Prediksi Harga Bitcoin Dengan Menggunakan Recurrent Neural Network

Redha Arifan Juanda[1], Jondri[2], Aniq Atiqi Rohmawati[3]

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1]redhajuanda@students.telkomuniversity.ac.id, [2]jondri@telkomuniversity.ac.id,
[3]aniqatiqi@telkomuniversity.ac.id

Abstrak

Bitcoin adalah salah satu kurensi elektronik yang bersifat terdesentralisasi dan tidak diatur atau dijamin oleh otoritas pusat. Sebagai sebuah sistem yang masih berusia muda, mengakibatkan harga Bitcoin sangat fluktuatif dan sering kali membuat resah pengguna dan investor Bitcoin. Oleh karena itu, diusulkan sebuah metode atau sistem prediksi harga Bitcoin dengan mempelajari pola dan tingkah laku data *time series* harga historisnya.

Sebelumnya telah dilakukan penelitian yang serupa yaitu prediksi harga Bitcoin dengan menggunakan *Support Vector Machines*. Teknik yang diusulkan pada penelitian ini yaitu prediksi harga Bitcoin dengan menggunakan salah satu arsitektur *Artificial Neural Network* (ANN) yaitu *Recurrent Neural Network* (RNN). Semakin optimal model yang dibangun maka akan semakin tinggi pula akurasi yang dihasilkan. Bobot RNN yang optimal dapat diperoleh dengan algoritma optimasi *Backpropagation Through Time* (BPTT).

Dari proses pelatihan dan pengujian, didapatkan akurasi terbaik sebesar 98.76% pada data latih dan 97.46% pada data uji.

Kata kunci : Bitcoin, Recurrent Neural Network, Data Time Series, Prediksi, Backpropagation Through Time

Abstract

Bitcoin is a decentralized digital currency, as the system works without a central bank or single administrator. As the system is still young, Bitcoin prices are very volatile and often make users and investors worried. Therefore, the author proposed a method or a system of Bitcoin prices prediction by studying the pattern and behavior of time series data.

Previously, a similar study has been conducted, the prediction of Bitcoin prices using Support Vector Machines. The technique proposed in this research is Bitcoin prices prediction using one of Artificial Neural Network architecture (ANN) which is Recurrent Neural Network (RNN). The more optimal the model is built the higher the accuracy will be produced. The optimal RNN weight can be obtained with Backpropagation Through Time (BPTT) optimization algorithm.

From the training and testing processes, the model archieves the highest accuracy of 98,76% on training data and 97,46% on testing data.

Keywords: Bitcoin, Recurrent Neural Network, Time Series Data, Forecasting, Backpropagation Through Time

1. Pendahuluan

Latar Belakang

Bitcoin adalah sebuah kurensi baru yang diciptakan pada tahun 2009 oleh seorang yang tidak diketahui dengan nama alias Satoshi Nakamoto. *Cryptocurrency* yang bersifat terdesentralisasi dan tidak diatur atau dijamin oleh otoritas pusat ini telah ramai digunakan untuk bertransaksi dan investasi di berbagai negara, termasuk Indonesia. Akan tetapi, meski memiliki banyak kelebihan dibandingkan dengan mata uang konvensional, perdagangan Bitcoin merupakan aktivitas berisiko tinggi, harga Bitcoin sangat fluktuatif, dimana harga dapat berubah secara signifikan dari waktu ke waktu.

Pada awal januari 2013 misalnya, Bitcoin dihargai 13 dollar AS per keping (1 BTC). Angka itu meroket ke lebih dari 1.100 dollar AS per keping pada Desember tahun yang sama, lalu terpangkas menjadi hanya setengahnya, hanya dalam beberapa jam setelah pelarangan transaksi Bitcoin di China. [1]

Oleh karena itu, prediksi atau peramalan harga sangat penting untuk diketahui agar meminimalisir resiko kerugian dalam investasi Bitcoin. Untuk dapat melakukan prediksi tersebut, diperlukan data historis yang dapat menunjukkan pola kejadian di masa lampau. Data historis yang digunakan adalah data harga perdagangan Bitcoin dengan interval 1 hari.

Telah banyak dilakukan penelitian terhadap peramalan data *time series* seperti ini dengan berbagai metode. [2, 3, 4] Dasar yang dipakai pada proses prediksi data *time series* adalah eksplorasi data non-linear yang sudah ada untuk membentuk suatu pola model sehingga bisa dijadikan acuan untuk memprediksi nilai selanjutnya.

Masalah prediksi *time series* adalah jenis pemodelan prediktif yang sulit, tidak seperti pemodelan predikftif regresi, *time series* juga menambah kompleksitas ketergantungan urutan antar variable input. Berdasarkan penelitian [5], *Recurrent Neural Network* terbukti berhasil digunakan untuk prediksi data *time series* karena RNN mampu menggunakan informasi yang telah direkam sebelumnya yang panjang urutannya atau *sequence*-nya beragam-ragam. Oleh karena itu, pembangunan sistem ini dibuat dengan metode *Recurrent Neural Network* dengan menggunakan algoritma *Backpropagation Through Time*.

Topik dan Batasannya

Adapun rumusan masalah yang dibahas pada tugas akhir ini adalah bagaimana mendapatkan bobot optimal yang dapat digunakan untuk memprediksi harga Bitcoin dengan menggunakan *Recurrent Neural Network* serta bagaimana performansi dari *Recurrent Neural Networ*k dalam memprediksi harga Bitcoin.

Prediksi dilakukan dengan mempelajari pola dan tingkah laku data *time series* harga historisnya. Faktor-faktor lain yang dapat menyebabkan perubahan harga, seperti situasi politik, berita baik dan buruk tentang Bitcoin, ataupun kejahatan *hacker* tidak diperhitungkan. Dan pada penelitan ini hanya memprediksi harga penutupan (*closing price*) Bitcoin

Tujuan

Tujuan dari penelitian ini adalah untuk mendapatkan bobot optimal dengan mempertimbangkan skenario komposisi data, jumlah pola input, jumlah *epoch*, nilai *learning rate* dan jumlah *hidden unit* di *hidden layer* sehingga dapat digunakan untuk memprediksi harga Bitcoin dengan hasil akurasi yang tinggi. Dan selanjutnya melakukan analisis hasil performansi dari hasil prediksi harga Bitcoin menggunakan *Recurrent Neural Network*.

2. Studi Terkait

2.1 Related Work

Penelitian tentang prediksi harga Bitcoin dengan menggunakan Machine Learning saat ini masih sangat kurang.

Georgoula dkk. [2] menyelidiki faktor-faktor penentu harga Bitcoin dan juga sekaligus mengimplementasikan *Sentiment Analysis* menggunakan *Support Vector Machines*. Penulis menemukan bahwa frekuensi view Wikipedia tentang Bitcoin dan *hash rate* jaringan memiliki korelasi positif terhadap harga Bitcoin.

Greaves dkk. [3] menganalisis *Blockchain* Bitcoin untuk memprediksi harga Bitcoin menggunakan SVM dan ANN. Penulis melaporkan akurasi sebesar 55 persen dengan regular ANN. Mereka menyimpulkan bahwa ada keterbatasan prediksi pada data Blockchain karena harga secara teknis didikte oleh bursa yang perilakunya terletak di luar wilayah Blockchain.

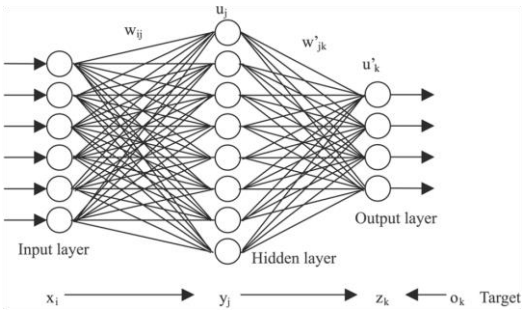
Matta dkk. [4] menganalisis hubungan antara harga Bitcoin, tweet dan view Bitcoin di Google Trends. Penulis menemukan korelasi yang lemah hingga sedang antara harga Bitcoin dan view Google Trends dan *tweet* positif di Twitter. Penulis menemukan ini sebagai bukti bahwa mereka dapat digunakan sebagai prediktor. Namun, satu keterbatasan dalam penelitian ini adalah ukuran sampel terbatas hanya 60 hari.

2.2 Artificial Neural Network

Artificial Neural Network atau dalam bahasa Indonesia yaitu Jaringan Syaraf Tiruan (JST) adalah sebuah model komputasi yang mempunyai karakteristik menyerupai jaringan syaraf manusia yang didasarkan atas asumsi sebagai berikut: [6]

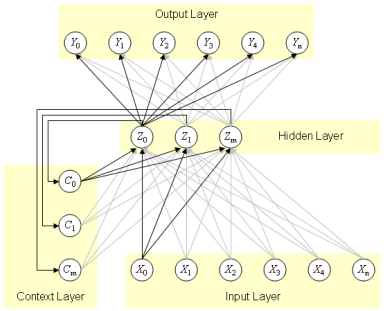
1. Pemrosesan informasi terjadi pada elemen sederhana yang disebut neuron.
2. Isyarat mengalir di antara sel saraf/neuron melalui satu sambungan penghubung.
3. Setiap sambungan penghubung memiliki bobot yang bersesuaian.
4. Setiap sel saraf akan menerapkan fungsi aktivasi terhadap isyarat hasil penjumlahan berbobot yang masuk kepadanya untuk menentukan isyarat keluarannya.

Artificial Neural Network terdiri dari banyak neuron buatan yang biasa disebut unit yang diatur dalam serangkaian lapisan. Berikut gambar arsitektur umum dari *Artificial Neural Network*:



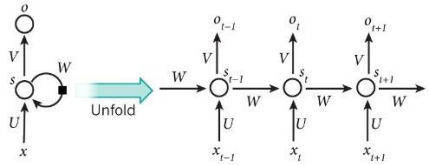
Gambar 2-1. Arsitektur ANN **[7]**

2.3 Recurrent Neural Network



Gambar 2-2. Arsitektur Recurrent Neural Network

Recurrent Neural Network (RNN) pertama dikembangkan oleh Jeff Elman pada tahun 1990 [8]. RNN merupakan variasi dari *Artificial Neural Network*, perbedaan utama yang terdapat pada model ini adalah sinyal dapat mengalir secara *forward* dan *backward* secara berulang. Untuk bisa melakukan hal tersebut, maka ditambahkan sebuah layer baru yang disebut dengan c*ontext layer*. Selain melewati *input* antar layer, *output* dari setiap layer juga menuju ke *context laye*r untuk digunakan sebagai inputan pada *timestep* berikutnya. RNN menyimpan informasi di *context layer*, yang membuatnya dapat mempelajari urutan data dan menghasilkan output atau urutan lain. Jika ditarik kesimpulan maka bisa dikatakan bahwa RNN memiliki memori yang berisikan hasil rekaman informasi yang dihasilkan sebelumnya.



Gambar 2-3. Proses RNN saat menghitung proses didepannya

Diagram diatas menunjukkan RNN pada posisi yang tidak dibuka ke *full network*. Dengan membuka gulungan RNN maka simpelnya kita menuliskan seluruh jaringan dengan urutan (sequence) secara lengkap. Berikut ialah keterangan simbol yang ada pada gambar diatas:

1. x_t ialah input pada setiap *time step*.
2. s_t adalah *hidden state* pada setiap *time step* t
3. o_t adalah output untuk setiap step t

2.4 Backpropagation Through Time

Backpropagation Through Time adalah sebuah algoritma yang digunakan untuk mengubah nilai bobot dalam metode *Recurrent Neural Network*. Algoritma pelatihan *Backpropagation Through Time* biasanya digunakan untuk data yang memiliki ketergantungan urutan seperti data *time series.*

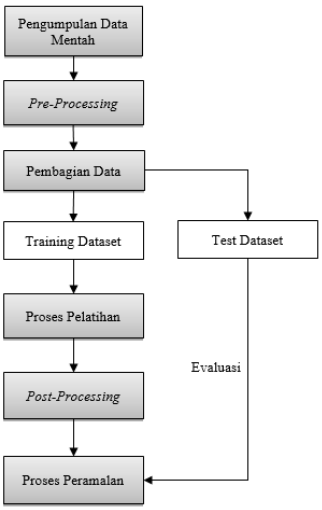
Konsep utama dari BPTT adalah membentangkan jaringan ke dalam waktu dengan meletakkan salinan yang sama dari *Recurrent Neural Network* dan mengatur kembali koneksi jaringan untuk mendapatkan koneksi antara salinan selanjutnya. Untuk menghasilkan peramalan yang akurat, parameter yang ada dalam RNN akan diuji seperti *learning rate*, jumlah neuron dan banyaknya data.

Singkatnya, algoritma berjalan seperti berikut: [9]

1) Berikan *input* dan *output* ke jaringan.
2) Buka gulungan jaringan lalu kalkulasi dan akumulasi *error* di setiap *timesteps*.
3) Gulung jaringan dan sesuaikan nilai bobot.
4) Ulangi.

3. Sistem yang Dibangun

3.1 Perancangan Sistem



Gambar 3-1. Blok Diagram

3.2 Pre-Processing dan Post-Processing

Pre-processing yaitu proses penskalaan pada data sedemikian sehingga keseluruhan data berada pada *range* yang sama. Dalam tugas akhir ini, data akan dinormalisasi ke dalam suatu nilai dengan *range* [0.1, 0.9]. Hal ini disesuaikan dengan fungsi sigmoid biner yang merupakan fungsi aktivasi yang akan digunakan yang nilainya tidak pernah mencapai 0 ataupun 1. Proses *preprocessing* dilakukan dengan persamaan 3.1.

$$x' = \frac{(0.9 - 0.1)(x - a)}{b - a} + 0.1 \tag{3.1}$$

Sedangkan data yang sudah dilakukan *pre-processing*, apabila ingin dikembalikan ke data aslinya disebut dengan proses *post-processing*. Prosesnya dilakukan dengan persamaan 3.2.

$$x = \frac{(x' - 0.1)(b - a)}{0.9 - 01} + a \tag{3.2}$$

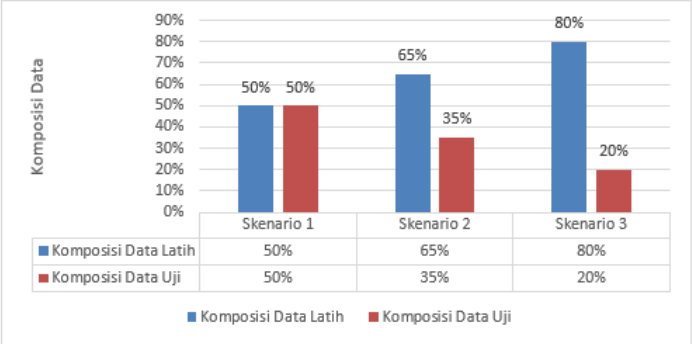
dimana *x* adalah data yang belum dinormalisasi, *x'* adalah data yang sudah dinormalisasi, *a* adalah nilai minimum dari keseluruhan data dan *b* adalah nilai maksimum dari keseluruhan data

3.3 Pembagian Data

Data yang digunakan berupa data harga Bitcoin yang diambil tiap hari mulai dari 3 Januari 2013 – 10 November 2017. Data berupa bilangan *real* dalam satuan mata uang US Dollar. Data diperoleh dari www.coindesk.com. Data sebelum tahun 2013 tidak disertakan karena penulis merasa sistem masih belum matang sehingga data masih relatif tidak berubah dan tidak merepresentasikan keseluruhan data yang sangat fluktuatif. Berikut adalah grafiknya:

Gambar 3-2. Diagram Datasets Harga Bitcoin

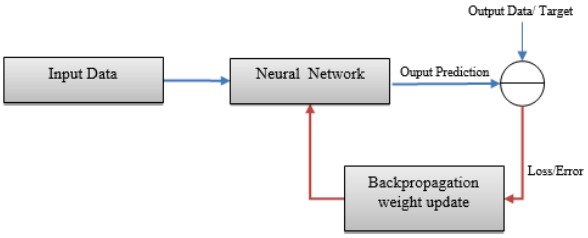
Total data terdiri dari 1.722 baris data yang kemudian dibagi menjadi 2 bagian yaitu data latih dan data uji. Berbagai pembagian komposisi data akan dilakukan untuk diobservasi. Hal ini bertujuan untuk menentukan komposisi data yang terbaik. Berikut ini adalah grafik komposisi data yang digunakan:



Gambar 3-3. Grafik komposisi data

3.4 Proses Pelatihan

Proses pelatihan terdiri dari 2 bagian utama yaitu *Fordward Pass* dan *Backward Pass*. Berikut tahapan dari proses pelatihan, panah biru dibawah ini adalah *Forward Pass* dan panah merah adalah *Backward Pass*.



Gambar 3-4. Pelatihan *Neural Network*

Pada saat *forward pass*, *input* akan di-*propagate* menuju *output layer* dan hasil dari *output* tersebut akan dibandingkan dengan data target yang benar dengan menggunakan fungsi *Loss Function.*

Pada proses ini, parameter-parameter RNN dimasukkan ke dalam sistem. Adapun parameter-parameter tersebut adalah jumlah neuron pada *input layer*, jumlah neuron pada *hidden layer*, dan jumlah neuron pada *output layer*. Adapun jumlah layer untuk *input layer* adalah 1, jumlah layer untuk *hidden layer* adalah 1, dan jumlah layer pada *output layer* adalah 1.

3.5 Proses Peramalan

Setelah didapatkan model terbaik dari proses pelatihan berdasarkan data *training* untuk sistem prediksi harga Bitcoin. Lalu model dan *knowledge* inilah yang digunakan pada proses peramalan sehingga proses peramalan yang dilakukan diharapkan menghasilkan hasil prediksi dengan tingkat akurasi yang tinggi.

3.6 Pengukuran Performansi

Pengukuran performansi dilakukan untuk mengetahui seberapa dekat hasil peramalan yang dilakukan oleh sistem dengan data yang sebenarnya. Pada Tugas Akhir ini pengukuran performansi dinyatakan dengan Mean Absolute Percentage Error (MAPE). Rumus MAPE sebagai berikut:

$$MAPE = \frac{100\%}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right| \tag{3.3}$$

dimana A_t adalah *actual value* atau nilai target yang diharapkan dan F_t adalah *forecast value* atau output yang dihasilkan oleh RNN. Perhitungan MAPE didasarkan pada data *preprocessing*. Sedangkan untuk nilai akurasi dari tiap solusi dihitung berdasarkan MAPE, yaitu:

$$accuracy = 100 - MAPE \tag{3.4}$$

4. Evaluasi

4.1 Hasil Pengujian

Berdasarkan percobaan yang telah dilakukan, diperoleh parameter yang optimal yang dapat digunakan untuk memprediksi harga Bitcoin dengan menggunakan *Recurrent Neural Network*. Parameter-parameter tersebut adalah:

- Jumlah Pola Input : 5
- Jumlah Epoch : 2000
- Nilai Learning Rate : 0.001
- Jumlah Hidden Unit : 50

4.2 Analisis Hasil Pengujian

Pada proses pengujian, dilakukan observasi terhadap kombinasi parameter untuk mendapatkan kombinasi parameter terbaik, meliputi jumlah pola input, jumlah *epoch*, nilai *learning rate* dan jumlah *hidden unit* di *hidden layer*. Berikut ini adalah nilai parameter-parameter yang diuji:

- Jumlah Pola Input : 1, 2, 3, 4, 5
- Jumlah Epoch : 100, 500, 1000, 2000, 5000
- Nilai Learning Rate : 0.0001, 0.001, 0.005, 0.01
- Jumlah Hidden Unit : 20, 50, 100, 200, 300

Untuk masing-masing parameter yang dikombinasikan dilakukan percobaan sebanyak 10 kali, hasil yang diambil adalah rata-rata dari 10 kali running tersebut. Hal ini dilakukan karena bobot awal diinisialisasi secara random, sehingga hasil yang didapatkan tidak akan sama setiap kali percobaan, bisa saja kebetulan baik, atau kebetulan buruk.

Agar sistem yang dibangun tidak mengalami kondisi *overfit*, maka error yang diperhitungkan adalah error pada proses pengujian. Parameter yang menghasilkan error pengujian terkecil dianggap sebagai parameter terbaik.

Berikut ini adalah parameter-parameter awal yang digunakan:

- Jumlah Pola Input : 3
- Jumlah Epoch : 1000
- Nilai Learning Rate : 0.001
- Jumlah Hidden Unit : 10

4.2.1 Analisis Komposisi Data

Berikut ini adalah hasil observasi untuk komposisi data pada sistem yang dibangun:

Table 4-1. Hasil pengujian komposisi data

Skenario	MAPE Training (Average)	MAPE Testing (Average)
Skenario 1	1.81 %	6.72 %
Skenario 2	1.58 %	7.40 %
Skenario 3	1.59 %	13.92 %

Berdasarkan tabel diatas, diketahui bahwa komposisi data yang baik digunakan adalah skenario 1. Hal ini karena nilai error MAPE *testing* pada skenario 1 lebih baik daripada nilai error pada skenario 2 dan 3. Skenario 2 dan 3 hanya baik diterapkan pada proses pelatihan, namun jika error pada proses pelatihan dan pengujian digabungkan, skenario 1 akan menghasilkan error terkecil.

4.2.2 Analisis Pola Input dan Jumlah Epoch

4.2.2.1 Analisis Pola Input

Prediksi time series pada *Neural Network* dapat dilakukan dengan menggunakan input berupa 2 series, 3 series, 4 series, dan sebagainya. Jumlah pola *time series* merepresentasikan pola yang akan dipelajari oleh RNN. Jika digunakan 3 series, berarti prediksi dilakukan dengan menggunakan 3 data sebelumnya, yaitu (t-1), (t-2) dan (t-3) dan begitu juga untuk series yang lain.

Tidak ada aturan pasti berapa jumlah pola input yang cocok dalam suatu prediksi data time series. Sehingga biasanya jumlah pola time series yang digunakan dalam suatu arsitektur jaringan didapatkan melalui percobaan (*trial error*).

4.2.2.2 Analisis Jumlah Epoch

Jumlah *epoch* merepresentasikan lamanya proses pembelajaran yang dilakukan terhadap jaringan yang sedang diobservasi. Jumlah *epoch* menentukan kapan proses pembelajaran dihentikan. Semakin besar nilai jumlah epoch, maka semakin lama pula proses pembelajaran berlangsung. Begitu juga sebaliknya.

Jumlah *epoch* yang terlalu sedikit mengakibatkan jaringan yang terbentuk bersifat terlalu general/umum. Artinya kemampuan jaringan dalam mengenali pola terlalu sedikit atau bahkan tidak ada sama sekali. Kondisi ini dinamakan *overweight*. *Overweight* akan mengakibatkan jaringan yang terbentuk akan kehilangan kemampuannya dalam mengenali pola dari data yang diberikan.

Jumlah *epoch* yang terlalu banyak akan mengakibatkan jaringan mengalami kondisi *overfit*. Kondisi *overfit* mempunyai arti bahwa jaringan bersifat terlalu spesifik terhadap data pelatihan. Akibat dari *overfit* adalah ketika proses pengujian berlangsung, jaringan akan mengalami kegagalan dalam menghasilkan output yang sesuai terhadap data pengujian dikarenakan jaringan kehilangan kemampuan generalisasinya. Jumlah *epoch* yang tepat akan menentukan kemampuan akhir jaringan dalam melakukan memorialisasi dan generalisasi. Memorialisasi adalah kemampuan jaringan dalam mengenali pola yang telah dilatihkan sedangkan generalisasi adalah kemampuan jaringan dalam mengenali pola yang belum pernah dilatihkan [10]

Tidak ada aturan pasti berapa jumlah *epoch* maksimal. Sehingga biasanya jumlah *epoch* yang digunakan dalam suatu arsitektur jaringan juga didapatkan melalui percobaan (*trial error*).

4.2.2.3 Analisis Kombinasi Pola Input dan Jumlah Epoch

Parameter jumlah pola input yang digunakan adalah 1, 2, 3, 4 dan 5 pola input. Sedangkan jumlah *epoch* yang digunakan adalah 100, 500, 1000, 2000 dan 5000 *epoch*.

Table 4-2. Hasil pengujian kombinasi parameter pola input dan jumlah epoch

Pola Input	Jumlah Epoch	Learning Rate	Hidden Unit	MAPE Training (Average)	MAPE Testing (Average)
1	100	0.001	100	14.57 %	23.76 %
2	100	0.001	100	15.63 %	26.32 %
3	100	0.001	100	14.68 %	24.09 %
4	100	0.001	100	12.72 %	21.12 %
5	100	0.001	100	13.43 %	22.44 %
1	500	0.001	100	2.70 %	6.75 %
2	500	0.001	100	1.82 %	6.91 %
3	500	0.001	100	1.84 %	6.83 %
4	500	0.001	100	1.82 %	6.48 %
5	500	0.001	100	1.92 %	6.32 %
1	1000	0.001	100	1.82 %	7.85 %
2	1000	0.001	100	1.82 %	7.14 %
3	1000	0.001	100	1.76 %	6.07 %
4	1000	0.001	100	1.73 %	5.96 %
5	1000	0.001	100	1.82 %	5.94 %
1	2000	0.001	100	1.78 %	7.62 %
2	2000	0.001	100	1.67 %	6.22 %
3	2000	0.001	100	1.62 %	5.08 %
4	2000	0.001	100	1.43 %	2.85 %
5	2000	0.001	100	1.31 %	2.65 %
1	5000	0.001	100	1.76 %	6.85 %
2	5000	0.001	100	1.29 %	3.03 %
3	5000	0.001	100	1.15 %	5.30 %
4	5000	0.001	100	1.07 %	4.98 %
5	5000	0.001	100	1.09 %	5.02 %

Berdasarkan hasil analisis kombinasi pola input dan jumlah *epoch*, hasil kombinasi terbaik adalah dengan menggunakan jumlah pola input 5 dan jumlah *epoch* 2000 yang menghasilkan error MAPE *testing* terkecil yaitu 2.65%.

Dengan menggunakan jumlah epoch sebanyak 5000, sistem mampu menghasilkan error MAPE pada data *training* yang lebih kecil yaitu 1.07%, akan tetapi tidak mampu menghasilkan error yang baik pada data *testing*. Kondisi seperti inilah yang dinamakan *overfit*, dimana jaringan terlalu sepesifik tehadap data *training*, dan kehilangan kemampuan generalisasinya.

4.2.3 Learning Rate

Learning rate adalah sebuah parameter yang mempengaruhi kecepatan Jaringan Syaraf Tiruan dalam mencari solusi minimum. Nilai *learning rate* yang terlalu kecil akan mengakibatkan laju perubahan pada bobot dalam mencari solusi menjadi pelan sedangkan nilai *learning rate* yang terlalu besar akan mengakibatkan laju perubahan bobot dalam langkah mencari solusi menjadi sangat besar dan sulit menemukan solusi. Bahkan jika terlalu besar maka sistem akan berosilasi dan tidak bias mencapai kondisi konvergen.

Sama seperti parameter-parameter sebelumnya, tidak ada aturan pasti berapa nilai *learning rate* yang cocok dalam suatu prediksi data *time series*. Nilai *learning rate* yang cocok umumnya didapatkan dengan percobaan (*trial error).*

Nilai *learning rate* yang digunakan dalam skenario pengujian ini adalah 0.0001, 0.001, 0.005 dan 0.01. Berikut hasil pengujian dengan menggunakan beberapa nilai *learning rate* tersebut:

Table 4-3. Hasil pengujian kombinasi parameter learning rate

Pola Input	Jumlah Epoch	Learning Rate	Hidden Unit	MAPE Training (Average)	MAPE Testing (Average)
5	2000	0.0001	200	1.88 %	7.42 %
5	2000	0.001	200	1.44 %	3.23 %
5	2000	0.005	200	1.51 %	5.58 %
5	2000	0.01	200	1.49 %	6.05 %

Berdasarkan pada tabel diatas menunjukan bahwa dengan penggunaan nilai *learning rate* 0.001, menghasilkan error MAPE terkecil pada data training maupun data testing dibandingakan dengan menggunakan nilai *learning rate* lainnya. Dengan menggunakan nilai learning rate 0.0001, sistem menghasilkan error terbesar karena dengan penggunaan nilai *learning rate* terlalu kecil, menyebabkan sistem lebih pelan dalam mencari solusi, dan akan membutuhkan proses pelatihan yang lebih lama. Sedangkan ketika menggunakan nilai *learning rate* 0,005 dan 0.01, sistem akan menghasilkan error yang kurang optimal karena nilai *learning rate* yang terlalu besar, sehingga sistem lebih sulit dalam mencari solusi optimal.

4.2.4 Hidden Unit

Jumlah *hidden unit* mempengaruhi penghitungan pada fungsi aktivasi suatu neuron pada *output layer* dari sistem yang dibangun yang pada akhirnya menentukan nilai *output* yang dihasilkan oleh RNN.

Tidak ada aturan pasti berapa jumlah *hidden unit* yang cocok dalam suatu prediksi data *time series*. Jumlah *hidden unit* yang besar pada *hidden layer* tidak dapat merepresentasikan bahwa hasil yang didapat akan semakin baik. Hal ini bergantung pada kasus yang dialami. Sehingga biasanya jumlah *hidden unit* yang digunakan melalui percobaan (*trial error).*

Parameter jumlah *hidden unit* yang digunakan adalah 20, 50, 100, 200 dan 300 *hidden unit*. Berikut merupakan tabel hasil pengujian dengan beberapa parameter tersebut:

Table 4-4. Hasil pengujian kombinasi parameter hidden unit

Pola Input	Jumlah Epoch	Learning Rate	Hidden Unit	MAPE Training (Average)	MAPE Testing (Average)
5	2000	0.001	20	1.19 %	3.54 %
5	2000	0.001	50	1.24 %	2.54 %
5	2000	0.001	100	1.38 %	2.88 %
5	2000	0.001	200	1.62 %	3.77 %
5	2000	0.001	300	1.93 %	7.39 %

Berdasarkan tabel diatas, diketahui bahwa penggunaan jumlah *hidden unit* sebanyak 50 buah menghasilkan error MAPE terkecil pada data *testing*. Sedangkan dengan menggunakan *hidden unit* sebanyak 20 buah, hanya menghasilkan error MAPE terkecil pada data *training* saja, namun penggunaan *hidden unit* sebanyak 20 buah tidak menjamin bahwa nilai error *testing* akan lebih kecil.

5. Kesimpulan

Prediksi harga Bitcoin dapat dilakukan menggunakan *Recurrent Neural Network*. Akurasi rata-rata terbaik yang didapatkan adalah 98.76% pada data latih dan 97.46% pada data uji, dengan parameter jumlah pola input terbaik adalah 5, jumlah *epoch* 1000, nilai *learning rate* 0.001 dan jumlah *hidden unit* 50.

Saran untuk penelitian selanjutnya adalah untuk menggunakan *sentiment analysis*, seperti menggunakan berita baik dan buruk tentang Bitcoin, Google Trends, *tweet* tentang Bitcoin, dan sebagainya. Serta berbagai pendekatan yang tidak diimplementasikan dalam penelitian ini seperti melakukan proses *sliding window validatio*n juga bisa jadi pertimbangan untuk penelitian selanjutnya.

Daftar Pustaka

[1] O. Yusuf, "Bitcoin, Aman atau Berisiko?," Kompas, 20 01 2014. [Online]. Available: http://tekno.kompas.com/read/2014/01/20/1743437/Bitcoin.Aman.atau.Berisiko.. [Accessed 27 11 2017].

[2] I. Georgoula, D. Pournarakis, C. Bilanakos, . D. N. Sotiropoulos and G. M. Giagli, "Using time-series and sentiment analysis to detect the determinants of bitcoin," 2015.

[3] A. Greaves and B. Au, "Using the bitcoin transaction graph to predict the price of," 2015.

[4] M. Matta, I. Lunesu and M. Marchesi, "Bitcoin spread prediction using social and web search media," *Proceedings of DeCAT*, 2015.

[5] S. D. Balkin, "Using Recurrent Neural Networks for Time Series Forecasting," *International Symposium on Forecasting*, 1997.

[6] A. Hermawan, Jaringan Saraf Tiruan Teori dan Aplikasi, Yogyakarta: ANDI, 2006.

[7] G. Templeton, "Artificial neural networks are changing the world. What are they," 12 October 2015. [Online]. Available: https://www.extremetech.com/extreme/215170-artificial-neural-networks-are-changing-the-world-what-are-they. [Accessed 29 November 2017].

[8] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 02, p. 179–211, 1990.

[9] J. Brownlee, "A Gentle Introduction to Backpropagation Through Time," 23 June 2017. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-backpropagation-time/. [Accessed 15 November 2017].

[10] I. T. Harsono, Analisis dan Implementasi Elman Recurrent Neural Netwok dan Tabu Search pada Prediksi Harga Perak., Bandung: Institut Teknologi Telkom, 2011.