Tugas Mandiri IF5181 Pengenalan Pola

Patrick Nugroho Hadiwinoto

Magister Informatika, Institut Teknologi Bandung, Bandung, Indonesia
23519022@std.stei.itb.ac.id

Abstract. This take-home task was given to give students more exploration time with the basic of Recurrent Neural Network (RNN), hands-on with RNN, doing Natural Language Processing (NLP) task using RNN, and to predict stock market using Long Short-Term Memory (LSTM) with Keras. RNN is useful to process sequential data. It is due to its ability to store information of the previous data in order to predict the value of next data. LSTM is one of the most popular RNN algorithms that is widely used to conduct prediction in many aspect, such as speech phoneme recognition and many more. LSTM is very powerful in analyzing time series and sequential data.

1 Pendahuluan

Recurrent Neural Network (RNN) adalah sebuah struktur jaringan syaraf buatan berdasarkan konsep *deep learning* yang memiliki kelebihan dalam menangani kasus data sekuensial. Data sekuensial adalah data yang nilainya bergantung kepada nilai dari data tersebut untuk periode waktu sebelumnya. Keterkaitan antar data ini menyebabkan *input* atau masukan dari struktur ini bukan merupakan data tunggal tetapi berupa *time series data* atau *sequence of inputs*. *Output* atau keluaran dari struktur ini dapat berupa sebuah nilai atau banyak nilai, bergantung kepada domain masalah yang ditangani.

RNN dapat memprediksi keluaran dari *time series data* karena memiliki *memory* yang menyimpan konteks dari setiap masukan. Hal ini yang tidak dimiliki oleh struktur jaringan syaraf buatan lainnya seperti pada *Convolutional Neural Network* (CNN). RNN menerima *input* x(t) misalnya berupa kata-kata setiap satu satuan waktu kemudian mengingat informasi/konteks menggunakan aktivasi *hidden layer* yang dilewatkan dari satu *time-step* ke *time-step* berikutnya. Di akhir, *output* dikeluarkan dengan mempertimbangkan nilai dari komponen-komponen sebelumnya ini.

2 RNN – Hands On

Tugas ini mengenalkan mahasiswa pada implementasi perdana RNN menggunakan

library ‘numpy’. Di awal sudah disediakan *file* ‘rnn_utils.py’ yang berisi implementasi dari fungsi-fungsi ‘softmax’, ‘sigmoid’, dan *optimizer* ‘Adam’ (fungsi ‘initialize_adam’ dan ‘update_parameters_with_adam’). Pada tugas Hands On ini diimplementasikan RNN-Cell dan LSTM Network. Untuk RNN-Cell dibuatlah fungsi ‘rnn_cell_forward’ untuk melangkah maju setiap satu *time-step* pada sel RNN dan ‘rnn_forward’ untuk propagasi maju pada RNN. Fungsi ‘rnn_forward’ menerima *input* berupa *hideen state* dari sel sebelumnya dan *input data* dari *time-step* sekarang. Fungsi ini akan memberikan *output* berupa *hidden state* dan prediksi dari *time-step* sekarang.

Pada LSTM Network pun mirip, yaitu terdapat fungsi ‘lstm_cell_forward’ untuk mengenali dan memperbarui (*update*) nilai *cell state* atau variabel memori c(t) pada setiap *time-step* pada sel LSTM dan ‘lstm_forward’ untuk kasus banyak *time-step* (*sequence of inputs*) yaitu untuk propagasi maju pada RNN menggunakan sel LSTM.

Selain itu, terdapat pula implementasi dari fungsi ‘rnn_cell_backward’ untuk melangkah mundur setiap satu *time-step* pada sel RNN dan ‘rnn_backward’ yang berisi perhitungan gradien dari *cost* setiap *time-step* pada propagasi balik (*backpropagation*) ke sel RNN sebelumnya serta ‘lstm_cell_backward’ untuk melangkah mundur setiap satu *time-step* pada sel LSTM.

3 *Character-Level Language Model – Dinosaurus Land*

Pada tugas ini diberikan dataset ‘dinos.txt’ yang berisi nama-nama dinosaurus yang ada. Tugas mahasiswa adalah membuat model bahasa untuk membangkitkan nama-nama dinosaurus baru dengan menggunakan algoritma RNN dan *input* berupa nama-nama pada *file* ‘dinos.txt’ tersebut. Pada tugas ini *file* ‘rnn_utils.py’ kembali digunakan.

Langkah pertama yang dilakukan adalah melakukan *loading* dan *preprocessing* pada dataset (‘dinos.txt’). Yang akan dilakukan di sini adalah mengekstraksi seluruh karakter yang ada pada dataset dan membuat python *dictionary* (misalnya tabel *hash*) untuk memetakan setiap karakter ke indeks 0 hingga 26. Selain itu dibuat pula *dictionary* kedua yang memetakan setiap indeks dengan karakter yang bersesuaian.

Setelah dilakukan inisialisasi parameter, kemudian dilakukan propagasi maju untuk menghitung *loss function*. Lalu, propagasi balik dilakukan untuk menghitung gradien berdasarkan *loss function*. Kemudian dilakukan *clipping* pada gradien. *Clipping* dilakukan untuk mencegah terjadinya pembengkakan nilai gradien, yaitu yang bernilai terlalu besar. Fungsi ‘*clip*’ menerima masukan berupa *dictionary* gradien dan memberikan keluaran berupa gradien yang sudah di-*clip* (jika diperlukan). Hal ini karena nilai gradien yang tidak melebihi ‘maxValue’ tidak akan di-*clip*. Nilai maxValue ditentukan sendiri oleh pemrogram sesuai kebutuhan.

Gradien yang sudah didapat kemudian digunakan untuk meng-*update* nilai parameter menggunakan aturan *gradient descent update*. Di akhir *loop* akan diperoleh nilai parameter baru. Hal ini akan dilakukan terus menerus hingga terjadi konvergensi.

Model yang sudah dibentuk akan digunakan untuk membangkitkan nama-nama dinosaurus baru. Yang dilakukan adalah mengimplementasikan fungsi 'sample' yang berisikan fungsi hiperbolik 'tanh' dan fungsi aktivasi 'softmax'. Kemudian dilakukan pembangkitan model bahasa menggunakan teknik stochastic gradient descent yaitu dengan mengimplementasikan fungsi 'optimize' yang memanfaatkan beberapa fungsi seperti 'rnn_forward', 'rnn_backward' dan 'update_parameters'.

Pelatihan model dilakukan dengan memasukan *input* berupa setiap baris (sebuah nama dinosaurus) dari 'dinos.txt'. Randomisasi sangat berperan penting di sini. Dengan menjalankan perintah 'parameters = model(data, ix_to_char, char_to_ix) dengan 'ix_to_char' dan 'char_to_ix' adalah kedua *dictionary* yang disebutkan di awal, maka akan dibangkitkan kata-kata acak hasil pembangkitan model bahasa.

4 Prediksi Pasar Modal – Dataset NSE-TATAGLOBAL

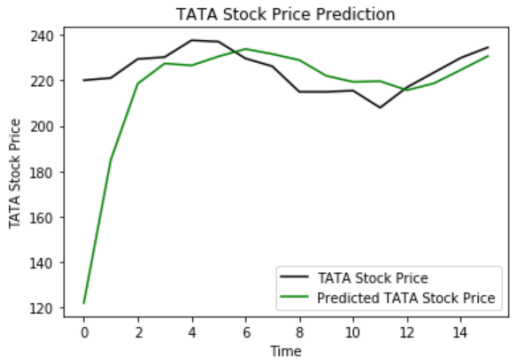
Pada tugas ini dilakukan prediksi pasar modal pada dataset NSE-TATAGLOBAL dari tahun 2010 hingga 2018 pada bagian 'Open' dengan menggunakan *input* berupa data dari 60 *time-steps* sebelumnya. Setelah dataset di-*load*, kemudian dilakukan *feature scaling* yaitu mengubah nilai 'Open' dari setiap baris menjadi berkisar antar 0 dan 1. *Scaling* ini menggunakan fungsi MinMaxScaler dari *library* ScikitLearn.

Struktur LSTM yang dibentuk terdiri dari sebuah *input layer*, 3 buah *hidden layer* dengan masing-masing terdiri dari 50 buah neuron dan nilai 'Dropout' sebesar 0.2 (20% dari total *layer*), dan sebuah *output layer* 'Dense' yang memberikan satu unit *output*. Jumlah *epoch* sebesar 100 kali dan nilai *batch_size* sebesar 32.

Dari model yang terbentuk dilakukan prediksi terhadap dataset_test. Dataset uji ini terdiri dari 16 baris data yang tidak *overlap* dengan dataset_train yang berjumlah 2.035 baris data. Ada konkatenasi pada awal proses pengujian karena data uji pertama harus memiliki *input* berupa 60 data *time-steps* sebelumnya.

Setelah dilakukan prediksi pada dataset menggunakan model yang terbentuk, hasilnya diubah kembali menjadi bentuk awal menggunakan fungsi 'inverse_transform'. Hal ini agar nilai hasil prediksi yang tadinya berkisar antar 0 dan 1 (untuk memudahkan perhitungan pada LSTM Keras) dikembalikan ke nilai sesungguhnya.

Hasil *plotting* menggunakan *MatPlotLib* memberikan gambaran di mana grafik dari hasil prediksi tidak jauh berbeda dengan grafik sesungguhnya. Hal ini membuktikan LSTM sangatlah baik dalam memprediksi *time series data* ataupun data sekuensial lainnya. Berikut adalah gambar hasil *plotting* kedua grafik.

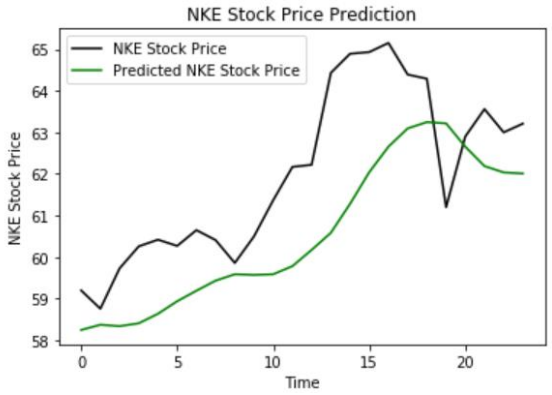


Gambar *plotting* kedua grafik pada dataset TATA

5 Prediksi Pasar Modal – Dataset NKE (Dataset baru)

Yang dilakukan pada tugas ini adalah mirip dengan tugas prediksi pasar modal sebelumnya, hanya saja kali ini dataset yang digunakan berupa hasil eksplorasi sendiri. Penulis menggunakan dataset NKE dari tautan berikut https://www.kaggle.com/szrlee/stock-time-series-20050101-to-20171231. Dataset NKE dibagi menjadi dua, training dataset yang terdiri dari 2.995 baris data dan 24 baris data untuk testing dataset. Data pasar modal yang ada berupa data per hari, berkisar dari tahun 2006 hingga 2017. Kolom yang digunakan adalah 'Open'. Time-steps yang digunakan sama dengan tugas sebelumnya yaitu 60, karena dianggap memberikan hasil yang cukup optimal (berupa data dua bulan sebelumnya).

Dilakukan *preprocessing* tambahan yaitu denganmenghapus sebuah baris data pada dataset latih karena memiliki *missing value* pada kolom 'Open'. Data latih sebenarnya berjumlah 2.996 baris data. Hal ini dipicu dari terjadinya fenomena nilai 'NaN' pada nilai *loss* pada setiap iterasi. Setelah dilakukan studi literatur pada beberapa literatur daring, ditemukanlah salah satu mitigasinya yaitu melakukan analisis pada data latih. Digunakan fungsi 'df.isnull().any()' dengan mengganti 'df' menjadi 'dataset_train'. Dari sini ditemukan fenomena adanya sebuah nilai 'Null' pada kolom 'Open' pada dataset_train. Baris tersebut kemudian dihapus. Setelah dilakukan *run* ulang, nilai *loss* sudah benar (tidak lagi 'NaN') dan memberikan hasil prediksi yang cukup baik. Berikut adalah gambar hasil *plotting* kedua grafik.



Gambar *plotting* kedua grafik pada dataset NKE (dataset baru)

References

1. https://datascience-enthusiast.com/DL/dlindex.html diakses 15 November 2019
2. https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/ diakses 2 Desember 2019
3. *Slide-slide* kuliah IF5181 Pengenalan Pola