

## DETEKSI SKEMA PONZI PADA BLOCKCHAIN MENGGUNAKAN ALGORITMA ADABOOST

Rizfi Syarif<sup>1)</sup>, Andi Sunyoto<sup>2)</sup>, Asro Nasiri<sup>3)</sup>

Magister Teknik Informatika, Universitas Amikom Yogyakarta

Email: <sup>1</sup>rizfi.1268@students.amikom.ac.id, <sup>2</sup>andi@amikom.ac.id, <sup>3</sup>asro@amikom.ac.id

### Abstrak

*Ponzi adalah skema penipuan yang sudah ada sejak 150 tahun lalu. Sekarang berkembang cara penawarannya dengan teknologi yang modern. Salah satu teknologi yang digunakan adalah cryptocurrency. Ethereum sebagai blockchain platform yang juga mengakomodasi cryptocurrency menjadi ajang beredarnya penipuan ponzi. Tidak sedikit nilai finansial yang menjadi kerugian pengguna Ethereum. Ponzi yang beredar sangat diuntungkan dengan adanya smart contract. Karena proses transaksi yang bersifat rahasia dan tanpa adanya campur tangan pihak ketiga. Sekali transaksi ponzi dilakukan tidak ada cara untuk mengembalikan ke pengirimnya. Penelitian yang ada mempunyai tingkat akurasi yang bagus tetapi kebanyakan harus berdasarkan data transaksi. Ini artinya deteksi ponzi baru bisa dilakukan jika adanya transaksi. Dengan kata lain sudah ada korban yang melakukan transaksi. Hal tersebut butuh adanya penelitian deteksi ponzi sebelum jatuh korban. Pada penelitian ini dikemukakan proses yang lebih cepat untuk pendeteksian skema ponzi pada Ethereum. Dengan menggunakan Adaboost dalam klasifikasi bytecode yang diubah menjadi matrik bobot dari algoritma word embedding. Ditemukan hasil paling maksimal adalah F1 Score sebesar 85%.*

**Keywords:** *Ethereum, smart contract, bytecode, word embedding, cryptocurrency*

## PENDAHULUAN

### 1.1 Latar Belakang

Lebih dari satu dekade popularitas cryptocurrency meningkat tajam, misalnya adalah Ethereum dan Bitcoin terutama pada ranah keuangan. Hal ini tidak lepas dari semakin meluasnya teknologi *blockchain* karena tingkat keamanan yang sangat tinggi. Selain itu *blockchain* juga sangat menjaga rahasia pribadi pengguna. Tidak hanya ranah keuangan, *blockchain* juga semakin banyak digunakan di ranah keamanan, transaksi, komputasi dan penyimpanan data. Jika dibandingkan dengan sistem keuangan konvensional, maka *blockchain* sangat diunggulkan karena data yang terdesentralisasi sehingga bisa fokus untuk pengamanan data dan tidak bisa diubah[1].

Salah satu fitur yang disediakan oleh beberapa *blockchain* platform adalah *smart*

*contract*. Hal ini membuat transaksi antar pengguna semakin aman dan transparan. *Smart contract* ini dibuat menggunakan bahasa pemrograman kemudian di-compile oleh *blockchain* platform menjadi *bytecode*. *Bytecode* tersebut disebar ke node seluruh dunia.

Simulasi dari penggunaan *smart contract* ini adalah jika pembeli suatu barang secara online, kemudian barang sudah sampai di lokasi tujuan. Maka *smart contract* akan tereksekusi mengambil deposit cryptocurrency pembeli dan mengirimkannya kepada penjual. Mekanisme ini akan semakin cepat dan efisien, jika dibandingkan dengan mekanisme konvensional. Secara konvensional, kadang membutuhkan waktu lebih serta melibatkan banyak pihak untuk transfer cryptocurrency tersebut[2].

Karena tingkat kerumitan pada *smart contract* yang tidak semua pengguna paham, mengakibatkan sebagian besar pengguna

mendapatkan informasi operasional *smart contract* hanya berupa penjelasan dari pembuat *smart contract* atau orang lain. Kondisi ini dimanfaatkan oleh orang yang tidak bertanggung jawab untuk melakukan penipuan. Salah satu penipuan yang adalah skema ponzi.

Terlebih lagi kelebihan dari *smart contract* yang tidak perlu pihak ketiga untuk perantara dan verifikasi, banyak digunakan untuk penipuan[3]. Antara tahun 2015 sampai dengan 2017 ditemukan 191 ponzi yang memanfaatkan *smart contract* pada Ethereum. Mengakibatkan kerugian sebesar \$500.000 berasal lebih dari 2000 pengguna Ethereum[4].

## 1.2 Penelitian Sebelumnya

Karena modus dari ponzi semakin banyak dan menyamar pada ajakan investasi, maka para peneliti harus mencari cara pendeteksian ponzi yang lebih efisien dan otomatis. Beberapa penelitian terkait pendeteksian skema ponzi antara lain Liang dkk [5] menggunakan *bytecode* dari *smart contract* dan transaksi sebagai dataset. Dari data yang ada dibuatkan graph yang menghubungkan antar *account* serta *opcode* hasil konversi dari *bytecode*. Graph yang sudah jadi dianalisa menggunakan Long Short Term Memory sebagai training deteksi ponzi. Yu dkk [6] mengambil dataset transaksi *Ethereum* untuk dijadikan data training dan uji. Feature yang telah diambil kemudian dianalisa menggunakan Graph Convolutional Network (GCN) untuk membedakan transaksi ponzi atau bukan.

Chen dkk [7] menggunakan source code *smart contract* untuk menjadi obyek penelitian. Source code yang didapat akan diurai dalam bentuk Abstract Syntax Tree (AST). Kemudian bentuk AST dikonversi menjadi urutan Structure-Based Traversal (SBT). Matrix dari bentuk SBT dimasukkan pada convolutional layer.

Penelitian Wang dkk [8] menggunakan *bytecode* dan informasi transaksi dari *account* terkait. Untuk *bytecode* dikonversi menjadi *opcode*. Karena dataset bersifat imbalance, maka dilakukan oversampling menggunakan SMOTE. Feature yang digunakan adalah sebanyak 83 buah, yang berasal dari *account* feature dan code feature.

Metode trainingnya menggunakan Long Short Term Memory.

Fan dkk [9] menggunakan *smart contract* berupa *bytecode* sebagai dataset. Konversi *bytecode* menjadi *opcode* menggunakan Pyevmasm library. Kemudian *opcode* dikonversi menjadi unigram. Pada proses pengurutan kata pada unigram ditentukan juga ‘kata henti’ dari *smart contract*. Pada unigram dihasilkan 1-gram, 2-gram, 3-gram dan 4-gram kemudian dikonversi menjadi vector berdasarkan bag of words. ANOVA F-value digunakan untuk memilih feature yang paling berpengaruh untuk klasifikasi. Untuk mengatasi data imbalance, maka digunakan borderline-SMOTE 2 untuk menghilangkan kekurangan tersebut.

Dari penelitian di atas mempunyai kesamaan antara lain dataset yang mengandung transaksi *Ethereum*, sehingga pembelajaran model bisa dilakukan jika ada transaksi. Artinya sudah ada korban ponzi sebelum *smart contract* tersebut dianggap ponzi atau tidak. Kesamaan yang lain adalah adanya konversi *bytecode* menjadi *opcode*, hal ini menambah panjang tahapan pendeteksian. Oleh karena itu perlu adanya penelitian untuk mendeteksi skema ponzi *smart contract Ethereum* yang lebih cepat.

## 1.3 Landasan Teori

### 1.3.1 Ponzi

Skema ponzi sudah ada sejak lama, yaitu sekitar 150 tahun yang lalu. Tercatat bahwa skema ponzi yang pertama kali berada di kota Boston pada tahun 1920. Skema ini biasanya adalah pembayaran laba kepada investor berasal dari pengumpulan dana dari investor baru. Sampai pada saat perputaran dana habis dikarenakan tidak ada investor baru yang mau bergabung. Maka ponzi tersebut akan bubar. Hal ini merugikan banyak pihak, terutama investor baru yang belum mendapat pembayaran laba.

Meskipun di berbagai negara sudah dianggap sebagai penipuan, tetapi skema ini masih bermunculan dengan media yang baru. Terutama pada media website yang sulit dibatasi oleh satu negara saja[10].

Skema ponzi pada *blockchain* tidak berbeda dengan skema ponzi konvensional (non-cryptocurrency). Konsep utamanya adalah investor yang telah mengirimkan ether ke *smart contract* akan mendapatkan keuntungan jika ada

investor baru yang masuk. Pola alur ponzi terdapat empat bentuk yaitu piramida berbasis array, piramida berbasis pohon, handover dan air terjun[**Error! Bookmark not defined.**].

### 1.3.2 Ethereum

Ethereum adalah *blockchain* platform yang berbasis *opensource*, sistem yang terdesentralisasi dengan fitur unggulan *smart contract*. Termasuk cryptocurrency dengan nama mata uang Ether. Dibuat oleh Vitalik Buterin pada tahun 2013, dan berjalan online pada 2015.

Berbeda dengan Bitcoin yang fokus pada pembayaran cryptocurrency, Ethereum bisa dijadikan platform dasar untuk membangun aplikasi desentralisasi yang lain. Misalnya adalah pembelian tiket konser atau marketplace, sehingga Ethereum diintegrasikan dengan aplikasi custom yang dibuat oleh pengembang lain[11].

### 1.3.3 Smart Contract

Smart contract adalah program yang bisa mengeksekusi dirinya sendiri ketika ada kondisi pemicu yang telah diatur. Disimpan secara publik pada semua node *Ethereum* sehingga tidak bisa diubah setelah diupload. Sebagai contohnya adalah ketika sebuah fungsi yang mengirim pesan kepada akun tertentu akan mengirimkan Ether pada akun lain jika kondisi logikanya terpenuhi[12].

### 1.3.4 AdaBoost

Adaboost atau Adaptive Boost merupakan algoritma learning yang dibuat oleh Freund dan Schapire pada tahun 1996. Dasar dari algoritma ini adalah meningkatkan kinerja learner dengan cara mengganti secara berulang learner yang lemah. Algoritma ini bisa digunakan untuk klasifikasi dan regresi.

Estimator yang terbaik untuk beberapa kasus adalah Tree. Sehingga parameter untuk proses training perlu ditentukan berapa estimator yang diperlukan. Selain itu perlu juga learning rate, jika ditentukan 0 maka tidak belajar sama sekali dan jika 1 maka dasar learning sebelumnya dipakai seluruhnya[13,14].

### 1.3.5 Word Embedding

Word embedding adalah metode yang mengubah urutan kata dalam kalimat menjadi vektor berdasarkan kedekatan dengan kata yang

terdekat. Hal ini akan bisa membedakan suatu kata yang disambung dengan kata lain secara semantik. Vektor yang tersusun berupa matrik yang setiap elemennya mempunyai bobot. Pada hasil vektornya akan bisa diketahui kata yang paling banyak korelasinya[15].

## METODE PENELITIAN

### 1.4 Metode Pengumpulan Data

Dataset yang digunakan adalah sama yang digunakan oleh Chen dkk [16]. Dataset tersebut berisi alamat *smart contract* Ethereum dan label. Label berisi 0 yang menandakan bahwa *smart contract* tersebut bukan ponzi. Data berisi 1 yang menandakan bahwa *smart contract* tersebut merupakan ponzi.

### 1.5 Variabel Penelitian

Variabel penelitian adalah *smart contract* yang berupa data *bytecode*. Data *bytecode* berisi program yang bisa dieksekusi oleh Ethereum Virtual Machine. Program tersebut berupa angka-angka heksadesimal. Bytecode tersebut merupakan rangkaian perintah dan data yang akan dieksekusi.

### 1.6 Langkah Analisis

Langkah-langkah analisis dalam penelitian ini diuraikan sebagai berikut:

- a. Mengambil data berupa alamat *smart contract* dan label dari Chen dkk, belum termasuk *bytecode*.
- b. Mengambil *bytecode* dari BigQuery Google dengan id tabel `bigquery-public-data:crypto_Ethereum.contracts` berdasarkan alamat *smart contract* yang didapat sebelumnya. Data yang didapat dikonversi menjadi CSV.
- c. Hasil data csv dimasukkan pada program Jupyter Notebook. Dataset *bytecode* dipecah menjadi rangkaian 2 karakter dan dipisah dengan spasi. Dan dilakukan *tokenize* untuk pembobotan word2vec dan glove. Untuk TF-IDF tidak menggunakan data *tokenize*.

- d. Selanjutnya dinilai pembobotan menggunakan Word2Vec, Glove, TF-IDF.
- e. Hasil pembobotan nilai beserta label ponzi diklasifikasikan menggunakan algoritma Adaboost.
- f. Melakukan pengujian akurasi.
- g. Menarik kesimpulan dari proses yang telah dilakukan.

## 2 HASIL DAN PEMBAHASAN

### 3.1 Pemrosesan Data

Data yang telah didapat dari penelitian sebelumnya berupa alamat *smart contract* dan label belum termasuk *bytecode*. Jumlah data sebesar 3793 terdiri dari 3590 alamat *smart contract* yang tidak mengandung ponzi serta 200 mengandung ponzi. Adapun 3 data mempunyai label error.

Berdasarkan data dari alamat *smart contract* tersebut, diambil data *bytecode smart contract* dari database BigQuery milik Google. Nama tabel yang dirujuk mempunyai ID bigquery-public-data:crypto\_Ethereum.contracts. Didapatkan data lengkap dari *smart contract* termasuk *bytecode*. Data yang didapat sebesar 2859 terdiri dari 2728 *smart contract* tanpa ponzi dan 131 mengandung ponzi. Hasilnya dikonversi menjadi format CSV. Jumlah data *bytecode* tidak sama dengan alamat *smart contract* yang didapat dari penelitian sebelumnya, hal ini dikarenakan beberapa *smart contract* mempunyai perintah menghancurkan dirinya sendiri sehingga tidak bisa diambil lagi.

**Tabel 1:** Contoh dataset berupa label dan *bytecode*

| label | Bytecode berupa heksadesimal                          |
|-------|---|
| 0     | 0x606060405236156100c35760003 ...<br>b6f297c8a40029   |
| 1     | 0x6060604052361561004a5760003 ...<br>cd0d83127ec60029 |
| 0     | 0x606060405236156100965763ffff ...<br>375a91b71eb0029 |

Dataset yang berupa CSV diupload pada aplikasi Jupyter Notebook untuk pengolahan data selanjutnya. Kolom selain *bytecode* dan label dihapus karena tidak diperlukan dalam proses penelitian. Pada kolom *bytecode*, data dipecah menjadi bagian-bagian kecil berupa dua karakter kemudian dipisahkan dengan spasi. Selanjutnya dilakukan *tokenize* pada data *bytecode* yang terpisah tadi.

Langkah selanjutnya adalah proses membuat *language model* menggunakan Word2Vec. Word2Vec akan menghitung bobot dari setiap gabungan dua karakter pada *bytecode*. Hasilnya adalah *matrik* bobot *language model* berdasarkan hubungan dua karakter *bytecode* dengan karakter sebelum dan sesudahnya. Hal tersebut juga dilakukan dalam proses pembuatan *language model* menggunakan Glove.

**Tabel 2:** Parameter yang digunakan untuk pembobotan Word2Vec dan Glove

| Parameter        | Word2Vec | Glove | TF-IDF |
|------------------|----------|-------|--------|
| Vector size      | 300      | 300   | -      |
| Window           | 5        | 5     | -      |
| Skip Gram        | 0        | -     | -      |
| Minimal Count    | 1        | -     | -      |
| Epoch            | 1000     | 1000  | -      |
| Maximal Features | -        | -     | 100000 |
| N-Gram Range     | -        | -     | 1-3    |

Sedangkan untuk pemrosesan TF-IDF data *bytecode* tidak perlu dilakukan *tokenize*. Cukup dengan data *bytecode* yang dipisah setiap dua karakter dengan spasi. Hal tersebut sudah cukup untuk proses pengujian.

Bytecode akan dinilai berdasarkan nilai bobot pada *language model* yang telah dibuat sebelumnya. Setiap dua karakter dari *bytecode* akan dinilai dan dimasukkan pada rangkaian *matrik*. Hasilnya adalah *matrik* penilaian *bytecode* menggunakan Word2Vec dan Glove. Sedangkan TF-IDF tidak melalui tahap ini karena langsung akan diproses pada pengujian.

### 3.2 Pengujian Data

1. Word2Vec  
Word2Vec akan menilai bobot setiap dua karakter dari *bytecode* kemudian menghasilkan *matrik language model*. Dengan *matrik* tersebut dilakukan pembobotan *bytecode* berdasarkan rata-rata posisi dengan karakter sebelum dan sesudahnya. Hasilnya juga berupa *matrik* hasil rata-rata bobot yang telah ditemukan. Dengan F1 score didapatkan hasil ujinya sebesar 68%
2. Glove  
Proses yang dilakukan adalah sama dengan Word2Vec. Adapun hasil pengujian menggunakan F1 Score mendapatkan hasil sebesar 44%
3. TF-IDF  
Setiap penggalan dua karakter akan dihitung seberapa banyak kemunculan dalam *bytecode* terkait. Hasil pembobotan oleh TF-IDF akan diklasifikasikan dengan AdaBoost. Pengujiannya menghasilkan nilai F1 Score 85%

**Tabel 3:** Hasil pengujian menggunakan AdaBoost

| Word Embedding | F1 Score |
|----------------|----------|
| Word2vec       | 68%      |
| Glove          | 44%      |
| TF-IDF         | 85%      |

### KESIMPULAN

Smart contract pada Ethereum berupa *bytecode* yang akan di-eksekusi pada EVM setiap node. Keuntungan dari *bytecode* ini adalah *smart contract* bisa dieksekusi lintas sistem operasi ataupun arsitektur komputer. Hal tersebut dikarenakan *bytecode* dibaca oleh Virtual Machine. Konsistensi data berupa karakter-karakter heksadesimal bisa menjadi keuntungan dalam pendeteksian ponzi. Dari penelitian-penelitian sebelumnya pendeteksian ponzi menggunakan menggunakan data *bytecode* dan data transaksi. Hal ini mengakibatkan *smart contract* tersebut dianggap ponzi jika sudah ada yang bertransaksi. Artinya sudah ada jatuh korban ponzi. Untuk penelitian yang hanya menggunakan *bytecode*, biasaya tahapnya sangat banyak. Sehingga butuh resource yang banyak untuk pendeteksian ponzi berdasar *bytecode* saja. Pada penelitian ini dilakukan pembuatan model berdasarkan *bytecode* saja. Tahap selanjutnya cukup *tokenize* ataupun cukup pemenggalan setiap dua karakter. Pembobotan token dan dua karakter dilakukan menggunakan Word2Vec, Glove dan TF-IDF. Kemudian hasil pembobotan diklasifikasikan menggunakan algoritma Adaboost dilanjutkan pengujian menggunakan F1 score. Hasil yang didapat adalah 68%, 44%, dan 85%. Hal ini membuktikan bahwa Adaboost bisa mengklasifikasikan *bytecode smart contract* yang mengandung penipuan ponzi atau tidak. Lebih dari itu, TF-IDF sebagai *word embedding* bisa paling unggul dalam pendeteksian.

- 1 Ferretti, Stefano, dan Gabriele D’Angelo. 2020. “On the Ethereum Blockchain Structure: A Complex Networks Theory Perspective.” *Concurrency and Computation: Practice and Experience* 32(12). doi: 10.1002/cpe.5493.
- 2 Was, Sanne, 2016, Landmark transaction merges *blockchain, smart contracts* and IoT, <https://www.gtreview.com/news/global/landmark-transaction-merges-blockchain-smart-contracts-and-iot/>, diakses tanggal 3 Januari 2022.
- 3 Tsankov, Petar, Andrei Dan, Dana Drachsler Cohen, Arthur Gervais, Florian Bueznli, dan Martin Vechev. 2018. “Securify: Practical Security Analysis of Smart Contracts.” ArXiv:1806.01143 [Cs]

- 
- 4 Bartoletti, Massimo, Salvatore Carta, Tiziana Cimoli, dan Roberto Saia. 2019. “Dissecting Ponzi Schemes on Ethereum: Identification, Analysis, and Impact.” *Future Generation Computer Systems* 102:259–77. doi: 10.1016/j.future.2019.08.014.
  - 5 Liang, Yuzhi, Weijing Wu, Kai Lei, dan Feiyang Wang. 2021. “Data-Driven Smart Ponzi Scheme Detection.” *ArXiv:2108.09305 [Cs]*.
  - 6 Yu, Shanqing, Jie Jin, Yunyi Xie, Jie Shen, dan Qi Xuan. 2021. “Ponzi Scheme Detection in EthereumTransaction Network.” *ArXiv:2104.08456 [Cs]*.
  - 7 Chen, Yizhou, Heng Dai, Xiao Yu, Wenhua Hu, Zhiwen Xie, dan Cheng Tan. 2021. “Improving Ponzi Scheme Contract Detection Using Multi-Channel TextCNN and Transformer.” *Sensors* 21(19):6417. doi: 10.3390/s21196417.
  - 8 Wang, Lei, Hao Cheng, Zibin Zheng, Aijun Yang, dan Xiaohu Zhu. 2021. “Ponzi Scheme Detection via Oversampling-Based Long Short-Term Memory for Smart Contracts.” *Knowledge-Based Systems* 228:107312. doi: 10.1016/j.knosys.2021.107312.
  - 9 Fan, Shuhui, Shaojing Fu, Haoran Xu, dan Xiaochun Cheng. 2021. “AI-SPSD: Anti-Leakage Smart Ponzi Schemes Detection in Blockchain.” *Information Processing & Management* 58(4):102587. doi: 10.1016/j.ipm.2021.102587.
  - 10 Moore, Tyler, Jie Han, dan Richard Clayton. 2012. “The Postmodern Ponzi Scheme: Empirical Analysis of High-Yield Investment Programs.” Hlm. 41–56 dalam *Financial Cryptography and Data Security*. Vol. 7397, *Lecture Notes in Computer Science*, disunting oleh A. D. Keromytis. Berlin, Heidelberg: Springer Berlin Heidelberg.
  - 11 Grincalaitis, Merunas. 2019. *MASTERING ETHEREUM: Implement Advanced Blockchain Applications Using Ethereum-Supported Tools, ... Services, and Protocols*. PACKT Publishing Limited.
  - 12 Atzei, Nicola, Massimo Bartoletti, dan Tiziana Cimoli. 2017. “A Survey of Attacks on Ethereum Smart Contracts (SoK).” Hlm. 164–86 dalam *Principles of Security and Trust*. Vol. 10204, *Lecture Notes in Computer Science*. Berlin, Heidelberg: Springer Berlin Heidelberg.
  - 13 Freund, Yoav, dan Robert E. Schapire. 1997. “A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting.” *Journal of Computer and System Sciences* 55(1):119–39. doi: 10.1006/jcss.1997.1504.
  - 14 Schapire, Robert E. 2013. “Explaining AdaBoost.” Hlm. 37–52 dalam *Empirical Inference*, disunting oleh B. Schölkopf, Z. Luo, dan V. Vovk. Berlin, Heidelberg: Springer Berlin Heidelberg.
  - 15 Jurafsky, D. (2007). *Speech and language processing*. Prentice Hall.
  - 16 Chen, Weili, Zibin Zheng, Edith C. H. Ngai, Peilin Zheng, dan Yuren Zhou. 2019. “Exploiting Blockchain Data to Detect Smart Ponzi Schemes on Ethereum.” *IEEE Access* 7:37575–86. doi: 10.1109/ACCESS.2019.2905769.