GLOSSARY

Internet of Things (IoT): a network on which billions of devices are connected over the Internet

LPWAN (Low Power Wide Area Network): a network that covers a large distance with low data rate

Wireless Sensor Nodes (WSNs): nodes used for monitoring certain parameters

System Requirements Specification (SRS): a document specifying the recommendations and requirements of the system being conducted

Define-Measure-Analyze-Improve-Control (DMAIC): a framework on improving a product or system

Long Short-Term Memory (LSTM): a deep learning model that is part of the recurrent neural network

Recurrent Neural Network (RNN): a type of neural network; an alternative to convolutional network

Stochastic Gradient Method (SGM): a stochastic algorithm

Root Mean Square Error (RMSE): a forecasting error type that calculates the difference between the test and predicted values

Normalized Root Mean Square Error (NRMSE): an indicator as to whether the predicted data are within acceptable range or not

REFERENCES

Fu Editor, M. C. (n.d.). *International Series in Operations Research & Management Science Handbook of Simulation Optimization*. http://www.springer.com/series/6161

Fuller, A., Fan, Z., Day, C., & Barlow, C. (2020). Digital Twin: Enabling Technologies, Challenges and Open Research. *IEEE Access*, *8*, 108952–108971. https://doi.org/10.1109/ACCESS.2020.2998358

Gulati, K., Kumar Boddu, R. S., Kapila, D., Bangare, S. L., Chandnani, N., & Saravanan, G. (2021). A review paper on wireless sensor network techniques in Internet of Things (IoT). *Materials Today: Proceedings*, *51*, 161–165. https://doi.org/10.1016/j.matpr.2021.05.067

Liu, M., Fang, S., Dong, H., & Xu, C. (2021). Review of digital twin about concepts, technologies, and industrial applications. *Journal of Manufacturing Systems*, *58*, 346–361. https://doi.org/10.1016/j.jmsy.2020.06.017

Michelinakis, F., Al-selwi, A. S., Capuzzo, M., Zanella, A., Mahmood, K., & Elmokashfi, A. (2020). *Dissecting Energy Consumption of NB-IoT Devices Empirically*. http://arxiv.org/abs/2004.07127

Singh, R. K., Puluckul, P. P., Berkvens, R., & Weyn, M. (2020). Energy consumption analysis of LPWAN technologies and lifetime estimation for IoT application. *Sensors (Switzerland)*, *20*(17), 1–22. https://doi.org/10.3390/s20174794

Srinidhi, N. N., Dilip Kumar, S. M., & Venugopal, K. R. (2019). Network optimizations in the Internet of Things: A review. In *Engineering Science and Technology, an International Journal* (Vol. 22, Issue 1, pp. 1–21). Elsevier B.V. https://doi.org/10.1016/j.jestch.2018.09.003

Tan, L. (n.d.). *Comparison of LoRa and NB-IoT in Terms of Power Consumption*.

Aswin, S., Geetha, P., & Vinayakumar, R. (2018). Deep Learning Models for the Prediction of Rainfall. *Proceedings of the 2018 IEEE International Conference on Communication and Signal Processing, ICCSP 2018*, 657–661. https://doi.org/10.1109/ICCSP.2018.8523829

Danapati, A. (2022). *ENERGY HARVESTING AT A 110-KV-CABLE FOR INDEPENDENT MONITORING PURPOSES*.

Hardt, M., Recht, B., & Singer, Y. (2016). *Train faster, generalize better: Stability of stochastic gradient descent*.

Mahmoud, M. S., & Mohamad, A. A. H. (2016). A Study of Efficient Power Consumption Wireless Communication Techniques/ Modules for Internet of Things

(IoT) Applications. *Advances in Internet of Things*, *06*(02), 19–29. https://doi.org/10.4236/ait.2016.62002

Shcherbakov, M. V., Brebels, A., Shcherbakova, N. L., Tyukov, A. P., Janovsky, T. A., & Kamaev, V. A. evich. (2013). A survey of forecast error measures. *World Applied Sciences Journal*, *24*(24), 171–176. https://doi.org/10.5829/idosi.wasj.2013.24.itmies.80032

Lauridsen, M., Krigslund, R., Rohr, M., & Madueno, G. (2018). An Empirical NB-IoT Power Consumption Model for Battery Lifetime Estimation. *IEEE Vehicular Technology Conference*, *2018-June*, 1–5. https://doi.org/10.1109/VTCSpring.2018.8417653

Product SIM7600A SIM7600SA SIM7600E SIM7600E-L1C Form Factor Dimensions(mm) Frequency Bands. (2020).

Shankar Basu, S., Kumar Sultania, A., Famaey, J., & Hoebeke, J. (n.d.). *Experimental Performance Evaluation of NB-IoT*.

SIM7600E. (n.d.).

SIM8262E-M2. (2020). www.simcom.com

Sinha, R. S., Wei, Y., & Hwang, S. H. (2017). A survey on LPWA technology: LoRa and NB-IoT. In *ICT Express* (Vol. 3, Issue 1, pp. 14–21). Korean Institute of Communications Information Sciences. https://doi.org/10.1016/j.icte.2017.03.004

"40.15US $ 11% OFF|Simcom 4g Lte Module | Module Sim 4g Simcom | Lte Module Sim | Simcom Lte Board - Sim7600e - Aliexpress." *Aliexpress.com*, id.aliexpress.com/item/4000591184815.html. Accessed 19 May 2023.

"REYAX RYLR896 Lora Module SX1276 UART 868MHz 915MHz Antenna at FCC NCC Command: Amazon.de: Electronics & Photo." *Www.amazon.de*, www.amazon.de/-/en/RYLR896-Module-SX1276-Antenna-Command/dp/B07NB3BK5H. Accessed 19 May 2023.

"SIM8262E-M2 SIMCom Original 5G Module, M.2 Form Factor, Qualcomm Snapdragon X62." *Www.waveshare.com*, www.waveshare.com/sim8262e-m2.htm. Accessed 19 May 2023.

"Simcom Sim7080g Modul Lpwa Baru Dan Asli Modul Cat-m Multi-Band Dan Nb-Iot - Buy Simcom,Sim7080g,Module Product on Alibaba.com." *Indonesian.alibaba.com*, indonesian.alibaba.com/product-detail/SIMCOM-SIM7080G-New-and-Original-LPWA-1600181233766.html.

"SIMCom SIM7080G CAT-M/NB-IoT - 12155 - S2-108HB-Z3037 |S2-108HB-Z30X8 |S2-108HB-Z312E - LPWA SMT - Techship." *Techship.com*, techship.com/products/simcom-sim7080g-cat-m-with-gnss/005/S2-108HB-Z312E/.

"SIMCom SIM8262E-M2 5G Sub6 - 11506 - S2-10ADV ES SAMPLE |S2-10C69-Z31DW |S2-10D26-Z31N8 - 5G NR M.2 - Techship." *Techship.com*, techship.com/products/simcom-sim8262e-m2-5g-sub6/.

"SimCom SIM900 GSM SMT - 10290 - S2-1040S-Z1K0K - GSM/GPRS Modules - Techship." *Techship.com*, techship.com/products/simcom-sim900-gsm-smt/.

"LoRA 868MHz Swivel Antenna, SMA-M - 10644 - JCG410-868MHZ - ISM Stub/Swivel Antennas - Techship." *Techship.com*, techship.com/products/lora-868mhz-swivel-antenna-sma-m/.

"Simcom Sim7600e 4g Module." *Indiamart.com*, www.indiamart.com/proddetail/simcom-sim7600e-4g-module-17977931973.html.

"SIM7080G Module LPWA Wireless Solutions | SIMCom Wireless Solutions Co.,Ltd." *Www.simcom.com*, www.simcom.com/product/SIM7080G.html.

Shurdi, O., Ruci, L., Biberaj, A., & Mesi, G. (2021). 5G Energy Efficiency Overview. *European Scientific Journal ESJ*, *17*(03). https://doi.org/10.19044/esj.2021.v17n3p315

"Train Network for Time Series Forecasting Using Deep Network Designer - MATLAB & Simulink." *Www.mathworks.com*, https://www.mathworks.com/help/deeplearning/ug/time-series-forecasting-using-deep-network-designer.html.

"5G Power: Creating a Green Grid That Slashes Costs, Emissions & Energy Use." *Huawei*, www.huawei.com/en/huaweitech/publication/89/5g-power-green-grid-slashes-costs-emissions-energy-use#:~:text=Energy%20consumption%20per%20unit%20of.

Mehedi, M. A. Al, Khosravi, M., Yazdan, M. M. S., & Shabanian, H. (2022). Exploring Temporal Dynamics of River Discharge Using Univariate Long Short-Term Memory (LSTM) Recurrent Neural Network at East Branch of Delaware River. *Hydrology*, *9*(11). https://doi.org/10.3390/hydrology9110202

Markidis, S. (2021). The Old and the New: Can Physics-Informed Deep-Learning Replace Traditional Linear Solvers? *Frontiers in Big Data*, *4*. https://doi.org/10.3389/fdata.2021.669097

Mehedi, M. A. Al, Khosravi, M., Yazdan, M. M. S., & Shabanian, H. (2022a). Exploring Temporal Dynamics of River Discharge Using Univariate Long Short-Term Memory (LSTM) Recurrent Neural Network at East Branch of Delaware River. *Hydrology*, *9*(11). https://doi.org/10.3390/hydrology9110202

Salman, S., & Liu, X. (2019). *Overfitting Mechanism and Avoidance in Deep Neural Networks*. http://arxiv.org/abs/1901.06566

APPENDIX

Generate Data Code (SIM7080G)

```
iterations=800;
x=[];
for i=1:iterations
x(i)=randi([567 1008]);
Results=num2cell(x);
end
```

Generate Data Code (SIM8262E)

```
iterations=800;
x=[];
for i=1:iterations
x(i)=randi([14076 17388]);
Results=num2cell(x);
end
```

Generate Data Code (SIM900)

```
iterations=800;
x=[];
for i=1:iterations
x(i)=randi([2880 5166]);
Results=num2cell(x);
end
```

Generate Data Code (LoRa)

```
iterations=800;
x=[];
for i=1:iterations
x(i)=randi([125 134]);
Results=num2cell(x);
end
```

Generate Data Code (SIM7600E)

```
iterations=800;
x=[];
for i=1:iterations
x(i)=randi([14076 17388]);
Results=num2cell(x);
end
```

Deep Learning Code (SIM7080G)

```
load nb_iot;
data = Results;
data = [data{:}];

figure
plot(data)
xlabel("Iterations")
ylabel("Power Consumption (mW)")
title("Power Consumption of NB-IoT")

numTimeStepsTrain = floor(0.6*numel(data));

dataValid = data(1:81);
dataTrain = data(81:numTimeStepsTrain+1);
dataTest = data(numTimeStepsTrain+1:end);

mu = mean(dataTrain);
sig = std(dataTrain);

dataTrainStandardized = (dataTrain - mu) / sig;
dataValidStandardized = (dataValid - mu)/ sig;

XTrain = dataTrainStandardized(1:end-1);
YTrain = dataTrainStandardized(2:end);
XValid = dataValidStandardized(1:end-1);
YValid = dataValidStandardized (2:end);

adsXTrain = arrayDatastore(XTrain);
adsYTrain = arrayDatastore(YTrain);
adsXValid = arrayDatastore(XValid);
adsYValid = arrayDatastore(YValid);


cdsTrain = combine(adsXTrain,adsYTrain);
cdsValid = combine(adsXValid, adsYValid);

dataTestStandardized = (dataTest - mu) / sig;

XTest = dataTestStandardized(1:end-1);
YTest = dataTest(2:end);

net = predictAndUpdateState(trainedNetwork_1,XTrain);

[net,YPred] = predictAndUpdateState(net,YTrain(end));

numTimeStepsTest = numel(XTest);
```

```
for i = 2:numTimeStepsTest
    [net,YPred(:,i)] = predictAndUpdateState(net,YPred(:,i-
1),'ExecutionEnvironment','cpu');
end

YPred = sig*YPred + mu;

rmse = sqrt(mean((YPred-YTest).^2))

figure
plot(dataTrain(1:end-1))
hold on
idx = numTimeStepsTrain:(numTimeStepsTrain+numTimeStepsTest);
plot(idx,[data(numTimeStepsTrain) YPred],'.-')
hold off
xlabel("Iterations")
ylabel("Power Consumption (mW)")
title("Forecast")
legend(["Observed" "Forecast"])

figure
subplot(2,1,1)
plot(YTest)
hold on
plot(YPred,'.-')
hold off
legend(["Observed" "Forecast"])
ylabel("Power Consumption (mW)")
title("Forecast")

subplot(2,1,2)
stem(YPred - YTest)
xlabel("Iterations")
ylabel("Error")
title("RMSE = " + rmse)
```

Deep Learning Code (SIM8282E)

```
load Data_5G.mat;
data = Results;
data = [data{:}];

figure
plot(data)
xlabel("Iterations")
ylabel("Power Consumption (mW)")
title("Power Consumption of SIM7600E")
```

```
numTimeStepsTrain = floor(0.6*numel(data));

dataValid = data(1:81);
dataTrain = data(81:numTimeStepsTrain+1);
dataTest = data(numTimeStepsTrain+1:end);

mu = mean(dataTrain);
sig = std(dataTrain);

dataTrainStandardized = (dataTrain - mu) / sig;
dataValidStandardized = (dataValid - mu)/ sig;

XTrain = dataTrainStandardized(1:end-1);
YTrain = dataTrainStandardized(2:end);
XValid = dataValidStandardized(1:end-1);
YValid = dataValidStandardized (2:end);

adsXTrain = arrayDatastore(XTrain);
adsYTrain = arrayDatastore(YTrain);
adsXValid = arrayDatastore(XValid);
adsYValid = arrayDatastore(YValid);


cdsTrain = combine(adsXTrain,adsYTrain);
cdsValid = combine(adsXValid, adsYValid);

dataTestStandardized = (dataTest - mu) / sig;

XTest = dataTestStandardized(1:end-1);
YTest = dataTest(2:end);

net = predictAndUpdateState(trainedNetwork_1,XTrain);

[net,YPred] = predictAndUpdateState(net,YTrain(end));

numTimeStepsTest = numel(XTest);
for i = 2:numTimeStepsTest
    [net,YPred(:,i)] = predictAndUpdateState(net,YPred(:,i-
1),'ExecutionEnvironment','cpu');
end

YPred = sig*YPred + mu;

rmse = sqrt(mean((YPred-YTest).^2))

figure
plot(dataTrain(1:end-1))
hold on
```

```
idx = numTimeStepsTrain:(numTimeStepsTrain+numTimeStepsTest);
plot(idx,[data(numTimeStepsTrain) YPred],'.-')
hold off
xlabel("Iterations")
ylabel("Power Consumption (mW)")
title("Forecast")
legend(["Observed" "Forecast"])

figure
subplot(2,1,1)
plot(YTest)
hold on
plot(YPred,'.-')
hold off
legend(["Observed" "Forecast"])
ylabel("Power Consumption (mW)")
title("Forecast")

subplot(2,1,2)
stem(YPred - YTest)
xlabel("Iterations")
ylabel("Error")
title("RMSE = " + rmse)
```

Deep Learning Code (SIM900)

```
load Data_SIM900;
data = Results;
data = [data{:}];

figure
plot(data)
xlabel("Iterations")
ylabel("Power Consumption (mW)")
title("Power Consumption of SIM900")

numTimeStepsTrain = floor(0.6*numel(data));

dataValid = data(1:81);
dataTrain = data(81:numTimeStepsTrain+1);
dataTest = data(numTimeStepsTrain+1:end);

mu = mean(dataTrain);
sig = std(dataTrain);

dataTrainStandardized = (dataTrain - mu) / sig;
dataValidStandardized = (dataValid - mu)/ sig;
```

```
XTrain = dataTrainStandardized(1:end-1);
YTrain = dataTrainStandardized(2:end);
XValid = dataValidStandardized(1:end-1);
YValid = dataValidStandardized (2:end);

adsXTrain = arrayDatastore(XTrain);
adsYTrain = arrayDatastore(YTrain);
adsXValid = arrayDatastore(XValid);
adsYValid = arrayDatastore(YValid);


cdsTrain = combine(adsXTrain,adsYTrain);
cdsValid = combine(adsXValid, adsYValid);

dataTestStandardized = (dataTest - mu) / sig;

XTest = dataTestStandardized(1:end-1);
YTest = dataTest(2:end);

net = predictAndUpdateState(trainedNetwork_1,XTrain);

[net,YPred] = predictAndUpdateState(net,YTrain(end));

numTimeStepsTest = numel(XTest);
for i = 2:numTimeStepsTest
    [net,YPred(:,i)] = predictAndUpdateState(net,YPred(:,i-
1),'ExecutionEnvironment','cpu');
end

YPred = sig*YPred + mu;

rmse = sqrt(mean((YPred-YTest).^2))

figure
plot(dataTrain(1:end-1))
hold on
idx = numTimeStepsTrain:(numTimeStepsTrain+numTimeStepsTest);
plot(idx,[data(numTimeStepsTrain) YPred],'.-')
hold off
xlabel("Iterations")
ylabel("Power Consumption (mW)")
title("Forecast")
legend(["Observed" "Forecast"])

figure
subplot(2,1,1)
plot(YTest)
hold on
```

```
plot(YPred,'.-')
hold off
legend(["Observed" "Forecast"])
ylabel("Power Consumption (mW)")
title("Forecast")

subplot(2,1,2)
stem(YPred - YTest)
xlabel("Iterations")
ylabel("Error")
title("RMSE = " + rmse)
```

Deep Learning Code (LoRa)

```
load LoRa_Data;
data = Results;
data = [data{:}];

figure
plot(data)
xlabel("Iterations")
ylabel("Power Consumption (mW)")
title("Power Consumption of LoRa")

numTimeStepsTrain = floor(0.6*numel(data));

dataValid = data(1:81);
dataTrain = data(81:numTimeStepsTrain+1);
dataTest = data(numTimeStepsTrain+1:end);

mu = mean(dataTrain);
sig = std(dataTrain);

dataTrainStandardized = (dataTrain - mu) / sig;
dataValidStandardized = (dataValid - mu)/ sig;

XTrain = dataTrainStandardized(1:end-1);
YTrain = dataTrainStandardized(2:end);
XValid = dataValidStandardized(1:end-1);
YValid = dataValidStandardized (2:end);

adsXTrain = arrayDatastore(XTrain);
adsYTrain = arrayDatastore(YTrain);
adsXValid = arrayDatastore(XValid);
adsYValid = arrayDatastore(YValid);


cdsTrain = combine(adsXTrain,adsYTrain);
```

```
cdsValid = combine(adsXValid, adsYValid);

dataTestStandardized = (dataTest - mu) / sig;

XTest = dataTestStandardized(1:end-1);
YTest = dataTest(2:end);

net = predictAndUpdateState(trainedNetwork_1,XTrain);

[net,YPred] = predictAndUpdateState(net,YTrain(end));

numTimeStepsTest = numel(XTest);
for i = 2:numTimeStepsTest
    [net,YPred(:,i)] = predictAndUpdateState(net,YPred(:,i-
1),'ExecutionEnvironment','cpu');
end

YPred = sig*YPred + mu;

rmse = sqrt(mean((YPred-YTest).^2))

figure
plot(dataTrain(1:end-1))
hold on
idx = numTimeStepsTrain:(numTimeStepsTrain+numTimeStepsTest);
plot(idx,[data(numTimeStepsTrain) YPred],'.-')
hold off
xlabel("Time (seconds)")
ylabel("Power Consumption (mW)")
title("Forecast")
legend(["Observed" "Forecast"])

figure
subplot(2,1,1)
plot(YTest)
hold on
plot(YPred,'.-')
hold off
legend(["Observed" "Forecast"])
ylabel("Power Consumption (mW)")
title("Forecast")

subplot(2,1,2)
stem(YPred - YTest)
xlabel("Iterations")
ylabel("Error")
title("RMSE = " + rmse)
```

Deep Learning Code (SIM7600E)

```
load SIM7600E_DATA;
data = Results;
data = [data{:}];

figure
plot(data)
xlabel("Iterations")
ylabel("Power Consumption (mW)")
title("Power Consumption of SIM7600E")

numTimeStepsTrain = floor(0.6*numel(data));

dataValid = data(1:81);
dataTrain = data(81:numTimeStepsTrain+1);
dataTest = data(numTimeStepsTrain+1:end);

mu = mean(dataTrain);
sig = std(dataTrain);

dataTrainStandardized = (dataTrain - mu) / sig;
dataValidStandardized = (dataValid - mu)/ sig;

XTrain = dataTrainStandardized(1:end-1);
YTrain = dataTrainStandardized(2:end);
XValid = dataValidStandardized(1:end-1);
YValid = dataValidStandardized (2:end);

adsXTrain = arrayDatastore(XTrain);
adsYTrain = arrayDatastore(YTrain);
adsXValid = arrayDatastore(XValid);
adsYValid = arrayDatastore(YValid);


cdsTrain = combine(adsXTrain,adsYTrain);
cdsValid = combine(adsXValid, adsYValid);

dataTestStandardized = (dataTest - mu) / sig;

XTest = dataTestStandardized(1:end-1);
YTest = dataTest(2:end);

net = predictAndUpdateState(trainedNetwork_1,XTrain);

[net,YPred] = predictAndUpdateState(net,YTrain(end));

numTimeStepsTest = numel(XTest);
```

```
for i = 2:numTimeStepsTest
    [net,YPred(:,i)] = predictAndUpdateState(net,YPred(:,i-
1),'ExecutionEnvironment','cpu');
end

YPred = sig*YPred + mu;

rmse = sqrt(mean((YPred-YTest).^2))

figure
plot(dataTrain(1:end-1))
hold on
idx = numTimeStepsTrain:(numTimeStepsTrain+numTimeStepsTest);
plot(idx,[data(numTimeStepsTrain) YPred],'.-')
hold off
xlabel("Iterations")
ylabel("Power Consumption (mW)")
title("Forecast")
legend(["Observed" "Forecast"])

figure
subplot(2,1,1)
plot(YTest)
hold on
plot(YPred,'.-')
hold off
legend(["Observed" "Forecast"])
ylabel("Power Consumption (mW)")
title("Forecast")

subplot(2,1,2)
stem(YPred - YTest)
xlabel("Iterations")
ylabel("Error")
title("RMSE = " + rmse)
```

MATLAB Code to Calculate Energy Consumption (SIM7080G, Series Upload)

```
function y = fcn(f)

y = ((f/0.01875)*(0.77753/600));
```

MATLAB Code to Calculate Energy Consumption (SIM8262E, Series Upload)

```
function y  = fcn(g)

y = ((g/25)*(15.708/600));
```

MATLAB Code to Calculate Energy Consumption (SIM900, Series Upload)

```
function y = fcn(h)

y = ((h/0.005375)*(4.0559/600));
```

MATLAB Code to Calculate Energy Consumption (LoRa, Series Upload)

```
function y = fcn(i)

y = ((i/0.00625)*(0.12969/600));
```

MATLAB Code to Calculate Energy Consumption (SIM7600E, Series Upload)

```
function y = fcn(j)

y = ((j/0.625)*(5.2279/600));
```

MATLAB Code to Calculate Energy Consumption (SIM7080G, Parallel Upload)

```
function a = fcn(f)

a = (((f/5)/0.01875)*(0.77753/600));
```

MATLAB Code to Calculate Energy Consumption (SIM8262E, Parallel Upload)

```
function y  = fcn(g)

y = (((g/5)/25)*(15.708/600));
```

MATLAB Code to Calculate Energy Consumption (SIM900, Parallel Upload)

```
function y = fcn(h)

y = (((h/5)/0.005375)*(4.0559/600));
```

MATLAB Code to Calculate Energy Consumption (LoRa, Parallel Upload)

```
function y = fcn(i)

y = (((i/5)/0.00625)*(0.12969/600));
```

MATLAB Code to Calculate Energy Consumption (SIM7600E, Parallel Upload)

```
function y = fcn(j)

y = (((j/5)/0.625)*(5.2279/600));
```

MATLAB Code to Assign Packet Size Values (General System)

```
function [a, b]= fcn(u)

a = u;
b = u;
```

MATLAB Code to Assign Packet Size Values (Series and Parallel Upload Subsystems)

```
function [a, b, c, d, e]= fcn(u)

a = u;
b = u;
c = u;
d = u;
e = u;
```

CURRICULUM VITAE

NIKLAS PARLINDUNGAN SIMORANGKIR

Bogor, Indonesia | +62 811 8826 563
niklassimorangkir@gmail.com | LinkedIn: Niklas Parlindungan Simorangkir |
Indonesian

EDUCATION

Swiss German University **2019 - present**
Tangerang, Indonesia
Sarjana Teknik in Industrial Engineering

- Expected to graduate in 2023

Double Degree Program:
South Westphalia University of Applied Sciences
Soest, Germany
Bachelor of Engineering in Industrial Engineering
Achievements:

- Non-Academic Achievement Scholarship Holder 2019 – present
- Student Ambassador 2019 – present
- 1st Place in Internal Chess Tournament 2 2023
- 2nd Place in Internal Chess Tournament 1 2023
- 1st Place in Ride or Run Competition (Running) 2023
- A Winner of the LLDIKTI 4 Representatives at the National University Debating Championship 2022
- Head of the Education Division of the Industrial Engineering Student Organization 2021
- Best Grade Scholarship (Semesters 4 and 5) 2021
- President of the SGUMUN Association 2019 – 2020
- President of the Photography Club 2020
- 2nd Place in Board Game Information Literacy Competition 2020

WORK EXPERIENCE

Swiss German University **Sept. 2022 – Feb. 2023**
Tangerang, Indonesia
Teaching Assistant
Key Responsibilities:

- Conducted six laboratory sessions on industrial automation
- Participated in a project on heijunka
- Improved marketing capabilities of the industrial engineering department
- Aided lecturers in preparing materials for lecturing sessions

Andros Deutschland GmbH **March – Aug. 2022**
Breuberg, Germany
Logistics Intern
Key Responsibilities:

- Conducted a project on optimizing the forklift system
- Created disposition plans
- Processed and organized customer information
- Sorted picking orders daily
- Conducted inventory management
- Monitored and updated monthly figures daily
- Sorted and scanned delivery orders on a weekly or monthly basis
- Translated an addendum from English to German

PT. Pharma Health Care **Feb. – April 2023**
Cikarang, Indonesia
Engineering Intern
Key Responsibilities

- OJT daily activity of utility operators and technicians
- Learned about utility systems such as HVAC, water system, and compressed air
- Conducted study of optimizing secondary packaging process to aim high capacity/output

TRAINING AND VOLUNTEER ACTIVITIES

Trainee at ATMI Cikarang **2021 – 2022**

- A six-week practical training programme between SGU and ATMI Cikarang
- Trained in mechanical benchwork, manual milling, manual turning/lathe, welding, grinding, assembly and reverse engineering, general maintenance

- Conducted final project in assembling a miniature windmill

Volunteer at Empower and Care Organization (EACO) Uganda **2020**

- Developed an overall fundraising strategy for EACO's water and sanitation project
- Searched for fundraising and grant opportunities
- Provided support to the Project Coordinator on strategic thinking and how to look for donors, foundations, and humanitarian agencies

SKILLS

Languages	**Indonesian** Native, **English** Bilingual, **German** Professional **French** Basic
Technical	**Microsoft Office** Advanced, **Python** Intermediate **AutoCAD** Intermediate, **MATLAB/Simulink** Intermediate **IBM SPSS Statistics** Intermediate, **Adobe** Intermediate **Mastercam** Basic