LAMPIRAN

Lampiran 1 *Dataset* penelitian Shiddiq *et al.* (2020)

TIME	HGTS	TEMP	UWND	VWND
2019-01-01 00:00:00	-14.157875	266.4074	-1.571225	4.6386719
2019-01-01 03:00:00	-14.83394	266.3074	-1.107568	4.3676684
2019-01-01 06:00:00	-20.80616	266.0186	-0.892544	4.725769
2019-01-01 09:00:00	-22.69066	265.5988	-1.035361	4.720188
2019-01-01 12:00:00	48.31031	265.7725	-1.445967	5.4046021

TIME	WWND	RELH	CO	CO_2
2019-01-01 00:00:00	0.0007362744	80.4	6.5885×10^{-9}	8.366125×10^{-7}
2019-01-01 03:00:00	0.0006813671	78.61018	5.10875×10^{-9}	6.712675×10^{-6}
2019-01-01 06:00:00	0	81.80206	1.345575 × 10-9	2.96975×10^{-6}
2019-01-01 09:00:00	0.0007043652	79.6441	6.70675 × 10-9	1.54675×10^{-7}
2019-01-01 12:00:00	0.0007931982	80.4	6.71075 × 10-9	0

Lampiran 2 *Source Code* model prediksi dengan menggunakan LSTM

```
# Inisiasi Param
batch_size = 1
timestep = dim(x_train)[2]
feature = dim(x_train)[3]

units = c(128, 85, 64)
epochs = c(25,50)
learning_rates = c(0.0005,0.00075,0.001, 0.0025, 0.005)
decays = c(1e-4,1e-6,1e-8)

params <- NULL
rmse_train <- NULL
mse_train <-NULL
rsquare_train <- NULL
cor_pearson_train <- NULL
rmse_test <- NULL
mse_test <-NULL
rsquare_test <- NULL
cor_pearson_test <- NULL
c <- 1

# Tuning for Model
for(e in 1:length(epochs)){
  for(u in 1:length(units)){
    for(l in 1:length(learning_rates)){
      for(d in 1:length(decays)){

        model <- keras_model_sequential()

        model %>%
          layer_lstm(units=units[u],
                     input_shape = c(timestep, feature),
                     batch_size = batch_size,
                     recurrent_dropout = 0.1,
                     dropout = 0.2,
                     stateful= TRUE,
                     activation = "tanh",
                     return_sequences = TRUE) %>%
          layer_dense(units = 1, activation = "sigmoid")

        model %>% compile(
          loss = 'mean_squared_error',
          optimizer = optimizer_adam(lr= learning_rates[l], decay = decays[d]),
          metrics = c('mean_absolute_error')
        )

        summary(model)
```

Lampiran 3 *Source Code* model prediksi dengan menggunakan LSTM (Lanjutan)

```
    model %>% fit(x_train,
                  y_train,
                  epochs=epochs[e],
                  batch_size=batch_size,
                  verbose=1,
                  shuffle=FALSE)
    # Prediksi nilai
    preds_test = model %>% predict(x_test, batch_size=batch_size)
    preds_train = model %>% predict(x_train, batch_size=batch_size)
    test$preds <- preds_test
    train$preds <- preds_train
    # Balikin ke normal
    test$preds <- invert_norm(test$preds, min_co, max_co)
    train$preds <- invert_norm(train$preds, min_co, max_co)
    # Evaluasi
    rmse_te <- sqrt(mean((test$co_act - test$preds)^2))
    mse_te <- mean((test$co_act - test$preds)^2)
    rsquare_te <- rsq(test$co_act, test$preds)
    corr_pea_te <- cor(test$co_act, test$preds)
    rmse_tr <- sqrt(mean((train$co_act - train$preds)^2))
    mse_tr <- mean((train$co_act - train$preds)^2)
    rsquare_tr <- rsq(train$co_act, train$preds)
    corr_pea_tr <- cor(train$co_act, train$preds)
    # Masukin ke sebuah list
    params[c] <- paste('units :', units[u],
                       'epoch :', epochs[e],
                       'learning_rate :', learning_rates[l],
                       'decay :', decays[d], sep = '_')
    rmse_test[c] <- rmse_te
    mse_test[c] <- mse_te
    rsquare_test[c] <- rsquare_te
    cor_pearson_test[c] <- corr_pea_te
    rmse_train[c] <- rmse_tr
    mse_train[c] <- mse_tr
    rsquare_train[c] <- rsquare_tr
    cor_pearson_train[c] <- corr_pea_tr
    c <- c+1
    filename <- paste(units[u], epochs[e], learning_rates[l],
decays[d], "model_co2.h5", sep = '_')
    model %>% save_model_tf(filename)
    model %>% reset_states()
  }
 }
}


result <- data.frame(params, rmse_test, rmse_train, mse_test, mse_train,
                     rsquare_test, rsquare_train, cor_pearson_test,
                     cor_pearson_train
)
```

Lampiran 4 *Source Code* model prediksi dengan menggunakan *Facebook's Prophet*

```
model <- prophet(daily.seasonality = TRUE,
                 yearly.seasonality = FALSE,
                 changepoints = c('2019-02-01 00:00:00', '2019-03-01 00:00:00'))

# model <- add_seasonality()
model <- add_regressor(model, name="HGTS")
model <- add_regressor(model, name="TEMP")
model <- add_regressor(model, name="UWND")
model <- add_regressor(model, name="VWND")
model <- add_regressor(model, name="WWND")
model <- add_regressor(model, name="RELH")
model <- add_regressor(model, name="dir_uwnd")
model <- add_regressor(model, name="dir_vwnd")
model <- add_regressor(model, name="dir_wwnd")
model <- add_regressor(model, name="co_t1")
model <- add_regressor(model, name="co_t2")
model <- add_regressor(model, name="co_t3")
model <- add_regressor(model, name="co_t4")
model <- add_regressor(model, name="co_t5")
model <- add_regressor(model, name="co_t6")
model <- add_regressor(model, name="co_t7")
model <- add_regressor(model, name="co_t8")

model <- fit.prophet(model, data_target)

# Buat Dataframe Kedepan
future <- make_future_dataframe(model, periods = 874, freq = 10800)
future$ds <- df$time
future$HGTS <- df$HGTS
future$TEMP <- df$TEMP
future$UWND <- df$UWND
future$VWND <- df$VWND
future$WWND <- df$WWND
future$RELH <- df$RELH
future$dir_uwnd <- df$dir_uwnd
future$dir_vwnd <- df$dir_vwnd
future$dir_wwnd <- df$dir_wwnd
future$co_t1 <- df$co_t1
future$co_t2 <- df$co_t2
future$co_t3 <- df$co_t3
future$co_t4 <- df$co_t4
future$co_t5 <- df$co_t5
future$co_t6 <- df$co_t6
future$co_t7 <- df$co_t7
future$co_t8 <- df$co_t8

forecast <- predict(model, future)
```