

Implementasi *Deep Neural Network* pada Perancangan Aplikasi Deteksi Token *Scam Blockchain* Ethereum

Implementation of Deep Neural Network in the Design of Ethereum Blockchain Scam Token Detection Applications

Dimas Arya Pamungkas, Ivana Lucia Kharisma*, Dwi Sartika Simatupang, Kamdan

Program Studi Teknik Informatika, Fakultas Teknik, Komputer dan Desain

Universitas Nusa Putra

Jalan Raya Cibolang No.21 Cisaat Sukabumi 43152, Indonesia

*e-mail: ivana.lucia@nusaputra.ac.id

(received: 22 Juli 2023, revised: 27 Juli 2023, accepted: 5 Agustus 2023)

Abstrak

Popularitas *blockchain* terus meningkat seiring dengan perkembangan teknologi, terutama dalam konteks Ethereum sebagai salah satu platform *blockchain* terkemuka. Namun, peningkatan ini juga diikuti dengan banyaknya kasus penipuan, terutama dalam bentuk token. Dalam teknologi *blockchain* Token sering mengacu pada *cryptocurrency* atau mata uang digital yang digunakan sebagai sarana pertukaran terkait dengan proyek atau platform tertentu. Penelitian ini merancang dan membangun sebuah sistem aplikasi yang dapat mendeteksi token kripto *scam* pada *blockchain* Ethereum, dikhususkan pada jenis token ERC-20 (*Ethereum Request for Comments 20*), yang diusulkan oleh Fabian Vogelsteller pada November 2015, merupakan standar token yang menerapkan API untuk token dalam Kontrak Pintar. Pembuatan aplikasi deteksi *scam* mengimplementasikan metode *deeplearning* dengan algoritma *DeepNeuralNetwork* (DNN) dan dilakukan evaluasi performa menggunakan dua skenario pengujian dengan pembagian *dataset* menjadi tiga rasio data latih dan data uji. *Output* dari aplikasi berupa JSON-RPC yang diintegrasikan dengan *website*. Dalam pengujian model DNN, dengan menggunakan 80% data latih dan 20% data uji, algoritma DNN memberikan akurasi sebesar 0.997558%. Selanjutnya, dilakukan pengujian sistem melibatkan berbagai skenario untuk memverifikasi fungsionalitasnya, termasuk validasi input, ekstraksi data, prediksi DNN, dan tampilan hasil prediksi, dimana memberikan hasil yang baik dari sistem yang dibuat. Aplikasi yang dibuat telah berhasil mengidentifikasi token *scam* dengan akurasi yang tinggi, meningkatkan keamanan pengguna dalam bertransaksi kripto.

Kata kunci: *Blockchain, Ethereum, Penipuan, Scam, Deep Learning.*

Abstract

The popularity of *blockchain* continues to increase as technology develops, especially in the context of Ethereum as one of the leading *blockchain* platforms. However, this increase was also followed by many cases of fraud, especially in the form of tokens. In *blockchain* technology, tokens often refer to *cryptocurrencies* or digital currencies used as a means of exchange related to a particular project or platform. This research designs and builds an application system that can detect *scam* crypto tokens on the Ethereum *blockchain*, specifically for the ERC-20 (*Ethereum Request for Comments 20*) token type, which was proposed by Fabian Vogelsteller in November 2015, is a token standard that implements APIs for tokens. in Smart Contracts. Making a *scam* detection application implements the deep learning method with the *Deep Neural Network* (DNN) algorithm and evaluates performance using two test scenarios by dividing the *dataset* into three ratios of training data and test data. The output of the application is JSON-RPC which is integrated with the *website*. In testing the DNN model, using 80% training data and 20% test data, the DNN algorithm provides an accuracy of 0.997558%. Furthermore, system testing was carried out involving various scenarios to verify its functionality, including input validation, data extraction, DNN prediction, and display of prediction results, which gave good results from the system created. The application has succeeded in identifying *scam* tokens with high accuracy. , increasing user security in crypto transactions.

<http://sistemasi.ftik.unisi.ac.id>

Keywords: *Blockchain, Ethereum, Fraud, Scam, Deep Learning.*

1 Pendahuluan

Dalam beberapa tahun terakhir, penggunaan teknologi *blockchain* telah meningkat pesat, adalah metode penyimpanan data dalam bentuk blok yang tidak dapat diubah yang menggunakan skema *hashing*[1]. *Blockchain* beroperasi secara terdesentralisasi, terdiri dari banyak *node* yang saling terhubung dengan tingkat transparansi tinggi. Data transaksi dalam *blockchain* disimpan dalam bentuk blok yang membentuk suatu rantai. Data transaksi yang sudah tersimpan dalam blok bersifat konsisten yang artinya data tersebut tidak dapat dihapus maupun diubah. Bitcoin merupakan salah satu jenis *cryptocurrency* yang paling dikenal, karena menggunakan teknologi kriptografi yang canggih dan memiliki sifat yang terdesentralisasi. Selain Bitcoin, terdapat banyak *cryptocurrency* lain yang dibangun dengan protokol *blockchain*-nya masing-masing. Salah satu contohnya adalah Ethereum, yang menjadi fokus penelitian ini. Ethereum adalah sebuah *platform open-source blockchain* dengan kapitalisasi pasar kedua setelah Bitcoin. Ethereum awalnya dibuat pada tahun 2015, dan sebagian besar penelitian dan sumber daya produksi untuk aplikasi terdesentralisasi (DApps) masih dalam tahap awal. Ethereum memungkinkan semua orang untuk membuat kontrak pintar (*smart contract*) [2]. Kontrak pintar merupakan suatu kode program yang di *compile* menjadi sebuah *bytecode* dalam *Ethereum Virtual Machine* (EVM) dan disimpan pada sebuah *block*.

Perkembangan ilmu dalam bidang teknologi dan komunikasi telah memberikan banyak dampak positif, namun juga tidak terlepas dari berbagai dampak negatif diantaranya merupakan kasus penipuan. Penipuan dalam transaksi *cryptocurrency* menjadi sebuah masalah serius yang perlu ditangani. Menurut laporan Lembaga Perlindungan Konsumen AS (FTC), pada periode 1 Januari 2021 sampai 31 Maret 2022 lebih dari 46.000 orang melaporkan kehilangan lebih dari \$1 miliar dalam bentuk mata uang kripto akibat penipuan [3]. Serangan *Cryptocurrency* dapat terjadi dalam berbagai bentuk. “*High Yield Investment Programs*” (HYIP) adalah contoh umum dari serangan pada *cryptocurrency*. Para penipu menjamin tingkat bunga yang besar bagi investor, seperti 1% hingga 2% setiap hari atau lebih pada penipuan dengan metode HYIP ini [4]. Dengan adanya Aplikasi Deteksi Token *Scam* menggunakan algoritma *Deep Neural Network* ini diharapkan para investor dan pelaku trader token kripto dapat terhindar dari penipuan dan menciptakan transaksi finansial yang aman. Selanjutnya diharapkan hasil dari penelitian ini akan berpengaruh terhadap kemajuan teknologi *blockchain* di Indonesia.

2 Tinjauan Literatur

Serangan yang umum dikenal dengan istilah “*Bitcoin Generator Scam*” (BGS) merupakan jenis penipuan yang menjanjikan *cryptocurrency* secara gratis kepada para calon korban untuk biaya penambangan kecil. Pada penelitian ini menggunakan pendekatan berbasis data serta metode klasifikasi untuk menemukan, memantau, dan menilai sebuah *scam* pada *bitcoin* generator [5]. Penerapan algoritma pembelajaran mesin digunakan untuk menentukan apakah sebuah akun *blockchain* masuk kedalam label berbahaya atau tidak dengan memeriksa kesesuaian antara nama domain (*Domain Name*) yang digunakan oleh pemilik akun *blockchain*. Sebanyak 73.769 DN *blockchain* yang menunjukkan perilaku berbahaya setidaknya sekali, dengan akurasi pemodelan sebesar 89.53% [6]. Salah satu jenis penipuan yang cukup banyak terjadi dalam dunia *crypto* adalah *wash trade*, merupakan kegiatan ilegal oleh satu atau banyak pihak, yaitu ketika pembeli serta penjual dalam suatu transaksi koleksi digital dilakukan oleh orang atau pihak yang sama. Kegiatan ini dianggap dapat memanipulasi dan menyesatkan kondisi pasar sebenarnya. Metode pemodelan *graph* perdagangan token diikuti oleh pengalamanan status sebuah akun mampu mendeteksi apakah sebuah token diperdagangkan dengan cara *wash trade* atau tidak. Pengujian menunjukkan bahwa tingkat *wash trade* melebihi 15% untuk sebagian besar mata uang kripto ERC20[7]. Metode algoritma XGBoost untuk klasifikasi akun berbahaya dari *cryptocurrency* ethereum dengan melihat sejarah transaksi dengan kombinasi 2502 akun yang normal yang telah terlibat dalam kegiatan ilegal, memperoleh akurasi rata-rata 0,963 (0,006) dan AUC rata-rata 0,994 (0,0007) dengan validasi silang 10 kali lipat[8].

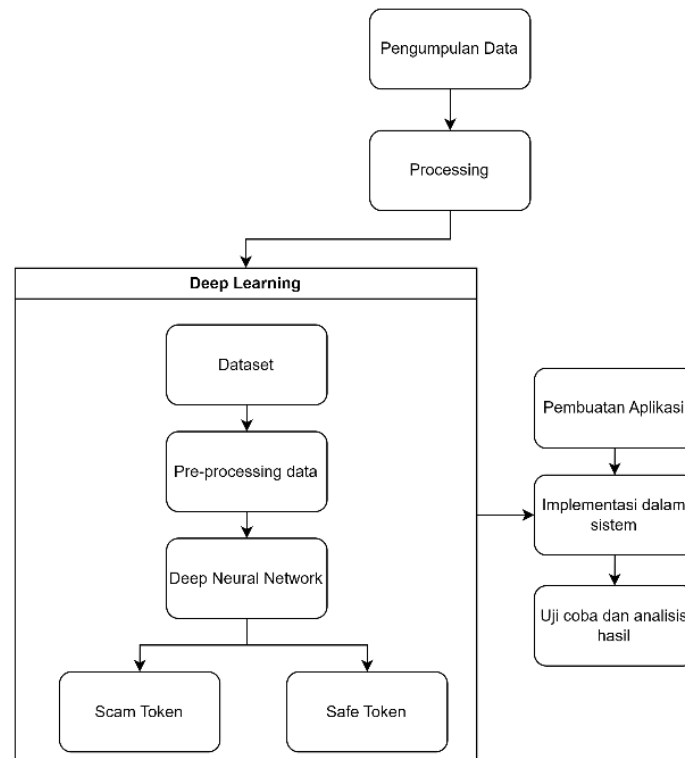
Berdasarkan penelitian terdahulu deteksi token *scam* pada *blockchain* telah dilakukan dengan berbagai metode, sehingga pada penelitian ini aplikasi deteksi *scam* akan menerapkan algoritma *deep*

<http://sistemasi.ftik.unisi.ac.id>

neural network untuk klasifikasi berdasarkan atribut dari set data berupa data token yang berhasil dikumpulkan dari proses *extractingblock* pada *blockchain* Ethereum.

3 Metode Penelitian

Adapun langkah gambaran umum dalam melaksanakan penelitian ini dapat dilihat pada Gambar 1 berikut :



Gambar 1. Langkah Penelitian

Berdasarkan gambar 1, tahap pertama yang dilakukan adalah mengumpulkan data yang akan dianalisa, data mengalami tahap *processing* data, menggunakan metode *deepneuralnetwork* pada deteksi penipuan token kripto. Hasil pemodelan pelatihan data kemudian akan diimplementasikan pada pembuatan aplikasi, pada tahap akhir adalah uji aplikasi deteksi token *scam*.

3.1 Pengumpulan Data

Pada tahap ini dilakukan pengumpulan data yang relevan untuk digunakan pada klasifikasi token *scam* dengan metode *deep neural network*. Data tersebut berupa data token yang berhasil dikumpulkan dari proses *extracting block* pada *blockchain* Ethereum, dengan atribut yaitu:

1. *TokenAddress*: Alamat token.
2. *PairAddress*: Alamat *pair*, merupakan alamat perdagangan yang dibuka oleh pembuat token atau biasa disebut *liquidity pool*.
3. *CreatorAddress*: Alamat dari pembuat token.
4. *TokenOwnershipPercentage*: Persentase kepemilikan token oleh pembuat token.
5. *PairOwnershipPercentage*: Persentase kepemilikan pasangan perdagangan oleh pembuat token.
6. *TotalFee*: Jumlah total persentase *fee* dari token yang dibebankan kepada pengguna saat melakukan *trading*.
7. *SmartContractCodeHash*: Kode program *smart contract* pada token yang di konversi kedalam bentuk *hash*.
8. *CountSimilarContractCode*: Jumlah token dengan *source code* yang sama.
9. *CountSimilarScamContractCode*: Jumlah token *scam* dengan *source code* yang sama.

Data tersebut disaring kembali dan dipilih hanya mencakup fitur-fitur berikut: *CreatorAddress*, *TokenOwnershipPercentage*, *PairOwnershipPercentage*, *TotalFee*, *SmartContractCodeHash*, *CountSimilarContractCode* dan *CountSimilarScamContractCode*. Contoh sebuah data set seperti ditunjukkan di Tabel 1 berikut:

Tabel 1. Contoh Data Set

Fitur	Nilai
CreatorAddress	0xb8f226ddb7bc672e27dff67e4adabfa8c0dfa08
TokenOwnershipPercentage	0
PairOwnershipPercentage	0
TotalFee	1
SmartContractCodeHash	0xd0caa0f9bc744c523933d44e6d8d07f868803d10bf16c8129e12f670296175ac
CountSimilarContractCode	493
CountSimilarScamContractCode	0

3.2 Preprocessing Data

Sebelum dilakukan penerapan model DNN pada data, dilakukan *preprocessing* atau serangkaian langkah yang digunakan untuk mempersiapkan data mentah menjadi bentuk yang lebih sesuai dan dapat digunakan secara efektif oleh model atau algoritma yang digunakan. Oleh karena itu, akan dilakukan tahapan *EncodingCategoricalData* atau pengkodean data kategori untuk mengubah data kategori menjadi format integer sehingga data dengan nilai kategori yang dikonversi dapat diberikan ke model yang berbeda [9]. *CreatorAddress* dan *SmartContractCodeHash* dikonversi dari format string hexadesimal menjadi bilangan desimal. Pada tahap ini juga terjadi proses penghapusan data yang tidak valid dan tidak relevan dan melakukan proses *labelling* “*Safe*” atau “*Scam*” terhadap *dataset* yang sudah dikumpulkan. Pada Tabel 2 merupakan contoh sampel data yang sudah dilakukan *preprocessing* dan pelabelan.

Tabel 2. Proses Preprocessing dan Labelling Token

<i>CreatorAddress</i>	X1	X2	Y	Z	<i>Safe</i>	<i>Scam</i>
854506163767955309106753415133821742388812760270	0	0	1	0	1	0
536874681163468227221112583237482569129145681369	51.97	0	100	6	0	1

3.3 Split Data

Tahap selanjutnya melakukan *split data* atau pemisahan data pada *dataset* menjadi dua *subset* yang terpisah, yaitu data latih (*training data*) dan data pengujian (*testing data*). Percobaan menggunakan rasio perbandingan 90:10, 80:20, dan 70:30 untuk data latih dan data pengujian.

3.4 Pembangunan Model DNN

Pada tahap ini, dilakukan pembangunan model *Deep Neural Network* (DNN) yang akan digunakan untuk mendeteksi token *scam*. Model DNN merupakan bagian inti dari sistem deteksi penipuan yang bertujuan untuk mengklasifikasikan token sebagai *scam* atau tidak berdasarkan fitur-fitur yang ada. Model DNN dilatih dan divalidasi menggunakan dataset yang telah disiapkan. Hal ini bertujuan untuk mengukur kinerja model pada data yang belum pernah ada dan memverifikasi kemampuan model dalam mendeteksi tindak penipuan.

3.4 Pelatihan Model

Proses pelatihan menggunakan akan menerapkan metode pembelajaran *supervised data* dengan menggunakan *binary cross-entropy* sebagai fungsi *loss* dan *optimizer Adam*[10]. *Early stopping* digunakan untuk mencegah *overfitting* dan menghentikan pelatihan ketika tidak ada peningkatan performa pada data validasi.

3.5 Evaluasi Model

Evaluasi dilakukan dengan menghitung matrik akurasi, presisi, *recall*, dan *F1-score*[11]. Selanjutnya, nilai – nilai tersebut akan ditampilkan pada *Confusion matrix* yang digunakan untuk mengevaluasi kinerja model terhadap validasi data, terdapat empat nilai yang dihasilkan tabel

confusionmatrix, yaitu *True Positive* (TP), *False Positive* (FP), *False Negative* (FN), dan *True Negative* (TN).

1. Akurasi

Nilai akurasi didapatkan dari jumlah data bernilai positif yang diprediksi sebagai positif dan data bernilai negatif yang diprediksi sebagai negatif dibagi dengan jumlah seluruh data dalam dataset. Rumus menghitung akurasi seperti pada rumus 1 berikut :

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (1)$$

2. Presisi

Presisi merupakan pembagian dari jumlah total contoh positif yang diklasifikasikan bernilai benar dengan jumlah total contoh positif yang diprediksi. Rumus menghitung presisi seperti rumus 2 berikut:

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

3. Recall

Recall merupakan peluang kasus dengan kategori positif yang dengan tepat diprediksi positif. Rumus menghitung *recall* ditunjukkan pada rumus 3 berikut:

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

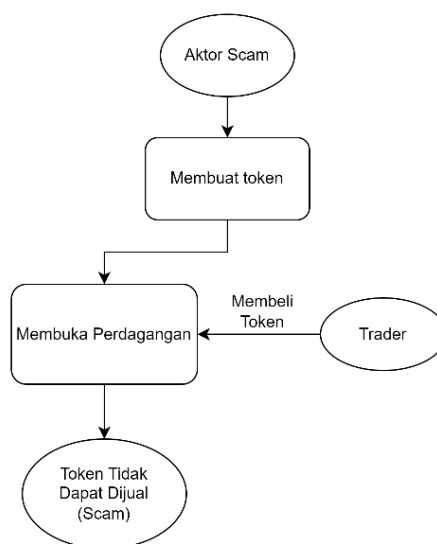
4. *F1-score*

Nilai *F1-score* atau dikenal juga dengan nama *F-measure* didapatkan dari hasil presisi dan *Recall* antara kategori hasil prediksi dengan kategori sebenarnya. Rumus menghitung *F1-score* ditunjukkan pada rumus 4 berikut :

$$F1 - score = \frac{2 \times Recall \times Precision}{Recall + Precision} \quad (4)$$

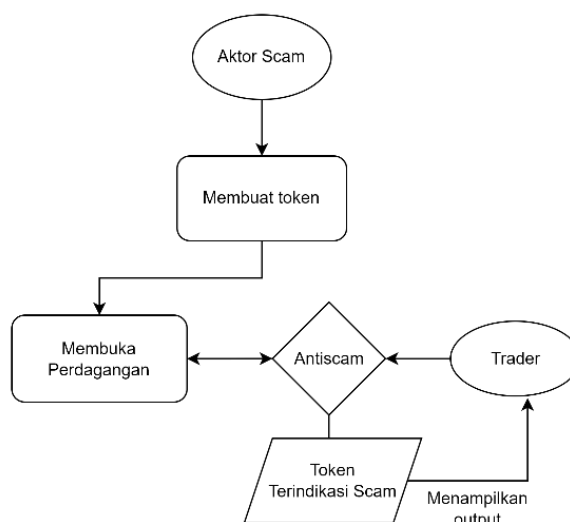
3.6 Perancangan Sistem

Pada tahap ini dilakukan analisis kebutuhan sistem terhadap bagaimana proses bisnis berjalan pada saat seorang *developer* membuat sebuah token *scam* yang diilustrasikan pada Gambar 2.



Gambar 2. Proses Bisnis Berjalan

Pada Gambar 2 *trader* melakukan pembelian token setelah aktor *scam* membuka perdagangan pada token, namun token yang dibeli tersebut tidak dapat dijual. Dari proses tersebut, kemudian dibuat usulan sebuah proses bisnis baru yang digambarkan pada Gambar 3 berikut:



Gambar 3. Rancangan Proses Bisnis Baru

Pada Gambar 3 *trader* terlebih dahulu menggunakan aplikasi deteksi *scam*, aplikasi akan melakukan proses pendeteksian dan simulasi untuk beli-jual pada token yang diperdagangkan untuk menentukan *scam* atau bukan, setelah itu aplikasi deteksi akan menampilkan *output*-nya kepada *trader*.

3.7 Pengujian Sistem

Pengujian ini bertujuan untuk memverifikasi kinerja dan fungsionalitas sistem dalam berbagai skenario, termasuk pengujian pada kasus input, ekstraksi data, prediksi dengan model DNN. Hasil pengujian akan membantu dalam memastikan bahwa sistem beroperasi dengan baik dan memberikan hasil prediksi yang akurat dan dapat diandalkan.

4 Hasil dan Pembahasan

Sistem deteksi *scam* token pada jaringan *blockchain* Ethereum dirancang menggunakan Bahasa pemrograman Golang yang dapat di implementasikan dengan program milik Ethereum Geth (*go-ethereum*), Bahasa Golang digunakan karena dapat membaca inputan angka yang sangat besar dan hasilnya cukup akurat, juga merupakan bahasa pemrograman sumber terbuka yang memudahkan pembuatan perangkat lunak yang sederhana, andal, dan efisien[12].

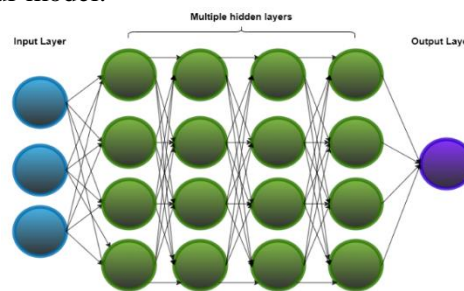
4.1 Dataset

Pada tahap *crawlingdataset*, penulis melakukan ekstraksi *block* untuk mendapatkan transaksi pembuatan *smart contract*. Setiap *block header* memiliki informasi berupa *timestamp*, *blockNumber*, *baseFeePerGas*, *difficulty*, *mixHash*, *parentHash*, *transactions*, *stateRoot* dan *nonce*. Proses ekstraksi dimulai dari nomor blok 0 sampai nomor blok 17564293 dengan *timestamp* Jul-30-2015 03:26:13 PM +UTC sampai Jun-26-2023 02:48:35 PM +UTC atau 2888 hari dengan bantuan aplikasi *geth* untuk proses ekstraksinya. Penulis mendapati 4.387.499 *smart contract* telah di dibuat, diantaranya terdapat 124.138 token yang masih bisa diperdagangkan. Token yang masih bisa diperdagangkan tersebut dijadikan sebagai dataset pada penelitian ini. Selanjutnya, pengelompokan data kedalam jenis *scam* berdasarkan beberapa indikator yaitu, persentase kepemilikan *creator* terhadap token, persentase kepemilikan *creator* terhadap *liquidity*, total *fee trading*, dan persamaan *source code* dengan token *scam* sebelumnya.

4.2 Arsitektur Model DNN

Arsitektur model Deep Neural Network (DNN) yang digunakan dalam penelitian ini terdiri dari empat lapisan, yaitu *Input Layer*, *Hidden Layers*, *Activation Function*, dan *Output Layer*. Pada *Input Layer*, terdapat 7 neuron yang mewakili fitur-fitur penting dari dataset, antara lain *CreatorAddress*, *TokenOwnershipPercentage*, *PairOwnershipPercentage*, *TotalFee*, *SmartContractCodeHash*, *CountSimilarContractCode*, dan *CountSimilarScamContractCode*. Fitur-fitur ini menjadi masukan awal bagi model untuk memproses informasi dan melakukan prediksi. Selanjutnya, terdapat 7

HiddenLayers pada arsitektur model DNN. *HiddenLayers* bertugas untuk memproses input dari *InputLayer* dengan menerapkan fungsi aktivasi ReLU (*Rectified Linear Unit*) pada setiap neuronnya. Fungsi aktivasi ReLU digunakan untuk mengaktifkan neuron[13]. Proses aktivasi pada setiap layer diatur oleh fungsi aktivasi ReLU, yang memastikan bahwa setiap neuron hanya akan mengirimkan output jika nilai inputnya lebih besar dari nol. Hal ini membantu model DNN dalam melakukan proses *non-linear mapping*, sehingga mampu menangkap pola dan kompleksitas data yang lebih tinggi. Pada *OutputLayer*, terdapat 1 neuron yang merepresentasikan dua kelas, yaitu "Safe" atau "Scam". Fungsi aktivasi *softmax* digunakan pada *OutputLayer* untuk menghasilkan probabilitas untuk setiap kelas. Dengan adanya fungsi *softmax*, model dapat memberikan prediksi dengan bentuk distribusi probabilitas, di mana probabilitas tertinggi akan menentukan klasifikasi akhir token sebagai "Safe" atau "Scam". Ilustrasi arsitektur pemodelan Deep Neural Network ini disajikan dalam Gambar 4 pada laporan penelitian. Gambar 4 di bawah ini menampilkan ilustrasi arsitektur model.



Gambar 4. Ilustrasi Arsitektur Model DNN

4.3 Hasil Evaluasi Performa

Hasil evaluasi performa dilakukan dengan dua skenario pengujian, *dataset* dibagi menjadi tiga rasio untuk data latih dan data uji dengan proporsi sebagai berikut : 90%:10%, 80%:20%, dan 70%:30%. Pada proses pelatihan, *callback EarlyStopping* digunakan untuk mencegah *overfitting* dan memperoleh model terbaik berdasarkan pengawasan *loss* pada data validasi[14]. Setelah melatih model dengan *epoch* sebanyak 20, model pada data uji di evaluasi dan mendapatkan hasil evaluasi performa yang dapat dilihat pada Tabel 3:

Tabel 3. Hasil Evaluasi Performa

Rasio	Confusion Matrix		Akurasi	Presisi	Recall	F1-Score
90:10	P (1)	N (0)	0.997103	0.995763	0.999875	0.997815
	4056	34				
	N (0)	1				
80:20	P (1)	N (0)	0.997558	0.996519	0.999813	0.998163
	8074	56				
	N (0)	3				
70:30	P (1)	N (0)	0.997351	0.996442	0.999585	0.998011
	12065	86				
	N (0)	10				

Pada Tabel 3, hasil evaluasi performa menunjukkan bahwa model yang dibuat mampu dengan sangat baik dalam mendeteksi penipuan dengan tingkat akurasi yang sangat tinggi, yaitu di atas 99%. Selain itu, nilai *precision*, *recall*, dan *F1-score* juga menunjukkan performa yang sangat baik, mendekati nilai 1. Hal ini mengindikasikan bahwa model yang dibuat memiliki kemampuan yang tinggi dalam mengklasifikasikan penipuan dengan sedikit kesalahan. Dengan demikian, hasil penelitian ini menunjukkan bahwa model DNN dengan arsitektur yang di implementasikan sangat efektif dalam mendeteksi penipuan berdasarkan fitur-fitur pada dataset yang digunakan.

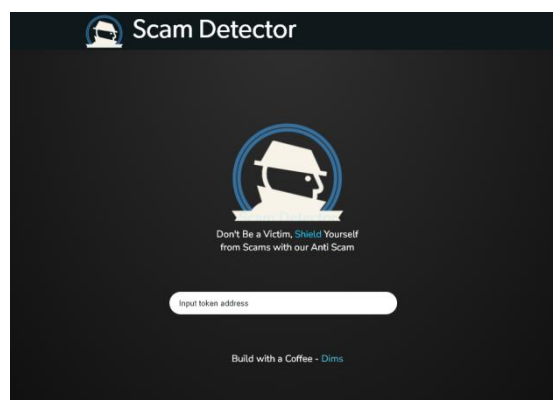
4.4 Implementasi Sistem dalam Aplikasi

JSON-RPC digunakan untuk menerima pemanggilan fungsi pada aplikasi deteksi *scam* yang sudah dibuat, JSON-RPC adalah protokol yang memungkinkan melakukan panggilan metode jarak jauh ke program lain berada di alamat yang berbeda menggunakan JSON sebagai pembungkus pesan[15]. Pemanggilan RPC perlu dilakukan agar dapat berinteraksi dengan aplikasi deteksi *scam*. Metode pemanggilan yang digunakan yaitu *eth_checkTokenIsScam* dengan parameter input alamat token *address* yang *outputnya* akan berupa tipe data JSON seperti pada Gambar 5.berikut:

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "result": {
    "TokenName": "SHIBA INU",
    "TokenSymbol": "SHIB",
    "TokenNameWithSymbol": "SHIBA INU (SHIB)",
    "TokenAddress": "0x95ad61b0a150d79219dcf64e1e6cc01f0b64c4ce",
    "PairAddress": "0x811beed0119b4afce20d2583eb608c6f7af1954f",
    "CreatorAddress": "0xb8f226ddb7bc672e27dff67e4adabfa8c03fa08",
    "CreatorHash": "0x0a4022e61c49c59b2538b78a6c7c9a0e4bb8c8fce2d1b4a725baef3c55fb7363",
    "TimeStampCreated": 1596220363,
    "TokenOwnershipPercentage": "0",
    "PairOwnershipPercentage": "0",
    "TotalFee": 1,
    "IsHoneyPot": false
  }
}
```

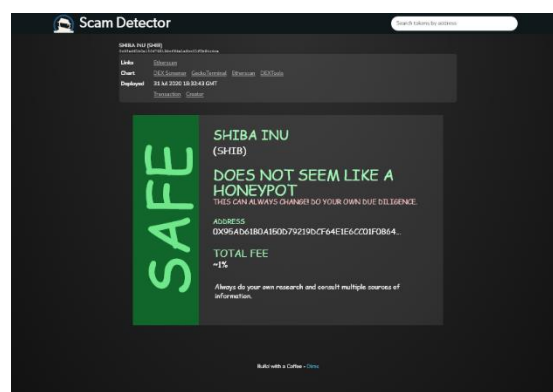
Gambar 5. Output Pemanggilan JSON-RPC

Pada tahap selanjutnya, dilakukan integrasi aplikasi yang sudah dibuat dalam bentuk *website* dengan tujuan agar mudah digunakan oleh *user*. *Website* yang dibangun menggunakan Bahasa PHP, untuk dapat mendeteksi token *scam* perlu menginputkan alamat token pada *website* dan akan menampilkan hasil dari aplikasi deteksi *scam*. Pada Gambar 6, Gambar 7, dan Gambar 8 merupakan tampilan aplikasi deteksi *scam* yang sudah terintegrasi dalam bentuk *website*.



Gambar 6. Halaman Depan Aplikasi Deteksi Scam

Pada Gambar 6 merupakan tampilan depan dari Aplikasi Deteksi Token Scam.



Gambar 7. Halaman Hasil Aplikasi Deteksi Scam (Safe)

Pada Gambar 7 merupakan tampilan dari contoh deteksi token yang termasuk kategori aman atau *safe*.



Gambar 8. Halaman Hasil Aplikasi Deteksi Scam (Scam)

Pada Gambar 8 merupakan tampilan dari contoh deteksi token yang termasuk kategori bahaya atau *scam*.

4.5Pengujian Sistem

Berdasarkan pengujian yang bertujuan untuk melihat fungsional dari sistem, ditunjukkan pada Tabel 4, untuk menguji input yang dimasukkan kedalam aplikasi. Sementara pada Tabel 5, proses ekstraksi data berhasil dilakukan pada aplikasi. Proses prediksi token apakah termasuk kedalam *scam* atau *safe* ditunjukkan pada Tabel 6, dengan memberikan contoh dataset token *scam* dan *safe*. Hal ini bertujuan untuk memverifikasi kinerja dan fungsionalitas sistem dalam berbagai skenario, termasuk pengujian pada kasus input, ekstraksi data, prediksi dengan model DNN. Hasil pengujian akan membantu dalam memastikan bahwa sistem beroperasi dengan baik dan memberikan hasil prediksi yang akurat dan dapat diandalkan dalam berbagai situasi penggunaan.

Tabel 4. Pengujian Input


No.	Pengujian	Input	Hasil	Keterangan
1	Input Alamat Token Valid	Alamat token valid	Sukses	Aplikasi menerima alamat token yang valid dan menampilkan hover alamat token, nama dan simbol token.
2	Input Alamat Token Tidak Valid	Alamat token tidak valid	Gagal	Aplikasi tidak dapat mengenali alamat token yang tidak valid dan menampilkan hover "no result".

Tabel 5. Hasil Uji Ekstraksi Data

No.	Pengujian	Input	Hasil	Keterangan
1	Ekstraksi Data Berhasil	Alamat token valid	Sukses	Sistem berhasil mengambil data dari <i>database</i> berdasarkan alamat token.
2	Ekstraksi Data Gagal	Alamat token tidak valid	Gagal	Sistem gagal mengambil data dari <i>database</i> karena alamat token tidak valid.

Tabel 6. Hasil Uji Prediksi dengan Model DNN

No.	Pengujian	Input	Hasil	Keterangan
1	Prediksi Scam	Dataset Token Scam 0xe053f7b5 513acab661 c50a279cc8 6dccc91b4b 9e	Prediksi Scam 	Model DNN memberikan hasil prediksi <i>scam</i> yang akurat.
2	Prediksi Safe	Dataset Token Safe 0x95aD61b0 a150d79219 dCF64E1E6	Prediksi Safe	Model DNN memberikan hasil prediksi <i>safe</i> yang akurat.

		Cc01f0B64 C4cE		
--	--	-------------------	---	--

5 Kesimpulan

Dari hasil dan pembahasan sebelumnya diperoleh sebuah kesimpulan bahwa metode DNN terbukti sangat efektif dalam mengklasifikasikan token *scam* berdasarkan fitur-fitur yang ada pada data token. Model DNN yang dibangun mencapai tingkat akurasi yang sangat tinggi dengan menggunakan 80% data latih dan 20% data uji, yaitu di atas 99%, dan memiliki nilai presisi, *recall*, dan *F1-score* yang mendekati nilai 1. Aplikasi deteksi *scam* yang dikembangkan telah diuji coba dan berhasil mengidentifikasi token *scam* dengan akurasi yang tinggi. Hal ini memungkinkan pengguna untuk melakukan *trading* dengan aman. Saran untuk pengembangan penelitian ini meliputi pengujian dan evaluasi yang komprehensif terhadap sistem deteksi penipuan yang dibangun, pengembangan mekanisme keamanan tambahan, dan eksplorasi implementasi sistem deteksi penipuan pada platform *blockchain* lainnya selain Ethereum.

Referensi

- [1] R. M. Jaya, V. D. Rakkhitta, P. Sembiring, I. S. Edbert, and D. Suhartono, "Blockchain applications in drug data records," *Procedia Comput. Sci.*, vol. 216, pp. 739–748, 2023, doi: 10.1016/j.procs.2022.12.191.
- [2] N. F. Samreen and M. H. Alalfi, "An empirical study on the complexity, security and maintainability of Ethereum-based decentralized applications (DApps)," *Blockchain Res. Appl.*, vol. 4, no. 2, 2023, doi: 10.1016/j.bcra.2022.100120.
- [3] F. T. Commission, "Remarks of Chair Lina M. Khan Regarding the 6(b) Orders Concerning Deceptive Advertising on Social Media - March 16, 2022," 2022.
- [4] K. Toyoda, P. Takis Mathiopoulous, and T. Ohtsuki, "A Novel Methodology for HYIP Operators' Bitcoin Addresses Identification," *IEEE Access*, vol. 7, pp. 74835–74848, 2019, doi: 10.1109/ACCESS.2019.2921087.
- [5] E. Badawi, G. Jourdan, and I. Onut, "Blockchain : Research and Applications The ' Bitcoin Generator ' Scam," *Blockchain Res. Appl.*, vol. 3, no. 3, p. 100084, 2022, [Online]. Available: <https://doi.org/10.1016/j.bcra.2022.100084>.
- [6] R. K. Sachan, R. Agarwal, and S. K. Shukla, "Identifying malicious accounts in blockchains using domain names and associated temporal properties," *Blockchain Res. Appl.*, p. 100136, 2023, doi: 10.1016/j.bcra.2023.100136.
- [7] W. Cui and C. Gao, "WTEYE: On-chain wash trade detection and quantification for ERC20 cryptocurrencies," *Blockchain Res. Appl.*, vol. 4, no. 1, p. 100108, 2023, doi: 10.1016/j.bcra.2022.100108.
- [8] S. Farrugia, J. Ellul, and G. Azzopardi, "Detection of illicit accounts over the Ethereum blockchain," *Expert Syst. Appl.*, vol. 150, p. 113318, 2020, doi: 10.1016/j.eswa.2020.113318.
- [9] Yugesh Verma, "A Complete Guide to Categorical Data Encoding," 2021. <https://analyticsindiamag.com/a-complete-guide-to-categorical-data-encoding/> (accessed Jul. 21, 2023).
- [10] S. Kohli, "Understanding a Classification Report For Your Machine Learning Model," <https://medium.com/>, 2019. <https://medium.com/@kohlishivam5522/understanding-a-classification-report-for-your-machine-learning-model-88815e2ce397>.
- [11] P. Palinggik Allorerung, J. K. Perintis Kemerdekaan, K. Tamalanrea, K. Makassar, and S. Selatan, "SISTEMASI: Jurnal Sistem Informasi Analisis Sentimen Pada Ulasan Aplikasi WeTV di Google Play Store Menggunakan Algoritma NBC dan SVM Sentiment Analysis on WeTV App Reviews on Google Play Store Using NBC and SVM Algorithms," vol. 12, no. 2, pp. 2540–9719, 2023, [Online]. Available: <http://sistemasi.ftik.unisi.ac.id>.
- [12] C. Yuan, J. Du, M. Yue, and T. Ma, "The design of large scale IP address and port scanning tool," *Sensors (Switzerland)*, vol. 20, no. 16, pp. 1–12, 2020, doi: 10.3390/s20164423.
- [13] S. SHARMA, "Activation Functions in Neural Networks Sigmoid, tanh, Softmax, ReLU,

<http://sistemasi.ftik.unisi.ac.id>

- Leaky ReLU EXPLAINED !!!,” <https://towardsdatascience.com>, 2017.
<https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>.
- [14] X. Ying, “An Overview of Overfitting and its Solutions,” *J. Phys. Conf. Ser.*, vol. 1168, no. 2, 2019, doi: 10.1088/1742-6596/1168/2/022022.
- [15] M. R. Iswardhana and S. Widiono, *Diplomasi Siber dan Teknologi Mobile Pada Multi disiplin*. 2021.