficient high-pressure turbine can be seen in “Engine 4” (97,67%). The most exergy efficient low-pressure turbine is observed in “Engine 5” (99,75%), while the least exergy efficient low-pressure turbine is observed in “Engine 1” (95,79%), Figure 7. “Engine 2” is the only engine of the five observed with an afterburner which runs with exergy efficiency of 70,65%. When comparing whole engines, “Engine 1” has the highest overall exergy efficiency of 54,57%, while “Engine 5” has the poorest overall exergy efficiency of only 10,38%, Figure 5.

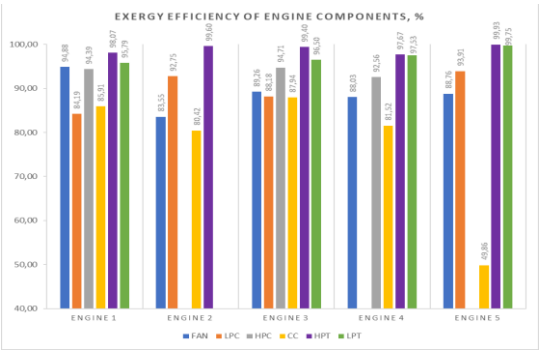


Figure 7. *Exergy efficiency of specific engine components.*

4. Conclusion

From the performed analysis, it can be concluded that “Engine 2” operates with the largest losses, both for energy and exergy. This was expected because of the afterburner fitted to this turbofan engine. Furthermore, “Engine 3” performed with the smallest losses both for energy and exergy. However, the most energy and exergy efficient engine was “Engine 1”, while “Engine 5” showed the poorest performance in relation to energy and exergy efficiency. The afterburner in “Engine 2” had the highest exergy loss of all the engine components.
This research shows which of the turbofan components could be further improved. For future work it would be interesting to examine the internal build of the turbofan components more closely to see where energy and exergy improvements could be made. Also, in the future it would be useful to make a study which investigates turbofan energy or exergy efficiencies periodically over the whole flight, as in research performed by Tona, Cesare et al. (2010) [11] or more recently Sohret et al. (2015) [12]. Moreover, optimization of each observed turbofan engine or any of the engine components can be performed in the future by the application of various artificial intelligence (AI) methods, which show good performance in the energy sector [13, 14].

Reference

[1] Brines, Gerald L. "The turbofan of tomorrow." *Mechanical Engineering* 112.8 (1990): 65.
[2] Tai, Vin Cent, Phen Chiak See, and Cristinel Mares. "Optimisation of energy and exergy of turbofan engines using genetic algorithms." *International Journal of Sustainable Aviation* 1.1 (2014): 25-42. https://doi.org/10.1504/IJSA.2014.062866
[3] Tuzcu, Halil, Yasin Sohret, and Hakan Caliskan. "Energy, environment and enviroeconomic analyses and assessments of the turbofan engine used in aviation industry." *Environmental Progress & Sustainable Energy* (2020): e13547. https://doi.org/10.1002/ep.13547
[4] Turgut, Enis T., T. Hikmet Karakoc, and Arif Hepbasli. "Exergetic analysis of an aircraft turbofan engine." *International Journal of Energy Research* 31.14 (2007): 1383-1397. https://doi.org/10.1002/er.1310
[5] Turgut, Enis T., et al. "Exergy analysis of a turbofan aircraft engine." *International Journal of Exergy* 6.2 (2009): 181-199. https://doi.org/10.1504/IJEX.2009.023997
[6] Aydin, Hakan, et al. "Exergetic sustainability indicators as a tool in commercial aircraft: a case study for a turbofan engine." *International journal of green energy* 12.1 (2015): 28-40. https://doi.org/10.1080/15435075.2014.889004
[7] Turan, Onder, and Hakan Aydin. "Exergy-based sustainability analysis of a low-bypass turbofan engine: A case study for JT8D." *Energy Procedia* 95 (2016): 499-506. https://doi.org/10.1016/j.egypro.2016.09.075
[8] Lemmon, E. W., M. L. Huber, and M. O. McLinden. "NIST standard reference database 23, NIST reference fluid thermodynamic and transport properties, REFPROP, version 9.0." Standard Reference Data Program (2010).
[9] Mrzljak, Vedran, et al. "Analysis of gas turbine operation before and after major maintenance." Pomorski zbornik 57.1 (2019): 57-70. https://doi.org/10.18048/2019.57.04.
[10] Mrzljak, Vedran, et al. "Exergy analysis of marine waste heat recovery CO2 closed-cycle gas turbine system." Pomorstvo 34.2 (2020): 309-322. https://doi.org/10.31217/p.34.2.12
[11] Tona, Cesare, et al. "Exergy and thermoeconomic analysis of a turbofan engine during a typical commercial flight." *Energy* 35.2 (2010): 952-959. https://doi.org/10.1016/j.energy.2009.06.052

[12] Şöhret, Yasin, Ali Dinç, and T. Hikmet Karakoç. "Exergy analysis of a turbofan engine for an unmanned aerial vehicle during a surveillance mission." *Energy* 93 (2015): 716-729.
https://doi.org/10.1016/j.energy.2015.09.081
[13] Baressi Šegota, Sandi, et al. "Improvement of Marine Steam Turbine Conventional Exergy Analysis by Neural Network Application." Journal of Marine Science and Engineering 8.11 (2020): 884.
https://doi.org/10.3390/jmse8110884
[14] Lorencin, I., Anđelić, N., Mrzljak, V., & Car, Z. (2019). Genetic algorithm approach to design of multi-layer perceptron for combined cycle power plant electrical power output estimation. Energies, 12(22), 4352.
https://doi.org/10.3390/en12224352

ISBN: 978-953-8246-22-7

Prediction of surface roughness with genetic programming

Matko Glučina [1*],Ammar Muminović [2]

[1].Tehnički Fakultet Sveučilišta u Rijeci, Vukovarska 58, 51000 Rijeka, Hrvatska, email: mglucina@riteh.hr,amuminovic@riteh.hr

Abstract: This paper represents the prediction of surface roughness after using an electrical machine. Using the given data set which is consisted of four parameters the depth of cut, feed rate, cutting speed, and surface roughness. The goal was to achieve a mathematical equation that can describe the regression function of trained and tested parameters. Python script consists of many libraries such as pandas, matplotlib, but most important are GP learn genetics which has implemented Symbolic Regression and sk learn which has tools to check R^2 score that is coefficient of determination and RMSE- Root Mean square Error. The regression model can be a well-established method for data analysis in various field applications.

Keywords: Artificial intelligence, Genetic Programing, Machine Learning, Python3, Surface roughness

1. Introduction

Engineers can experience two big practical problems in manufacturing processes. The first one is the determination merit of a process that can represent and yield desired product results, the second one is to increase and maximize the performance of manufacturing systems using resources available at that moment. Decisions made by engineers are not based only on their experience and knowledge of the profession but are also based on conventions during the phenomena that are placed during that specific processing. Researchers need to have the ability to overcome these kinds of problems with their knowledge and expertise. To resolve the above-mentioned difficulties, researchers derive models that can simulate conditions during the process of machining and can simulate cause and effect relationships between multiple factors and desirable product characteristics as well. Surface roughness is an important index for product selection and prerequisite for mechanical products. Prediction and acquiring targeted surface roughness quality is important for normal operation in manufacturing systems and workshops. This work aims to apply artificial intelligence (AI) and genetic programming (GP) for prediction and possible consumption of surface waste which will obtain a good idea of necessary replacements and service of materials if necessary for normal operation and work in manufacturing processes [3]. The problem of prediction of surface roughness can be solved using the AI method and GP algorithms. This kind of algorithm can offer a possible solution of creating a mathematical expression that can make a correlation between input and output data [1]. Many actions can disintegrate surfaces, one of them is milling. Milling has been proved as one of the most important machining processes that can improve some characteristics of the material such as fatigue strength and lower the possibility of corrosion of the material. Additionally, the roughness of the surface also affects friction of the surface itself, the ability to hold lubricant, and light reflection. Important parameters can be sorted as controllable and non-controllable. However, the existence of non-controllable cutting parameters is present, such as vibration, tool wear, machine motion. Many of the given research reveals that a lot of efforts are devoted to the determination best model for surface roughness prediction [2]. One of the methods for testing and determination for surface roughness can be GP. The authors of this paper will test and try the following method to get a possible solution for its regression.

2. Methodology

The second section of this paper represents the methodology used for this. Initially given data set is briefly described for research-specific GP settings and a candidate for quality evaluation solution. A method that is used for this research is named Symbolic regression. The application of this method is used to generate mathematical expressions which can be solved using GP [4]. The main principle of GP serves for generating equations that are formed in the shape of trees, [5] that are tested for the determination of solution quality. Relations between inputs $\boldsymbol{X} = [\boldsymbol{x_1}, \boldsymbol{x_2}, \dots, \boldsymbol{x_n}]$ and output $\boldsymbol{Y}$ are described from before mentioned generated trees [6]. The appliance of the equations to a given data set provides a prediction of values, which when compared to real values, have a small amount of deviation. Symbolic regression is a feed-forward neural network that can be characterized by a single output and the possibility of multiple inputs [7].
The solution of symbolic regression is an equation that consists of complex functions. A visual example of one solution is given in picture 1.

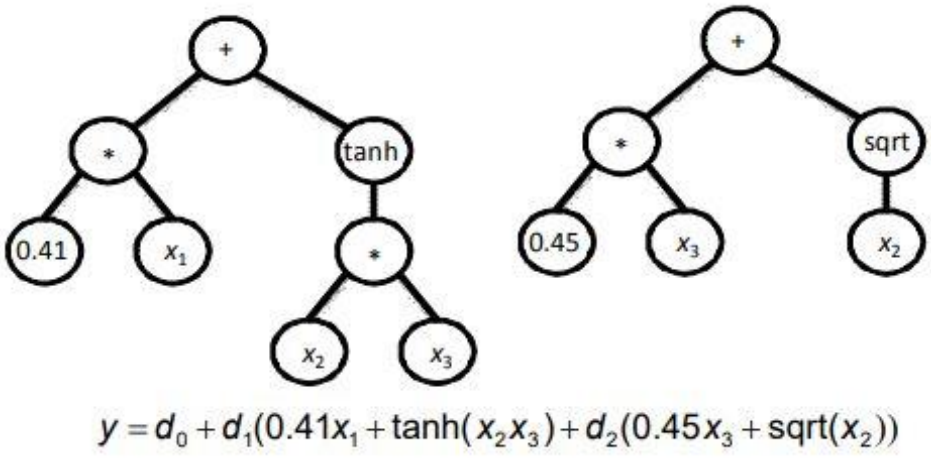


Figure 1. *Example of symbolic regression solution y [11]*

Evolutionary computing operations can be briefly explained as follows [8,9]:

- **reproduction** focus is on the performance of a single candidate solution. The candidate solution is then copied to the next population without any variations or modifications.

- **crossover** selects two random nodes from different trees and swaps them. The connection is made, but the root node in the first tree stays the same.

- **mutation** generates entire new trees in GP, those new trees are modified versions of previous trees.

Another important set of parameters are hyperparameters and their values which are described later in this section.
Data set within this research is publicly available on a website to be obtained from the repository by the University of Engineering & Technology Lahore. Fragment of a before-mentioned data set is presented in table 1. The data set used in this research constitutes three control variables i.e. feed rate, depth of cut, and cutting speed. The output variable in this experiment is surface roughness.

Table 1. Part of u*sed data set from Lahore university.*

DOC (MM)	CS (M/MIN)	FR (MM/REV)	SURFACE ROUGHNESS (µM)
1	70	0,06	1,01
1,5	70	0,06	1,4
2	70	0,06	2,05
1	80	0,06	0,816
1,5	80	0,06	0,828
2	80	0,06	1,726
1	80	0,08	0,599
2	90	0,1	1,3

As shown in Table 1, DOC, CS, and FR are input variables, and Surface roughness is the output variable. In total, we have 54 values for each of the parameters. Those parameters are categorized as training and testing parameters in the ratio of 80:20. 43 instances are in the Training set, and the other 11 remaining variables have a role for the test model which can obtain symbolic expression and calculate the coefficient of determination ($\mathbf{R^2}$) score. The entire data set is randomly shuffled, afterward, it's divided into a section for training each possible GP execution, and then testing data is set to prevent overfitting. $\mathbf{R^2}$ is a calculation for a value in a range from 0 to 1 $\left(\boldsymbol{R^2} \in [0,1] \right)$. Solutions are divided into 2 sets– real data in form y and model data $\hat{y}$. The meaning of $\mathbf{R^2}$ equals 1.0 is that there is no variance left unexplained between two compared sets, while the value that equals 0.0 has meant that none of the variances contains any fragment of real data y that is explained in the model data $\hat{y}$ [1].
The $\mathbf{R^2}$ score is calculated using the following mathematical equation which can be presented in the following form (1):

$$\boldsymbol{R^2} = 1 - \frac{\sum_{i=0}^{m}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{m}\left(y_i - \frac{1}{m}\sum_{i=0}^{m} y_i\right)^2}, (1)$$

where $y_i, \hat{y}_i$ are a number of samples data set used for testing and m represents the length of vectors. Defining precise values for hyperparameters is important because they define the regression. Even a little change in those parameters can cause drastic changes in results.

In this paper, standard values for hyperparameters are selected. The specified obtained model must be sufficiently precise, otherwise, the value of used hyperparameters has to be changed. If better parameters are selected, the result will also improve. The combination of hyperparameters used in this paper is shown in table 2.

Table 2. List of hyperparameters used in Phyton.

HYPERPARAMETER	1ST RUN	2ND RUN	3RD RUN
Population	10000	10000	10000
Generations	10	100	1000
Stopping criteria	0,01	0,01	0,01
p crossover	0,7	0,7	0,7
p subtree mutation	0,05	0,05	0,05
p hoist mutation	0,01	0,01	0,01
p point mutation	0,01	0,01	0,01
max. samples	0,9	0,9	0,9

"Population" represents the number of possible candidate solutions that are generated in each iteration. "Generation" is the number of times symbolic regression will be computed. "Crossover" is probabilities of EC operations with the possible probability of reproduction that equals the difference between 1.0 and the sum of some listed probabilities. "Stopping criteria" represents and defines the value of error that has to be the score to stop further executions. "Max. samples" is a percentage of candidate solutions that are used for the crossover and mutating operations[1].
After finishing training, the quality of obtained solutions is evaluated in the next section which represents results and discussion chapter.

3. Results and Discussion

This paragraph represents the results and discussion of a given task. The result is a symbolic expression of the surface roughness. The procedure for obtaining the symbolic expression is performed on the given datasheet which has been split into two modules which are training and testing portions. Portions are split in a ratio of 80:20, meaning that 80% of the data set was used for training (for symbolic regression) and the remaining 20% of the data set was for $\mathbf{R^2}$ score using earlier mentioned Equation (1). In each set of iteration in the GP algorithm, parameters were chosen from the pre-defined values given in table (2). Different values of generations gave a different value of $\mathbf{R^2}$ as shown in tables 1,2 and 3.
In the first case, we took a number of generations equal to 10, as shown in table 3.

Table 3. Number of generations is 10.

GENERATIONS	$\mathbf{R^2}$
10	0.3113

Regression based on these parameters is the fastest executing regression in this paper. The result of $\mathbf{R^2}$ is 0.3113 which is the highest $\mathbf{R^2}$ value of all examples. Furthermore, the expression we got from a regression that is the approximate equation is:

$$\mathbf{y} = X_0 * 0.826 \ , (2)$$

Where X_0 represents the depth of cut (DOC), and *y* is the surface roughness approximation.

In the next case, the same parameters are used except for the change in the number of generations, as shown in table 4. More generations include more algorithms to be executed by the program, therefore we expect a longer time of calculation.

Table 4. Number of generations is 100.

GENERATIONS	$\mathbf{R^2}$
100	-0,8996

Regression, in this case, is lower than in the case with 10 generations. Furthermore, regression contains a negative value, which is a result of poor parameter selection for approximating given surface roughness values.
An approximate equation in this example is shown below in equation 3:

$$\mathbf{y} = 0.996 \ , (3)$$

which is constant. Hence, this also gives us a bad approximation.

The last step is to evaluate the regression with 1000 generations and the result of $\mathbf{R^2}$ is shown in table 5.

Table 5. Number of generations is 1000.

GENERATIONS	R^2
1000	0,2639

As it can be seen from the attached results, once again the $\mathbf{R^2}$ value is not satisfactory. Even for 1000 generations, regression is having poor results. An approximate equation for this set of parameters is shown in equation 4.

$$\mathbf{y} = X_0 * 0.79 \quad , (4)$$

In the attached results with the given parameters, the conclusion is that the Symbolic regression method doesn't give a good approximate value.
Therefore, regression is repeated with different parameters of crossover, subtree mutation, and hoist mutation, however, the results are still not satisfactory.

4. Conclusion

Symbolic regression is a good method of analysis for mathematical equations and expressions for finding a model that fits best for given data set in terms of accuracy or simplicity. It randomly builds mathematical blocks that are trivial like addition, subtraction, multiplication, division in combination with trigonometry functions and constants like π x, y. In this case, symbolic regression is not a good method for approximating values of Surface roughness. The reason may be due to the lack of values in given data sets. For better symbolic regression results it is recommended to have a larger data set. Reasons for that include more precise measurements and better combinations of variables in the GP algorithm. The authors of this paper encourage other students and engineers to try other possible methods for achieving better results than those given in this specific work.

Acknowledgments

We would like to express our sincere gratitude to prof. dr.sc Zlatan Car and assistants Sandi Baressi Šegota, Ivan Lorencin, and Nikola Anđelić for their invaluable guidance and assistance through the course of this project. We were extremely fortunate to have had their suggestions and mentorship.

Reference

[1] Anđelić, Nikola, et al. "Estimation of covid-19 epidemiology curve of the united states using genetic programming algorithm." International Journal of Environmental Research and Public Health 18.3 (2021): 959.
https://doi.org/10.3390/ijerph1D030959
[2] Brezocnik, M., Kovacic, M., & Ficko, M. (2004). Prediction of surface roughness with genetic

programming. Journal of materials processing technology, 157, 28-36. https://doi.org/10.1016/j.jmatprotec.2004.09.004

[3] Benardos, P. G., and G-C. Vosniakos. "Predicting surface roughness in machining: a review." *International journal of machine tools and manufacture* 43.8 (2003): 833-844. https://doi.org/10.1016/S0890-6955(03)00059-2

[4] Walker, M." Introduction to genetic programming". Tech. Np: University of Montana, 2001.

[5] Cormen, TH, Leiserson, CE, Rivest, RL, et al. "Introduction to algorithms". Cambridge: MIT Press, 2009.

[6] Anđelić, N., Baressi Šegota, S., Lorencin, I., Mrzljak, V., & Car, Z. (2021). "Estimation of COVID-19 epidemic curves using genetic programming algorithm". *Health Informatics Journal*, *27*(1), 1460458220976728. https://doi.org/10.1177/1460458220976728

[7] Car, Zlatan, et al. "Modeling the spread of COVID-19 infection using a multilayer perceptron." *Computational and mathematical methods in medicine* 2020 (2020). https://doi.org/10.1155/2020/5714714

[8] Eiben, AE, Smith, JE. "Introduction to evolutionary computing", volume 53. Cham: Springer, 2003.,

[9] Koza, JR, Andre, D, Keane, MA, et al. Genetic programming III: Darwinian invention and problem solving, volume 3. Burlington: Morgan Kaufmann, 1999.

[10] Sipper, M, Fu, W, Ahuja, K, et al. "Investigating the parameter space of evolutionary algorithms". BioData Min 2018; 11(1): 2.

[11] Searson, D. P., Leahy, D. E., & Willis, M. J. (2011). Predicting the toxicity of chemical compounds using GPTIPS: a free genetic programming toolbox for MATLAB. In *Intelligent control and computer engineering* (pp. 83-93). Springer, Dordrecht.

ISBN: 978-953-8246-22-7

Face mask classification using MLP Classifier

Tin Ladić [1*], Ante Mandekić [2]

[1] Tehnički Fakultet Sveučilišta u Rijeci, Vukovarska 58, 51000 Rijeka, Hrvatska, tladic@riteh.hr, amandekic@riteh.hr.

Abstract: In the trying times of the Covid-19 pandemic a need was identified to regulate the usage of masks in public places. This can be done by detecting if a person is wearing a mask or not using a camera and a detection algorithm, the camera could be placed for example at the entrance to a venue or a building. The goal of this work is to create a detection algorithm using an MLP Classifier and achieve satisfactory accuracy. Image data preparation was done using Open Computer Vision library in python, and the detection and metrics of the results were done using the Scikit-learn python library. The best-achieved accuracy was 83.85%, and it was concluded that maybe the MLP Classifier was not the best method for detecting image data.

Keywords: Artificial intelligence, Face mask, Machine Learning, Multi-Layer Perceptron.

1. Introduction

In the light of the recent COVID-19 pandemic, a need was presented to detect if someone is wearing a mask or not. COVID-19 is a contagious disease caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). Since the spread of disease began in the Wuhan province, China in late 2019, according to [1], efforts have been made to implement different AI methods in modeling the spread of disease and patient outcome. Methods such as MLP [2,3] and genetic programming [4,5] were used to estimate the number of active cases for specific countries and on a global scale. Convolutional neural networks (CNN) were used on X-ray images in [6] for the classification of the patient`s clinical picture. Face mask detection would be useful for example at an entrance to a building or a venue. Omkar Gurav created a dataset [7] containing images of people with and without facemasks. There have been many implementations of AI algorithms on this dataset, including Convolutional neural networks (CNN) and transfer learning. This work focuses on implementing a Multi-layer perceptron classifier on the aforementioned dataset and getting the best accuracy from this method.

2. Methodology

The selected dataset [7] consists of two folders named “with_mask” and “without_mask”, these folders contain RGB images of people with or without a mask. The images must be converted to grayscale, and then to numeric matrices in order to prepare the image data for the MLP classifier. Open computer vision python library [8] was used to accomplish this, additionally, all images were converted to size 64×64. The prepared data was shuffled and then split into train and test matrices using “train_test_split” from Scikit-learn [9]. 25% of the data was used to test the MLP after training. Multilayer perceptron (MLP) is a type of fully connected, feed-forward artificial neural network (ANN), consisting of neurons arranged in layers. At least three layers make up MLP: an input layer, an output layer, and one or more hidden layers. The output layer consists of a single neuron, the value of which is the output of the MLP ANN [10]. The output neuron in this case represents the detected class, either with a mask or without a mask. The number of and sizes of hidden layers, as well as the type of activation function and alpha, were determined through trial and error to produce the best accuracy of classification. Five different MLPClassifire parameters were set for purpose of comparing activation functions. Two hidden layers

of 200 neurons each were placed for the test. The test showed that the best accuracy is obtained with the ReLU activation function that can be seen in Table 1.

Table 1. *Comparison of activation functions*

Parameters	tanh	identity	logistic	relu
solver:adam alpha:0.001	0.49814	0.49444	0.49073	0.80201
solver:adam alpha:0.00001	0.49073	0.60349	0.50873	0.61302
solver:adam alpha:0.03	0.49168	0.57967	0.50926	0.78083
solver:lbfgs alpha:0.001	0.75277	0.62361	0.73161	0.75436
solver:lbfgs alpha:0.0142	0.75913	0.62096	0.73848	0.76113

The activation function ReLU is set as the best choice for training the neural network to obtain the best possible accuracy. The following test compares the different hidden layers of neurons and their impact on accuracy that can be seen in Tabel 2.

Table 2. *Comparison of different hidden layers of neurons*

Parameters	10,68	256,302	400,20, 660	56,136, 100
solver:adam alpha:0.001	0.49021	0.61461	0.49073	0.66649
solver:adam alpha:0.00001	0.72525	0.66013	0.49085	0.76283
solver:adam alpha:0.03	0.63208	0.51402	0.49056	0.49073
solver:lbfgs alpha:0.001	0.64531	0.76761	0.75383	0.73213
solver:lbfgs alpha:0.0142	0.64478	0.75331	0.74483	0.75807

From a comparison of the hidden layers of neurons, it can be concluded that increasing the number of hidden layers and the number of neurons in them does not necessarily result in better accuracy. After conducting tests with different parameters, the best accuracy was achieved with the hyperparameters that can be seen in Table 3.

Table 3. *Hyperparameters*

Number of hidden layers	Sizes of hidden layers	Alpha	Activation function
8	128,256, 128,256, 512,256,128,64	0.021	“relu”

To help visualize the results training loss and a confusion matrix were plotted. The confusion matrix was plotted using the metrics library from Scikit-Learn [9].

3. Results and Discussion

The best accuracy achieved in this work after optimizing the hyperparameters is 83,85%. This is not satisfactory because other methods have produced accuracies as high as 99%, for example, Ritwek Khosla achieved this using Fastai [11]. The training loss curve can be seen in Figure 1.

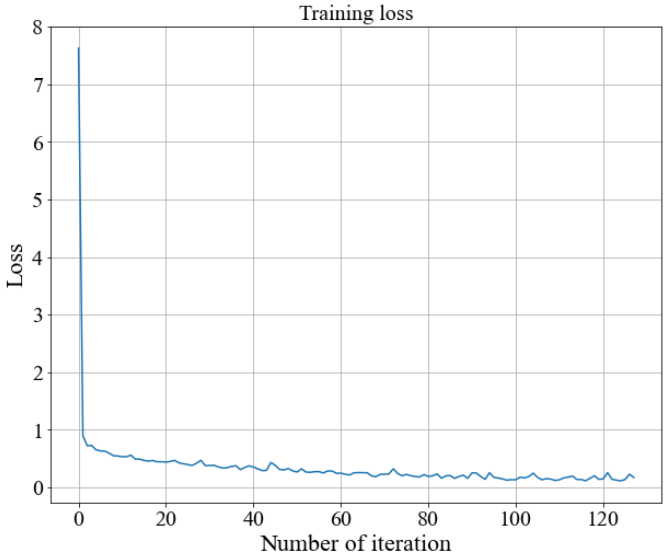


Figure 1. *Training loss curve*

The ANN converges relatively quickly with minor rises in the training loss curve. The confusion matrix is shown in Figure 2.

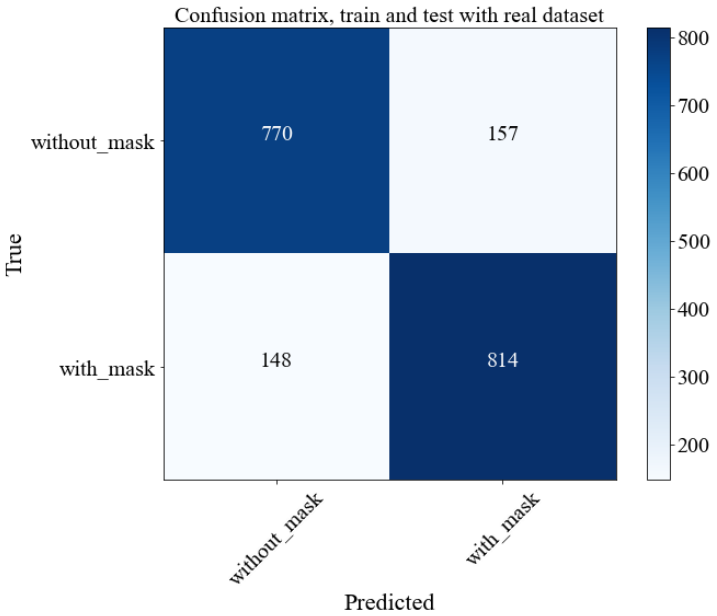


Figure 2. *Confusion matrix*

As can be seen from the confusion matrix, most of the data is classified correctly, but also there are a lot of false detections and undetected masks.

4. Conclusion

Face masks were successfully detected, but with limited accuracy. It may be possible to further improve the accuracy of the MLP Classifier in future work by using some better method of optimization for hyper-parameters, for example, Random search for hyper-parameter optimization [12]. If these methods do not yield better results, then it may be possible that MLP just is not well suited for image data classification. Many other methods can be used for face mask detection, like CNN [13] or support vector machine (SVM) [14].

Reference

[1] Musulin, J., Baressi Šegota, S., Štifanić, D., Lorencin, I., Anđelić, N., Šušteršič, T., Blagojević, A., Filipović, N., Ćabov, T., and Markova-Car, E. (2021). Application of Artificial Intelligence-Based Regression Methods in the Problem of COVID-19 Spread Prediction: A Systematic Review. International Journal of Environmental Research and Public Health, 18(8), 4287.
[2] Car, Z., Baressi Šegota, S., Anđelić, N., Lorencin, I., & Mrzljak, V. (2020). Modeling the spread of COVID-19 infection using a multilayer perceptron. Computational and mathematical methods in medicine, 2020.
[3] Šegota, S. B., Lorencin, I., Anđelić, N., Štifanić, D., Musulin, J., Vlahinić, S., Šušteršič, T., Blagojević, A., & Car, Z. (2021). Automated Pipeline for Continual Data Gathering and Retraining of the Machine Learning-Based COVID-19 Spread Models.
[4] Anđelić, N., Baressi Šegota, S., Lorencin, I., Mrzljak, V., & Car, Z. (2021). Estimation of COVID-19 epidemic curves using genetic programming algorithm. Health Informatics Journal, 27(1), 1460458220976728.
[5] Anđelić, N., Baressi Šegota, S., Lorencin, I., Jurilj, Z., Šušteršič, T., Blagojević, A., Protić, A., Ćabov, T., Filipović, N., and Car, Z. (2021). Estimation of covid-19 epidemiology curve of the united states using genetic programming algorithm. International Journal of Environmental Research and Public Health, 18(3), 959.
[6] Lorencin, I., Baressi Šegota, S., Anđelić, N., Blagojević, A., Šušteršić, T., Protić, A., Arsenijević, M., Ćabov, T., Filipović, N., and Car, Z. (2021). Automatic Evaluation of the Lung Condition of COVID-19 Patients Using X-ray Images and Convolutional Neural Networks. Journal of Personalized Medicine, 11(1), 28.
[7] Face Mask Detection Dataset, accessed: 28.05.2021; from: https://www.kaggle.com/omkargurav/face-mask-dataset
[8] Basic operations on images in OpenCV, accessed: 28.05.2021; from: https://docs.opencv.org/master/d3/df2/tutorial_py_basic_ops.html
[9] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
[10] Car, Zlatan, et al. "Modeling the spread of COVID-19 infection using a multilayer perceptron." *Computational and mathematical methods in medicine* 2020 (2020). https://doi.org/10.1155/2020/5714714
[11] Fastai Tutorial 1: Image Classification 99%val acc, accessed: 28.05.2021; from: https://www.kaggle.com/vanvalkenberg/fastai-tutorial-1-image-classification-99-val-acc
[12] Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." Journal of machine learning research 13.2 (2012).
[13] Chavda, Amit, et al. "Multi-stage cnn architecture for face mask detection." 2021 6th International Conference for Convergence in Technology (I2CT). IEEE, 2021. https://doi.org/10.1109/I2CT51068.2021.9418207
[14] Loey, Mohamed, et al. "A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic." Measurement 167 (2021): 108288. https://doi.org/10.1016/j.measurement.2020.108288

ISBN: 978-953-8246-22-7

Prediction of tool wear after machining

Filip Shahini [1*], Nikola Grgurić [1*]

[1*] Faculty of Engineering RITEH University of Rijeka, Vukovarska 58, 51000 Rijeka, Hrvatska, e-mail:

fshahini@riteh.hr, ngrguric1@riteh.hr

Abstract: The purpose of this article is to regress and predict tool wear from predefined datasets. Given datasets consists of six different parameters: experiment number, depth of cut (DOC, mm), cutting speed (CS, m/min), feed rate (FR, mm/rev), tool wear (µm), surface roughness (µm). The goal is to merge measurements, eliminate unnecessary elements from datasets, use genetic programming method to regress the value of the ‘’tool wear’’ column, predict values based on the given dataset and to evaluate regress quality using R^2 and RMSE metrics. Many of the python libraries were used to successfully solve a given problems. To point out, *pandas* for processing given datasets, *gplearn* for genetic programming algorithm, *sklearn* for metrics evaluation and split test and train samples and *matplotlib* for graphical representation of the result. In addition, comparison to Random Forest Regressor algorithm and Decision Tree Regressor algorithm was made in order to confirm the results. The conclusion of this research is that the two given datasets were not compatible for merging into one measurement so the regression and tool wear prediction were not usable.

Keywords: Genetic Programming, Regression, Tool wear prediction

1. Introduction

Prediction od tool wear after machining is a very important part of diagnostic process used to find relevant time to change the machine tool, what results in higher quality products and lower cost of machining process. Tool wear prediction has been researched with regression mathematical and artificial neural network (ANN) models. Flank wear was taken as the response (output) variable measured during milling, while cutting speed, feed and depth of cut are taken as input parameters. The experimental values have been used to develop a regression model and feed forward back propagation artificial neural network model for the prediction of tool wear with different numbers of nodes in the hidden layer. The experimentally determined tool wear values are compared with predicted values obtained from the regression and ANN models. The predictive ANN model is found to be capable of better predictions of tool flank wear within the range that they had been trained [1].
To preserve the diversity of knowledge, the authors used genetic programming method. Genetic programming has been successfully implemented in power systems [2] and COVID-19 disperse model [3,4]. Also, Genetic programming has been used in steelmaking and oil and gas industry [5,6].
The input parameters of dataset [1] as the same given authors of this paper. In addition, dataset in this research had experiment number and surface roughness column. Genetic programming is compared to other methods, Random Forest Regressor (RFR) and Decision Tree Regressor (DTR) to confirm the results of this research. In addition, R^2 and RMSE metrics were evaluated to display the quality of, so called, Symbolic Regressor and alternative methods.

2. Methodology

To start with, two datasets from Critical process parameters for machining process conducted at UET lab [7] were given to the authors to accomplish the solution. Both datasets had 27 samples and were given in format .xlsx file. The .xlsx file was not appropriate for analysis so the conversion into

suitable,.csv format, was done using Microsoft Office® Excel. In order to read data from .csv datasets, pandas library file was used.
In the given datasets 6 columns were given, but two of them, experiment number and surface roughness column, were rejected since they were not needed for the regression model. After converting to .csv format, a classification of data from dataset was made. Each dataset was assigned to a unique variable to be able to test and train every dataset individually or merged. Furthermore, input parameters (depth of cut, cutting speed, feed rate) were separated into variable so called X. On the other hand, variable y contains only output value, tool wear. After completion of the data preparation, additional action needed was to separate data for training and testing the model. To add, 80% of the dataset was randomly separated as training data, while 20% of the data was used as test data.
To solve the regression model and predict tool wear Genetic programming method was applied. Genetic programming (GP) is an evolutionary approach that extends genetic algorithms to allow the exploration of the space of computer programs. Like other evolutionary algorithms, GP works by defining a goal in the form of a quality criterion (or fitness) and then using this criterion to evolve a set (or population) of candidate solutions (individuals) by mimicking the basic principles of Darwinian evolution. GP breeds the solutions to problems using an iterative process involving the probabilistic selection of the fittest solutions and their variation by means of a set of genetic operators, usually crossover and mutation [8].
Genetic programming starts with an initial population of
randomly generated computer programs composed of functions. Each individual computer program in the population is measured in terms of how well it performs in the particular
problem environment. This measure is called the fitness
measure [9].
Simplified genetic programming algorithm is shown on Figure 1.

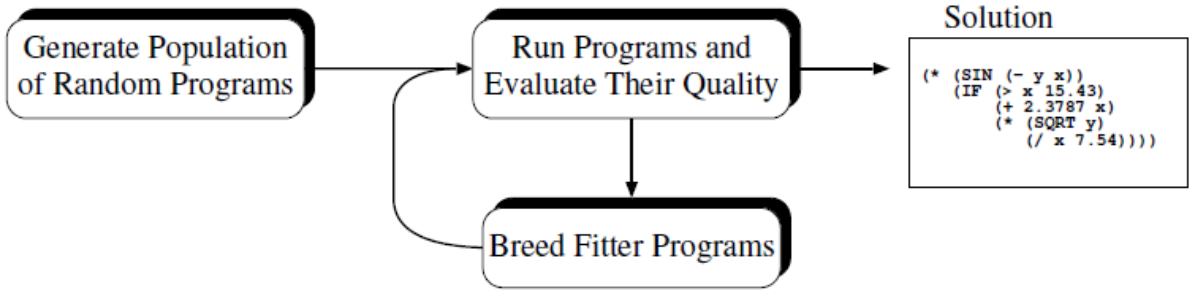


Figure 1. *Simplified genetic programming algorithm flowchart [10]*

Using python library *GPLearn*, main parameters which should be adjusted to maximize the regression and predictions are: population size (the number of programs in each generation), number of generations (the number of generations to evolve), p_crossover(the probability of performing crossover on a tournament winner) p_subtree_mutation(the probability of performing subtree mutation on a tournament winner), p_point_mutation(the probability of performing point mutation on a tournament winner) and max_samples(the fraction of samples to draw from X to evaluate each program on).
In achieving the best results, authors have concluded that the most important parameter is number of generations. By modifying population size, which could be assumed as an important parameter, evaluation was deteriorating. To add, training was too extensive with population greater than 10000. To conclude with, population size was set to be fixed at 5000. To add, above mentioned parameters were set to be fixed for training. Figure 2 shows genetic programming parameters relevant for each generation which are displayed on the Python kernel.

	Population Average		Best Individual			
Gen	Length	Fitness	Length	Fitness	OOB Fitness	Time Left
0	33.42	9.45624e+11	63	20.4871	41.6519	3.24m
1	19.75	103936	17	18.9939	50.3727	2.92m
2	27.80	18338.5	17	17.628	60.4119	3.05m

Figure 2. *Genetic programming algorithm kernel output*

To sum up, Decision Tree and Random Forest Tree were also used for training to confirm or refute the results of GP. Result of the metric parameters are shown as 2D graph, while the predicted value and test values for the largest generations were shown on a 3D scatter plot in correlation to as mentioned dataset input parameters.

3. Results and Discussion

Each dataset was used for trained individually and at last merged datasets was trained. As mentioned, results of the regression and prediction mostly depend on the number of generations. Therefore, different values of the generations, R^2 metrics, mean squared error values (RMSE metrics) were plotted to graphically show the dependence of metrics due to the number of generations. In addition, true and predicted values for each used method are displayed.

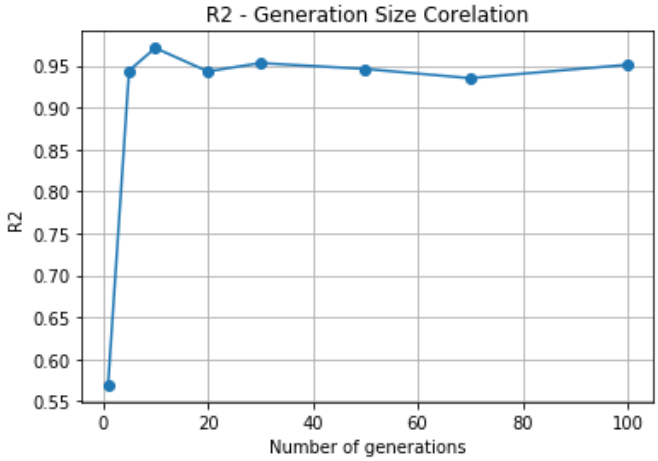


Figure 3. *Dataset A – R2 & generation size corelation*

Figure 3 shows the corelation of R^2 metrics and the number of generations of dataset A. Figure shows that the best value of R^2 metrics is within 10 generations and it is valued at 0.971.

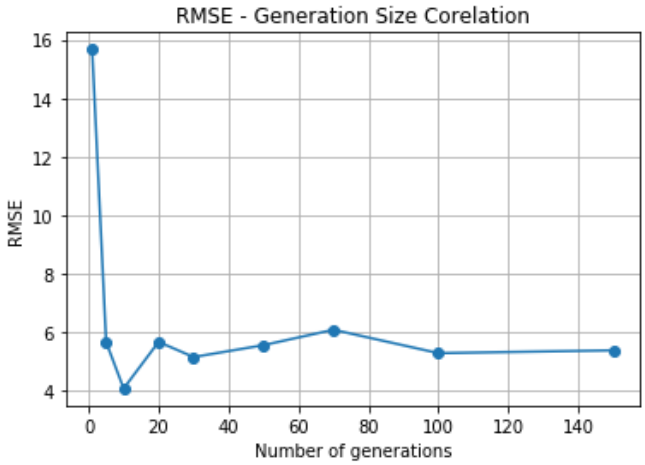


Figure 4. *Dataset A – RMSE & generation size corelation*

Figure 4 shows the dependency of RMSE factor due to generation size. As in case of R^2, the best value of RMSE factor is at generation size equal to ten.
Table 1 shows R^2 and RMSE metrics evaluated from Genetic Programming (GP), Decision Tree Regressor (DTR) and Random Forrest Regressor (RFR) for Dataset A. Genetic Programming has the best metrics evaluations and tool wear predictions.

Table 1. *Different methods comparison for Dataset A*

Metrics	GP	DTR	RFR
R^2	0.971	0.824	0.848
RMSE	4.082	9.991	9.284

True values: [85. 79. 30. 30. 80. 44.]
Predicted values Genetic programming:
[87.867, 74.442, 37.435, 37.5132, 80.747, 49.155]

Predicted values Decision Tree Regressor:
[80, 69, 48, 35, 69, 42,]

Predicted values Random Forest Tree
[81.5, 62.69, 33.63, 36.67, 66.65, 45.75]

Figure 5 shows comparison of true and GP predicted values in dependence of input parameters. Since predicted and true values do not vary a lot on a 3D plot, the colour difference may be hard to notice.

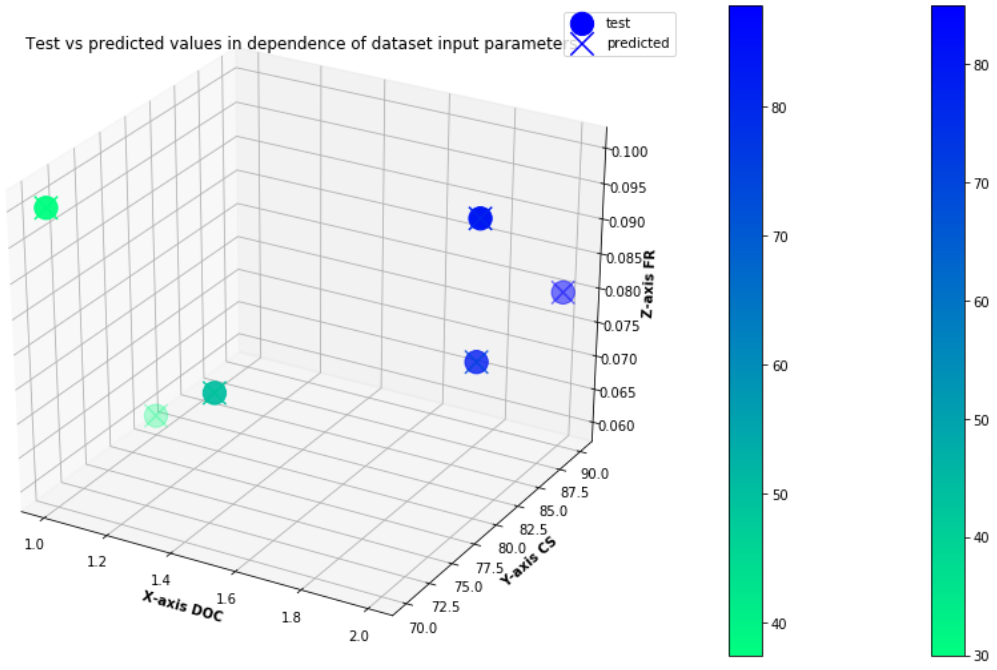


Figure 5. *Dataset A – Test vs predicted values Genetic Programming*

Same procedure has been made for dataset B. Figure 6 shows the corelation of R^2 metrics and the number of generations of dataset B. Figure shows that the best value of R^2 metrics is within 100 generations and it is valued at

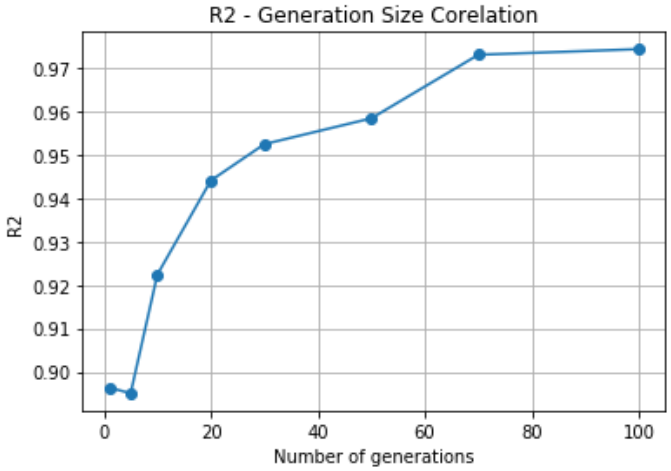


Figure 6. *Dataset B – RMSE & generation size corelation*

Figure 7 shows the dependency of RMSE factor due to generation size. As in case of R^2, the best value of RMSE factor is at generation size equal to hundred.

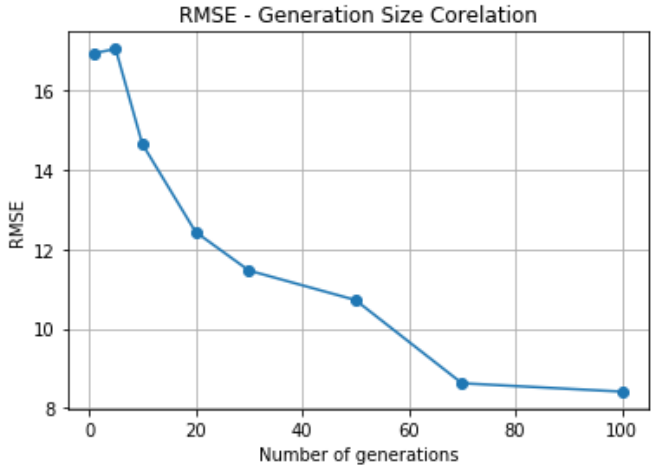


Figure 7. *Dataset B – RMSE & generation size corelation*

Table 2 shows R^2 and RMSE metrics evaluated from Genetic Programming (GP), Decision Tree Regressor (DTR) and Random Forrest Regressor (RFR) for Dataset A. Genetic Programming has the best metrics evaluations and tool wear predictions.

Table 2. *Different methods comparison for Dataset B*

Metrics	GP	DTR	RFR
R^2	0.974	0.943	0.937
RMSE	8.415	12.523	13.094

True values: [165, 162, 44, 46, 160, 108.]
Predicted values Genetic programming:
[170.45, 148.249, 45.729, 44.250, 162.66, 94.110]

Predicted values Decision Tree Regressor:
[161, 140, 43, 52, 140, 110,]

Predicted values Random Forest Tree

[166.12, 134.67, 43.52, 49.86, 142.94, 103.55]

Figure 8 shows comparison of true and GP predicted values in dependence of input parameters.

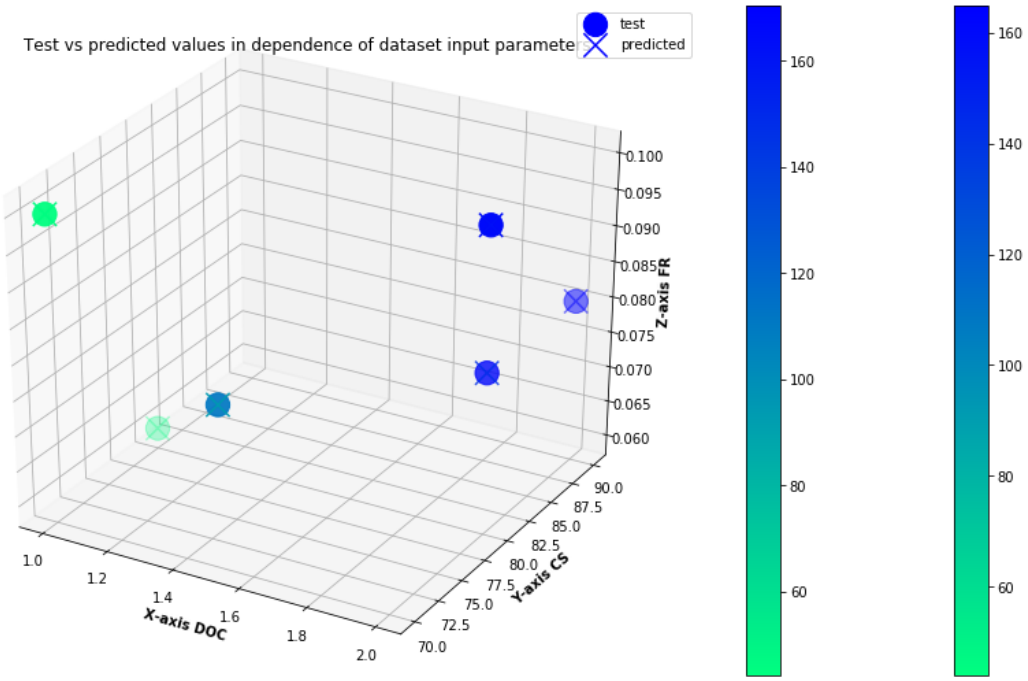


Figure 8. *Dataset B – Test vs predicted values Genetic Programming*

At last, merged datasets have been trained. As not expected, the genetic programming evaluation parameters were quite misleading. Figure 8 shows the corelation of R^2 metrics and the number of generations of merged datasets where R^2 metrics has negative values.

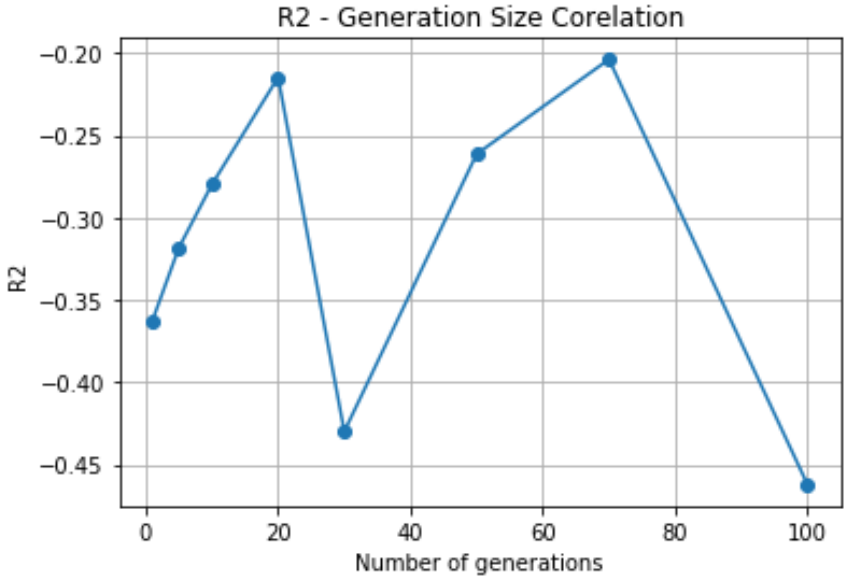


Figure 9. *Dataset A+B – RMSE & generation size corelation*

Figure 6 shows the dependency of RMSE factor due to generation size. RMSE metrics were also misleading, having quite high values.

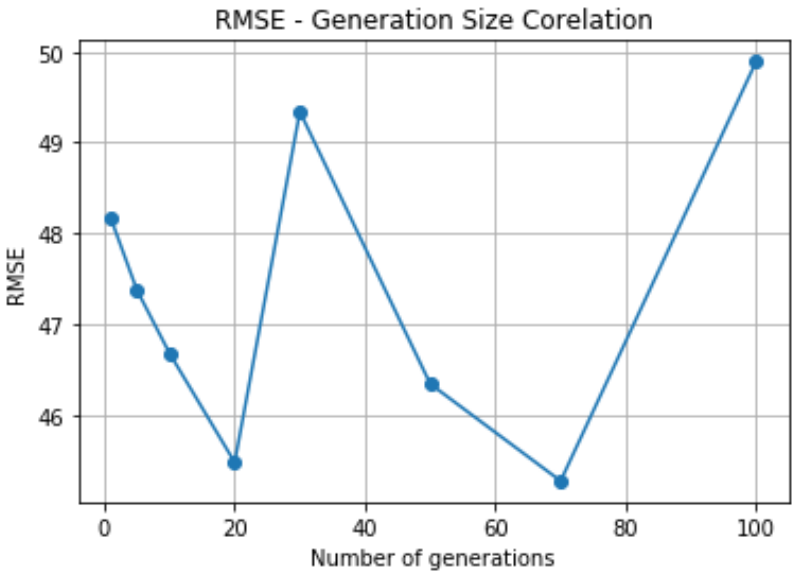


Figure 10. *Dataset A+B – RMSE & generation size corelation*

Table 3 shows R^2 and RMSE metrics evaluated from Genetic Programming (GP), Decision Tree Regressor (DTR) and Random Forrest Regressor (RFR) for Dataset A. All methods have bad metrics outcome.

Table 3. *Different methods comparison for Dataset A+B*

Metrics	GP	DTR	RFR
R^2	-0.20375	-0.5456	-0.2549
RMSE	45.269	51.2968	46.2218

True values: [46, 55, 165, 140, 97, 132, 130, 126, 110, 54,44]
Predicted values Genetic programming:
[45.5315, 59.179, 84.680, 70.713, 61.956, 64.196, 70.854, 64.252, 56.737, 56.737, 55.514]

Predicted values Decision Tree Regressor:
[30, 114, 108, 103, 50, 60, 63, 83, 108, 108, 108]

Predicted values Random Forest Tree
[34.483, 95.982, 103.35, 83.467, 54.625, 82.4875, 72.63, 72.939, 96.316, 96.316, 92.584]

Figure 10 shows comparison of true and predicted values in dependence of input parameters. Predicted and true values are not in corelation and genetic programming nor other methods could not be used for given dataset.

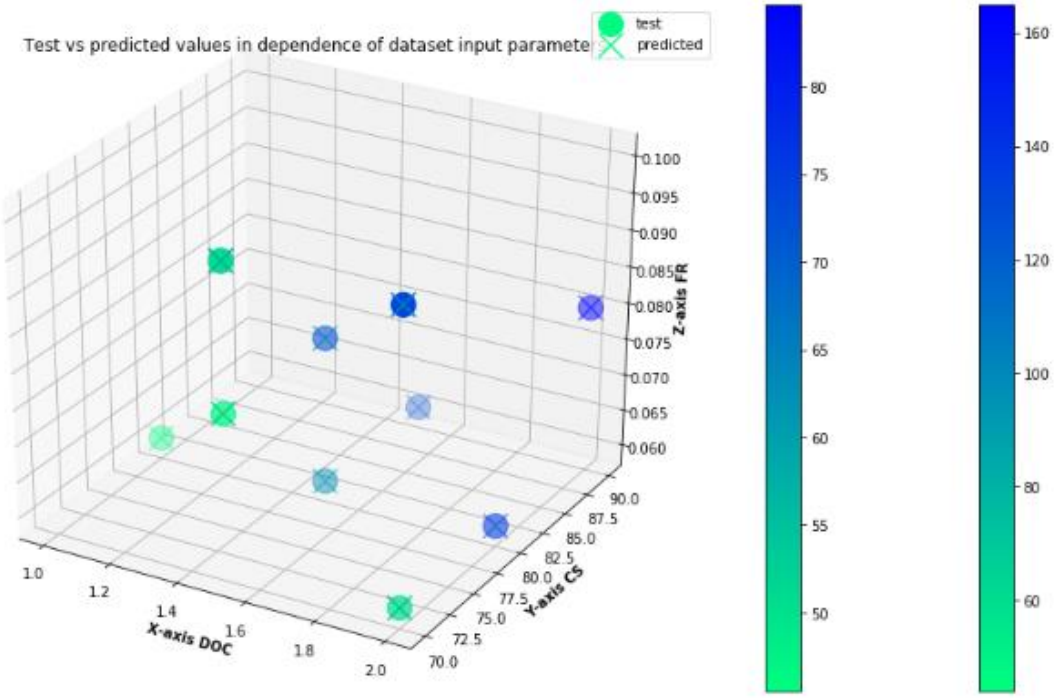


Figure 11. *Dataset A+B – Test vs predicted values Genetic programming*

4. Conclusion

After the experiment with datasets and trying other methods, it can be concluded that the merged dataset was badly defined and genetic programming could not be applied. Also, the prediction of the tool wear cannot be taken as correct. In addition, when training single dataset, it can be assumed that the symbolic regressor does not depend on the number of generations. The best metric and prediction were not at the largest generation size while training on dataset A. To add, the authors have quite a thrill working on this project and are looking forward to implementing genetic programming on another, better defined, dataset.

Acknowledgments
We thank assistant Sandi Baressi Šegota for his help during the work on this article.
"*One general law, leading to the advancement of all organic beings, namely, multiply, vary, let the strongest live and the weakest die.*"
Charles Darwin, On the Origin of Species (1859)

Reference
[1] Palanisamy, P., Rajendran, I. & Shanmugasundaram, S. "Prediction of tool wear using regression and ANN models in end-milling operation". *Int J Adv Manuf Technol 37, 29–41 (2008).*
https://doi.org/10.1007/s00170-007-0948-5
[2] Anđelić, N., Baressi Šegota, S., Lorencin, I., & Car, Z. "Estimation of gas turbine shaft torque and fuel flow of a CODLAG propulsion system using genetic programming algorithm". *Pomorstvo, 34(2), 323-337* (2020).
https://doi.org/10.31217/p.34.2.13
[3] Anđelić, N., Baressi Šegota, S., Lorencin, I., Mrzljak, V.
Car, Z.*" Estimation of COVID-19 epidemic curves using genetic programming algorithm". Health Informatics Journal, 27(1), (2021)*
https://doi.org/10.1177/1460458220976728

[4] Anđelić, N., Baressi Šegota, S., Lorencin, I., Jurilj, Z., Šušteršič, T., Blagojević, A., Protić, A., Ćabov, T., Filipović, N. and Car, Z. *"*Estimation of covid-19 epidemiology curve of the United States using genetic programming algorithm*." Journal of Environmental Research and Public Health 18(3),959, (2021)*
https://doi.org/10.3390/ijerph18030959
[5] Kovačič, M., Župerl, U. "Genetic programming in the steelmaking industry". *Genet Program Evolvable Mach 21, 99–128* (2020).
https://doi.org/10.1007/s10710-020-09382-5
[6] Luchian H., Băutu A., Băutu EGenetic Programming Techniques with Applications in the Oil and Gas Industry. In: *Cranganu C., Luchian H., Breaban M. (eds) Artificial Intelligent Approaches in Petroleum Geosciences. Springer, Cham*. (2015)
https://doi.org/10.1007/978-3-319-16531-8_3
[7] Navid Usama, Muhammad , " Critical process parameters for machining process conducted at UET lab."(2020).Accessed: 28.04.2021;
https://www.kaggle.com/muqadir1/machining?fbclid=IwAR0RTI3iLhWkSYUhU3OhFNbZGjySSXhzz07Bz3GSl9laInhqbCmonY1G7cs
[8] Vanneschi L., Poli R. Genetic Programming — Introduction, Applications, Theory and Open Issues. (2012) In: *Rozenberg G., Bäck T., Kok J.N. (eds) Handbook of Natural Computing*
https://doi.org/10.1007/978-3-540-92910-9_24
[9] Koza, J.R., "Genetic programming as a means for programming computers by natural selection." (1994) *Stat Comput 4, 87–112*
https://doi.org/10.1007/BF00175355
[10] O'Neill, M. Riccardo Poli, William B. Langdon, Nicholas F. McPhee: "A Field Guide to Genetic Programming." (2009). https://doi.org/10.1007/s10710-008-9073-y

ISBN: 978-953-8246-22-7

Investigation of the association between polymorphisms in the circadian CLOCK and NPAS2 genes and cancer by using methods of AI

Ana Mioč[1], Barbara Fabulić[1], Jelena Musulin[2], Daniel Štifanić[2], Elitza Markova-Car[1]*

[1]University of Rijeka, Department of biotechnology, Radmile Matejčić 2, 51000 Rijeka, Croatia; ana.mioc@student.uniri.hr (A.M.),
barbara.fabulic@student.uniri.hr (B.F.)
[2]Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka; jmusulin@riteh.hr (J.M.), dstifanic@riteh.hr (D.Š.)
* Correspondence: elitza@biotech.uniri.hr (E.M.C.)

Abstract: Circadian rhythm is a natural process in all living organisms that regulates the sleep–wake cycle and repeats roughly every 24 hours. On molecular level, that process is controlled by so-called clock genes. Disruption of circadian rhythms or expression of clock genes is emerging as a novel and potentially modifiable cancer risk factor. Single nucleotide polymorphism (SNP) stands for single base change in a DNA sequence, with an usual alternative of two possible nucleotides at a given position. In this study, the data from case-control studies containing available genotype frequencies of the SNPs in two clock genes (CLOCK and NPAS2) were collected. Based on that data, the association between genetic variations in clock genes and the risk of developing cancer was investigated. Furthermore, Artificial Intelligence (AI) algorithm was developed to predict the type of cancer (breast or mixed).

Keywords: Artificial Intelligence, Cancer, Circadian rhythm, Clock genes, SNP

1. Introduction

Circadian rhythms are daily oscillations in metabolic, psychological and physiological processes in all living organisms. Those rhythms are endogenously driven during the evolution due to day-night changes in nature with a period length of about 24-h. Deregulation of these rhythms is associated with a number of pathological conditions including depression, diabetes, metabolic syndrome and cancer. Those conditions could appear due to deregulations in so-called clock genes which control circadian rhythm on the molecular level by transcriptional/translational feedback loops [1-2]. Single nucleotide polymorphism (SNP) stands for single base change in a DNA sequence, with an usual alternative of two possible nucleotides at a given position. SNPs could appear in coding regions of genes, regions between genes or non-coding regions of genes [3]. Current studies which are focused on observing the impact of SNPs in clock genes on the risk of developing cancer are using standard statistical tools to predict the outcome.

The aim of this study was developing Artificial Intelligence (AI) algorithm which can be used to find a correlation between SNPs in two main clock genes (NPAS2 and CLOCK) and cancer. Algorithms like Support Vector Machine (SVM) and K-Nearest Neighbours (KNN) are commonly used in various fields of medicine and computing biology because of their high precision, ability to deal with large databases and versatility in modelling diverse data sources [4-6].

2. Methodology

Detailed analysis of the literature available online from 2008. till 2021. was conducted. Eligible studies were considered those who had, within their data, detailed information about genotype of the respondents with enquired SNP. Data for AI was retrieved from [7] Supplementary Table S3, using data

only for CLOCK and NPAS2 genes. Since the data from that study only contains studies conducted until 2015., the table was complemented with updated information from additional studies up to 2021. After collecting all the data, they were put together and the frequency of the alleles belonging to corresponding SNP was calculated as shown in Figure 1. To assess susceptibility of developing cancer, if having any of the studied SNPs, the statistical analysis using Website for Statistical Computation: VassarStats (http://vassarstats.net/) was conducted. Calculated parameters were Odds ratios (OR), 95% confidence interval (CI) and Pearsons chi-square p (χ^2) value. Criteria used for statistical significance of p-value was α=0.05. In addition, the protective role or increased predisposition of genotypes under dominant, codominant, recessive and overdominant model was calculated in order to assess if any of those in particular had impact on cancer development. Models which have had 95% CI upper limit under 1 were found to be associated with decreased predisposition, while those who have had p- value equal or lower than 0.05 were found to be associated with either decreased or increased predisposition of developing cancer.

		GENOTYPE (N)							GENOTYPE (N)			
Study	Total	Case			Total	Control			Case		Control	
rs2305160 A>G		AA	AB	BB		AA	AB	BB	A	B	A	B
Madden 2014	522	221	232	69	546	253	227	66	674	370	727	441
rs3749474 (C>T)												
Zienolddiny 2013	542	223	251	68	596	228	260	108	697	387	716	476
Benna 2018	162	62	80	20	608	259	266	83	204	120	784	432

Figure 1. Example of genotype data collection for CLOCK and NPAS2 polymorphisms, rs represents SNP while the name underneath stands for applied study.

In this research, two AI algorithms were used: SVM and KNN. The KNN algorithm is a non-parametric algorithm that classifies new input data into one of the classification classes [8]. The algorithm is simple and easy to implement, which is one of its advantages. The SVM, in terms of classification, uses the highest possible margin to find a hyperplane of separation between different classification classes [9] SVM for linearly separable binary sets works by creating a hyperplane that divides all training vectors into two classes (class1 = breast cancer and class2 = mixed cancer).

3. Results and Discussion

Even though the potential relationship between polymorphisms in clock genes and cancer risk has only recently emerged as an interesting field of research, here, it is affirmed that the connection does exist and that it can be substantially significant. Out of all the investigated variants it was found that several SNPs under given genotype model were associated with lower susceptibility to developing cancer. In Table 1. are shown inheritance models of SNP rs11133373 for the CLOCK gene and rs2305160 for NPAS2 genes. Genetic variations in those genes are already known to have an impact on cancer, and those SNPs in particular [10-11]. For rs11133373 was found that CA genotype under overdominant model was associated with decreased predisposition to cancer (CC+AA vs. CA: OR=0.88; 95%CI 0.78-0.99, p=0.03) and that presence of the A allele also has protective role under dominant model (CC vs. CA+AA: OR=0.84; 95%CI 0.75-0.95, p=0.005). Protective role of the A allele was also confirmed by comparing inheritance models per-allele (C vs. A: OR=0.89; 95%CI 0.81-0.97, p=0.01). As for rs2305160, it is associated with decreased risk under dominant (GG vs. GA+AA: OR=0.78, 95%CI 0.72-0.84, p<.0001), codominant (GG vs. AA: OR=0.49, 95%CI 0.44-0.55, p<.0001) and recessive (GA+GG vs. AA: OR=0.51, 95%CI 0.45-0.56, p<.0001) model.

Table 1. Example of calculated models for each genotype with corresponding statistical data of two selected SNPs.

SNP	N	Case/Control	Comparisons	OR	95%CI	p value
rs11133373	2	2179/2125	**C vs. A**	**0.89**	**0.81-0.97**	**0.01**
		codominant	CC vs. AA	0.82	0.68-1.01	0.07
		dominant	**CC vs. CA + AA**	**0.84**	**0.75-0.95**	**0.005**
		recessive	CC + CA vs. AA	0.90	0.74-1.09	0.29
		overdominant	**CC+AA vs CA**	**0.88**	**0.78-0.99**	**0.03**
		codominant	CA vs. AA	0.98	0.80-1.19	0.82
rs2305160	10	4622/4595	G vs.A	1.08	1.02-1.14	0.01
		codominant	**GG vs.AA**	**0.49**	**0.44-0.55**	**<.0001**
		dominant	**GG vs GA + AA**	**0.78**	**0.72-0.84**	**<.0001**
		recessive	**GG +GA vs. AA**	**0.51**	**0.45-0.56**	**<.0001**
		overdominant	GG+AA vs.GA	1.14	1.05-1.23	0.001
		codominant	GG vs.GA	0.94	0.86-1.02	0.14

To expand our search, as said before, dataset from [7] was used. 80% of data points were used as a training set, while 20% of them were used as a testing set. Afterwards, binary classification was performed. Optimal results for SVM were given by the Linear kernel function and C = 1 and as for KNN, optimal results were achieved with value of variable k = 3. Here, the Accuracy and Precision values of different AI algorithms are compared which is shown in Figure 2. It can be noticed that the Accuracy of 94.44% is the same for both algorithms. On the contrary, the Precision value is higher when the KNN algorithm is used with the value of 99.99%.

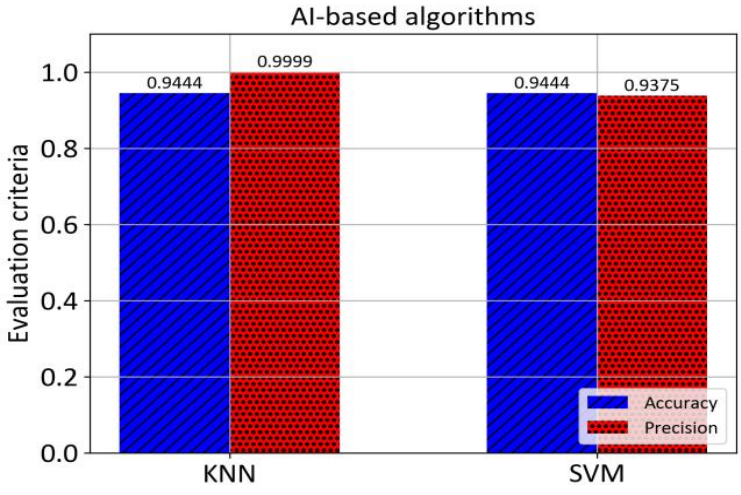


Figure 2. Accuracy and Precision values for KNN and SVM algorithm

4. Conclusion

The present work proved that the association between clock genes polymorphisms and risk of cancer does exist. Also, it showed that the relationship could be useful for future investigations of cancer predisposition and its prediction. Merging the fields of biotechnology and engineering has also given some valuable insight in the field of cancer risk assessment. Nevertheless, this study has its limitations. Because the parameters strictly related to the onset of the disease cannot be accurately determined, and the initial distribution of parameters is relatively unknown, the implementation of KNN and SVM has proven to be successful with satisfactory values of performance measures.

Acknowledgments
This research has been has supported by the University of Rijeka scientific grant uniri-biomed-18-257.

Reference

[1] Korkmaz, T., Aygenli, F., Emisoglu, H., Ozcelik, G.,Canturk, A., Yilmaz, S., & Ozturk, N. (2018).Opposite carcinogenic effects of circadian clockgene BMAL1. Scientific reports, 8(1), 1-11., https://doi.org/10.1038/s41598-018-34433-4
[2] Markova-Car, E. P., Jurišić, D., Ilić, N., & Pavelić, S. K. (2014). Running for time: circadian rhythms and melanoma. Tumor biology, 35(9), 8359-8368., https://doi.org/10.1007/s13277-014-1904-2
[3] Vignal, A., Milan, D., SanCristobal, M., & Eggen, A. (2002). A review on SNP and other types of molecular markers and their use in animal genetics. Genetics selection evolution, 34(3), 275-305. https://doi.org/10.1186/1297-9686-34-3-275
[4] Lee, B. H., & Scholz, M. (2006). A comparative study: Prediction of constructed treatment wetland performance with k-nearest neighbors and neural networks. Water, Air, and Soil Pollution, 174(1), 279-301.
https://doi.org/10.1007/s11270-006-9113-2
[5] Musulin, Jelena, et al. "Comparison of Three Artificial Intelligence Algorithms for Sepsis Prediction." 2020.
[6] Vieira, S. M., Mendonça, L. F., Farinha, G. J., & Sousa, J. M. (2013). Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients. Applied Soft Computing, 13(8), 3494-3504.
https://doi.org/10.1016/j.asoc.2013.03.021
[7] Benna, C., Helfrich-Förster, C., Rajendran, S., Monticelli, H., Pilati, P., Nitti, D., & Mocellin, S. (2017). Genetic variation of clock genes and cancer risk: a field synopsis and meta-analysis. Oncotarget, 8(14), 23978.
https://doi.org/10.18632/oncotarget.15074
[8] Zhang, Z. (2016). Introduction to machine learning: k- nearest neighbors. Annals of translational medicine, 4(11).
doi: https://doi.org/10.21037/atm.2016.03.37
[9] Soman, K. P., Loganathan, R., & Ajay, V. (2009). Machine learning with SVM and other kernel methods. PHI Learning Pvt. Ltd..
[10] Zhu, Y., Stevens, R. G., Hoffman, A. E., Fitzgerald, L. M., Kwon, E. M., Ostrander, E. A., ... & Stanford, J. L. (2009). Testing the circadian gene hypothesis in prostate cancer: a population-based case-control study. Cancer research, 69(24), 9315-9322.
[11] Wang, B., Dai, Z. M., Zhao, Y., Wang, X. J., Kang, H. F., Ma, X. B., ... & Dai, Z. J. (2015). Current evidence on the relationship between two common polymorphisms in NPAS2 gene and cancer risk. International journal of clinical and experimental medicine, 8(5), 7176.
https://doi.org/10.1007/s11033-012-2018-9

ISBN: 978-953-8246-22-7

Brain tumor detection based on MRI images using multilayer perceptron

Marko Mataija [1], Dorijan Sablić-Nemec [1*]

[1] Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Hrvatska, e-mail: mmataija@riteh.hr, dnemec@riteh.hr.

Abstract: The problem of brain tumor detection was approached with the classification of MRI images via the application of a multilayer perceptron, feedforward artificial neural network. Dataset was artificially expanded for training needs and some corrections were done after which images were processed using the Gaussian blur and threshold method. The experiment regarding the fine tuning of the main parameters and neural network architecture was conducted and lead to satisfying results upon application of optimal parameters and architecture.

Keywords: Artificial intelligence, Brain tumor, Machine Learning, Multilayer perceptron.

1. Introduction

Rapid development of computer vision and artificial intelligence leads to numerous innovations in a broad spectrum of fields. These innovations can overcome issues in situations where human error could lead to negative consequences in fields of quality control, inspection, and such. In medical fields, errors like these could lead to devastating consequences due to misdiagnosis caused by a failure in recognizing a defining feature or pattern on some sort of scan or a medical result. It is in human nature to get tired upon executing a repetitive task for an extended period of time. Artificial intelligence is capable of helping human operators with monotonous but important tasks, in some cases even replacing the operator. Many applications of artificial intelligence systems based on multilayer perceptron have been successfully deployed in fields of medicine [1,2] and energetics [3,4,5,6].

To this date, several artificial intelligence decision-making and medical condition diagnostic systems have been developed and proposed in the medical industry, even by some technological giants such as Siemens [7], Philips [7], and many others. Implementations of similar systems are found in numerous medical fields; medical imaging, cardiovascular, hematology, and many others.

The subject of this research is to propose a machine learning model for the detection of brain tumors based on a patient's MRI images. Some of the techniques previously proven to be effective in tasks like this are support vector machines, fuzzy clustering means, and artificial neural networks. These techniques are used for regional segmentation, which is needed to extract the information from a given MRI image. [8]

According to Damodharan and Raghavan, methods of using neural network based classifiers have given an accuracy of 83%. [9] Support vector machine based classifier has given researchers Alfonse and Salem the accuracy of 98,9% due to the accuracy improvement induced by using fast Fourier transform for feature extraction and further reduction of features due to application of Minimal-Redundancy-Maxima-Relevance method. Bidirectional Associative Memory (BAM) Artificial Neural Network (ANN) used as a classifier and segmentation tool whilst utilizing texture-primitive features has given Sharma et al. claimed stellar accuracy of 100%. [9]

Our attempt will utilize multilayer perceptron (MLP) feedforward Artificial Neural Network (ANN) as a classifier.

2. Dataset

The dataset used for this model training is a series of pre-classified brain MRI images where qualified practician classified the imagery based on a binary approach, is the tumor present or not. Specifics regarding the tumor type or stage were ignored. Images are of a high variety of sizes and qualities, also not all backgrounds were black but some were white. According to the author of the dataset, there is no guarantee of image source regarding the patient, some may be scans from the same patient.

Regarding the scans, there are few types in practice, most commonly T1, T2 weighted, and Flair scans. T1-weighted images produce the image where tumor tissue is displayed as a brighter section of an image, whereas in T2-weighted the tumor tissue is displayed as a darker section. Fluid-attenuated inversion recovery (*i.e. Flair)* is used for suppression of the effect that cerebrospinal fluid brings to the image. [10] T1 weighted images are commonly used in brain tumor detection since this method allows for the easier distinguishment of necrotic core and active cell region. [11]

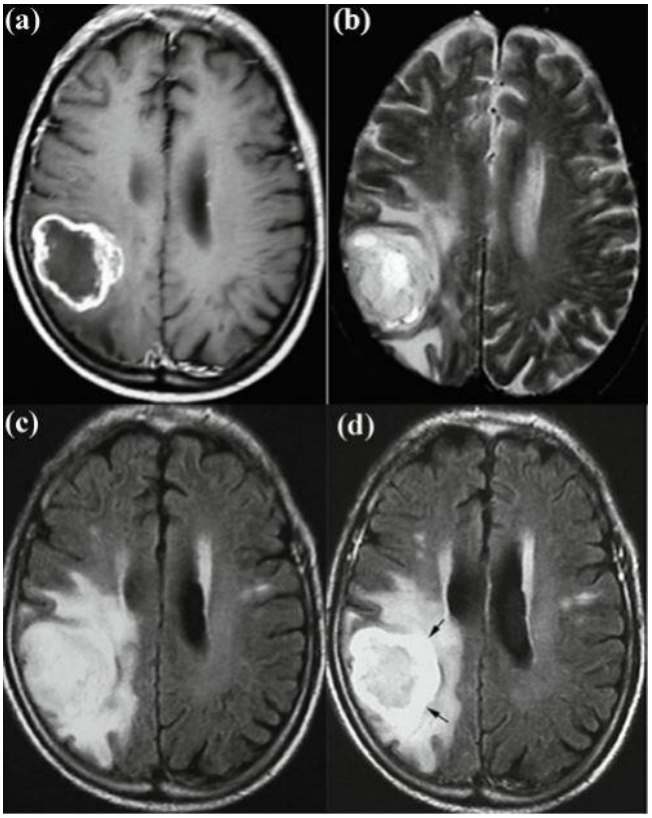


Figure 1. *(a) T1-weighted, (b) T2-weighted, (c) Flair, (d) Flair with contrast enhancement.* [12]

Large dataset diversity is not wanted therefore the images had to be processed and adjusted to conform to a single type. This dataset also had a problem of low image count, only 253 images. This is a problem that can manifest in a falsely over-accurate model which was confirmed by confusion matrix stating a 100% accuracy with default MLP model parameters using only one hidden layer with 100 neurons.

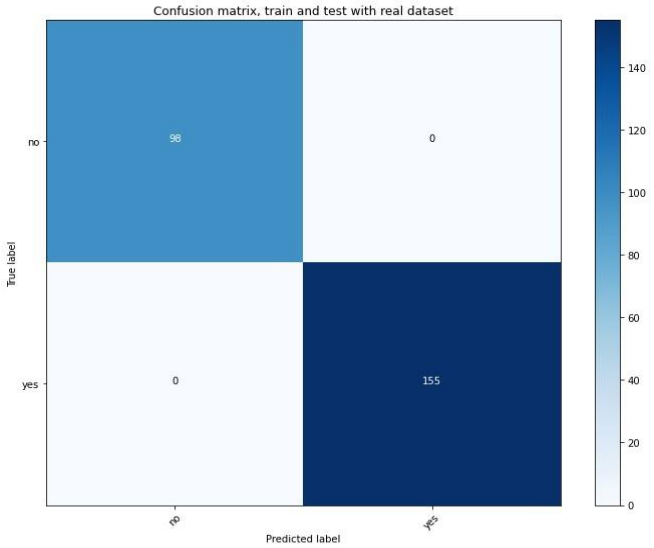


Figure 2. *Confusion matrix of initial training.*

This problem can be approached by artificially expanding the dataset by mirroring the images and rotating them in multiple positions. Dataset expansion using this technique has given the 2669 image dataset from its initial size of only 253 images.

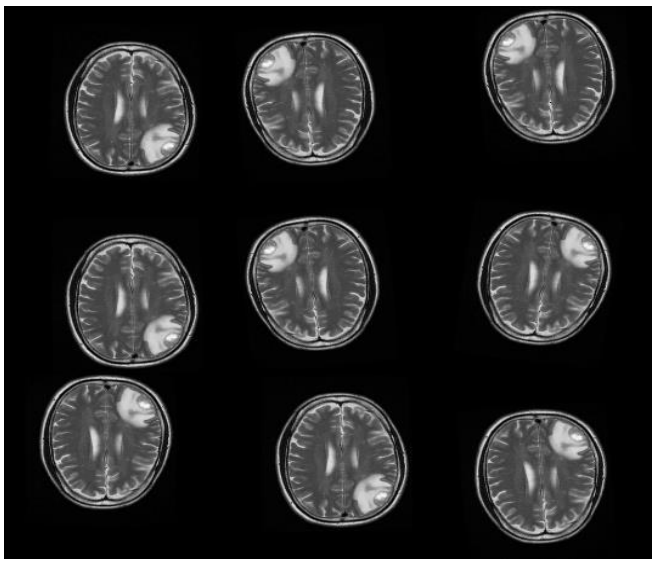


Figure 3. *Example of dataset expansion.*

For model training, each of the dataset images were transformed to a grayscale 64X64 array.

3. MLP classifier main parameters comparison

There are several available solvers and activation functions used in MLP classification via MLPClassifier function of the scikit-learn Python library. For each of the activation functions performance of each solver will be compared so the optimal combination of solver and activation

function can be determined. Every combination in this run of experiments will use 2 hidden layers, each composed of 100 neurons.

Linear *identity* activation function is used in the first run of MLP classification. As the function is linear, unlike the dataset, which is highly nonlinear, the expected outcome is that this method will not be effective in the task.

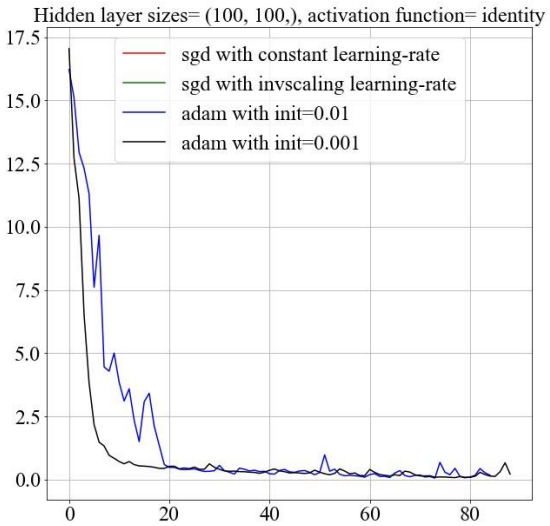


Figure 4. *MLPClassifier training loss for identity activation function.*

It is noticeable that *sgd* solver couldn't converge due to activation function linearity, therefore the result is not displayed.

Table 1. *MLPClassifier training results for identity activation function.*

Solver	Iterations	Training loss	Testing score
sgd - constant	200	nan	0.387352
sgd - invscaling	200	nan	0.387352
adam - init=0.01	57	0.242794	0.648221
adam - init=0.001	118	0.039542	0.683794
lbfgs - alpha=0.0001	200	0.153843	0.652174
lbfgs - alpha=0.00001	200	0.116240	0.628458

Next combination to be initiated was the implementation of *tanh* activation function.

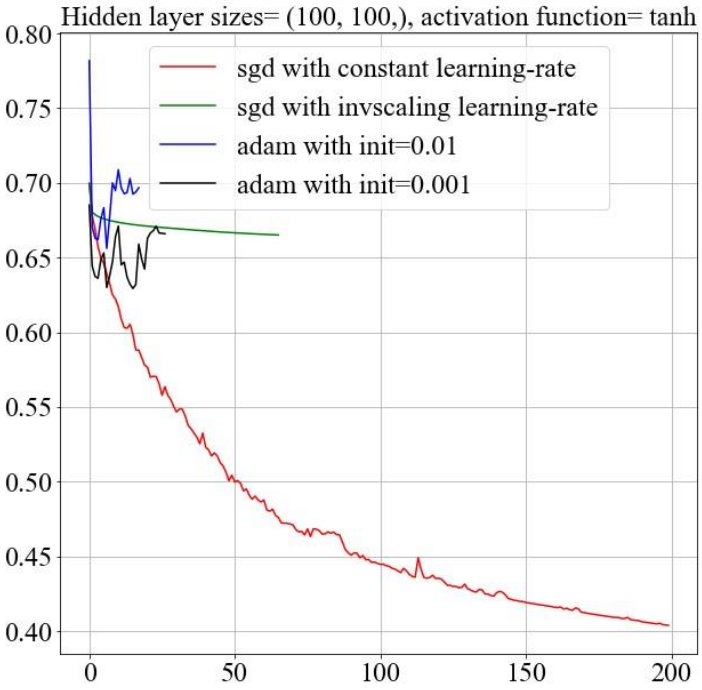


Figure 5. *MLPClassifier training loss for tanh activation function.*

Table 2. *MLPClassifier training results for tanh activation function.*

Solver	Iterations	Training loss	Testing score
sgd - constant	200	0.396053	0.679842
sgd - invscaling	68	0.657834	0.592885
adam - init=0.01	30	0.696097	0.612648
adam - init=0.001	16	0.674090	0.450593
lbfgs - alpha=0.0001	200	0.063070	0.735178
lbfgs - alpha=0.00001	200	0.073599	0.691700

The following method is using the *logistic*, *sigmoid* activation function.

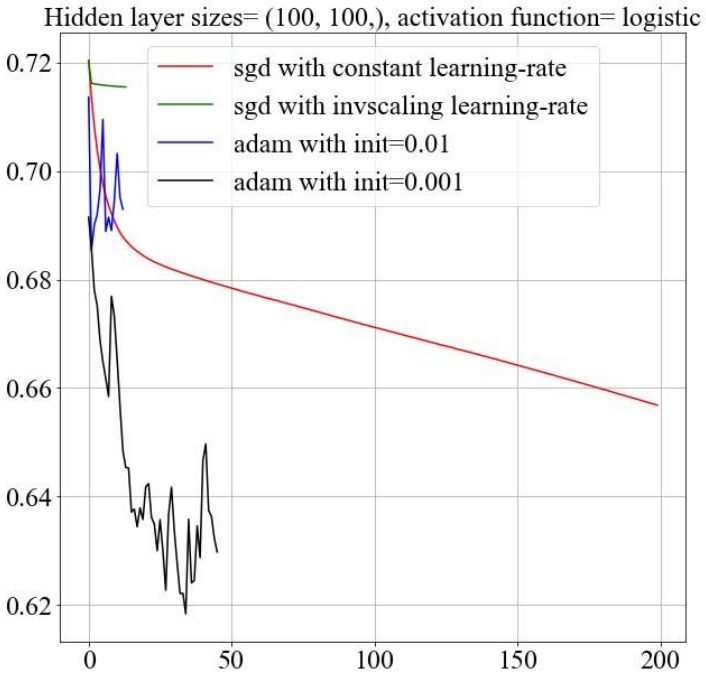


Figure 6. *MLPClassifier training loss for logistic activation function.*

Table 3. *MLPClassifier training results for logistic activation function.*

Solver	Iterations	Training loss	Testing score
sgd - constant	200	0.653397	0.727273
sgd - invscaling	14	0.714827	0.387352
adam - init=0.01	16	0.688389	0.660079
adam - init=0.001	37	0.636158	0.723320
lbfgs - alpha=0.0001	200	0.352143	0.667984
lbfgs - alpha=0.00001	200	0.369488	0.703557

The last activation function to be used will be *relu* activation function.

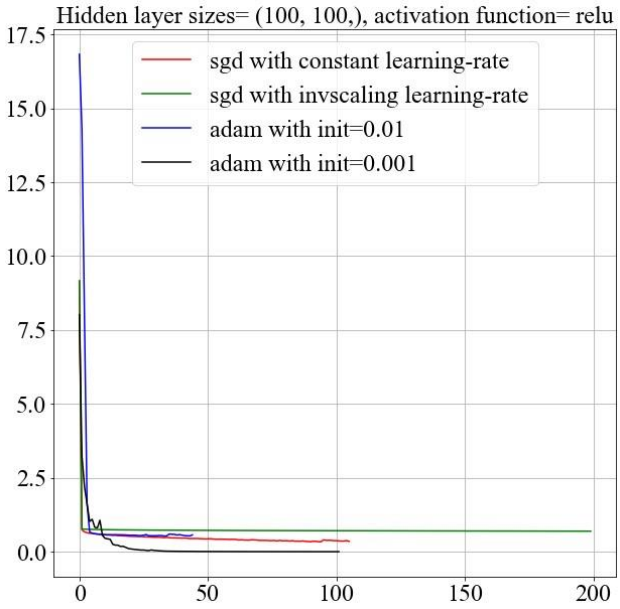


Figure 7. *MLPClassifier training loss for relu activation function.*

Table 4. *MLPClassifier training results for relu activation function.*

Solver	Iterations	Training loss	Testing score
sgd - constant	200	0.156805	0.691700
sgd - invscaling	146	0.430830	0.700803
adam - init=0.01	67	0.523427	0.687747
adam - init=0.001	101	0.002098	0.707510
lbfgs - alpha=0.0001	200	0.000015	0.735178
lbfgs - alpha=0.00001	200	0.000008	0.743083

After analysis of the given results conclusion is that so far the best results are given from combination of *relu* activation function and *lbfgs* solver. Using the *relu* activation function not only gives the best result in one case of solver but gives overall the best average testing score. Due to this conclusion in following comparisons only the *relu* activation function will be used for MLP classifications.

4. MLP classifier architecture parameters comparison

Varying the hidden layer sizes can give different output results using the same activation function and solver. In following experiments the hidden layer sizes and neuron count will be varied in multiple combinations after which the results will be compared.

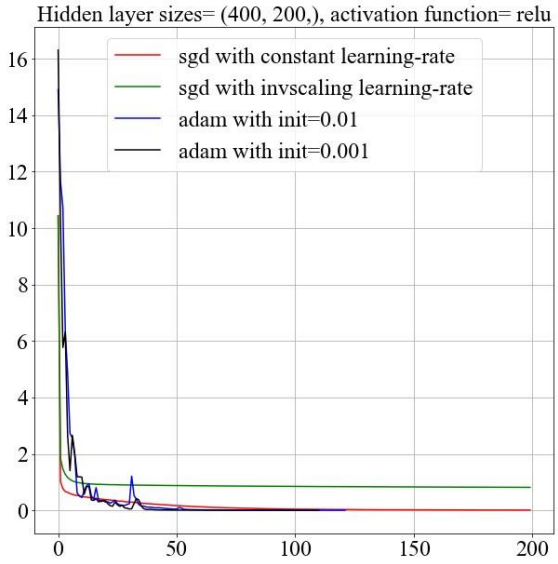


Figure 8. *MLPClassifier training loss for neuron count of 400 in first hidden layer and 200 in second.*

Table 5. *MLPClassifier training results for neuron count of 400 in first hidden layer and 200 in second.*

Solver	Iterations	Training loss	Testing score
sgd - constant	200	0.010521	0.660079
sgd - invscaling	200	0.963506	0.604743
adam - init=0.01	92	0.392435	0.679842
adam - init=0.001	108	0.002551	0.683794
lbfgs - alpha=0.0001	102	0.000028	0.719368
lbfgs - alpha=0.00001	86	0.000011	0.731225

In the following experiment architecture of neural network will be changed from two hidden layers to three hidden layers, each composed of 200 neurons.

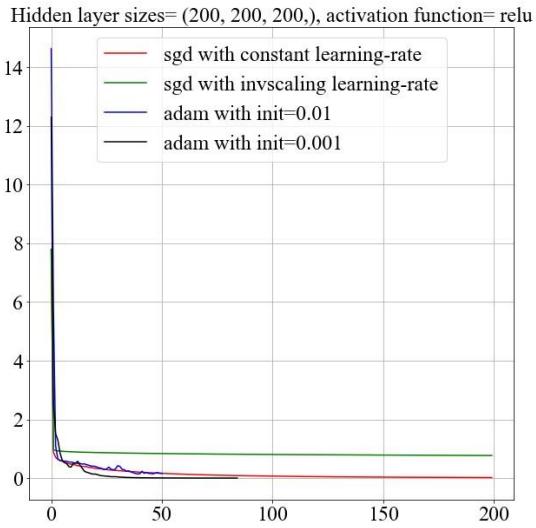


Figure 9. *MLPClassifier training loss for neural network with three hidden layers, each composed of 200 neurons.*

Table 6. *MLPClassifier training results for neural network with three hidden layers, each composed of 200 neurons.*

Solver	Iterations	Training loss	Testing score
sgd - constant	200	0.018041	0.695652
sgd - invscaling	200	0.769314	0.592885
adam - init=0.01	69	0.333450	0.699605
adam - init=0.001	96	0.000965	0.731225
lbfgs - alpha=0.0001	115	0.000024	0.762846
lbfgs - alpha=0.00001	116	0.000009	0.766798

The next experiment will use the same three-layer network but the neuron count of each layer will be altered, now the first layer will be composed of 800 neurons, second 400 and third will be composed of 200 neurons.

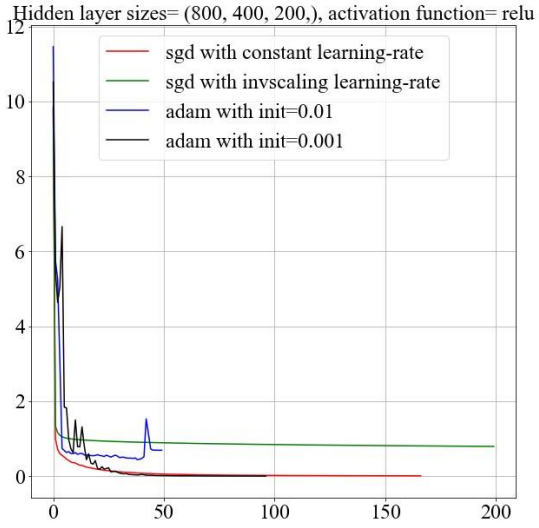


Figure 10. *MLPClassifier training loss for neural network with three hidden layers, first composed of 800, second 400 and third 200 neurons.*

Table 7. *MLPClassifier training results for neural network with three hidden layers, first composed of 800, second 400 and third 200 neurons.*

Solver	Iterations	Training loss	Testing score
sgd - constant	191	0.007907	0.707510
sgd - invscaling	200	0.770376	0.604743
adam - init=0.01	94	0.052771	0.695652
adam - init=0.001	112	0.001718	0.747036
lbfgs - alpha=0.0001	76	0.000063	0.687747
lbfgs - alpha=0.00001	78	0.000022	0.719368

The last experiment will use the three hidden layer architecture with each layer composed of 800 neurons.

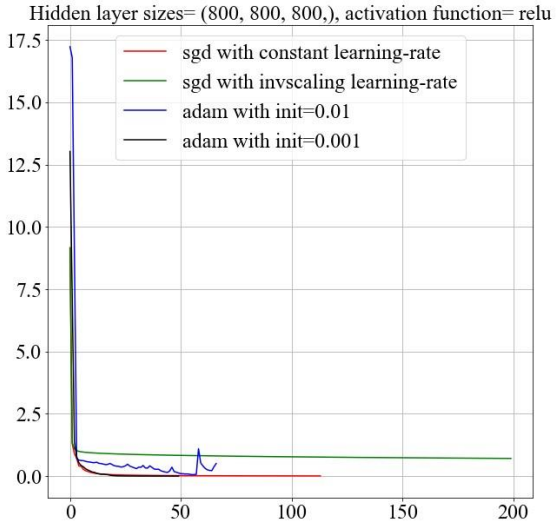


Figure 11. *MLPClassifier training loss for neural network with three hidden layers, each composed of 800 neurons.*

Table 7. *MLPClassifier training results for neural network with three hidden layers, each composed of 800 neurons.*

Solver	Iterations	Training loss	Testing score
sgd - constant	111	0.008159	0.620553
sgd - invscaling	200	0.700378	0.608696
adam - init=0.01	70	0.640721	0.683794
adam - init=0.001	53	0.001142	0.743083
lbfgs - alpha=0.0001	91	0.000066	0.679842
lbfgs - alpha=0.00001	90	0.000011	0.687747

Analysis of given results concludes that there is in fact a connection in altering the neural network architecture and accuracy of classification as the results changed in each architecture alteration, but no clear pattern of which architecture and main parameter combination give the best results was concluded in these experiments. In preceding experiments *adam* solver with an initial learning rate of 0.001 has given the best results, therefore it will be used in the following experiments.

5. Image processing and dataset correction

The idea for application of *threshold* function and Gaussian blur filter in MRI image processing comes from the fact that images are grayscale and there is some noise present in the images. Particularly MRI images are usually corrupted by Rician noise [13]. Gaussian blur can be used to suppress the noise floor of the image without heavy alteration the image which could lead to alteration of the key features.

Threshold function is able to "*cut out"* the image section that fits into the luminance levels we chose while blackening the remains of the image.

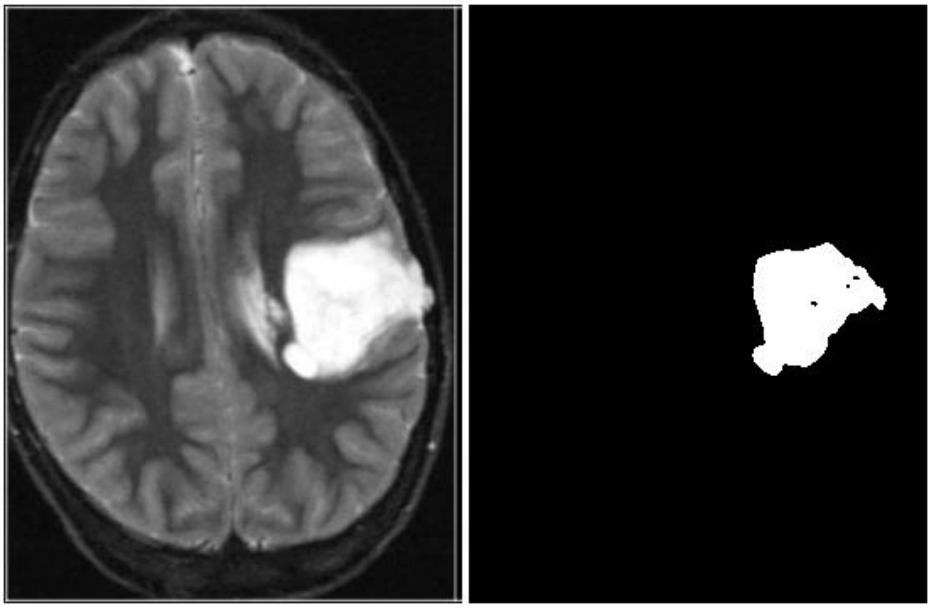


Figure 12. *Unprocessed image (left) and same image processed with threshold function (right)*

The effects of this image processing method is clear, as a result only the highlighted part of the source image is left on the output image.

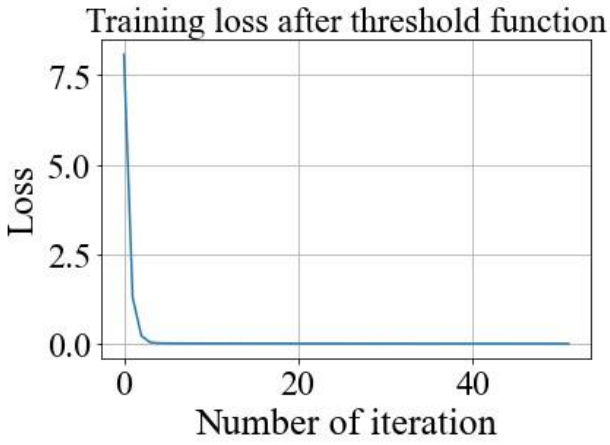


Figure 13. *MLPClassifier training loss after gaussian blur and treshold applied to the dataset.*

Application of this image processing method has given the best classification results so far, 78,66% while using the *relu* activation function, adam solver, and three hidden layer architecture with every layer composed of 800 neurons.

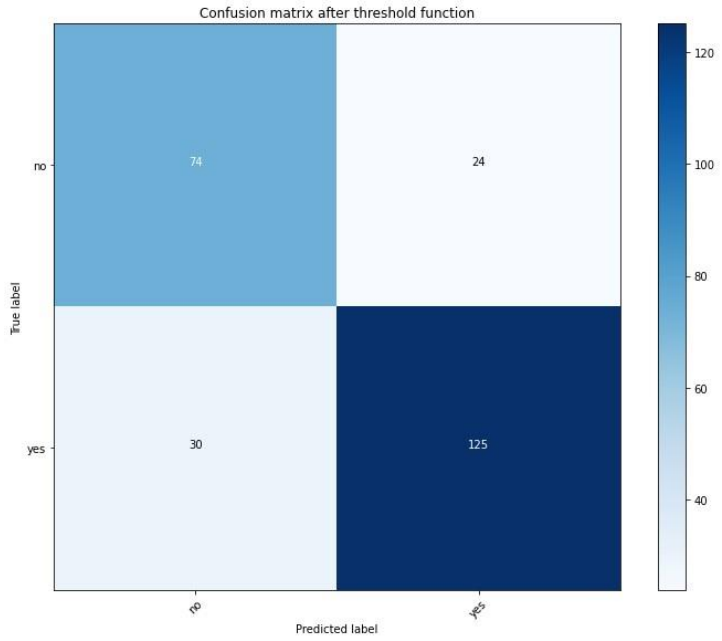


Figure 14. *Confusion matrix after application of treshold function to the dataset.*

The confusion matrix also gives a clear indication that the method is effective, with 30 false negatives, and 24 false positives.

The dataset had some obvious flaws, multiple images were present in both positive and negative sets. Some of the images were photographed and not MRI images coming from the source of the scan. After the further sorting of the dataset and excluding the flawed images the dataset was reduced to 50 positively and 50 negatively diagnosed scans. After the reduction, the dataset has been expanded using the expansion method previously laid out. The expansion has given a total of 1000 images in the dataset.

A new dataset has been fed into the same neural network as in the previous case, using the *relu* activation function, adam solver, and three hidden layer architecture with every layer composed of 800 neurons.

Figure 15. *MLPClassifier training loss after the dataset correction.*

After the dataset correction, classification accuracy has jumped to 95% which is a great upgrade considering that the previous dataset with flaws present gave 78,66% accuracy.

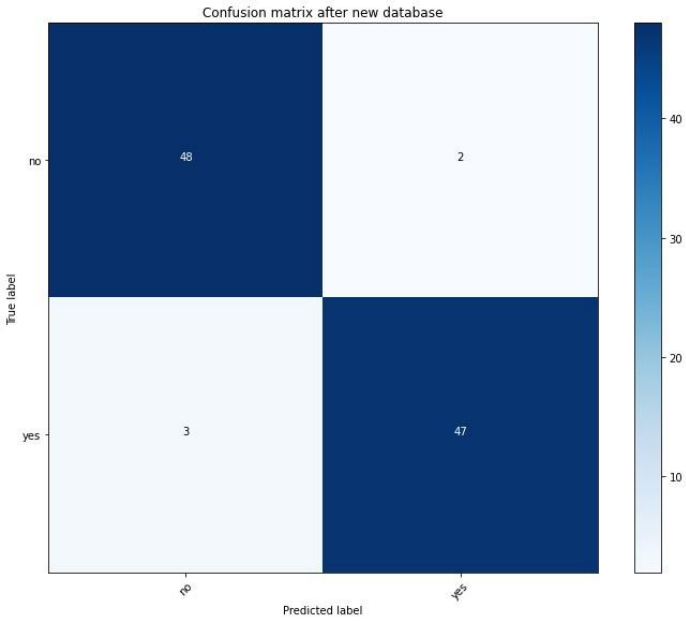


Figure 16. *Confusion matrix loss after the dataset correction.*

The confusion matrix gives a clear view of the dataset correction, only 3 false negatives out of the 50 total negatives and 2 false positives out of the 50 total positives in the dataset.

6. Conclusion

Analysis of all results given by this research leads to the conclusion that in fact, a multilayer perceptron artificial neural network can give satisfying results in this application. However, there are still some

unavoidable false negatives in the results, which could lead to misdiagnosis if the MRI scans are automatically discarded via the software and not re-evaluated by the medical personnel trained for such task. An important comparison parameter would be comparing the artificial intelligence based diagnosis performance with "*real"* medical diagnosis conducted by a professional, but such comparison is not in the scope of this research.

Results of the classification varied throughout the research and some conclusions regarding the correct main parameters and architecture were given.

Besides correct tuning of the classifier, the greater leap in accuracy was introduced upon correction and processing of the dataset itself. Such observation leads to the conclusion that dataset quality itself is a greater factor in overall accuracy than the fine-tuning of parameters.

A key thing to note is that one method and parameters of classification may be optimal for one type of dataset and use case, but completely incorrect for some other [14], that is why this type of research is important and should be conducted for each different use case and dataset.

Acknowledgments

We thank professor Zlatan Car and assistants Nikola Anđelić, Sandi Baressi Šegota and Ivan Lorencin for help and guidance during this research.

References

[1] Lorencin, Ivan, et al. "Using multi-layer perceptron with Laplacian edge detector for bladder cancer diagnosis." Artificial Intelligence in Medicine 102 (2020): 101746. https://doi.org/10.1016/j.artmed.2019.101746
[2] Car, Zlatan, et al. "Modeling the spread of COVID-19 infection using a multilayer perceptron." Computational and mathematical methods in medicine 2020 (2020). https://doi.org/10.1155/2020/5714714
[3] Baressi Šegota, Sandi, et al. "Improvement of Marine Steam Turbine Conventional by Neural Network Application." Journal of Marine Science and Engineering 8.11 (2020) 884. https://doi.org/10.3390/jmse8110884
[4] Lorencin, Ivan, et al. "Genetic algorithm approach to design of multi-layer perceptron for combined cycle power plant electrical power output estimation." Energies 12.22 (2019): 4352. https://doi.org/10.3390/en12224352
[5] Lorencin, Ivan, et al. "Multilayer perceptron approach to condition-based maintenance of marine CODLAG propulsion system components." Pomorstvo 33.2 (2019): 181-190. https://doi.org/10.31217/p.33.2.8
[6] Baressi Šegota, Sandi, et al. "Artificial neural network for predicting values of residuary resistance per unit weight of displacement." Pomorski zbornik 57.1 (2019): 9-22. https://doi.org/10.18048/2019.57.01.
[7] Wang, Bo, et al. "AI-assisted CT imaging analysis for COVID-19 screening: Building and deploying a medical AI system." Applied Soft Computing 98 (2021): 106897. https://doi.org/10.1016/j.asoc.2020.106897
[8] Nadeem, Muhammad Waqas, et al. "Brain tumor analysis empowered with deep learning: A review, taxonomy, and future challenges." Brain sciences 10.2 (2020): 118. https://doi.org/10.3390/brainsci10020118

[9] Bahadure, Nilesh Bhaskarrao, Arun Kumar Ray, and Har Pal Thethi. "Image analysis for MRI based brain tumor detection and feature extraction using biologically inspired BWT and SVM." International journal of biomedical imaging 2017 (2017). https://doi.org/10.1155/2017/9749108
[10] Bakshi, Rohit, et al. "Fluid-attenuated inversion recovery magnetic resonance imaging detects cortical and juxtacortical multiple sclerosis lesions." Archives of neurology 58.5 (2001): 742-748. https://doi.org/10.1001/archneur.58.5.742
[11] Liu, Jin, et al. "A survey of MRI-based brain tumor segmentation methods." Tsinghua science and technology 19.6 (2014): 578-595. http://tst.tsinghuajournals.com/Y2014/V19/I6/578
[12] Drevelegas A., Papanikolaou N., "Imaging modalities in brain tumors, in Imaging of Brain Tumors with Histological Correlations." Springer, 2011, pp. 13-33.
[13] Chen, Wensheng, et al. "A sparse representation and dictionary learning based algorithm for image restoration in the presence of Rician noise." Neurocomputing 286 (2018): 130-140. https://doi.org/10.1016/j.neucom.2018.01.066
[14] Mehrotra, Rajat, M. A. Ansari, and Rajeev Agrawal. "Neural Network and Wavelet-Based Study on Classification and Analysis of Brain Tumor using MR Images." 2019 2nd International Conference on Power Energy, Environment and Intelligent Control (PEEIC). IEEE, 2019. https://doi.org/10.1109/PEEIC47157.2019.8976555

ISBN: 978-953-8246-22-7

Expert fuzzy system for estimating risks of hypertension

Jovana Nikolić [1,*]

[1] Fakultet inženjerskih nauka u Univerziteta u Kragujevcu, Sestre Janjić 6, 34000 Kragujevac, Srbija, joxikg@gmail.com

Abstract: These days medical application, especially diagnosis of some diseases, has rapidly increased because of its importance and effectiveness to detect diseases and classify patients. In this article, the design of an expert fuzzy system that aims at providing diagnosis of hypertension is presented. The hypertension is one of the most dangerous diseases that seriously threat the health of people and communities worldwide. One of the most dangerous aspects of the hypertension is that you may not know that you have it. In this study, expert system using fuzzy for diagnosis of hypertension is proposed. The input parameters include gender, age, systolic blood pressure (SBP), diastolic blood pressure (DBP), blood glucose level, body mass index (BMI), heart rate, smoking, genetic factors. The diagnosis process, linguistic variables and their values were modeled based on expert's knowledge and from existing database. It is expected that proposed Fuzzy Expert System can provide a faster, cheaper and more accurate result compared with other traditional methods.

Keywords: Body mass index, Defuzzification, Diastolic blood pressure, Expert System, Fuzzy logic, Fuzzification, Hypertension, Systolic blood pressure

1. Introduction

The level of application of expert system in the fields of medicine is continuously growing all over the world. Diagnosis of the disease is of special importance because nowadays, timely detection of the presence of the disease is very important in the treatment of the patient. Nowadays, the methods of artificial intelligence have largely been used in the medical applications. In the medicine area, many expert systems were designed to diagnose and treat the disease (Neshat et al., 2008). Medical researchers and doctors cannot precisely define how the disease prevents the normal functioning of the body. Developed countries, but also developing countries face various forms of health crisis, which in addition to many diseases includes hypertension - a disease that threatens the lives of millions of people. Hypertension may be caused by different factors and be so unpredictable, which is why doctors often make decisions based on intuitions and experiences. Therefore, the biggest factor for the appearance of a mistake is the man himself. Doctor's fatigue, carelessness, lack of expertise and similar causes can lead to serious failures. This article presents the procedure for designing an expert system for predicting the risk of hypertension.

2. Methodology

The complexity of medical practices makes traditional quantitative approaches of analysis inappropriate (Rahim et al., 2007). Every trustworthy expert knows that his/her medical knowledge and resulting diagnosis are pervaded by uncertainty with imprecise formulations. Medical processes can be so complex and unpredictable that physicians sometimes must make decisions based on intuition. Computers are capable of making calculations at high and constant speed and of recalling large amounts of data and can, therefore, be used to manage decision networks of high complexity (Merouani et al., 2009).

A. Expert System

An expert system is an intelligent computer program that uses knowledge and reasoning procedures in the process of solving problems that require a high degree of expertise and experience in the domain that the expert system deals with (Guzmán et al., 2015). Due to their reliability, scalability and speed, these systems are increasingly being used in modern applications.
Based on the defined rules in a given system, data and the range of data values are processed and based on them the system executes next steps and generates the appropriate output. Implementation and construction process of expert system includes a set of methods, rules and procedures such as collection, presentation and memorization of knowledge, but also the use of human resources to solve complex problem situations. The engineer is in charge of communicating with the expert, in this case with a team of doctors, from whom procedures, strategies and procedures were taken for problem solving and then that knowledge is built into a given expert system. The result process is a set of rules that generate quality and accurate data output that solves a problem in the way that an expert person does.

B. Fuzzy logic

Fuzzy logic plays an important role in medicine. Fuzzy logic is a method that renders precise what is imprecise in the world of medicine using natural language. Fuzzy logic systems are excellent in handling ambiguous and imprecise information prevalent in medical diagnosis. Fuzzy set and fuzzy logic founded by (Zadeh, 1965) makes it possible to define inexact medical entities as fuzzy set. Fuzzy logic is one of the methods of Soft Computing. Soft Computing is a computational method that is tolerant to sub-optimality, impreciseness, vagueness and thus giving quick, simple and sufficiently good solutions (Chen and Chen, 1994).
Inputs and outputs can have different linguistic names and their values. The transformation of such expressions into the form of a mathematical representation is made possible by the theory of fuzzy sets. Within the given expert system inputs that are important for diagnosing the risk of hypertension and their possible values are used. In this system, the following are taken as input variables: gender, age, systolic blood pressure (SBP), diastolic blood pressure (DPB), blood glucose level, body mass index (BMI), heart rate, smoking, genetic factors.

C. System design

A fuzzy logic system is a collection of membership functions and fuzzy rules that are used to determine the diagnosis (Das et al., 2013). This design has been divided into several steps. The steps are fuzzification, rule evaluation and finally defuzzification (Sivanandam et al., 2007). To design the system, the FIS (Fuzzy inference system) tool in open-source library JFuzzyLogic is used. Each FIS consists of one or more function blocks. Fuzzification is done by determining the linguistic variables and corresponding membership functions for various parameters by the aid of experts, statistical analysis of patient data and literature review. It will then convert the input that is crisp into a fuzzy input. The heart of a fuzzy logic expert system is the Rule Base. The Rule Base consists of a set of Fuzzy Propositions and is derived from the Knowledge Base of the Medical Expert. The Rule Base consists of the IF-THEN rules of fuzzy. Defuzzification is the operation of computing a crisp value given some membership function. It converts the fuzzy quantities into crisp quantities.

D. Input data

The linguistic variable gender is input and has values: male/female. It is estimated that men get sick more often, while in women the risk increases after the age of 65. Variable age is the input. For test examples, samples of people of all ages were considered, which are defined by belonging to one of the groups shown in Table 1.

Table 1. *Input of age*

RANGE	INDICATOR
0 – 35	Young
30 – 65	Middle age
60 – 100	Old

Table 2 shows the input of systolic blood pressure (SBP) while Table 3 shows the input of diastolic blood pressure (DBP).

Table 2. *Input of SBP*

RANGE	INDICATOR
90 – 120 mmHg	Normal
118 – 150 mmHg	High
148 – 200 mmHg	Very High

Table 3. *Input of DBP*

RANGE	INDICATOR
60 – 80 mmHg	Normal
78 – 120 mmHg	High
110 – 150 mmHg	Very High

Table 4 shows input of heart rate.

Table 4. *Input of heart rate*

RANGE	INDICATOR
70 – 90 bpm	Low
75 – 115 bpm	Moderate
100 – 120 bpm	High

The input of cigarette consumption is the average of cigarette consumption on a weekly basis level. Health problems related to smoking are a consequence of the toxicity of tobacco and its harmful compounds and elements. The health risks of active smoking are causally related to most cardiovascular and lung diseases, as well as most malignancies. Smoking tobacco leads to activation of the sympathetic nervous system which increases blood pressure, narrowing peripheral artery and cardiac dysfunction. Table 5 shows input of cigarettes intake.

Table 5. *Input of cigarettes intake*

CIGARETTES	INDICATOR
0 – 5	Very Low
4 – 10	Low
9 – 20	High
19 – 40	Very High

The input of body mass index (BMI) is an indicator of nutritional status of one person. Body mass index is calculated by taking a person's body weight in kilograms divided by the square of the height in meters. Excess weight, especially when it is associated with increased visceral obesity, is a major cause of arterial hypertension. Table 6 shows input of BMI.

Table 6. *Input of BMI*

BMI	INDICATOR
0 – 18.5 (kg/m^2)	Underweight
18.4 – 24.9 (kg/m^2)	Normal
24.8 – 29.9 (kg/m^2)	Overweight
29.8 – 50.0 (kg/m^2)	Obesity

Hypertension or high blood pressure also increases the risk of other cardiovascular diseases. By definition, blood pressure is “the force with which circulating blood acts on walls of an artery as the heart pumps blood.” Hypertension occurs when that force on the walls of the arteries is too high. Diabetes is a disease that affects how the body uses glucose. Type 2 diabetes, which is the most common form of the disease, reduces the production of insulin, a hormone that regulates levels of blood sugar. When this happens, the blood sugar level rises, increasing the risk of heart attack. Table 7 shows input of glucose.

Table 7. *Input of glucose*

RATE	INDICATOR
4 – 5.9 mmol/l	Normal
5.8 – 6.9 mmol/l	Pre-diabetes
6.8 – 30 mmol/l	Diabetes

For the output variable, the degree of risk is defined, which represents the estimated percentage risk of hypertension. Depending on the range to which the output belongs for each person the corresponding risk is defined based on the values of the input variables. Table 8 shows output of system. The percentage risk can be divided into four categories; low, medium, high, very high. Low has percentage between 0% and 25%, medium (24-50) %, high (49-75) % and very high (74-100) %.

Table 8. *Output variable – the degree of risk*

RANGE	INDICATOR
0 – 25%	Low
24 – 50%	Medium
49 – 75%	High
74 – 100%	Very high

E. Fuzzification

Fuzzy sets can be defined as a set without a crisp, clearly defined boundary. It can contain only a partial degree of membership. After determining fuzzy sets, the next step is to determine the membership function. Fuzzification, which includes membership function, plays greater role for fuzzification data. This function can be set such that the value assigned to the elements of the universal set fall within a specific range. These elements will indicate membership grade in the set of question. After defining the input variables and their values, all variables go through fuzzification. All input variables are fuzzified individually and each one of them is described as specified membership function for which the reserved words are used: TRIAN (triangular function), TRAPE (trapezoidal function).

F. Defuzzification

Defuzzification is a process to compute the outcome values corresponding to each label. The function of defuzzification is to find the single crisp value that summarizes the fuzzification of fuzzy sets. In this project, the ‘center of gravity’ method is being used.

G. Rules

A rule block is also defined within the function block. The rules represent one of the most commonly used methods for presenting knowledge within the expert system. It provides a link between some known information and information for which based on the accuracy of the first, it can be concluded that it is also valid. The theoretical background of this methods is based on expressive logic and its extension by predicate calculus. There are one or more statements within the IF part on which the conclusion is derived in the THEN section. If a rule contains multiple premises they are connected by logical operators (AND, OR). There is a conclusion within the THEN part. As the number of rules increases, the complexity of the system increases.

3. Results and Discussion

A GUI (Graphical User Interface) enables user adequate communication with the computer using graphic icons and visual indicators (Figure 1) (Zanino et al., 1994). Data validation is enabled. When the data type provided is not entered, the user will be required to enter the data in the correct format. In this project, nine inputs were used for this expert system and one output was generated based on them (Milošević-Georgiev et al., 2016). A test example for one patient and the values of his parameters are presented on Figure 1.

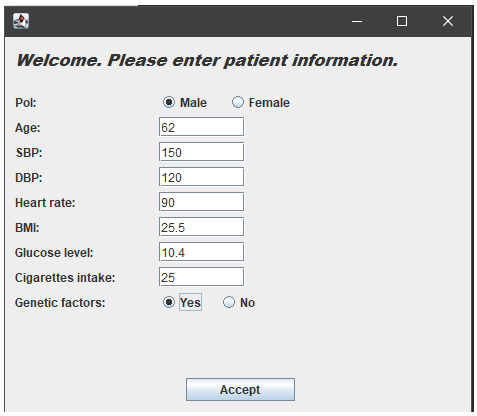


Figure 4. *Graphical User Interface*

The graphs generated for patient data are as follows. Figure 2 shows the membership function of age, based on input, which is given in Table 1.

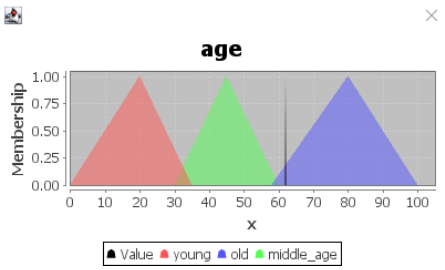


Figure 5. *Membership function of age*

Figure 3 shows the membership function of systolic blood pressure (SBP), based on input, which is given in Table 2.

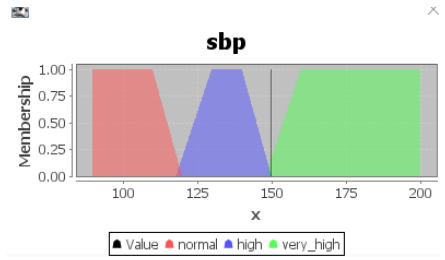


Figure 6. *Membership function of SBP*

Figure 4 shows the membership function of diastolic blood pressure (DBP) based on input, which is given in Table 3.

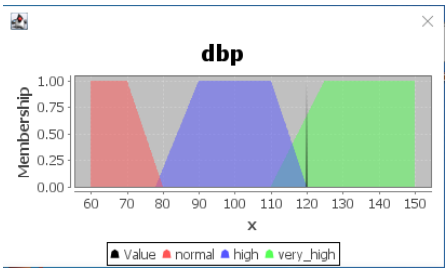


Figure 7. *Membership function of DBP*

Figure 5 shows the membership function of heart rate, based on input, which is given in Table 4.

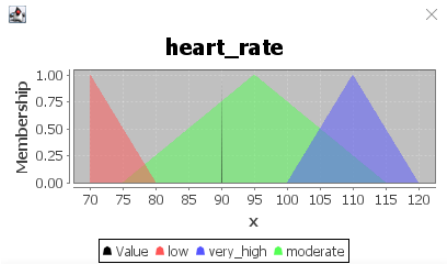


Figure 8. *Membership function of heart rate*

Figure 6 shows the membership function of body mass index (BMI), based on input, which is given in Table 6.

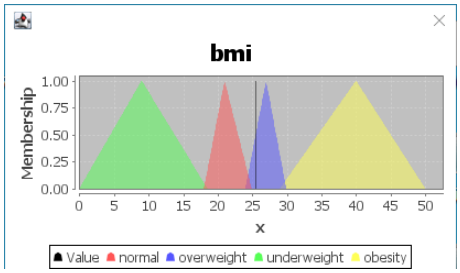


Figure 9. *Membership function of BMI*

Figure 8 shows the membership function of glucose level, which was developed based on input shown in Table 7.

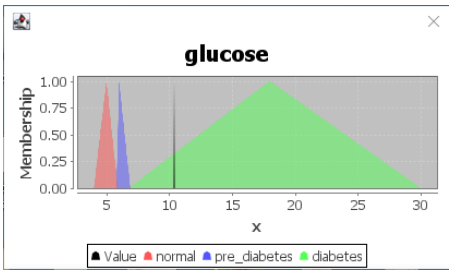


Figure 10. *Membership function of glucose*

Figure 8 shows the membership function of cigarettes intake, which was developed based on input shown in Table 5.

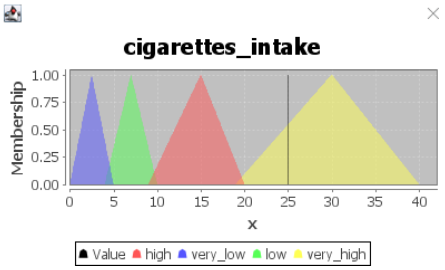


Figure 11. *Membership function of cigarettes intake*

Figure 9 shows the membership function of output variable, which is given in Table 8.

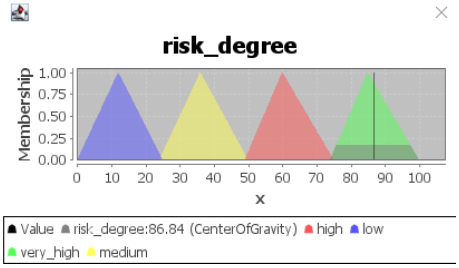


Figure 12. *Membership function of output variable*

The result of patient risk estimation is shown on Figure 10.

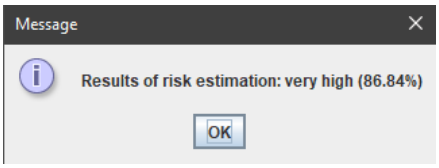


Figure 13. *Membership function of output variable*

In this study, the final hypertension diagnosis result is shown on our developed GUI. By inserting the data, and pressing the button, the percentage of risk for getting hypertension will be shown. By using this developed fuzzy expert system, the result can be known within few second and thus can save the diagnosis time. When a person gets a reading between 74% and 100%, he is confirmed to have hypertension. The result will be more accurate because more inputs were used. The inputs that have been used are selected based on experiences from medical practitioners, nurses and doctors and also based on scientific literatures (Abdullah et al., 2011).

4. Conclusion

This study is aimed to develop a system to diagnose hypertension and to know the percentage risk for getting hypertension by just entering the input required and to develop a user-friendly layout by using fuzzy logic and Graphic User Interface (GUI).
The effects achieved by applying an adequately designed, implemented expert system are multiple, of which the following can be emphasized: increasing the efficiency of treatment by reducing the time required for diagnosis, more complete recording of medical services, quality knowledge base and data on each patient individually, more efficient health care protection, better and more comprehensive elements for scientific research.
The aspiration of artificial intelligence and thus of expert systems as a subfield of artificial intelligence is the development of a system capable of making decisions in an identical way as a human being. The future of such systems seems very certain and promising. The number of experts who work every day on the development of expert systems in various disciplines as well as companies interested in them, is increasing every day. All this, as well as the need for more efficient management, quality decision making and storing data indicates that such applications will be developed and used in the future even more than today.

Acknowledgments

I would like to thank my mentor Tijana Šušteršič for her thoughtful giudance during the studies. She was engaged in all the stages of article preparation and revision and I am very thankful for this.

Reference

[1] Rahim, F., Deshpande, A., and Hosseini, A. 2007. “Fuzzy Expert System for Fluid Management in General Anesthesia”. Journal of Clinical and Diagnostic Research. 256-267.
[2] Merouani, M., Guignard, B., Vincent, F., Borron, S.W., Karoubi, P., Fosse, J.P., Cohen, Y., Clec'h, C., Vicaut, E., Marbeuf-Gueye, C., Lapostolle, F., and Adnet, F. 2009. “Can Fuzzy Logic Make Things More Clear?”. Critical Care. 13:116.
[3] Chen, C.L. and Chen, W.C. 1994. “Fuzzy Controller Design by Using Neural Network Techniques”. IEEE Transactions on Fuzzy Systems. 2(3):235-244.

[4] Michael C. Zanino, Ritu Agarwal, Jayesh Prasad 1994. “Graphical user interfaces and ease of use: some myths examined”
https://doi.org/10.1145/186281.186313
[5] M. Neshat, M. Yaghobi, M. B Naghibi, A. Esmaelzadeh, Fuzzy Expert System Design for Diagnosis of liver disorders, Department of Computer Engineering, Azad University of Mashhad Iran 2008 International Symposium on Knowledge Acquisition and Modeling, pp.252-256
[6] S. N. Sivanandam, S. Sumathi and S. N. Deepa, (2007). Introduction to Fuzzy Logic using MATLAB, pp. 304-309.
[7] Milošević-Georgiev, A., & Krajnović, D. [2016]. Faktori rizika za razvoj hipertenzije u vezi sa navikama u ishrani studenata Univerziteta u Beogradu. *Timočki medicinski glasnik*, 41(3), 203-207.
[8] S. Das, P. K. Ghosh and S. Kar, "Hypertension diagnosis: A comparative study using fuzzy expert system and neuro fuzzy system," *2013 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2013, pp. 1-7, doi: 10.1109/FUZZ-IEEE.2013.6622434.
[9] Guzmán J.C., Melin P., Prado-Arechiga G. (2015) Design of a Fuzzy System for Diagnosis of Hypertension. In: Melin P., Castillo O., Kacprzyk J. (eds) Design of Intelligent Systems Based on Fuzzy Logic, Neural Networks and Nature-Inspired Optimization. Studies in Computational Intelligence, vol 601. Springer, Cham.
https://doi.org/10.1007/978-3-319-17747-2_40
[10] A. A. Abdullah, Z. Zakaria and N. F. Mohamad, "Design and Development of Fuzzy Expert System for Diagnosis of Hypertension," *2011 Second International Conference on Intelligent Systems, Modelling and Simulation*, 2011, pp. 113-117, doi: 10.1109/ISMS.2011.27.

ISBN: 978-953-8246-22-7

Comparison of open and closed gas turbine cycles

Dominik Salma*, Vedran Mrzljak

Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia
E-mail: dsalma@riteh.hr, vedran.mrzljak@riteh.hr

Abstract: This paper presents an analysis and comparison of a real open and closed cycle gas turbine process. Analysis is performed by using operating parameters of six different working mediums (air, helium, CO_2, neon, argon, nitrogen). The most notable difference is that a real open cycle process uses air and contains a combustion chamber and its dissipated heat is released into the environment, while a real closed cycle process uses helium, CO_2, neon, argon and nitrogen while consisting of a heater and a cooler where its heat is dissipated. The air had the lowest required driving turbocompressor power (45816,6 kW), while the highest was achieved by neon (198855,27 kW). The air had again the lowest total power developed by the turbine (113768,1 kW), while the highest turbine total power was also achieved by neon (266807,32 kW). Neon had the lowest share of effective power in the total turbine power (25,5%), while the highest share of effective power in the total turbine power was achieved by air (59,7%). Helium had the lowest mass flow (74,84 kg/s), while argon had the highest value of mass flow (731,84 kg/s).

Keywords: Gas Turbine, Heat Exchange, Turbine, Turbocompressor, Various Working Mediums

1. Introduction

Generating electricity with gas turbines was first seen in the year 1939. The gas turbine is a type of engine in which burning of an air-fuel mixture produces hot gas that spins a turbine to produce power. Combustion occurs continuously in gas turbines. Gas turbines can usually be found in a power production as an individual device, in cogeneration systems or as a part of a complex combined-cycle power plants [1, 2].
There are also two types of gas turbines: open-cycle and closed-cycle gas turbine. Closed-cycle gas turbines offer various benefits in comparison to open-cycle gas turbines. They can use various non-corrosive operating mediums, which offers better thermodynamic characteristics (when compared with air) while not having any connection with the environment. Produced useful power in this process can easily be regulated by a rate change of operating medium mass flow. Disadvantages of closed-cycle gas turbines lie in usually high-cost operating mediums and their inaccessibility in the market, this kind of cycle requires additional space for operating medium storage because of the huge dimensions that heaters and coolers possess [3, 4]. This paper provides an analysis with six different operating mediums. In the open-cycle operating medium is air, while in the closed-cycle gas turbine there are: helium, CO_2, neon, argon and nitrogen.

2. Methodology

During the operation of the gas turbine in the open process, the compressor takes clean air from the atmosphere with temperature T_1 and pressure p_1 and compresses it to pressure p_2. During compression, there occurs an increase in friction between the air molecules, which is manifested by an increase in the temperature, so the air at the outlet of the compressor has a temperature of T_2, Figure 1.
The air is further brought to the combustion chamber where the fuel is sprayed and mixed with air, the mixture burns and the flue gas temperature increases to the temperature T_3, which is also the highest

temperature of the process. Flue gases were delivered to turbine and after expansion in the turbine, a state of flue gases is denoted through the temperature T_4 and pressure p_1, Figure 1.

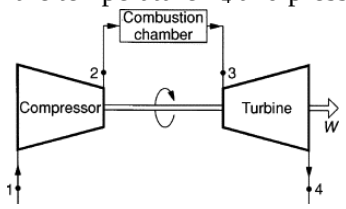


Figure 1. *Open-cycle gas turbine scheme [5]*

After the calculation of the open process, the same plant needs to be upgraded and the work in the closed process. The closed system works on a similar principle as the open one, the difference is that the closed system has a heater instead of a combustion chamber in which the working medium is heated by heat supplied from the outside, Figure 2. Also, instead of being released into the environment, working medium is cooled through a cooler so that, through a closed process of structure, the same working medium is present all the time. Closed process achieves significantly higher reliability of the turbine and extend its lifespan because the working medium does not contain substances that could cause corrosion or erosion on the parts of the turbine.

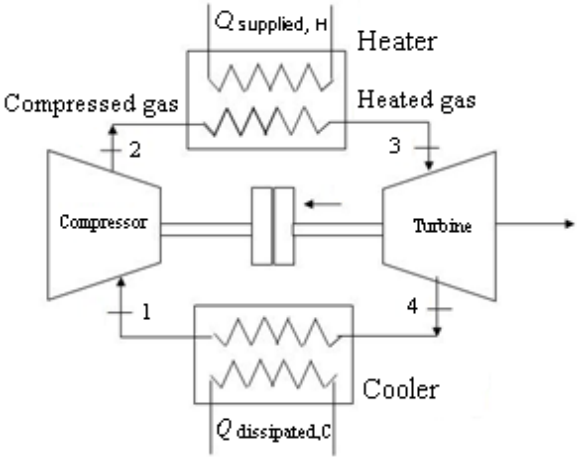


Figure 2. *Closed-cycle gas turbine scheme*

3. Equations required for the analysis

All the equations required for the main operating parameters calculation, regardless if the process is open or closed one, are based on the operating points presented in Figure 3 (temperature-specific entropy diagram). The real process is presented with red lines, Figure 3.

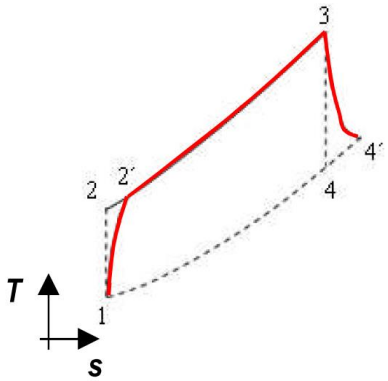


Figure 3. *T-s diagram of the real gas turbine process*

3.1. Real open cycle gas turbine process

- The power required to drive the turbocompressor:

$$P_{\text{T-C}} = \dot{m}_{\text{flow}} \cdot (T_{2'} \cdot c_{p2',\text{air}} - T_1 \cdot c_{p1,\text{air}}) \quad (1)$$

- Total power developed by the turbine:

$$P_{\text{T}} = \dot{m}_{\text{flow}} \cdot (T_3 \cdot c_{p3,\text{air}} - T_{4'} \cdot c_{\text{p4'},\text{air}}). \quad (2)$$

- Difference of the two powers gives useful power:

$$P_{\text{useful}} = P_{\text{T}} - P_{\text{T-C}}. \quad (3)$$

- The share of useful power in the total turbine power:

$$P_{\text{useful,t}} = \frac{P_{\text{useful}}}{P_{\text{T}}}. \quad (4)$$

- Turbocompressor power share in the total power:

$$P_{\text{turbocom,t}} = \frac{P_{\text{T-C}}}{P_{\text{T}}}. \quad (5)$$

- Heat supplied to the combustion chamber:

$$Q_{\text{supplied,CC}} = \dot{m}_{\text{flow}} \cdot (T_3 \cdot c_{p3,\text{air}} - T_{2'} \cdot c_{p2',\text{air}}). \quad (6)$$

- Heat dissipated into the environment:

$$Q_{\text{dissipated,C}} = \dot{m}_{\text{flow}} \cdot (T_{4'} \cdot c_{p4',\text{air}} - T_1 \cdot c_{p1,\text{air}}). \quad (7)$$

- Efficiency of the whole turbine cycle:

$$\eta_{\text{t,cycle}} = \frac{P_{\text{useful}}}{Q_{\text{supplied,CC}}}. \quad (8)$$

3.2. Real closed cycle gas turbine process

Open and closed cycle gas turbine have the same useful power. For every individual working medium firstly is calculated mass flow:

$$\dot{m}_{\text{flow}} = \frac{P_{\text{useful}}}{(h_3 - h_{4'} - h_{2'} + h_1)}. \quad (9)$$

- The power required to drive the turbocompressor:

$$P_{\text{T-C}} = \dot{m}_{\text{flow}} \cdot (h_{2'} - h_1). \quad (10)$$

- Total power developed by the turbine:

$$P_{\text{T}} = \dot{m}_{\text{flow}} \cdot (h_3 - h_{4'}). \quad (11)$$

- Difference of the two powers gives useful power:

$$P_{\text{useful}} = P_{\text{T}} - P_{\text{T-C}}. \quad (12)$$

- The share of useful power in the total power:

$$P_{\text{useful,t}} = \frac{P_{\text{useful}}}{P_{\text{T}}}. \quad (13)$$

- Turbocompressor power share in the total power:

$$P_{\text{turbocom,t}} = \frac{P_{\text{T-C}}}{P_{\text{T}}}. \quad (14)$$

- Heat supplied to the heater:

$$Q_{\text{supplied,H}} = \dot{m}_{\text{flow}} \cdot (h_3 - h_{2'}). \quad (15)$$

- Heat dissipated into the cooler:

$$Q_{\text{dissipated,C}} = \dot{m}_{\text{flow}} \cdot (h_{4'} - h_1). \tag{16}$$

- Efficiency of the whole turbine cycle:

$$\eta_{\text{t,cycle}} = \frac{P_{\text{useful}}}{Q_{\text{supplied,H}}}. \tag{17}$$

In the above equations P is mechanical power, c_p is specific heat at constant pressure, T is temperature, $\dot{m}$ is operating medium mass flow and h is specific enthalpy.

4. Operating parameters of each observed process

Air temperature, pressure and middle specific heat in each operating point for the real open cycle gas turbine process are shown in Table 1. Operating points are related to Figure 3.

Table 1. *Data for the real open cycle process*

Operating points (Air)	Temperature [°C]	Pressure [bar]	Middle specific heat [kJ/kgK]
1.	15	1	1.0047
2.	312.77	12	1.021
2´.	353.38	12	1.0246
3.	1200	11.6	1.1091
4.	403.91	1	1.0294
4´.	459.63	1	1.0354

In each observed gas turbine process (open and closed ones) isentropic efficiencies of turbocompressor and turbine are equal to 88% and 93%, while in the combustion chamber (heater) pressure drop is equal to 4%. Therefore, all the losses which occur in the reality were taken into account, for each observed process.

All the thermodynamic data related to each analyzed closed cycle gas turbine processes, for all working mediums are presented in Table 2. Again, presented operating points are related to operating points from Figure 3. It should be highlighted that specific enthalpies, specific entropies and all other required data in each operating point are calculated from known pressures and temperatures by using NIST-REFPROP software [6].

Table 2. *Data for the real closed cycle processes*

Helium				
Operating points	Temperature (°C)	Pressure (bar)	Enthalpy (kJ/kg)	Entropy (kJ/kgK)
1	33	11,00	1598,50	23,133
2	353,26	66,00	3278,80	23,133
2'	397,35	66,00	3507,60	23,486
3	880	63,36	6011,70	26,305
4	299,54	11	2962,60	26,305
4'	340,37	11	3194,60	26,743

CO_2				
Operating points	Temperature (°C)	Pressure (bar)	Enthalpy (kJ/kg)	Entropy (kJ/kgK)
1	33,00	11,00	503,55	2,2878
2	180,22	66,00	622,77	2,2878
2'	194,84	66,00	639,03	2,3231
3	800,00	63,36	1456,30	3,3937
4	607,25	11,00	1119,4	3,3937
4'	626,85	11,00	1143,00	3,4202

Neon				
Operating points	Temperature (°C)	Pressure (bar)	Enthalpy (kJ/kg)	Entropy (kJ/kgK)
1	33,00	11,00	375,35	4,7230
2	353,12	66,00	709,18	4,7230
2'	397,22	66,00	754,70	4,7932
3	775,00	63,36	1148,90	5,2708
4	247,49	11,00	596,61	5,2708
4'	284,66	11,00	634,92	5,3419

Argon				
Operating points	Temperature (°C)	Pressure (bar)	Enthalpy (kJ/kg)	Entropy (kJ/kgK)
1	33,00	11,00	157,30	3,3856
2	354,10	66,00	325,05	3,3856
2'	397,04	66,00	347,93	3,4209
3	880,00	63,36	602,13	3,7154
4	299,36	11,00	297,50	3,7154
4'	340,13	11,00	318,82	3,7514

Nitrogen				
Operating points	Temperature (°C)	Pressure (bar)	Enthalpy (kJ/kg)	Entropy (kJ/kgK)
1	33,00	11,00	315,51	6,1486
2	238,35	66,00	530,75	6,1486
2'	265,35	66,00	560,10	6,2045
3	880,00	63,36	1262,00	7,0888
4	460,59	11,00	773,13	7,0888
4'	491,37	11,00	807,35	7,1265

5. Results and discussion

Comparisons between various calculated variables of six different gas turbines (one open cycle and five closed cycle) are presented in this section.
Power that is required to drive the turbocompressor was first compared with the total power developed by the turbine. Working medium that had the lowest required driving turbocompressor power was air (45816,6 kW), while the highest turbocompressor power was achieved by neon (198855,27 kW), Figure 4. Respectively, working medium that had the lowest total power developed by the turbine was also air (113768,1 kW), while the highest turbine total power was achieved by neon (266807,32 kW), Figure 4.

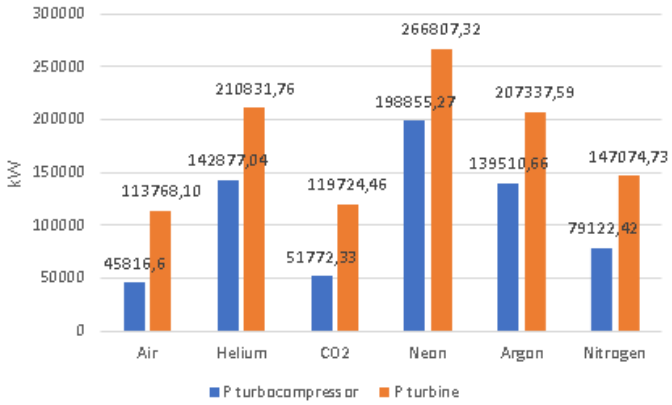


Figure 4. *Turbocompressor and turbine power comparison*

Next comparison is between the share of useful power in the total turbine power (effective power is used for the power consumer drive) and the turbocompressor power share in the total power (turbocompressor power is used for the turbocompressor drive), Figure 5. Working medium that had the lowest share of useful power in the total turbine power was neon (25,5%), while the highest share of useful power in the total turbine power was achieved by air (59,7%), Figure 5. Respectively, working medium that had the lowest turbocompressor power share in the total power was also air (40,3%), while the highest turbocompressor power share in the total power has neon (74,5%), Figure 5. This comparison shows that the total share (the sum of both mentioned shares, for each observed turbine) should always be equal to 100%.

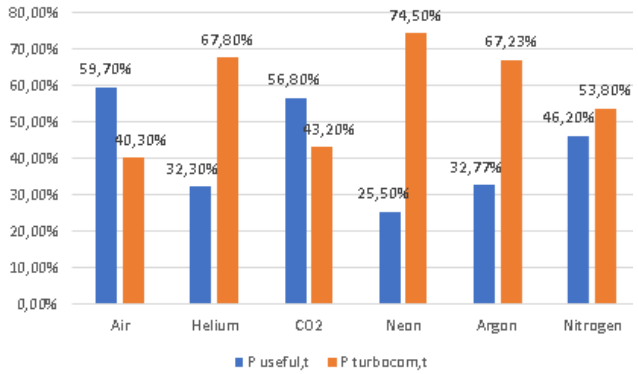


Figure 5. *The share of (useful / turbocompressor) power in the total turbine power*

Third comparison is between the heat supplied to the heater (combustion chamber) and heat that is dissipated to cooler (released to the environment). Of all six working mediums, five of them (helium, CO_2, neon, argon and nitrogen) are used in a real closed cycle process which contains heater and cooler, while only air is used in a real open cycle process which contains combustion chamber and dissipated heat is released into the environment. It is implied for the sake of simplicity that on the chart these two values are the heater and cooler, Figure 6.
The air had the lowest supplied heat (128949 kW), while the CO_2 had the highest value of supplied heat (312311,56 kW). Lowest dissipated heat again belong to air (60997,5 kW), while the highest belong to

CO_2 (244359,42 kW), Figure 6.

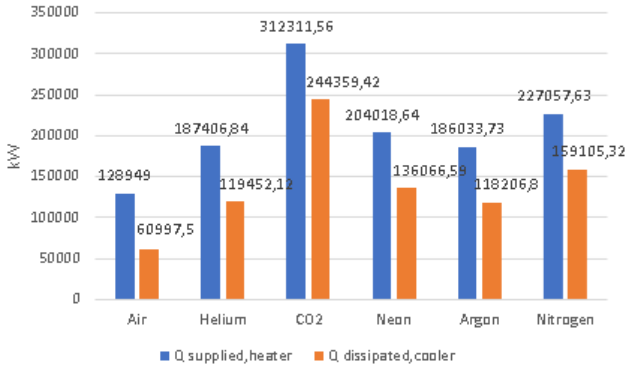


Figure 6. *Comparison of supplied and dissipated heat*

The fourth comparison shows the relationship between specific turbocompressor power and specific turbine power. The lowest specific turbocompressor power can be seen for argon (304,63 kJ/kg), while the highest can be seen for helium (3029,1 kJ/kg). CO_2 has the lowest specific turbine power (119,22 kJ/kg), while helium has the highest specific turbine power (1680,3 kJ/kg), Figure 7. From the data shown in Figure 7 can be expected that the lowest working medium mass flow will have closed cycle gas turbine which operates with helium (due to the highest specific power of turbocompressor and turbine). Similarly, high mass flow of operating medium can be expected in closed cycle gas turbines which operates by using CO_2, neon, argon and nitrogen.

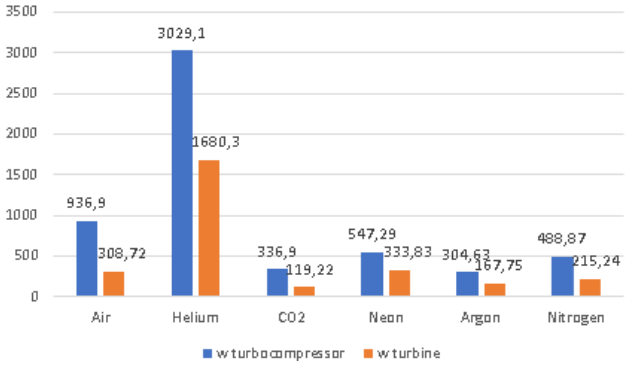


Figure 7. *Specific power of turbocompressor and turbine (in kJ/kg)*

Fifth comparison shows the thermodynamic efficiency of the whole process, which shows the relationship between the obtained useful power and the thermal energy invested in the system. Working medium CO_2 had the lowest process efficiency (21,8%), while the air had the highest process efficiency of all observed working mediums (52,7%), Figure 8.

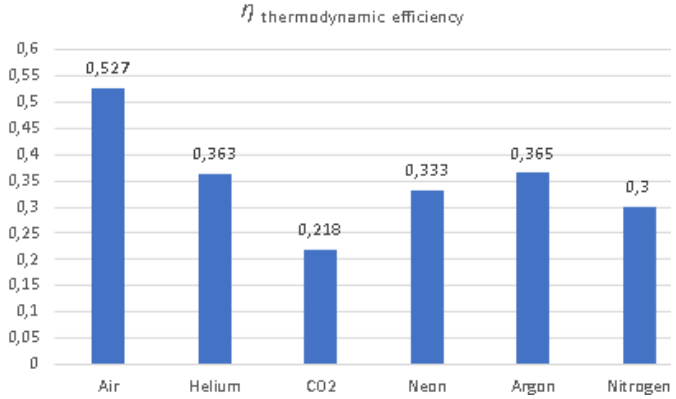


Figure 8. *Thermodynamic efficiency of the whole process for all six observed working mediums (x100%)*

Final comparison represents the mass flow of working medium in each observed process. Helium had the lowest mass flow of all six working mediums (74,84 kg/s), while argon had the highest value of mass flow (731,84 kg/s), Figure 9.

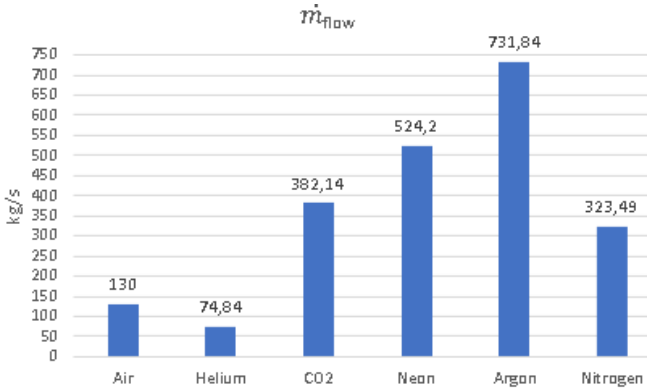


Figure 9. *Mass flow of working mediums*

6. Conclusions

While heat input for the real open cycle process has a combustion chamber and heat is dissipated at the later stage of the process into the environment, real closed cycle process differs in two parts: heat is supplied to the heater, while heat is dissipated to cooler, ensuring the same working medium is present all the time by a closed process structure. Reliability of the turbine is significantly higher and its lifespan is greatly prolonged if the working medium does not contain substances that could cause corrosion or erosion on the parts of the turbine unlike the air. Useful power is equal for both real open and closed cycles which enables comparison between them.
This paper presented an analysis of a real open and closed cycle gas turbine process. The analysis was performed by using six different working mediums which consist of: air, helium, CO_2, neon, argon and nitrogen. The presented analysis provided insight into the operation of each cycle by comparing its: the turbocompressor and turbine power, the total share of effective and useful turbocompressor power in total produced power, supplied and dissipated heat, specific turbocompressor and turbine power, the thermodynamic efficiency of the whole cycle and lastly the mass flow of working mediums. The most important conclusions of this analysis are:

- Working medium that had the lowest required driving turbocompressor power was air (45816,6 kW), while the highest turbocompressor power was observed during operation by neon (198855,27 kW). Working medium that had the lowest total power developed by the turbine was also air (113768,1 kW), while the highest turbine total power was again achieved by neon (266807,32 kW).

- Working medium that had the lowest share of useful power in the total turbine power was neon (25,5%), while the highest useful power share in the total power was achieved by air (59,7%). Working medium that had the lowest share of turbocompressor power in the total turbine power was air (40,3%), while the highest turbocompressor power share in the total power has neon (74,5%).

- Air had the lowest supplied heat (128949 kW) and the lowest dissipated heat (60997,5 kW). The highest value of the supplied heat (312311,56 kW) and the highest dissipated heat (244359,42 kW) belong to CO_2.

- Lowest specific turbocompressor power can be seen for argon (304,63 kJ/kg), while the highest can be seen for helium (3029,1 kJ/kg). CO_2 has the lowest specific turbine power (119,22 kJ/kg), while helium has the highest value of specific turbine power (1680,3 kJ/kg).

- Working medium CO_2 had the lowest value of overall thermodynamic efficiency (21,8%), while the air had the highest overall thermodynamic efficiency (52,7%) in comparison to all the other operating mediums.

- Helium had the lowest mass flow of all six working mediums (74,84 kg/s), while argon had the highest

mass flow (731,84 kg/s).

Further research of all processes presented in this paper will be based firstly on the exergy analysis which will show the influence of the ambient parameters (pressure and temperature) on the overall efficiency [7, 8]. The next step will be optimization of the components and the whole processes which will be performed by using various artificial intelligence methods which proved to be a good solution for optimization purposes in energy sector [9-11].

7. References

[1] Aliyu, Mansur, et al. "Energy, exergy and parametric analysis of a combined cycle power plant." *Thermal Science and Engineering Progress* 15 (2020): 100450.
https://doi.org/10.1016/j.tsep.2019.100450
[2] Mrzljak, Vedran, et al. "Analysis of gas turbine operation before and after major maintenance." *Pomorski zbornik* 57.1 (2019): 57-70.
https://doi.org/10.18048/2019.57.04.
[3] Olumayegun, Olumide, Meihong Wang, and Greg Kelsall. "Closed-cycle gas turbine for power generation: A state-of-the-art review." *Fuel* 180 (2016): 694-717.
https://doi.org/10.1016/j.fuel.2016.04.074
[4] Mrzljak, Vedran, et al. "Exergy analysis of marine waste heat recovery CO2 closed-cycle gas turbine system." *Pomorstvo* 34.2 (2020): 309-322.
https://doi.org/10.31217/p.34.2.12
[5] Cengel, Yunus A., and Michael A. Boles. Thermodynamics: An Engineering Approach 6th Editon (SI Units). The McGraw-Hill Companies, Inc., New York, 2007.
[6] Lemmon, E. W., M. L. Huber, and M. O. McLinden. "NIST standard reference database 23, NIST reference fluid thermodynamic and transport properties, REFPROP, version 9.0." Standard Reference Data Program (2010).
[7] Rosen, Marc A. "Energy-and exergy-based comparison of coal-fired and nuclear steam power plants." *Exergy, An International Journal* 1.3 (2001): 180-192.
https://doi.org/10.1016/S1164-0235(01)00024-3
[8] Anđelić, Nikola, et al. "Comparison of Exergy and Various Energy Analysis Methods for a Main Marine Steam Turbine at Different Loads." *Pomorski zbornik* 59.1 (2020): 9-34.
https://doi.org/18048/2020.59.01.
[9] Ebrahimgol, H., et al. "A novel approach in exergy optimization of a WWER1000 nuclear power plant using whale optimization algorithm." *Annals of Nuclear Energy* 145 (2020): 107540.
https://doi.org/10.1016/j.anucene.2020.107540
[10] Sultan, Khalid F., and Raheel J. Abd-Kadhum. "Evaluation and improvement performance of a boiler in a thermal power plant using artificial neural network." *Engineering and Technology Journal* 36.6 Part A (2018).
http://dx.doi.org/10.30684/etj.36.6A.10
[11] Baressi Šegota, Sandi, et al. "Improvement of Marine Steam Turbine Conventional Exergy Analysis by Neural Network Application." *Journal of Marine Science and Engineering* 8.11 (2020): 884.
https://doi.org/10.3390/jmse8110884

ISBN: 978-953-8246-22-7

Breast cancer classification

Nikola Radovanović [1,*]

[1] Fakultet inženjerskih nauka Univerziteta u Kragujevcu, Sestre Janjić 6, 34000 Kragujevac, Srbija, nradovanovic@kg.ac.rs.

Abstract: Breast cancer is most frequently diagnosed cancer and the second most frequent cause of death in women. Early detection and treatment is the key for the patient's health and reduced mortality. Machine learning classification can be used to help physicians in early diagnosis and classification of breast tumours. In this paper, three types of classifiers are presented: Random forest, Decision tree, and K-nearest neighbours for breast cancer classification. The classifiers are evaluated and compared. Techniques for improvement: feature selection, grid search and hyperparameter tuning reduce training time and improve performances of the classifiers. The highest accuracy is achieved by Random forest classifier (99.41%).
Keywords: Breast cancer, Classification, Cross-validation, Decision tree, Feature, selection, K-Nearest neighbours, Random forest

1. Introduction

Today, breast cancer is the most frequently diagnosed life-threatening cancer in women and the leading cause of cancer death among women along with lung cancer [1, 2]. The success of personalized breast cancer care depends on access to clinical information and risk factors, optimal imaging findings, well-established morphologic features, and traditional and contemporary testing. The recent decline in mortality from breast cancer in rich countries is attributed to increased public awareness, advances in breast imaging and screening and to the new innovations in breast cancer therapy [3]. Machine Learning algorithms can be used for precise classification of breast cancer.
Amrane et al. [4] present two supervised learning classifiers: Naïve Bayes (NB) classifier and K-Nearest neighbours classifier (k-NN) and use accuracy metric to compare two classifiers where k-NN classifier achieved the highest accuracy of 97.51% on Wisconsin Breast Cancer dataset.
Asri et al. [5] present 4 classifiers: Support Vector Machine(SVM), k-NN, NB and C4.5 and compare them using accuracy, precision, recall and f1-score metrics on Wisconsin Breast Cancer dataset.
In this article, feature selection, cross-validation and grid search techniques are used along with Random forest, Decision tree and k-NN classifiers and using Wisonsin Breast Cancer dataset. The classifiers are compared using accuracy, precision, recall and f1-score metrics. The Random forest classifier after grid searching for optimal hyperparamteres and feature selection is performed on the data samples and achieves the accuracy of 99.4% on test set.

2. Methodology

Samples in dataset consist of 30 numeric features and one class label (malign and benign). The values of features are derived from digitalized images of samples extracted by FNA (Fine-Needle aspiration) method. FNA is a diagnostic method that consists of taking a sample of tissue using a hollow needle.
Values of different features fall within various ranges and are scaled using procedure of normalization. Because data contains many features, feature selection is performed to reduce model complexity and improve its performance. Feature selection is performed by keeping only one of the highly correlated features in a group, thus removing the redundant information which can slow down and confuse

a model. After feature selection there are 16 features remaining in the model.

A. Decision tree classifier

Decision tree classifier is a supervised learning method used for classification and regression. The decision tree is constucted of nodes starting from root node that has no incoming edges. All other nodes have exactly one incoming edge. A node with outgoing edges is called an internal or test node. All other nodes are called leaves (also known as terminal or decision nodes). Each internal node splits the instance space into two or more sub-spaces according to a certain discrete function of the input attributes values. In the most frequent case, each test considers a single attribute, such that the instance space is partitioned according to the attribute's value. Each leaf is assigned to one class representing the most appropriate target value. Instances are classified by navigating them from the root of the tree down to a leaf, according to the outcome of the tests along the path [6]. Maximum depth of a tree is defined before training and can impact the performance of classifier so this parameter will be grid searched for optimal value.

B. Random forest classifier

Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest [7]. Random forest classifier is an ensemble learning method that works by combining the predictions of many decision tree classifiers and the final decision is the one with the most votes among individual decision trees. Total number of estimators will be grid searched.

C. K-Nearest neighbours classifier

K-NN (K-Nearest Neighbours) classifier is an algorithm to classify unlabelled data based on their similarity to labelled examples. The sample is classified into the class that is the most common among its k nearest neighbours. The procedure for selecting k nearest neighbours uses a preselected metric for distance calculation [8]. Euclidian distance (Equation 1) is most commonly used metric for distance calculation but there are other metrics to calculate distance such as Manhattan distance (Equation 2) or Chebyshev distance (Equation 3) where samples have n features and p_i and q_i represent values of i-th feature of sample p and q respectively.

$$dist(p,q) = \sqrt{(p_1 - q_1)^2 + \cdots + (p_n - q_n)^2}, \quad (1)$$

$$dist(p,q) = |p_1 - q_1| + \cdots + |p_n - q_n|, \quad (2)$$

$$dist(p,q) = \max(|p_1 - q_1|, \ldots, |p_n - q_n|), \quad (3)$$

Selection of parameter k has a very big impact on the model performance and it will be grid searched.

D. Cross-validation

Cross-validation is implemented to reduce the risk of overfitting and track performance of the classifiers during training. The basic form of cross-validation is k-fold cross-validation. In k-fold cross-validation, the data is first partitioned into k equally (or nearly equally) sized segments or folds. Subsequently k iterations of training and validation are performed such that within each iteration a different fold of the data is used for validation while the remaining k − 1 folds are used for learning [9].
The data is split into training and test set in a 70:30 ratio. During training, 10-fold cross validation is performed.

E. Hyperparameter tuning

Selection of the classifiers' hyperparameters can have a significant impact on the performances so grid search is performed to find the most optimal values for hyperparameters of classifiers [10].
Grid search is used to automatize the process of manually searching for the best parameters. Parameters that are searched are k value for the k-NN classifier, Minimum number of samples required to split an internal node and minimum number of samples required to be at a leaf node for Decision tree classifier and number of estimators and maximum depth for Random forest classifier.
Models are evaluated and compared using different metrics: accuracy, precision, recall and f1-score.

3. Results and Discussion

The classifiers are evaluated and compared on a test set. Evaluation metrics before any improvements are shown in Table 1.

Table 1. *Metrics of classifiers*

Metric	Random forest	decision tree	k-NN
Accuracy	0.99	0.95	0.98
Precision	0.98	0.89	1.0
Recall	0.98	0.98	0.95
F1-Score	0.98	0.93	0.97

Removing highly correlated features results in improved metrics shown in Table 2 and significant decrease in training time. Random forest classifier does not have any false negative samples now so its recall is 1.0.

Table 2. *Metrics of classifiers after feature selection*

Metric	Random forest	decision tree	k-NN
Accuracy	0.99	0.95	0.96
Precision	0.98	0.87	0.93
Recall	1.0	1.0	0.97
F1-Score	0.99	0.93	0.94

Grid searching for the optimal parameters improves the performances of every classifier as shown in Table 3. k-NN classifier has significant improvements in precision, recall and f1-score. Decision tree classifier does not have 1.0 recall anymore but other metrics are improved.

Table 3. *Metrics of classifiers after grid search*

Metric	Random forest	decision tree	k-NN
Accuracy	0.99	0.96	0.99
Precision	1.0	0.96	0.96
Recall	0.98	0.93	1.0
F1-Score	0.99	0.94	0.98

4. Conclusion

Looking at the metrics, it can be concluded that Random forest classifier has the best performance and is the optimal classifier for this problem. Even without any techniques for improvements (grid search and feature selection) this classifier has satisfying results. After feature selection the training of the classifiers is much faster and the performances are improved. After grid searching for optimal parameters, Random forest classifier provides best accuracy of 99.4%.

Reference

[1] Sharma, G. N., Dave, R., Sanadya, J., Sharma, P., & Sharma, K. K. (2010). Various types and management of breast cancer: an overview. Journal of advanced pharmaceutical technology & research, 1(2), 109–126.

[2] Feng, Y., Spezia, M., Huang, S., Yuan, C., Zeng, Z., Zhang, L., Ji, X., Liu, W., Huang, B., Luo, W., Liu, B., Lei, Y., Du, S., Vuppalapati, A., Luu, H. H., Haydon, R. C., He, T. C., & Ren, G. (2018). Breast cancer development and progression: Risk factors, cancer stem cells, signaling pathways, genomics, and molecular pathogenesis. Genes & diseases, 5(2), 77–106. https://doi.org/10.1016/j.gendis.2018.05.001

[3] Masood, S. (2016). Breast Cancer Subtypes: Morphologic and Biologic Characterization. Women's Health, 103–119. https://doi.org/10.2217/whe.15.99

[4] Amrane, M., Oukid, S., Gagaoua, I., & Ensari, T. (2018). Breast cancer classification using machine learning. 2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT). doi:10.1109/ebbt.2018.8391453

[5] Hiba Asri, Hajar Mousannif, Hassan Al Moatassime, Thomas Noel, Using Machine Learning Algorithms for Breast Cancer Risk Prediction and Diagnosis, Procedia Computer Science,Volume 83, 2016, 1064-1069.
https://doi.org/10.1016/j.procs.2016.04.224.

[6] Breiman, L. Random Forests. Machine Learning 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324

[7] Rokach L., Maimon O. (2005) Decision Trees. In: Maimon O., Rokach L. (eds) Data Mining and Knowledge Discovery Handbook. Springer, Boston, MA. https://doi.org/10.1007/0-387-25465-X_9

[8] Zhang Z. (2016). Introduction to machine learning: k-nearest neighbours. Annals of translational medicine, 4(11), 218. https://doi.org/10.21037/atm.2016.03.37
[9] Refaeilzadeh P., Tang L., Liu H. (2009) Cross-Validation. In: LIU L., ÖZSU M.T. (eds) Encyclopedia of Database Systems. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-39940-9_565
[10] Jiménez Á.B., Lázaro J.L., Dorronsoro J.R. (2007) Finding Optimal Model Parameters by Discrete Grid Search. In: Corchado E., Corchado J.M., Abraham A. (eds) Innovations in Hybrid Intelligent Systems. Advances in Soft Computing, vol 44. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-74972-1_17

ISBN: 978-953-8246-22-7

Deep learning methods for detection of carotid artery wall

Miloš Anić [1,2,*], Branko Arsić[2,3], Smiljana Đorović[1,2], Nenad Filipović[1,2]

[1] Faculty of Engineering, University of Kragujevac, Sestre Janjic 6, 34000 Kragujevac, Serbia, anic.milos@kg.ac.rs.
[2] Bioengineering Research and Development Center (BioIRC), Kragujevac, Serbia
[3] Faculty of Science, University of Kragujevac, Sestre Janjic 6, 34000 Kragujevac, Serbia

[*] Corresponding author

Abstract: Carotid artery is the main artery located in human neck. Its main role is to deliver blood to the neck and face muscles as well as, most importantly, to the brain. Carotid artery stenosis is one of many fatal carotid artery diseases involving carotid artery. Development of stenosis on artery wall can cause brain stroke if plaque breaks. Convolutional neural networks (CNNs) proved to be successful in object classification on images as well as object detection on same images. In the field of segmentation of clinical images, U-Net and SegNet architectures proved to have good performances. The aim of this paper was to use CNN to detect carotid artery wall in order to separate artery tissue from stenosis. Automatic segmentation of carotid artery wall was done via SegNet CNN and was compared with modified U-Net based deep convolutional network. Proposed model was evaluated on the images of real patients which were acquired through ultrasound. Experimental results show that this model outperforms models of other deep neural networks.

Keywords: Carotid artery, deep neural networks, U-Net, segmentation, medical images, stenosis, SegNet

1. introduction

Carotid artery is one of the main arteries located in human neck. Its main role is to deliver blood to the neck and face muscles as well as, most importantly, to the brain. Human body has two carotid arteries located in either side of neck. Both arteries have bifurcations. One of the bifurcated arteries, inner carotid artery, has the main role to deliver blood to the brain, while other, exterior carotid artery delivers blood to face and neck [1]. As well as other arteries, carotid arteries can succumb to a number of fatal diseases including carotid artery vasculitis, stroke, stenosis, aneurysms, atherosclerosis etc. The main resemblance of these diseases is their high mortality rate. Thus, great efforts are made to recognize these problems, solve them and even predict their development [1].

Main reasons for high mortality rate of carotid artery diseases is their detection and recognition by clinicians. Even though it is relatively easy detect aneurysms on carotid artery through ultrasound or X-ray, the problem occurs when doctors try to recognize plaque formation, to separate it from the artery wall and to recognize which type of plaque has formed (figure 1).

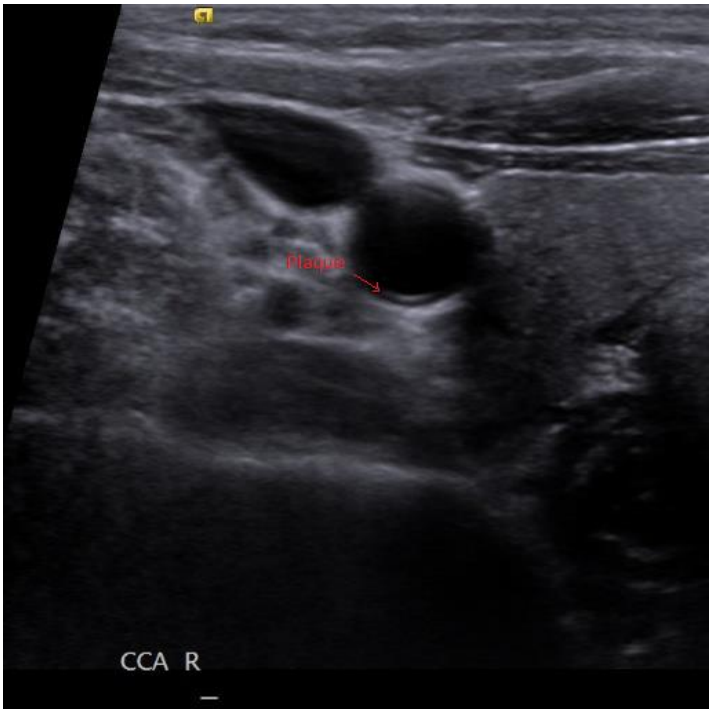


Figure 1. *Transversal cross section of carotid artery and plaque*

With development of technology in medicine, clinicians tried to describe diseases in simpler ways [2-4], while, with the development of neural networks, many attempts to automate the process of disease detection have occurred. As of the start of 21th sanctuary many attempts on automatic characterisation of plaque on blood vessel walls were attempted [5-9] with the previous steps of lumen and wall detection.

In this paper, attempt on automatic carotid wall detection was made as pre-step for plaque detection on carotid artery walls as well as their characterisation. This segmentation was applied on lungs of real clinical patients using SegNet based CNN and was compared with U-Net model. Models were evaluated on a set of ultrasound images with the resolution 512x512. Segmentation results were evaluated in the terms of dice coefficient. Experimental results show that SegNet model outperforms other models.

2. Methodology

In this study, the U-Net [10] and SegNet [11] architectures have been applied on a set of ultrasound images of carotid arteries. Images were separated into two sets, coloured images and black-white (BW) images (figure 2).

Figure 2. Different images from the dataset, coloured (left) and BW (right)

In order to use CNN models, segmented images had to be acquired for validation set. Manual segmentation was done in open source image editing software. Result of the manual segmentation is shown on figure 3.

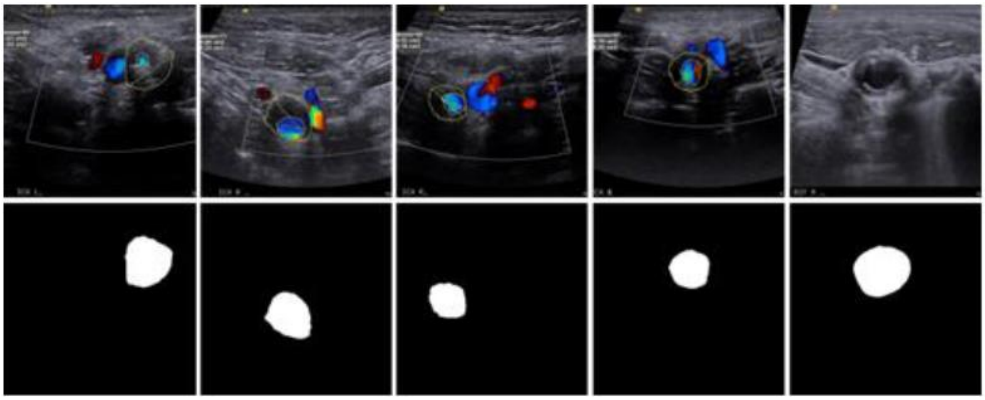


Figure 3. Manually segmented images from the dataset

Modified U-Net based CNN and baseline SegNet architectures were used on the dataset. U-Net architecture used in this study was modified from the aspect of depth. An additional block was added to encoder and decoder figure 4 as it was shown in Arsic et al. 2019 [12].

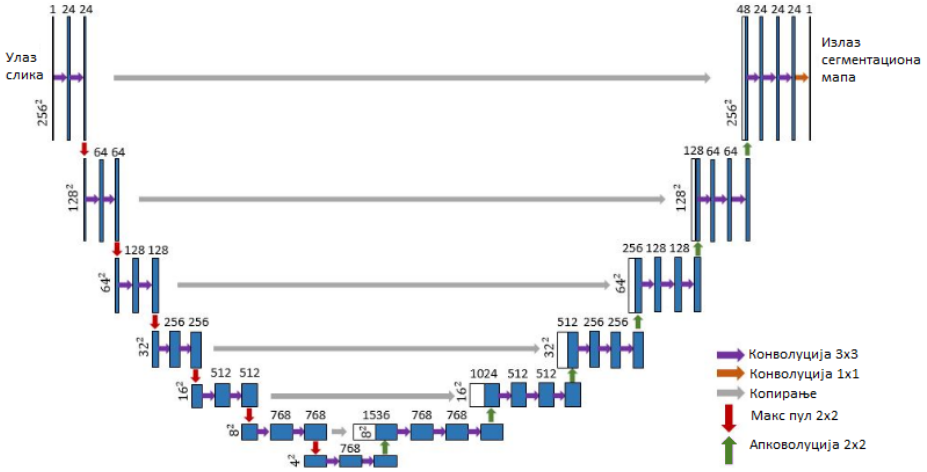


Figure 4. Modified U-Net architecture

Figure 4 shows that every encoder block has two convolutional layers with filter 3x3 which are followed by 2x2 max pull layer and that every decoder block has 2x2 upconvolution layer and a skip connection followed by three 1x1 convolutional layers with sigmoid activation function. All convolution layers are arranged so that height and width of output image is the same as it was on input. Additional difference between baseline and presented U-Net architecture is that modified architecture uses serial normalization after each convolution layer which proved to give better results [13]. Models were trained with a combination of binary crosentropy and dice coefficient.

3. Results and Discussion

In this paper, task of binary classification of segmented images was taken in consideration. Results have been evaluated using dice coefficient (F1 score) on two separated sets (colour and BW). U-Net model achieved F1 score of 0.91 and 0.93 on BW and coloured images respectively while SegNet achieved 0.92 and 0.94. While scores are not very different, SegNet has achieved slightly better results. This is a result of the fact that modified U-Net uses twice as much training parameters compared to SegNet thus getting slightly worse results in the process.

Figure 5 shows a comparison of original and SegNet segmented images.

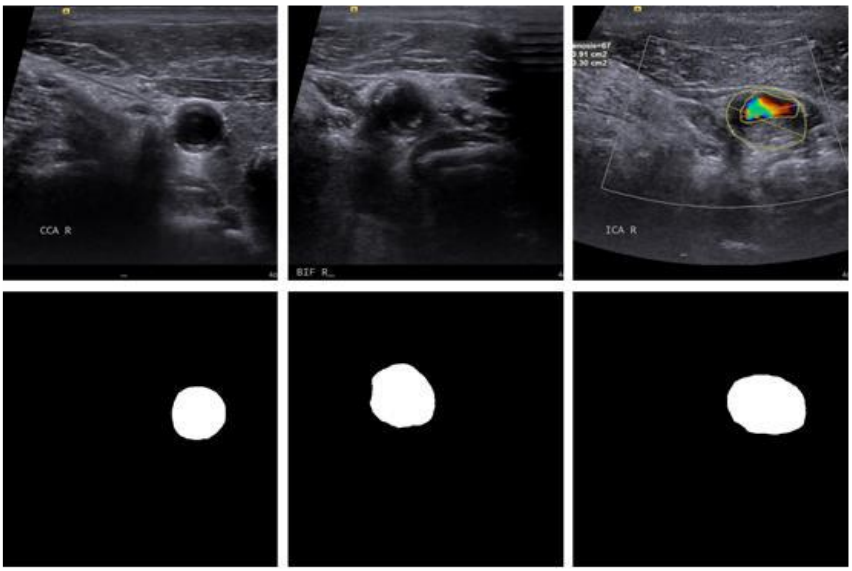


Figure 5. Results of SegNet architecture

As it can be seen in figure 5, SegNet architecture has smoot edges compared to manually segmented images shown on figure 3.

5. Conclusion

This study has shown the use of deep learning methods for detection of carotid artery wall. Set task of was achieved with SegNet neural network with the dice coefficient of 0.92 for BW and 0.94 for coloured images.
Future work will include lumen detection as well as plaque characterization.

Acknowledgements

This paper is supported by the TAXINOMISIS project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 755320. This article reflects only the author's view. The Commission is not responsible for any use that may be made of the information it contains. Research is funded by Serbian Ministry of Education, Science, and Technological Development [451-03-9/2021-14/200107 (Faculty of Engineering, University of Kragujevac)].

Reference

[1] Ashrafian H. Anatomically Specific Clinical Examination of Carotid Arterial Tree. Anatomical Science International, 2007; 82 (1): 16-23.
https://doi.org/10.1111/j.1447-073X.2006.00152.x
[2] Touboul P, J, Hennerici M, G, Meairs S, Adams H, Amarenco P, Bornstein N, Csiba L, Desvarieux M, Ebrahim S, Fatar M, Hernandez Hernandez R, Jaff M, Kownator S, Prati P, Rundek T, Sitzer M, Schminke U, Tardif J, -C, Taylor A, Vicaut E, Woo K, S, Zannad F, Zureik M. Mannheim Carotid Intima. Media Thickness Consensus 2004–2006. Cerebrovasc Dis 2007; 23: 75-80. doi: 10.1159/000097034
[3] Howard G, Sharrett A. R, Heiss G, Evans G. W, Chambless L. E, Riley W. A, Burke G. L. Carotid artery intimal-medical thickness distribution in general populations as evaluated by B-mode ultrasound. ARIC Investigators. Stroke. 1993, 24: 1297-1304.
doi: 10.1161/01.str.24.9.1297

[4] Lorenz M.W., Polak J. F., Kavousi M., Mathiesen E. B., Volzke H., Tuomainen T.-P., Sander D., Thompson S.G. Carotid intima-media thickness progression to predict cardiovascular events in the general population (the PROG-IMT collaborative project): A meta-analysis of individual participant data. *The Lancet*. 2012, 379 (9831), pp. 2053-2062. DOI: 10.1016/S0140-6736(12)60441-3
[5] K. Yoshida, O. Narumi, M. Chin, K. Inoue, T. Tabuchi, K. Oda, M. Nagayama, N. Egawa, M. Hojo, Y. Goto, Y. Watanabe and S. Yamagata. Characterization of Carotid Atherosclerosis and Detection of Soft Plaque with Use of Black-Blood MR Imaging. American Journal of Neuroradiology. 2008, 29 (5) 868-874;
https://doi.org/10.3174/ajnr.A1015
[6] Tae Ho Park. Evaluation of Carotid Plaque Using Ultrasound Imaging. J Cardiovasc Ultrasound. 2016 Jun; 24(2): 91–95;
[7] Amer M. Johri, Vijay Nambi, Tasneem Z. Naqvi, Steven B. Feinstein, Esther S.H. Kim, Margaret M. Park, Harald Becher, Henrik Sillesen. Recommendations for the Assessment of Carotid Arterial Plaque by Ultrasound for the Characterization of Atherosclerosis and Evaluation of Cardiovascular Risk: From the American Society of Echocardiography. Guidelines and Standards. 33, 8, P917-933;
[8] Ran Zhou, Aaron Fenster, Yujiao Xia J., David Spence, Mingyue Ding. Deep learning-based carotid media-adventitia and lumen-intima boundary segmentation from three-dimensional ultrasound images. Medical Pythisics, 2019, 46, 7:3180-3193;
[9] Lekadir, K., Galimzianova, A., Betriu, A., Del Mar Vila, M., Igual, L., Rubin, D. L., Fernandez, E., Radeva, P., & Napel, S. A Convolutional Neural Network for Automatic Characterization of Plaque Composition in Carotid Ultrasound. IEEE journal of biomedical and health informatics, 2017, 21(1), 48–55.
[10] Ronneberger O, Fischer P, and Brox T. U-net: Convolutional networks for biomedical image segmentation. in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2015, doi: 10.1007/978-3-319-24574-4_28.
[11] Badrinarayanan V, Kendall A, and Cipolla R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. IEEE Trans. Pattern Anal. Mach. Intell., 2017, doi: 10.1109/TPAMI.2016.2644615.
[12] Arsic B, Obrenovic M, Anic M, Tsuda A and Filipovic N. Image Segmentation of the Pulmonary Acinus Imaged by Synchrotron X-Ray Tomography. 2019 IEEE 19th International Conference on Bioinformatics and Bioengineering (BIBE), 2019, pp. 525-531, doi: 10.1109/BIBE.2019.00101.
[13] Zhou X and Yang G. Normalization in training u-net for 2-d biomedical semantic segmentation. IEEE Robotics and Automation Letters, vol. 4, no. 2, pp. 1792–1799, April 2019.

ISBN: 978-953-8246-22-7

Forest Covertype Prediction based on cartographic parameters using neural network

Lazar Dašić [1*]

[1] Faculty of Engineering University of Kragujevac, Sestre Janjić 6, 34000 Kraguejvac, Serbia, ldasis345@gmail.com.

Abstract: The study shows success of Artificial Neural Network in making right prediction of forest cover type of certain area based on that area cartographic parameters. The study evaluated area of Roosevelt National Forest in Colorado. Artificial neural network used for making predictions is Multilayer perceptron using Stochastic Gradient Descent as an optimizer and Sparse categorical cross-entropy as loss function. The result of the study showed that right neural network model can make over 90% correct predictions, which vastly transcend traditional statistical models. This success means that similar model can be used for making predictions in other areas around the world, which can decrease costs of managing natural resource inventory and even make managing possible for inaccessible areas.
Keywords: Artificial intelligence, Forest cover type, Multiclass classification, Sparse categorical cross-entropy

1. Introduction

Management of natural resource inventory information is key part of any private or federal property. One of the most important parts of that inventory is forest cover type. Traditional way of cover type recording is by field personnel. This is both time and cost consuming and even in some cases not possible due to inaccessibility of wilderness areas. Alternative method is using predictive models.
J.A. Blackard, D.J. Dean et al. (1999) [1] applied a feedforward three-layers artificial neural network model with the goal of improving accuracy of prediction over traditional statistical model based on Gaussian discriminant analysis.
A feedforward neural network is an artificial neural network wherein connections between the nodes do not form a cycle [2].
Model used by Blackard and Dean had architecture of three fully connected layers, where number of neurons in each layer is as follows: 54-100-7. Activation function for input layer was linear and activation function for hidden and output layer was sigmoid function. Loss function of choice was mean square error. Learning algorithm used for model was Backpropagation. Optimal learning rate found from their research was 0.05. Model was trained for 1000 epochs. Their model achieved accuracy of 70.58% for test data set.
Research question of this study is to see if it is possible to improve J.A. Blackard, D.J. Dean et al. (1999) neural network model [1] and show what does it take to do so.
The goal of this study is to improve their model using more modern approach and show progress of both technology and Artificial Intelligence in the last two decades.

2. Methodology

Dataset used to feed the neural network model comes from these areas of Roosevelt National Forest: Rawah (29 628 hectares), Comanche Peak (27 389 hectares), Neota (3904 hectares) and Cache la Poudre (3817 hectares). Full rights for database belong to Jock A. Blackard and Colorado State University [11].

Total area was divided into 581 012 raster cells dimension 30 × 30-m, where each cell is described with following variables(with their units of measure):
1. Elevation (m),
2. Aspect (azimuth from true north),
3. Slope (°),
4. Horizontal distance to nearest surface water feature (m),
5. Vertical distance to nearest surface water feature (m),
6. Horizontal distance to nearest roadway (m),
7. A relative measure of incident sunlight at 09:00 h on the summer solstice (index),
8. A relative measure of incident sunlight at noon on the summer solstice (index),
9. A relative measure of incident sunlight at 15:00 h on the summer solstice (index),
10. Horizontal distance to nearest historic wildfire ignition point (m),
11. Wilderness area designation (four binary values, one for each wilderness area),
12. Soil type designation (40 binary values, one for each soil type).

Total number of independent variables is 54 which is the number of inputs neural network is going to have. Output of the model will be integer number representing dominant forest type for observed 30 × 30-m raster cell. Types of forest are as following:
1. Spruce/fir – 211 840 samples
2. Lodgepole pine – 283 301 samples
3. Ponderosa pine – 35 754 samples
4. Cottonwood/Willow – 2 747 samples
5. Aspen – 9 434 samples
6. Douglas-fir – 17 367 samples
7. Krummholz – 20 510 samples

Upon understanding the goal of the study and observing data, it is clear that neural network will have to solve the problem of multiclass classification. Multiclass or multinomial classification is the problem of classifying instances into one of three or more classes [3]. Choosing the right hyperparameters is now significantly easier knowing type of the problem.

Before actually building and training neural network model it is important to do data preprocessing. First step in data preprocessing is to normalize data. This will result in samples being formatted the way that will give better accuracy of predictions, as well as faster and more stable training process. Normalization is a rescaling of the data from the original range so that all values are within the range of 0 and 1 [4]. Input parameters represented by numbers 1-10 are given in integer form. This results in values for different parameters to be on a different scale. To put all of this integer parameters on the same scale, rescaling(min-max normalization) method was used.

Following successful data normalization it is necessary to split data into following data sets: training data set, validation data set and test data set. After experimenting with different proportions among data sets relatively to the size of total data set it was discovered that 60% of total data for training(348 606 samples) and 20% each for validation and test data set(116 203 samples) was optimal way to split data. It is important to note that data samples were shuffled before splitting into different sets, because originally total data set was sorted by wilderness area designation areas, which causes training, validation and test data set to not be representative of the overall distribution of the data. Shuffling also reduces variance and makes sure that model remain general and overfit less. Overfitting is the production of an analysis that corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably [10].

With data processed neural network modelling could begin. Artificial neural network models require several architectural and training parameters to be selected prior to analysis. The optimal number of hidden layers and the number of nodes per hidden layer are generally not known a priori for a specific data set, and must be empirically determined through an examination of different parameter settings [5][6]. After extensive experimentation with different network architectures, best results were accomplished using four-layer network with input and output layer accompanied by two hidden layers.

Number of neurons in each layer is as follows: 54-120-60-7. Each neuron in output layer represents one forest type and it will give probability of that forest type to be on certain land cell. Total number of trainable parameters is 17 257.
For input layer and hidden layers activation function of choice is ReLU, and for output layer activation function is Softmax. This combination of activation functions for multiclass classification have become pretty common in recent years. For years most popular output layer activation function was sigmoid, but it was never good for classification problems where classes are mutually exclusive, which is case for forest cover types. The softmax function is a function that turns a vector of K real values into a vector of K real values that sum to 1 [7], using equation #(1).

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_i}}, \#(1)$$

where z is output value, i index of i -th neuron in output layer and K number of neurons in output layer. This way each forest type gets evaluation in a form of chance to be best prediction.
ReLU function is obvious choice given the benefits it provides. It does not saturate, it has linear behaviour which makes training process fast and it can learn complex relationships in the data.
As for loss function clear choice is Sparse Categorical Cross-entropy, since problem is of multiclass classification type and outputs are given in integer format, rather than being one-hot encoded. Advantage of using sparse categorical cross entropy is it saves time in memory as well as computation because it simply uses a single integer for a class, rather than a whole vector. This loss function is computed using equation #(2).

$$J(w) = -\sum_{i=1}^{N} y_i \log(\hat{y}_i), \#(2)$$

where N is number of neurons in a layer, w refer to the model parameters, y_i is the true label and $\hat{y}_i$ is the predicted label. Training algorithm, or optimizer, used for neural network learning is Stochastic Gradient Descent(SGD). During research and experimentation with other optimizers, SGD came on par with the more modern algorithm that are based on SGD, like Adam and AdaGrad. With the right learning rate SGD achieved better generalization without sacrificing accuracy or too much of the training time. Optimal learning rate is found to be larger than for most problems with the value of 0.1. Using SGD, it was crucial to find right learning rate since too high learning rate will make the learning jump over minima but a too low learning rate will either take too long to converge or get stuck in an undesirable local minimum [8].
Neural network model has been trained for 50 epochs giving accuracy on validation set of 90.19%. It was experimented with different numbers of epochs, but after the 50th epoch it was seen that loss function plateaus, thus making following epochs pointless. Graph of training process is showed on Figure 1, where change of loss function on training set(blue colour), loss function on validation set(green colour), accuracy on training set(orange colour) and accuracy on validation set(red colour) can be seen during 50 epochs training period.

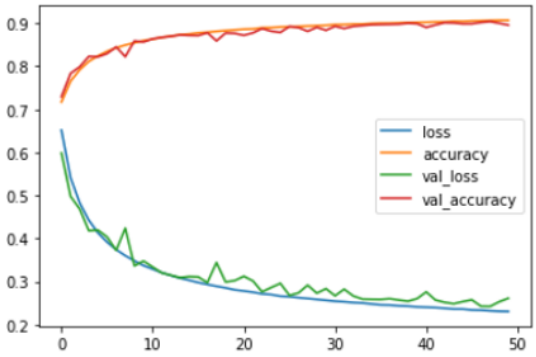


Figure 1. *Values of important functions during training process*

Training process took approximately 17 seconds per epoch, which contributes to around 15 minutes training time.

3. Results and Discussion

Following successful training, final evaluation of neural network model was made using test data set. Time to make predictions on 116 203 samples was 2.6231977 seconds. Precision of predictions for each forest type is shown in Table 1.

Table 1. *Accuracy of predictions*

Forest type	Precision
Spruce/fir	0.93
Lodgepole pine	0.89
Ponderosa pine	0.93
Cottonwood/Willow	0.72
Aspen	0.72
Douglas-fir	0.76
Krummholz	0.90

From the Table 1 it is clear that precision for Cottonwood/Willow, Aspen and Douglas-fir forest types is not as high as for the rest of types. This is a consequence of low number of samples in data set that contain those three forest types. It could mean that with bigger data set precision could be even better. Another important metric that shows neural networks performance is a confusion matrix. A confusion matrix summarizes the classification performance of a classifier with respect to some test data. It is a two-dimensional matrix, indexed in one dimension by the true class of an object and in the other by the class that the classifier assigns [9]. Confusion matrix for this model is shown in Figure 2.

Spruce/fir	36090	5646	0	0	79	27	370
Lodgepole pine	2580	53365	184	4	416	270	30
Ponderosa pine	2	224	6032	154	63	619	0
Cottonwood/Willow	0	0	41	492	0	36	0
Aspen	22	349	12	0	1488	15	0
Douglas-fir	3	99	216	38	7	3139	0
Krummholz	297	56	0	0	1	0	3737
	Spruce/fir	Lodgepole pine	Ponderosa pine	Cottonwood/Willow	Aspen	Douglas-fir	Krummholz

Figure 2. *Confusion matrix of test data*

From the Figure 2 it is easy to see what forest types neural network confused most often: Lodgepole pine for Spruce/fir, Spruce/fir and Aspen for Lodgepole pine, Douglass-fir for Ponderosa pine, Lodgepole pine for Aspen and Spruce/fir for Krummholz.

4. Conclusion

This study successfully showed that model that J.A. Blackard, D.J. Dean et al. (1999) [1] used, can be improved drastically with change of hyperparameters for more modern options that are better suited for multiclass classification problem. Besides showing progress made in the field of Neural Networks, progress in computation power is showed in the time spent on training neural network. Training time was reduced from 45 hours [1] to just 15 minutes. With accuracy of over 90% this neural network model can be confidentially used for managing natural inventory of Roosevelt National Forest. For future work this neural network model can be modified in way that will make it usable for predicting forest cover types of other wilderness areas around the world with different forest cover types.

Acknowledgments
I would like to offer my special thanks to professor Tijana Šušteršič for her patient guidance, enthusiastic encouragement and useful critiques of this research work.

Reference

[1] Blackard, J. A., & Dean, D. J. (1999). Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. Computers and electronics in agriculture, 24(3), 131-151.
https://doi.org/10.1016/s0168-1699(99)00046-0
[2] Zell, A. (1994). Simulation neuronaler netze (Vol. 1, No. 5.3). Bonn: Addison-Wesley.
https://doi.org/10.1007/978-3-642-78486-6_88
[3] Ou, G., & Murphey, Y. L. (2007). Multi-class pattern classification using neural networks. Pattern Recognition, 40(1), 4-18.
https://doi.org/10.1016/j.patcog.2006.04.041
[4] Sola, J., & Sevilla, J. (1997). Importance of input data normalization for the application of neural networks to complex industrial problems. IEEE Transactions on nuclear science, 44(3), 1464-1468.
https://doi.org/10.1109/23.589532
[5] Haykin, S., & Lippmann, R. (1994). Neural networks, a comprehensive foundation. MacMillan College, New York 696 pp
https://doi.org/10.1017/s0269888998214044
[6] Marzban, C., & Stumpf, G. J. (1996). A neural network for tornado prediction based on Doppler radar-derived attributes. Journal of Applied Meteorology and Climatology, 35(5), 617-626.
https://doi.org/10.1175/1520-0450(1996)035%3C0617:annftp%3E2.0.co;2
[7] Ren, Y., Zhao, P., Sheng, Y., Yao, D., & Xu, Z. (2017, August). Robust softmax regression for multi-class classification with self-paced learning. In Proceedings of the 26th International Joint Conference on Artificial Intelligence (pp. 2641-2647).
https://doi.org/10.24963/ijcai.2017/368
[8] Buduma, N., & Locascio, N. (2017). Fundamentals of deep learning: Designing next-generation machine intelligence algorithms. " O'Reilly Media, Inc.".
https://dl.acm.org/doi/book/10.5555/3161223
[9] Sammut, C., & Webb, G. I. (Eds.). (2010). Encyclopedia of machine learning. Springer Science & Business Media.
https://doi.org/10.1007/978-0-387-30164-8_157
[10] Leinweber, D. J. (2007). Stupid data miner tricks: overfitting the S&P 500. The Journal of Investing, 16(1), 15-22.
https://doi.org/10.3905/joi.2007.681820
[11]Covertype Data Set, accessed: 14.05.2021.; from: https://archive.ics.uci.edu/ml/datasets/Covertype

ISBN: 978-953-8246-22-7

CO_2 emissions calculation from steam generator during combustion of different fuel types

Danijel Marjanović [1], Vedran Mrzljak [2]

[1] Tehnički Fakultet Sveučilišta u Rijeci, Vukovarska 58, 51000 Rijeka, Hrvatska, dmarjanovic@riteh.hr
[2] Tehnički Fakultet Sveučilišta u Rijeci, Vukovarska 58, 51000 Rijeka, Hrvatska, vedran.mrzljak@riteh.hr

Abstract: The paper presents calculation results of carbon dioxide emissions from a ship's steam generator during the usage of different fuel types for 23 different loads. The calculation is done for the combined usage of HFO (Heavy Fuel Oil) and natural gas which is the standard working mode for the observed steam generator. After this, the calculation of CO_2 emissions is done separately for HFO, natural gas, 10 types of lignite coal, 4 types of anthracite coal and for bark wood. It is assumed that the steam generator produces an equal amount of heat in each load using every fuel type. The results are compared between the combined usage of HFO and natural gas, their respective separate usage, the average emissions of lignite coal, the average emissions of anthracite coal and for the emissions of bark wood fuel. It is found that the emissions are lowest for the usage of natural gas and it is closely followed by the combined usage of HFO and natural gas, while the use of only HFO, lignite, anthracite, or bark wood significantly increases the CO_2 emissions.
Keywords: Carbon dioxide, Coal, Emissions, Fuel, Generated heat, Heavy fuel oil, Natural gas

1. Introduction

Carbon dioxide emissions are the main reason why the global climate changes [1]. The combustion of fossil fuels is the largest anthropogenic influence on climate change mainly caused by emitted greenhouse gasses [2]. That makes the problem of CO_2 emissions from fuel combustion of great importance nowadays. That is the reason why the amounts and the impact of emissions have to be known, and it should be strived to reduce those amounts as much as possible. Also, the requirements of reducing greenhouse gas emissions by legislation are becoming more rigorous. By 2050 international shipping will be required to reduce the emissions by at least 50% [3]. These emission standards are continuously getting more rigorous for all areas of human influence, not only for maritime steam generators but for all fossil fuel powered machines. That makes the data regarding emissions from fossil fuel combustion even more relevant for future development. A proper selection of fuel for combustion can notably reduce all the emissions (not only CO_2 emissions analyzed in this paper) [4]. In this paper is presented a comparison of the impact that different fuel types have on the amount of CO_2 emissions in a ship's steam generator. The cross section of the marine steam generator is presented in Figure 1. The steam generator operates using two fuels at the same time - it uses HFO (Heavy Fuel Oil) and natural gas. Emissions of CO_2 are going to be calculated for the combustion of different fuel types in this steam generator. For reference, the amount of heat energy that the generator provides will be always equal at each load (regardless of used fuel). Therefore, the amount of each used fuel will be calculated from a constant released heat energy (released heat energy is constant at each load, but it varies for a different loads). The data of fuel consumption for HFO and natural gas in combined combustion are available for 23 different load capacities of the steam generator. Also, the lower heating value and the carbon mass fractions in each fuel are available [4]. The fuels that are going to be observed are the following: HFO (Heavy Fuel Oil) and natural gas in combined combustion as well as various fuels in an independent combustion: HFO, natural gas, kangal lignite, cayirhan lignite, elbistan lignite, orhaneli lignite, seyitomer lignite, tuncbilek lignite, yatagan lignite, soma lignite, yenikoy lignite, can lignite, armutcuk anthracite, catalgazi anthracite, karadon anthracite, amasra anthracite and bark wood. The required calculation data for each fuel (instead of HFO and natural gas) are found in the literature [6]. The use of coal on ships has mostly been substituted by oil which is already giving way for alternate

fuels such as natural gas. The use of natural gas is expected to increase drastically in the ship industry in the future due to higher emission standards [7, 8].

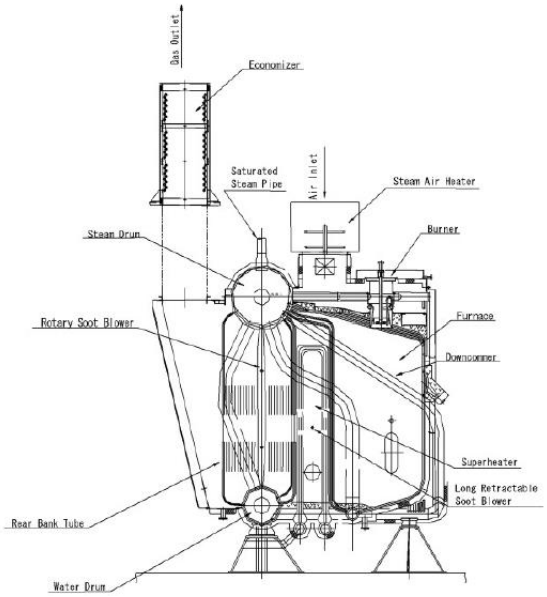


Figure 1. *Cross section of the analyzed marine steam generator [5]*

2. Methodology

For the combustion of any fuel, an approximate calculation of CO_2 emissions can be made according to the Eq. 1:

$$CO_2\left(\frac{kg}{h}\right) = Fuel\ consumption\left(\frac{kg}{h}\right) \cdot Carbon\ mass\ fraction\ in\ the\ fuel \cdot \frac{44}{12}. \quad (1)$$

The above-mentioned equation is derived from the chemical reaction of the elemental carbon combustion derived on 1 kg of carbon, Eq. 2:

$$12\ kg\ C + 32\ kg\ O_2 \rightarrow 44\ kg\ CO_2. \quad (2)$$

If multiple fuel types are used as in this marine steam generator, the calculation should be done for each fuel and then summarized. The comparison of different emissions for each fuel type makes sense only if the referent element is defined. In this paper the referent element is the amount of heat produced by steam generator which remains constant at each load, regardless of used fuel. The amount of heat is calculated using the Eq. 3 where Q represents the produced heat amount and is calculated in a way that the lower heating value of each fuel is multiplied by the fuel consumption used for the combustion purpose. That expression can be altered to calculate the fuel consumption which depends of the generated heat and the lower heating value of any fuel, as shown in Eq. 4.

$$Q\ (MW) = Lower\ heating\ value\left(\frac{MJ}{kg}\right) \cdot Fuel\ consumption\left(\frac{kg}{s}\right). \quad (3)\ (4)$$

$$Fuel\ consumption\ (kg/s) = Q\ (MW)/Lower\ heating\ value\ (MJ/kg). \quad (4)$$

Firstly, the generated heat is calculated for each load of the steam generator using the fuel consumptions of the combined HFO and natural gas combustion [4]. Obtained heat amount at each load is used for the emissions comparison of all the other different fuels [6]. Then, it is possible to calculate the fuel consumption for all the other fuel types based on the data of the generated heat. That is done using the Eq. 4. The CO_2 emissions of the combined combustion of HFO and natural gas at each steam generator load are calculated using the Eq. 1. Finally, the CO_2 emissions for every fuel type in independent combustion at each steam generator load are calculated by using Eq. 1 again.

3. Results and discussion

The results of the calculations are presented in Figures 2, 3, 4 and 5. Because of the condition that the amount of heat produced by steam generator at each load remains the same, the amount of the fuel differs and, in that way, leads to similar CO_2 emissions for different coal types. The similar CO_2 emissions for 10 different lignite coal types are presented in Figure 2.

In Figure 3, the amount of heat generated by the steam generator is calculated using Eq. 3. In all steam generator loads, HFO consumption is almost the same (increase is evident at low load during steam generator startup), while the natural gas consumption continuously increases during increase in steam generator load.

The steam generator produces the least CO_2 emissions using natural gas, followed by the combined usage of HFO and natural gas. During the usage of only HFO the CO_2 emissions are significantly higher in comparison to combined usage of two fuels or when only natural gas is used, Figure 4.

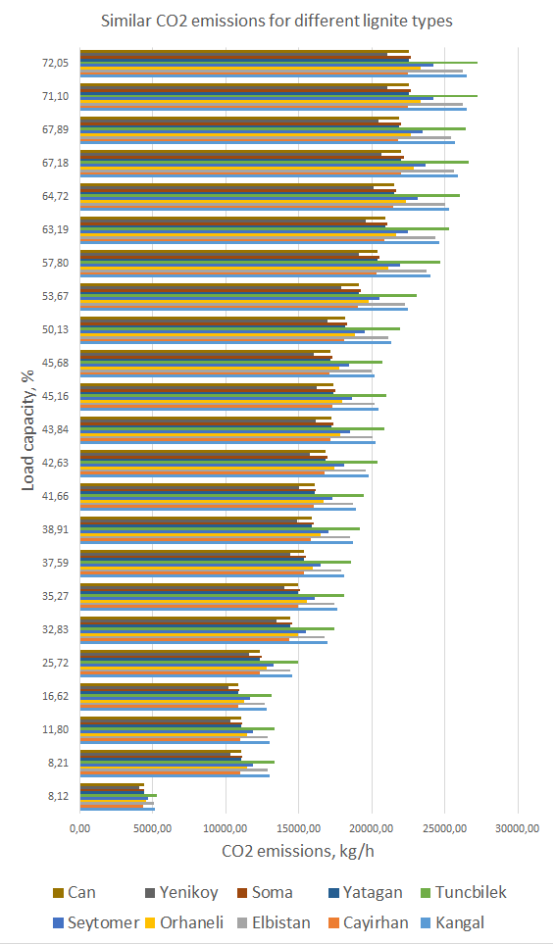


Figure 2. Similar *CO_2 emissions for 10 different lignite coal types at different steam generator loads*

In Figure 5, the CO_2 emissions of the combined combustion of HFO and natural gas, the individual combustion of natural gas, HFO, bark wood and the averaged emissions for different lignite coal types and anthracite coal types (10 lignite coal types and all 4 anthracite coal types) are compared. It is clear from Figure 4 and 5 that the lowest emissions are obtained for the individual combustion of natural gas thanks to its high lower heating value and the combustible hydrogen that reduces the carbon mass fraction in fuel. Natural gas generates lower CO_2 emissions for the same amount of produced heat in comparison to all the other observed fuels. The combined combustion of HFO and natural gas gives a slightly higher CO_2 emissions than just the combustion of natural gas, but still, mentioned CO_2 emissions are notably lower in comparison to the individual combustion of HFO. The combustion of all coal types and bark wood produces significantly higher CO_2 emissions in comparison to HFO and natural gas. Therefore, it can be concluded that removal of coal or bark wood from marine propulsion systems notably reduces CO_2 emissions. Further reducing of CO_2 emissions can be obtained only by using natural gas as an independent fuel.

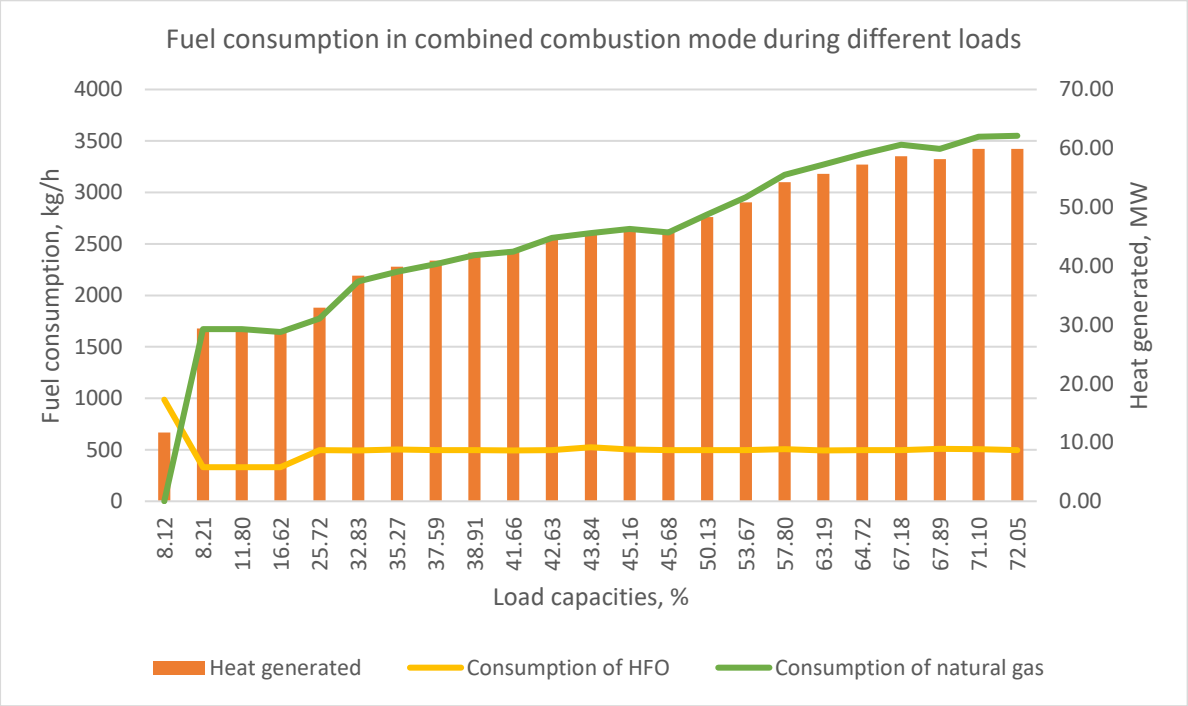


Figure 3. *Consumption of HFO and natural gas for different steam generator load capacities along with generated heat*

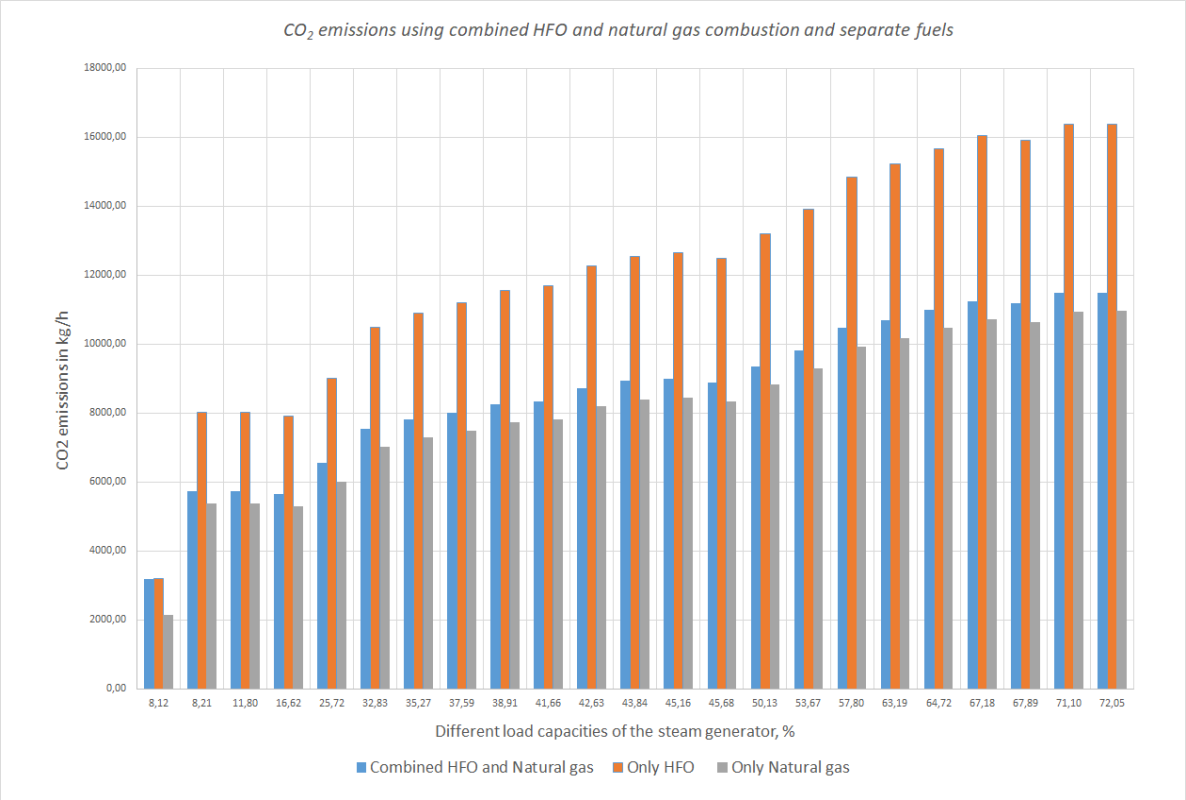


Figure 4. *CO_2 emissions using combined HFO and natural gas combustion and separately HFO and natural gas*

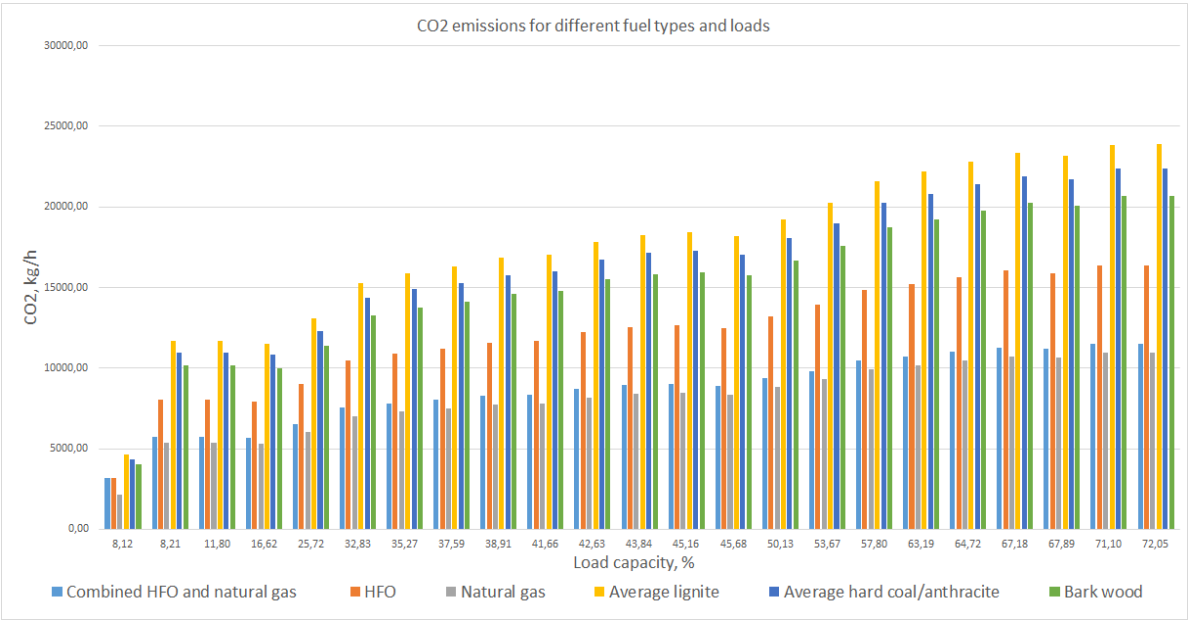


Figure 5. *CO_2 emissions using combined HFO and natural gas combustion and separate fuel combustion*

4. Conclusion

In this paper the presented calculations confirm the advantages of proper fuel choice for the marine steam generator with an aim to reduce CO_2 emissions. In many cases, the price of fuel is the main reason

why newer and cleaner approaches are not taken in consideration. With more rigorous regulations regarding emissions, the choice of less polluting fuels becomes a necessity. In the marine environment, analysis performed in this paper shows the dominance of natural gas in regards to CO_2 emissions when compared to all the other observed fuels. At any steam generator load, CO_2 emissions of natural gas are notably lower than emissions of any other analyzed fuel. In future works it is possible to consider different methods of fuel production and extraction and therefore the emissions produced during the whole operation life cycle. Also, the usage of alternative fuels and renewable energy sources can be an interesting way for the reduction of all emissions (not only CO_2 emissions).
Along with the mentioned, in the fuel production and its usage in various energy systems, the application of contemporary numerical methods can bring many benefits [9, 10]. Usually, the most important benefits of such numerical methods are fuel consumption reduction and increasing overall process efficiency – which are directly related to CO_2 emissions reduction.

References

[1] https://ourworldindata.org/co2-emissions
[2] Quadrelli, Roberta, and Sierra Peterson. "The energy–climate challenge: Recent trends in CO2 emissions from fuel combustion." *Energy policy* 35.11 (2007): 5938-5952. https://doi.org/10.1016/j.enpol.2007.07.001
[3]IMO 2018 Low Carbon Shipping and Air Pollution Control (www.imo.org/en/MediaCentre/HotTopics/GHG/Pages/default.aspx) (London: International Maritime Organization) (Accessed: 10 May 2021)
[4] Mrzljak, Vedran, Igor Poljak, and Vedran Medica-Viola. "Dual fuel consumption and efficiency of marine steam generators for the propulsion of LNG carrier." *Applied Thermal Engineering* 119 (2017): 331-346. https://doi.org/10.1016/j.applthermaleng.2017.03.078
[5] Main boilers operation and maintenance instructions (MB-4E-KS), Mitsubishi Heavy Industries Ltd, Nagasaki Shipyard & Machinery Works, Nagasaki, Japan, 2005.
[6] Oruc, Onur, and Ibrahim Dincer. "Environmental impact assessment of using various fuels in a thermal power plant." *International Journal of Global Warming* 18.3-4 (2019): 191-205. https://doi.org/10.1504/IJGW.2019.101082
[7] Žaglinskis, Justas, Paulius Rapalis, and Nadezda Lazareva. "An overview of natural gas use in ships: necessity and engine supply." *Periodica Polytechnica Transportation Engineering* 46.4 (2018): 185-193. https://doi.org/10.3311/PPtr.11708
[8] Endresen, Øyvind, et al. "A historical reconstruction of ships' fuel consumption and emissions." *Journal of Geophysical Research: Atmospheres* 112.D12 (2007). https://doi.org/10.1029/2006JD007630
[9] Baressi Šegota, Sandi, et al. "Improvement of Marine Steam Turbine Conventional Exergy Analysis by Neural Network Application." *Journal of Marine Science and Engineering* 8.11 (2020): 884. https://doi.org/10.3390/jmse8110884
[10] Lorencin, Ivan, et al. "Genetic algorithm approach to design of multi-layer perceptron for combined cycle power plant electrical power output estimation." *Energies* 12.22 (2019): 4352. https://doi.org/10.3390/en12224352