Terakreditasi SINTA Peringkat 2
Surat Keputusan Dirjen Penguatan Riset dan Pengembangan Ristek Dikti No. 10/E/KPT/2019
masa berlaku mulai Vol. 1 No. 1 tahun 2017 s.d Vol. 5 No. 3 tahun 2021

Terbit online pada laman web jurnal: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 4 No. 3 (2020) 447 - 453 ISSN Media Elektronik: 2580-0760

Penerapan *Long Short Term Memory* untuk Memprediksi *Flight Delay* pada Penerbangan Komersial

Muhammad Genta Ari Shandi[1], Rifki Adhitama[2], Amalia Beladinna Arifa[3]
[1,3] Teknik Informatika, Fakultas Informatika, Institut Teknologi Telkom Purwokerto,
[2] Rekayasa Perangkat Lunak, Fakultas Informatika, Institut Teknologi Telkom Purwokerto
[1]16102125@ittelkom-pwt.ac.id, [2]rifki@ittelkom-pwt.ac.id, [3]amalia@ittelkom-pwt.ac.id

Abstract

Delay in airline services, become an unpleasant experience for passengers who experience it. This study aims to build a model that can predict flight delay (departure) using the Long Short Term Memory method and can find out its performance. In this study there are two scenarios that have different ways of preprocessing. Both of these scenarios produce predictions with error values calculated using Root Mean Squared Error (RMSE), respectively from the first to the second scenario namely: 41, 21. Between the two, the second scenario is better than the first scenario due to extreme data deletion (anomaly) in the second scenario with an error value using RMSE of 0.116.

Keywords: *Departure delay, Flight, Long short term memory, Model, Recurrent neural network.*

Abstrak

Keterlambatan pada layanan maskapai penerbangan, menjadi pengalaman yang tidak menyenangkan bagi penumpang yang mengalaminya. Penelitian ini bertujuan untuk membangun sebuah model yang dapat memprediksi *flight delay* (*departure*) dengan menggunakan metode *Long Short Term Memory* dan dapat mengetahui performansinya. Dalam penelitian ini terdapat dua skenario yang memiliki perbedaan cara *preprocessing*. Kedua skenario tersebut menghasilkan prediksi dengan nilai *error* yang dihitung menggunakan *Root Mean Squared Error* (RMSE), secara berturut-turut dari skenario pertama sampai kedua yaitu: 41, 21. Diantara keduanya, skenario kedua lebih baik dari skenario pertama dikarenakan adanya penghapusan data ekstrim (anomali) pada skenario kedua dengan nilai error menggunakan RMSE sebesar 0.116.

Kata kunci: *Departure delay, Flight, Long short term memory, Model, Recurrent neural network.*

1. Pendahuluan

Pesawat adalah salah satu kendaraan yang biasa digunakan untuk berpindah tempat melalui jalur udara. Terdapat banyak jenis pesawat yang beroperasi di dunia ini baik pesawat konvensional, pemerintah, ataupun pesawat untuk tujuan lain seperti pesawat tempur. Salah satu media transportasi ini tidak jauh dengan yang namanya delay atau penundaan baik itu saat keberangkatan ataupun saat tiba.

Delay merupakan keadaan di mana pesawat mengalami keterlambatan berangkat atau keterlambatan tiba. Keterlambatan Penerbangan adalah terjadinya perbedaan waktu antara waktu keberangkatan atau kedatangan yang dijadwalkan dengan realisasi waktu keberangkatan. atau kedatangan [1]. *Delay* sangat sering terjadi pada layanan maskapai penerbangan dengan berbagai sebab, antara lain faktor manajemen maskapai, teknis operasional, faktor cuaca, faktor bencana alam dan faktor lainnya.

New York merupakan salah satu kota sebagai contoh yang memiliki 3 dari 40 bandara tersibuk di Amerika Serikat, antara lain Newark Liberty International Airport, LaGuardia Airport, John F. Kennedy International Airport [2]. Pada 7 Agustus 2019, 426 penerbangan telah dibatalkan dari bandara Newark, LaGuardia dan Kennedy, dan 500 lainnya ditunda - hampir setengah dari total penerbangan terjadwal [3]. Lebih dari 1.000 penerbangan di bandara Newark, LaGuardia, dan JFK ditunda pada hari Senin karena

Diterima Redaksi : 01-03-2020 | Selesai Revisi : 12-05-2020 | Diterbitkan Online : 20-06-2020

badai musim dingin yang menumpahkan beberapa inci salju di seluruh wilayah. Ada juga lebih dari 300 penerbangan dibatalkan di bandara-bandara itu, meninggalkan sejumlah penumpang berjuang untuk menemukan cara lain untuk pulang ketika salju besar pertama musim ini terus turun hingga Senin malam [4].

Penelitian ini dilakukan karena banyaknya penumpang yang mengalami kerugian akibat *flight delay* (*departure*). Oleh sebab itu diperlukan sebuah model yang dapat memprediksi *flight delay* (*departure*) agar penumpang dapat mempersiapkan apabila terjadi *flight delay* (*departure*). Tidak menutup kemungkinan, dengan adanya model ini pula, setiap penyedia layanan transportasi udara atau maskapai dapat mempersiapkan segala kemungkinan yang akan terjadi dan mengurangi tingkat delay pesawat. Untuk itu, diperlukan model yang dibangun berdasarkan algoritma tertentu yang dapat melakukan klasifikasi dan memprediksi sesuai dengan fitur yang ada dalam dataset. Dalam hal ini, metode yang digunakan adalah *Recurrent Neural Network*. *Dataset* yang digunakan dalam penelitian ini diperoleh dari *TranStats by BTS Bureau of Transportation Statistics* dari tahun 2019 [5].

Deep Learning (DL) adalah teknik dalam NN yang menggunakan teknik tertentu seperti *Restricted Boltzmann Machine* (RBM) untuk mempercepat proses pembelajaran dalam NN yang menggunakan lapis yang banyak atau lebih dari 7 lapis[6]. Beberapa jenis DL antara lain *Deep Auto Encoder*, *Deep Belief Nets*, *Convolutional NN*, *Recurrent Neural Network*, dan lain lain[6]. Salah satu teknik *recurrent neural network* (RNN) yang digunakan dalam penelitian ini adalah *Long Short Term Memory* (LSTM). Model ini lebih baik dibandingkan dengan model RNN sederhana[7]. LSTM merupakan salah satu varian dari *model recurrent neural network*, yang memiilki isi cell lebih kompleks[8]. Jaringan LSTM, yang dikembangkan [9] pada tahun 1997, adalah struktur jaringan saraf tiruan yang tidak seperti RNN yang tidak memiliki masalah vanishing gradient. Algoritma ini bekerja ketika ada penundaan yang lama, dan dapat menangani sinyal yang memiliki campuran komponen frekuensi rendah dan tinggi. LSTM RNN lebih bagus dibandingkan metode lain dalam berbagai aplikasi seperti halnya belajar bahasa dan *connected handwriting recognition*.

Penelitian yang dilakukan oleh Zheng et al. [10] memilki hasil yang menunjukkan bahwa metode prakira berbasis *Long Short Term Memory* (LSTM) dapat mengungguli metode prakira tradisional dalam masalah prakira beban listrik. Hasil *Root Mean Square Error* (RMSE) dan *Mean Absolute Percentage Error* (MAPE) metode LSTM adalah yang terendah jika dibandingkan dengan metode lain (0.0702 & 0.0535).

Recurrent Neural Network (RNN) merupakan jenis dari *deep learning* yang pemrosesannya dipanggil berulang-ulang untuk memproses input berupa data sekuensial. *Recurrent Neural Network* biasa digunakan pada *Natural Language Processing* yaitu bagian dari *Artificial Intelligence* yang berhubungan dengan interaksi antara komputer dan manusia menggunakan bahasa alami manusia. Inti dari NLP adalah membaca, menguraikan, dan memahami bahasa manusia dengan cara yang dapat dinilai. NLP banyak diterapkan di berbagai bidang, antara lain *language translator* (google translate), *Word Processors* (Grammarly & Microsoft Word), *Interactive Voice Response* (IVR), *Personal assistant applications* (OK Google, Siri, Cortana).

Long Short-Term Memory (LSTM) adalah tipe khusus RNN yang berisi sistem gating units yang mengontrol aliran informasi [11]. Dari pengertian sebelumnya, dapat diketahui yang membedakan LSTM dengan RNN adalah *gate function* yang ada pada LSTM dan hal tersebut meningkatkan kemampuan LSTM secara signifikan dibandingkan dengan RNN. Dan alasannya mungkin fakta bahwa gate function yang digunakan dalam LSTM dapat memungkinkannya untuk menangkap ketergantungan jangka panjang lebih baik daripada RNN [12].

Digunakannya algoritma LSTM dalam penelitian ini karena LSTM dapat digunakan untuk prediksi tme series data dan juga dalam pemrosesannya digunakan *memory* untuk menyimpan data terakhir serta terdapat *gate function* sehingga proses prediksi berdasarkan fitur yang ada dan beberapa data terakhir yang lebih jauh.

Berdasarkan latar belakang di atas, maka rumusan masalah pada penelitian ini, adalah seberapa tinggi performansi algoritma *Long Short Term Memory* jika digunakan sebagai model untuk memprediksi *flight delay.*

Maka dapat diketahui tujuan dari penelitian ini adalah membangun sebuah model yang dapat memprediksi *flight delay* (*departure*) dengan menggunakan metode *Long Short Term Memory* dan dapat mengetahui performansinya.

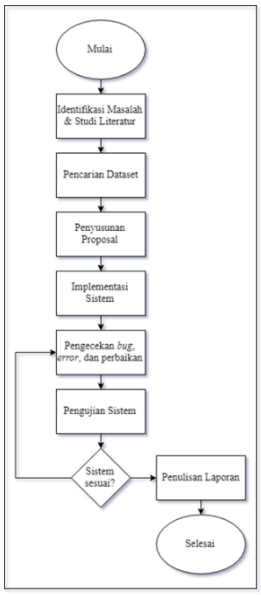
2. Metode Penelitian

Gambar 1 merupakan metode penelitian yang bertujuan untuk menguraikan seluruh kegiatan yang akan dilakukan selama penelitian berlangsung dan digunakan sebagai pedoman dalam pelaksanaan penelitian agar hasil dicapai tidak menyimpang dari tujuan.

2.1. Identifikasi Masalah dan Studi Literatur

Pada tahap awal ini topik diambil tentang masalah pada bidang penerbangan. Terdapat berbagai masalah pada bidang penerbangan, antara lain kecelakaan pesawat, kerusakan pesawat, pelayanan yang kurang baik, dan flight delay hingga terjadi pembatalan penerbangan. Di antara beberapa kasus tersebut, ada kasus yang sering terjadi yaitu *flight delay*. Permasalahan ini diangkat

karena flight delay sering terjadi namun belum banyak penelitian yang mengangkat tentang prediksi *flight delay*.



Gambar 1. Tahapan Penelitian

2.2. Pengumpulan Dataset

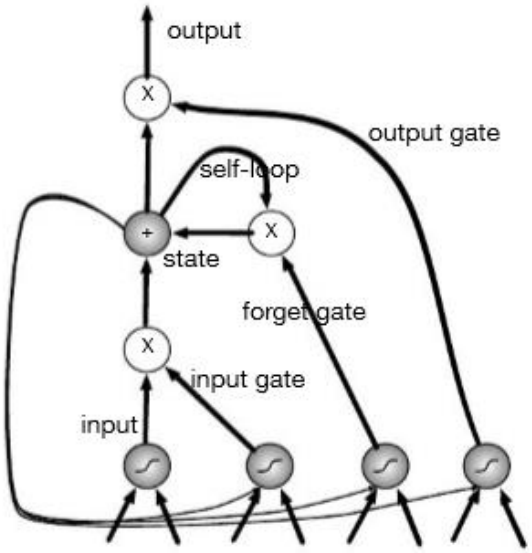
Pada penelitian ini, data dikumpulkan dengan observasi dataset terbuka di berbagai website yang membahas tentang penerbangan. Salah satu website yang menyediakan dataset terbuka dan yang digunakan di penelitian ini berasal dari *Transtat by BTS*.

2.3. Pembangunan Model

Dalam pembangunan model menerapkan tahapan-tahapan yang perlu dilakukan pada teknik deep learning dengan algoritma *Long Short Term Memory*. Langkah awal yang dilakukan dengan memasukan dan mengolah dataset yang akan digunakan, kemudian training data pada hidden layer dan dari hasil training data akan didapatkan output berupa data *flight delay prediction.*

Ide dasar mengondisikan *the forgetting* atau melupakan sesuai dengan konteks merupakan inti dari algoritma *Long Short Term Memory*. Tujuan utamanya adalah untuk selalu menerapkan *linear self-loop* yang mana *gradient* dapat berjalan di durasi yang lama.

Blok diagram "sel" *LSTM Recurrent Networks* pada Gambar 2. 3. Sel terhubung secara berulang-ulang satu sama lain, menggantikan hidden units yang biasa dari *reccurent networks*. Fitur input dihitung dengan regular artificial neuron unit, dan nilainya dapat diakumulasikan ke dalam keadaan jika gerbang input sigmoidal memungkinkannya. *State* unit memiliki *linear self-loop* dengan *weight* dikendalikan oleh *forget gate*. Output sel dapat ditutup oleh *output gate*. Semua *gating units* memiliki *sigmoid non-linearity*, sedangkan input unit dapat memiliki *squashing non-linearity*. *State* unit juga dapat digunakan sebagai input tambahan di *gating units*.



Gambar 2. Blok Diagram LSTM

Dari pada menerapkan unit dengan *squashing function* di inputs dan recurrent units, LSTM networks memiliki LSTM cell. Setiap cell memiliki input dan output yang sama dengan *vanilla recurrent network*, namun memiliki parameter dan gating units system yang mengontrol aliran dari informasi. Komponen paling penting dari LSTM adalah state units yang memiliki *linear self-loop* dimana *self-loop weight* dikontrol oleh *forget gate unit.*

2.4. Pengecekan *bug*, *error*, dan perbaikan

Pengecekan *bug* dan *error* dilakukan dengan menggunakan fitur *compiling* pada *Integrated Development Environment (IDE) jupyter notebook* sehingga bila terjadi *error* akan mudah terdeteksi. Setelah *error* dapat dideteksi maka akan dilakukan perbaikan dengan cara melakukan analisa pada *error* yang terjadi guna model dapat dijalankan dengan baik.

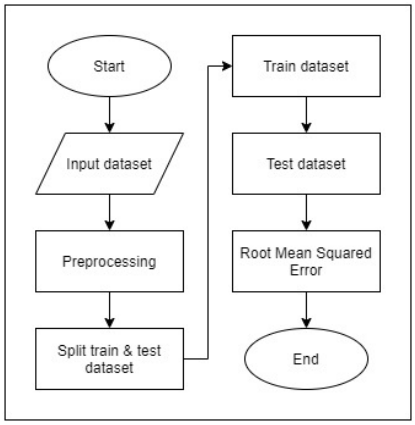
2.5. Pengujian Model

Pengujian model dilakukan dengan cara *testing* data agar memperoleh hasil prediksi *flight delay*. Dengan begitu, peneliti dapat mengetahui bahwa model yang dibuat dapat berjalan dan sesuai dengan tujuan awal.

2.6. Tahapan Implementasi

Gambar 2 merupakan tahapan implementasi yang bertujuan untuk menguraikan seluruh tahap yang perlu dilakukan dalam pembuatan dan pengimplementasian model yang akan dibuat dalam penelitian ini dan digunakan sebagai pedoman dalam pembuatan model penelitian agar hasil yang dicapai tidak menyimpang dari tujuan.

Input dataset merupakan salah satu aspek paling penting pada penelitian ini karena peran dataset sebagai tolak ukur untuk melakukan tahapan awal. Dalam penelitian ini, dataset yang digunakan berasal *TranStats by BTS Bureau of Transportation*. Dalam penelitian ini, terdapat 3 skenario yang akan dilakukan. Skenario pertama, menggunakan dataset satu tahun yang sudah di-*preprocessing* tanpa menghapus data anomali. Untuk skenario kedua, menggunakan dataset dua tahun yang sudah di *preprocessing* dengan data anomali yang belum dihapus, sedangkan skenario ketiga, menggunakan dataset dua tahun yang sudah di *preprocessing* dan sudah dihapus. Data anomali dalam penelitian ini merupakan atribut *departure delay* yang memiliki *value* yang terlalu ekstrim atau terlalu jauh selisihnya dengan rata-rata *value* dari atribut *departure delay*.



Gambar 3. Tahap Implementasi

Setelah dilakukan *preprocessing data*, pada tahap ini dataset dibagi menjadi 2, yaitu *training data* & *testing data*. Untuk *training data* akan menggunakan dataset dari bulan januari 2017 sampai bulan september 2019, sedangkan untuk *testing data* akan menggunakan dataset bulan terakhir yaitu bulan november 2019. Data yang sudah dibagi menjadi *training* & *testing* selanjutnya dipilih atribut yang akan digunakan pada model dalam penelitian ini serta disimpan masing-masing menjadi *clean_training* & *clean_testing*. Atribut dataset yang dipilih untuk model dalam penelitian ini, yaitu: *CRS_ELAPSED_TIME, DAY_TYPE, IS_HOLIDAY, DEP_FROM_NY, T_1, T_2, T_3, T_4, T_5, DEP_DELAY*. Berikut keterangan dari masing-masing atribut yang digunakan:

CRS_ELAPSED_TIME, waktu lepas landas-mendarat sesuai jadwal.
ORIGIN_AIRPORT_ID, ID dari bandara keberangkatan/asal.
ORIGIN_CITY_NAME, Kota keberangkatan/asal.
DEST_AIRPORT_ID, ID dari bandara pendaratan/tujuan.
DEST_CITY_NAME, Kota pendaratan/tujuan.
FLIGHTS, Jumlah penerbangan.
DISTANCE, Jarak dari bandara asal ke tujuan.
DAY_TYPE, tipe hari, weekday atau weekend.
IS_HOLIDAY, hari libur nasional atau tidak.
DEP_FROM_NY, keberangkatan dari New York atau tidak.
T_1, dep delay satu penerbangan sebelumnya.
T_2, dep delay dua penerbangan sebelumnya.
T_3, dep delay tiga penerbangan sebelumnya.
T_4, dep delay empat penerbangan sebelumnya.
T_5, dep delay lima penerbangan sebelumnya.
DEP_DELAY, waktu keterlambatan lepas landas.

Berikut merupakan *training data* yang digunakan pada penelitian ini yang digambarkan pada Tabel 1 dan Tabel 2 dengan urutan atribut yang dijelaskan di atas, antara lain:

Tabel 1. *Traning Data* Atribut ke 1-7

No	1	2	3	4	5	6	7
0	316	12892	57	12478	70	1.0	2475
1	181	12265	74	11697	36	1.0	1176
2	210	14025	85	11697	36	1.0	1334
…	…	…	…	…	…	…	…
391101	317	13891	79	12478	70	1.0	2429
391102	226	11292	31	12478	70	1.0	1626

Tabel 2. *Training Data* Atribut ke 8-16

No	8	9	10	11	12	13	14	15	16
0	0	0	0	-2	4	2	16	29	15
1	0	0	1	15	-2	4	2	16	-11
2	0	0	1	-11	15	-2	4	2	67
…	…	…	…	…	…	…	…	…	…
391101	0	0	0	-3	-2	0	7	0	-9
391012	0	0	0	-9	-3	-2	0	7	-3

Kemudian berikut merupakan *testing data* yang digunakan pada penelitian ini yang digambarkan pada Tabel 3 dan Tabel 4 dengan atribut seperti *training data*, antara lain:

Tabel 3. *Testing Data* Atribut ke 1-7

No	1	2	3	4	5	6	7
391103	331	12892	57	12478	70	1.0	2475
391104	333	12892	57	12478	70	1.0	2475
391105	236	14254	86	12478	70	1.0	1617
…	…	…	…	…	…	…	…
411003	234	10140	3	12478	70	1.0	1826
411004	216	11292	31	12478	70	1.0	1626

Tabel 4. *Testing Data* Atribut ke 8-16

No	8	9	10	11	12	13	14	15	16
391103	0	0	0	71	93	-10	-9	-1	32
391104	0	0	0	32	71	93	-10	-9	-4
391105	0	0	0	-4	32	71	93	-10	-18
…	…	…	…	…	…	…	…	…	…
411003	0	0	0	136	65	28	169	198	78
411004	0	0	0	78	136	65	28	169	97

Tahap selanjutnya dilakukan *training dataset* menggunakan LSTM dengan 20 *neurons*, 20 *epoch*. Jumlah epoch ditentukan setelah melakukan berbagai percobaan sehingga mendapatkan jumlah *epoch* yang efisien dalam menjalankan model tersebut. Artinya dari setiap satu *epoch* data dibagi menjadi *batch* berukuran

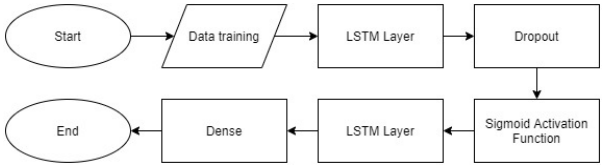
72 dan untuk menyelesaikan satu *epoch* tersebut dilakukan iterasi sejumlah dataset dibagi batch size.

Setelah *training dataset* menggunakan model LSTM, tahap selanjutnya adalah melakukan *testing dataset* untuk menguji hasil dari model yang sudah dilatih (*train*) sebelumnya.

3. Hasil dan Pembahasan

3.1. Penerapan

Setelah memastikan dataset sudah di-*preprocessing* dan dataset telah siap diproses, tahap selanjutnya yang dilakukan adalah mendeklarasikan metode yang digunakan dalam penelitian ini yaitu *Long-Short Term Memory* (LSTM). *Long-Short Term Memory* (LSTM) yang merupakan jaringan saraf (RNN). Berikut merupakan alur dari penerapan model yang dilakukan.



Gambar 4. Alur Penerapan Model

Seperti pada Gambar 3, konfigurasi model yang digunakan dalam LSTM antara lain, LSTM layer, *Dropout layer*, *Sigmoid Activation Function*, LSTM layer, dan Dense.

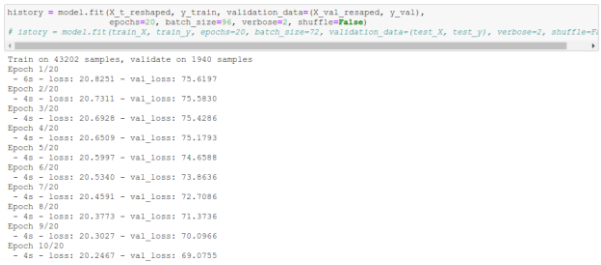
LSTM *layer*, terdapat beberapa parameter yang menjadi susunan dari LSTM *layer*. Dalam penelitian ini, LSTM *layer* menggunakan 200 *units*/*hidden layer*, dengan *input shape* = (1, 15) yang artinya ukuran atribut dari atribut yang akan digunakan dalam proses *training* adalah 15 kolom (*input*) dan 1 kolom sebagai target yang akan diprediksi (*output*).

Pada *dropout layer*, memilih secara acak *neuron* yang akan di *dropped out* atau diabaikan dalam proses *training.* Hal ini mengakibatkan neuron lain harus masuk dan menangani represenasi yang diperlukan untuk membuat prediksi untuk neuron yang sebelumnya hilang.

Sigmoid Activation Function, LSTM memiliki arsitektur khusus yang memungkinknnya untuk melupakan (*forget*) informasi yang tidak perlu. *Sigmoid activation function layer* mengambiil input dan dalam memutuskan bagian mana dari *output* lama yang harus dihapus (dengan cara mengeluarkan *output* 0).

Dense Layer, setiap *neuron* menerima input dari semua *neuron* di lapisan sebelumnya, sehingga terhubung dan menghasilkan *output* dalam satu *neuron*.

Kemudian, *training dataset* menggunakan LSTM dengan 20 *epoch*. Artinya dari setiap satu *epoch* data dibagi menjadi *batch* berukuran 96 dan untuk menyelesaikan satu *epoch* tersebut dilakukan iterasi sejumlah dataset dibagi *batch size* yang digambarkan dalam Gambar 5.



```
history = model.fit(X_t_reshaped, y_train, validation_data=(X_val_resaped, y_val),
                    epochs=20, batch_size=96, verbose=2, shuffle=False)
# istory = model.fit(train_X, train_y, epochs=20, batch_size=72, validation_data=(test_X, test_y), verbose=2, shuffle=F
```

```
Train on 43202 samples, validate on 1940 samples
Epoch 1/20
 - 6s - loss: 20.8251 - val_loss: 75.6197
Epoch 2/20
 - 4s - loss: 20.7311 - val_loss: 75.5830
Epoch 3/20
 - 4s - loss: 20.6928 - val_loss: 75.4286
Epoch 4/20
 - 4s - loss: 20.6509 - val_loss: 75.1793
Epoch 5/20
 - 4s - loss: 20.5997 - val_loss: 74.6588
Epoch 6/20
 - 4s - loss: 20.5340 - val_loss: 73.8636
Epoch 7/20
 - 4s - loss: 20.4591 - val_loss: 72.7086
Epoch 8/20
 - 4s - loss: 20.3773 - val_loss: 71.3736
Epoch 9/20
 - 4s - loss: 20.3027 - val_loss: 70.0966
Epoch 10/20
 - 4s - loss: 20.2467 - val_loss: 69.0755
```

Gambar 5. Tahap Training Data

Mean Absolute Error (MAE) merupakan salah satu cara menghitung nilai error pada suatu model regresi. Tahapan menghitung MAE adalah pertama menghitung selisih nilai data hasil prediksi dengan data aktual dan hasilnya dikuadratkan (dipangkatkan dengan 2), kemudian lakukan hal ini sebanyak baris data yang ada pada dataset dan jumlahkan hasil semuanya, setelah itu hasil dari tahap sebelumnya dibagi dengan jumlah baris data (n) yang ada pada dataset.

3.2. Pengujian

Setelah training dataset menggunakan model LSTM, tahap selanjutnya adalah melakukan testing dataset dengan memprediksi target yang sudah ditentukan dalam hal ini departure delay berdasarkan model yang sudah dilatih (*train*) sebelumnya.

Kemudian hasil prediksi akan dibandingkan dengan data aktual yang ada dan dihitung nilai *loss*-nya menggunakan *Root Mean Squared Error* (RMSE).

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}} \quad (1)$$

Keterangan:

$\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n$:P*redicted values*, nilai yang diprediksi

$y_1, y_2, \ldots, y_n$:O*bserved values*, nilai observasi atau asli

n :The *number of observations*, nomor observasi ke-n

Berdasarkan persamaan (1) di atas, tahap yang dilakukan pada RMSE yaitu pertama menghitung selisih dari setiap data yang diprediksi dengan data aktual untuk setiap baris data dalam dataset dan dihitung totalnya. Kemudian hasilnya dipangkatkan dengan dua dan dibagi dengan jumlah baris data. Terakhir hasil tahap sebelumnya diakar pangkat dua sehingga hasil perhitungan ini dapat diartikan dengan rata-rata dari nilai error (selisih antara data yang diprediksi dengan data actual) keseluruhan baris data sebanyak n baris dalam dataset.

3.3. Analisis Hasil

Model yang dibuat menghasilkan nilai RMSE yang berbeda-beda targantung dengan jenis *dataset* yang berbeda dalam setiap skenario. Dalam perbedaan antar skenario itu beberapa perubahan menunjukan perbedaan yang signifikan terhadap nilai RMSE. Diurutkan dari yang memiliki nilai RMSE paling kecil adalah skenario ketiga yang bernilai 21, dinormalisasi menjadi 0.116, diikuti skenario kedua dengan nilai 39, sedangkan untuk urutan terakhir dengan nilai yang paling besar yaitu skenario pertama. Berikut dijelaskan perbedaan yang lebih rinci antar skenario dalam bentuk tabel 5.

Tabel 5. Perbandingan skenario

Skenario	Jumlah data	*Preprocessing* 1-5	6	*Training time*	RMSE
Kedua	440822	Ya	Tidak	± 10 menit	41
Ketiga	429917		Ya	± 10 menit	21

Keterangan tabel:

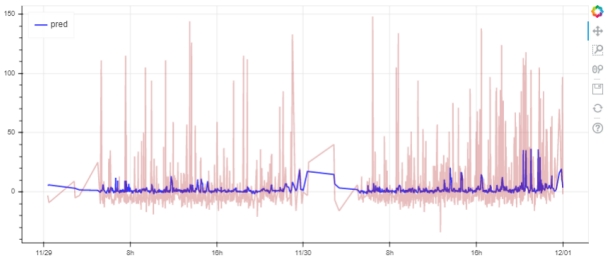
Preprocessing 1 : Mengubah tipe data atribut yang berupa tanggal dan waktu dari *string* menjadi *datetime.*
Preprocessing 2 : Memperbaiki *wrong value.*
Preprocessing 3 : Normalisasi *city name.*
Preprocessing 4 : Menambahkan beberapa atribut yang berkaitan dengan *dataset.*
Preprocessing 5 : Menambahkan *record* dari 5 *departure delay* terakhir.
Preprocessing 6 : Menghapus data ekstrim atau *value* dari atribut *departure delay* yang selisihnya jauh dari nilai rata-rata.
Proses *training* menggunakan model yang sama.

Dapat dilihat pada Tabel 1, skenario pertama dengan dataset yang berjumlah 440822 baris yang di-*preprocessing*, namun tanpa menghapus data ekstrim pada atribut target atau departure delay. Skenario pertama merupakan percobaan pertama pada penelitian ini, dimulai dengan menggunakan dataset dengan jangka satu tahun terakhir. Kemudian data tersebut di preprocessing karena masih banyak kejanggalan pada dataset mentah yang didapat dari *Beurau by TransStat*. *Preprocessing* pertama kali yang dilakukan adalah mengubah atribut yang berupa keterangan waktu/tanggal seperti atribut FL_DATE, CRS_DEP_TIME, DEP_TIME, DEP_DELAY, CRS_ARR_TIME, ARR_TIME, dan ARR_DELAY. Tahap kedua preprocessing yaitu mengubah wrong value pada atribut tertentu seperti DEP_TIME (waktu keberangkatan aktual) yang tidak sesuai dengan CRS_DEP_TIME (waktu keberangkatan yang terjadwal) dikurangi DEP_DELAY, karena pada dasarnya DEP_DELAY merupakan selisih dari CRS_DEP_TIME dan DEP_TIME. Tahap ketiga menormalisasi city name dari bentuk string ke int dengan menggunakan *Label Encoder*. Tahap keempat menambahkan beberapa atribut yang sesuai dengan dataset seperti atribut *weekend*/*weekday*, atribut hari libur nasional, dan atribut keberangkatan dari New York atau pendaratan ke New York. Untuk tahap *preprocessing* yang terakhir pada skenario ini adalah menambahkan 5 record terakhir dari atribut departure delay, menjadi atribut T_1 untuk *record* terakhir dari departure delay, T_2 untuk *record* terakhir kedua dari *departure delay*, dan seterusnya sampai 5 *record* terakhir. Setelah dataset sudah preprocessing, dilakukan penerapan model menggunakan dataset tersebut. Konfigurasi layer yang digunakan pada model ini adalah LSTM layer, *Dropout layer*, *Sigmoid Activation Function*, LSTM layer, dan Dense. Setelah model dilatih, dilakukan proses pengujian dengan memprediksi test_data. Hasil prediksi akan dibandingkan dengan hasil asli dari test_data. Perbandingan tersebut menghasilkan RMSE dengan nilai 41.

Skenario kedua dilakukan percobaan menggunakan dataset yang sama dengan skenario pertama. Perbedaan dalam skenario ini adalah adanya tambahan dalam tahap preprocessing. Yaitu menghapus data ekstrim atau value yang selisihnya terlalu jauh dengan rata-rata pada atribut *departure delay*. Model dilatih menggunakan dataset tersebut dan diuji. Proses pengujian menghasilkan RMSE dengan nilai 21.

Dapat dilihat dari dua skenario yang dijelaskan di atas, kebersihan dari dataset sangat berpengaruh dalam model LSTM yang digunakan dalam penelitian ini. Selain tingkat kebersihan data, ukuran dataset juga memiliki pengaruh yang signifikan dalam menurunkan nilai error dari RMSE.

Kemudian tahap setelah diketahui nilai error menggunakan RMSE adalah menampilkan grafik perbandingan setiap baris data yang diprediksi yang berwarna biru dengan setiap baris data aktual yang berwarna merah pada dataset. Hal tersebut dapat dilakukan menggunakan modul bokeh pada python seperti yang ada pada Gambar 6.



Gambar 6. Grafik perbandingan hasil prediksi dengan data aktual

4. Kesimpulan

Pada dataset yang digunakan dalam penelitian ini, terdapat waktu keberangkatan dengan hari yang sama dan jam yang sama, sehingga diperlukan proses *drop* guna memperbaiki nilai *loss* dari model.

Berdasarkan penelitian mengenai prediksi *departure*

delay menggunakan model LSTM, dapat ditarik kesimpulan bahwa performa *Long Short Term Memory* cukup baik sebagai model prediksi *flight delay*, dibuktikan dengan nilai *error* menggunakan RMSE sebesar 0.116, yang dapat diindikasikan baik bila dibandingkan dari penelitian yang dilakukan oleh Zheng et al. [10],. Performa tersebut dapat dicapai dengan epoch sebanyak 20 dan tahap *preprocessing drop* pada data yang memiliki waktu keberangkatan dengan hari dan jam yang sama serta menghapus nilai anomali pada dataset. Dengan menggunakan modul *bokeh* dalam *python* hasil prediksi dapat divisualisasikan dalam bentuk grafik garis.

Setelah penelitian mengenai prediksi *departure delay* menggunakan model LSTM dilakukan, maka saran untuk penelitian selanjutnya adalah proses pengecekan validitas data harus dilakukan dengan teliti guna menghindari data yang tidak valid & memaksimalkan *split* dataset dengan menggunakan *time series split* yang dapat membantu meningkatkan performansi model.

Daftar Rujukan

[1] Kemenhub, *Penanganan Keterlambatan Penerbangan (Delay Management) Pada Badan Usaha Angkutan Udara Niaga Berjadwal Di Indonesia*. Indonesia: Menteri Perhubungan Republik Indonesia, 2015, hal. 20.

[2] "US Top 40 Airports," *World Airport Codes*, 2016. [Daring]. Tersedia pada: https://www.world-airport-codes.com/us-top-40-airports.html. [Diakses: 13-Feb-2020].

[3] D. Meyer, "Heavy rains ground hundreds of flights at NYC airports," *New York Post*, 2019. [Daring]. Tersedia pada: https://nypost.com/2019/08/07/heavy-rains-ground-hundreds-of-flights-at-nyc-airports/. [Diakses: 13-Feb-2020].

[4] "Hundreds of NYC-Area Flights Delayed, Canceled Second Day in a Row," *NBC New York*, 2019. [Daring]. Tersedia pada: https://www.nbcnewyork.com/news/local/winter-storm-covers-new-jersey-roads-delays-new-york-flights/2235023/. [Diakses: 13-Feb-2020].

[5] U. S. D. of Transportation, "Reporting Carrier On-Time Performance," 2019. [Daring]. Tersedia pada: https://transtats.bts.gov/. [Diakses: 01-Feb-2020].

[6] A. Ahmad, "Mengenal Artificial Intelligence, Machine Learning, Neural Network, dan Deep Learning," *J. Teknol. Indones.*, no. October, hal. 3, 2017.

[7] P. Sugiartawan, R. Pulungan, dan A. Kartika, "Prediction by a Hybrid of Wavelet Transform and Long-Short-Term-Memory Neural Network," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 2, 2017.

[8] P. Sugiartawan, A. A. Jiwa Permana, dan P. I. Prakoso, "Forecasting Kunjungan Wisatawan Dengan Long Short Term Memory (LSTM)," *J. Sist. Inf. dan Komput. Terap. Indones.*, vol. 1, no. 1, hal. 43–52, 2018.

[9] S. Hochreiter, "Long Short-Term Memory," vol. 1780, hal. 1735–1780, 1997.

[10] J. Zheng, C. Xu, Z. Zhang, dan X. Li, "Electric load forecasting in smart grids using Long-Short-Term-Memory based Recurrent Neural Network," *2017 51st Annu. Conf. Inf. Sci. Syst. CISS 2017*, hal. 1–6, 2017.

[11] L. Chen dan C. Min, "A Comparisons of BKT, RNN and LSTM for Learning Gain Prediction," *Aied 2017*, vol. 1, hal. 650–655, 2017.

[12] R. Zhao, J. Wang, R. Yan, dan K. Mao, "Machine health monitoring with LSTM networks," *Proc. Int. Conf. Sens. Technol. ICST*, 2016.