JRIIN : Jurnal Riset Informatika dan Inovasi
Volume 1, No. 2, Juli 2023
ISSN 9999-9999 (media online)
Hal 409-413

PENGUJIAN PERANGKAT LUNAK UNTUK *MACHINE LEARNING*

Theola Putra Djunaedy[1], Musthofa Lutvi[2], Firda Siti Humaidah[3], Muhamad Ridzal Kahsbullah[4], Aries Saifudin[5]

[1-5] Ilmu Komputer, Teknik Informatika, Universitas Pamulang, Tangerang Selatan, Indonesia
Email: [1]thldjunaedy3@gmail.com , [2]musthofalutvi128 @gmail.com , [3]firdasitihumaidah14@gmail.com , [4]ridzalaos@gmail.com , [5]aries.saifudin@unpam.ac.id

Abstrak− *Machine Learning* telah umum di berbagai penggunaan aplikasi. Sayangnya, *Machine learning* juga terbukti sangat rentan terhadap penipuan, mengacu kepada error dan bahkan kesalahan yang fatal, keadaan ini membuat pertanyaan tentang penggunaan *machine learning* secara luas, terutama dalam aplikasi yang sangat kritis, kecuali kita dapat memastikan kebenaran dan kepercayaannya, verifikasi dan pengujian perangkat lunak adalah Teknik yang di tetapkan untuk memastikan sifat sifat tersebut, misalnya dengan mendeteksi kesalahan. Namun, tantangan pengujian perangkat lunak untuk *Machine Learning* sangatlah luas dan banyak dan penting untuk diatasi. Pembahasan ringkas ini membahas tentang pengujian perangkat lunak canggih saat ini untuk *Machine learning*. Lebih khusus lagi, pembahasan ini membahas enam bidang tantangan utama untuk pengujian perangkat lunak *system Machine learning*, memeriksa pendekatan saat ini untuk tantangan dan menyoroti keterbatasannya. Jurnal ini memberikan agenda penelitian dengan arahan yang diuraikan untuk mbuat kemajuan menuju kemajuan pengujian *Machine Learning* yang canggih.

Kata Kunci: Pembelajaran Mesin (*Machine Learning*), Pengujian Perangkat Lunak, Perangkat Lunak Canggih

Abstract− *Machine Learning has become common across a wide range of application uses. Unfortunately, machine learning has also proven to be very vulnerable to fraud, leading to errors and even fatal errors, this circumstance raises questions about the widespread use of machine learning, especially in highly critical applications, unless we can verify its correctness and reliability, verify and test the device. Software is a technique that is set to ensure these properties, for example by detecting errors. However, the software testing challenges for Machine Learning are vast and numerous and important to overcome. This brief discussion covers today's state-of-the-art software testing for Machine learning. More specifically, it examines six key challenge areas for machine learning system software testing, examines current approaches to challenges and highlights their limitations. This journal provides a research agenda with directions outlined for making progress toward advancing advanced Machine Learning testing.*

Keywords*: Machine Learning, Software Testing, Advanced Software*

1. PENDAHULUAN

Software Testing adalah sebuah metode yang dijalankan perusahaan untuk memeriksa apakah aplikasi sudah sesuai dengan persyaratan yang diharapkan atau belum. Tak hanya itu, *software testing* juga dilakukan untuk memastikan bahwa produk bebas dari cacat. Metode tersebut melibatkan proses pemeriksaan komponen dalam sistem *software* menggunakan alat manual atau otomatis. Meskipun istilah ini sudah lama digunakan, beberapa *developer* lebih suka menganggap *software testing* sebagai *white hat* dan *black hat testing*. Namun, akhirnya para profesional setuju untuk memberikan proses kerja ini dengan tajuk *software testing*. Tujuan dari *software testing* sendiri adalah supaya perusahaan bisa mengidentifikasi kesalahan atau fitur yang tidak sesuai dengan persyaratan yang sebenarnya. Pasalnya, jika ada bug atau kesalahan dalam *software* yang tidak teridentifikasi, perusahaan harus memulai kembali ke proses development. Tak hanya itu, perusahaan juga harus mengemban kerugian sumber daya dan keuangan karena pengiriman produk yang harus diundur. Produk *software* yang diuji dengan benar dapat memastikan keandalan, keamanan dan kualitas yang selanjutnya bisa menghasilkan penghematan waktu, efektivitas biaya, dan kepuasan pelanggan.

Seiring berkembangnya jaman, kebutuhan akan teknologi informasi semakin meningkat. Begitu pula dengan sebuah system perangkat lunak yang tumbuh secara cepat menjadi sangat kompleks, semakin kompleks suatu perangkat lunak. Semakin banyak biaya yang dibutuhkan untuk

JRIIN : Jurnal Riset Informatika dan Inovasi
Volume 1, No. 2, Juli 2023
ISSN 9999-9999 (media online)
Hal 409-413

proses testing. Pada tahun 2014 sebuah perangkat lunak diseluruh dunia menghabiskan $3.8 Triliun dengan *testing* dan *quality assurance* lainya sebesar 23% dari keseluruhan (Arar & Ayan,2015). Hal ini menjadikan testing perangkat lunak menjadi tahap dan kompleks dalam *software development life cycle* (SDLC)

Karena tekonologi *Machine Learning* semakin persuasive, memungkinkan fungsionalitas system otonom, semakin penting untuk memastikan kualitas penalaran otonom yang di dukung oleh *machine learning*. Pengujian perangkat lunak adalah kegiatan penjaminan kualitas yang bertujuan (dalam arti luas) untuk menentukan kebenaran dari system yang diuji. Misalnya dengan memeriksa apakah system merespon input dengan benar dan untuk mengidentifikasi kesalahan yang dapat menyebabkan kegagalan. Menafsirkan "Pengujian ML": Dua komunitas berbeda telah mempelajari konsep pengujian ML, komunitas ilmiah ML (MLC) dan komunitas pengujian perangkat lunak (STC). Namun, karena kedua komunitas mempelajari algoritme ML dari perspektif yang berbeda, mereka menginterpretasikan istilah pengujian ML secara berbeda dan menurut kami perlu diperhatikan perbedaannya. Dalam MLC, pengujian model ML dilakukan untuk mengestimasi akurasi prediksinya dan meningkatkan prediksinya.

Tantangan pengujian ML berasal dari kompleksitas bawaan dari penalaran stokastik yang mendasarinya. Tidak seperti sistem tradisional yang kodenya dibuat secara deduktif, sistem ML dibuat secara induktif. Perilaku sistem yang mendefinisikan logika disimpulkan dari data pelatihan. Akibatnya, kesalahan bisa berasal tidak hanya dari kode perangkat lunak yang salah, tetapi juga kesalahan dalam data pelatihan. Namun, pendekatan yang ada sering menganggap bahwa kumpulan data berkualitas tinggi diperlukan, tanpa menerapkan evaluasi kualitas yang sistematis. Selain itu, sistem ML membutuhkan kemampuan penalaran dan pembelajaran lanjutan yang dapat memberikan jawaban dalam kondisi di mana jawaban yang benar sebelumnya tidak diketahui (Murphy, Kaiser, dan Arias 2007). Meskipun ini mungkin terjadi pada sistem tradisional, sistem ML memiliki sifat non-determinisme yang melekat yang membuatnya terus-menerus mengubah perilaku seiring semakin banyaknya data yang tersedia, tidak seperti sistem tradisional. Selain itu, untuk sistem yang berisi beberapa model ML, model tersebut akan memengaruhi pelatihan dan penyetelan satu sama lain, berpotensi menyebabkan propagasi kesalahan non-monotonik (Amershi, Begel, dan Bird 2019).

Validasi dan uji dataset, untuk mengevaluasi kecocokan model pada dataset pelatihan. Di STC, pengujian sistem ML memiliki cakupan yang lebih umum yang bertujuan untuk mengevaluasi perilaku sistem untuk berbagai atribut kualitas. Misalnya, dalam kasus integrasi atau pengujian level sistem, komponen ML diuji dalam interaksi dengan komponen sistem lain untuk persyaratan fungsional dan non-fungsional, seperti kebenaran, ketahanan, keandalan, atau efisiensi.

2. METODE

2.1. PENGGUNAAN METODE *MISSING ORACLE*

Tidak seperti system tradisional yang mengoperasikan instruksi deterministic terprogram sebelumnya, Sistem *Machine learning* beroprasi berdasarkan penalaran stokastik. Penalaran stokastik atau berbasis probabilitas seperti itu memperkenalkan ketidakpastian dalam respons sistem, yang menimbulkan perilaku non-deterministik, termasuk perilaku yang tidak dapat diprediksi atau tidak ditentukan. Karena non-determinisme, sistem ML dapat mengubah perilaku saat mereka belajar dari waktu ke waktu. Implikasi untuk pengujian adalah bahwa keluaran sistem dapat berubah dari waktu ke waktu untuk masukan pengujian yang sama.

2.2. Pengujian Metamorfik

Pengujian metamorfik adalah pendekatan lain untuk pengujian perangkat lunak tanpa test oracle. Dalam pendekatan ini, fungsi transformasi digunakan untuk memodifikasi masukan kasus uji yang ada, dan menghasilkan keluaran baru. Jika output aktual untuk input yang dimodifikasi berbeda dari output yang diharapkan, hal ini mengindikasikan adanya kesalahan pada perangkat lunak yang diuji. Pengujian metamorfik telah diterapkan pada pengklasifikasi pembelajaran mesin (Xie, Ho, dan et al. 2011) (Dwarakanath et al. 2018). Namun, dalam menguji sistem ML dengan ruang input yang besar, menulis transformasi metamorfik sangat melelahkan, dan ada potensi besar bagi ML untuk menghindari kesulitan ini dengan mengotomatiskan pembuatan hubungan metamorfik.

JRIIN : Jurnal Riset Informatika dan Inovasi
Volume 1, No. 2, Juli 2023
ISSN 9999-9999 (media online)
Hal 409-413

2.3. Uji Prioritas Data

Karena oracle otomatis biasanya tidak tersedia untuk pengujian model ML yang besar dan realistis, ada upaya besar yang terlibat dalam pelabelan manual data pengujian untuk model ML. Deep-Gini (Shi et al. 2019) adalah pekerjaan awal untuk mengurangi upaya pelabelan data pengujian untuk DNN dengan memprioritaskan pengujian yang cenderung menyebabkan kesalahan klasifikasi. Asumsi yang dibuat oleh DeepGini adalah bahwa sebuah tes cenderung salah klasifikasi jika DNN mengeluarkan probabilitas serupa untuk setiap kelas. Keterbatasan dari pendekatan ini adalah bahwa hal itu membutuhkan menjalankan semua tes terlebih dahulu, untuk mendapatkan vektor keluaran yang digunakan untuk menghitung kemungkinan kesalahan klasifikasi.

3. ANALISA DAN PEMBAHASAN

Untuk mengurangi kerentanan pengklasifikasi ML terhadap musuh, upaya penelitian dilakukan untuk mempelajari dan mengevaluasi kekokohan model ML secara sistematis, serta menyediakan kerangka kerja untuk pembandingan kekokohan model ML.

3.1. MATRIKS KETANGGUHAN

Kurangnya ketangguhan dalam jaringan saraf menimbulkan kekhawatiran yang valid tentang keamanan sistem yang mengandalkan jaringan ini, terutama dalam domain keselamatan-kritis seperti transportasi, robotika, obat-obatan, atau peperangan. Pendekatan tipikal untuk meningkatkan ketangguhan jaringan saraf adalah dengan mengidentifikasi contoh musuh yang membuat jaringan gagal, kemudian menambah dataset pelatihan dengan contoh ini dan melatih jaringan saraf lainnya. Ketangguhan jaringan baru adalah rasio antara jumlah contoh permusuhan yang gagal di jaringan asli dan yang ditemukan untuk jaringan baru (Goodfellow, Shlens, dan Szegedy 2015). Keterbatasan pendekatan ini adalah kurangnya ukuran ketahanan objektif (Bastani et al. 2016). Oleh karena itu, metrik untuk mengukur ketahanan DNN menggunakan pemrograman linier (Bastani et al. 2016) diusulkan. Pendekatan lain termasuk mendefinisikan batas atas kekokohan pengklasifikasi terhadap gangguan permusuhan (Fawzi, Fawzi, dan Frossard 2018). Batas atas ditemukan tergantung pada ukuran pembedaan antara kelas, dan dapat dibuat secara independen dari algoritma pembelajaran. Dalam karyanya, Fawzi et al. laporkan dua temuan: pertama, pengklasifikasi non-linier lebih kuat terhadap gangguan permusuhan daripada pengklasifikasi linier, dan kedua, kedalaman (daripada napas) jaringan saraf memiliki peran kunci untuk ketahanan permusuhan.

3.2. BENCHMARK UNTUK KOKOHNYA APLIKASI

Ada kesulitan mereproduksi beberapa metode yang dikembangkan untuk meningkatkan ketahanan jaringan saraf atau metode untuk membandingkan hasil eksperimen, karena berbagai sumber contoh permusuhan dalam proses pelatihan dapat membuat pelatihan permusuhan lebih atau kurang efektif (Goodfel-low, Papernot, dan McDaniel 2016). Untuk mengatasi tantangan ini, Cleverhans (Goodfellow, Papernot, dan McDaniel 2016) dan Foolbox (Rauber, Brendel, dan Bethge 2017) adalah pustaka contoh permusuhan untuk mengembangkan dan membandingkan serangan dan pertahanan permusuhan, sehingga tolok ukur yang berbeda dapat dibandingkan. Keterbatasan kedua kerangka kerja ini adalah bahwa mereka tidak memiliki strategi generasi musuh yang defensif (Yuan et al. 2019). Robust Vision Benchmark 2 memperluas gagasan Foolbox, dengan mengizinkan pengembangan serangan baru yang digunakan untuk semakin memperkuat pengukuran ketahanan model ML. Inisiatif lain termasuk kompetisi yang diselenggarakan pada konferensi NIPS 2017 oleh Google Brain, di mana para peneliti didorong untuk mengembangkan metode baru untuk menghasilkan contoh permusuhan dan metode baru untuk pertahanan melawan mereka (Kurakin et al. 2018).

3.3. JAMINAN FORMAL ATAS KE KOKOHAN

Untuk domain kritis keamanan yang perlu mematuhi regulasi dan sertifikasi keselamatan, sangat penting untuk memberikan jaminan formal kinerja ML di bawah gangguan masukan musuh. Memberikan jaminan tersebut merupakan tantangan nyata dari sebagian besar pendekatan pertahanan, termasuk pendekatan yang dibahas di atas. Upaya yang ada ke arah ini termasuk (Hein dan Andriushchenko 2017), dengan menggunakan regularisasi dalam pelatihan, dan (Sinha, Namkoong, dan Duchi 2018), dengan memperbarui tujuan pelatihan untuk memenuhi kendala

JRIIN : Jurnal Riset Informatika dan Inovasi
Volume 1, No. 2, Juli 2023
ISSN 9999-9999 (media online)
Hal 409-413

ketahanan. Meskipun pendekatan awal ini menarik, pendekatan ini terbukti hanya dapat mencapai tingkat kekokohan sedang, yaitu memberikan jaminan perkiraan. Oleh karena itu, diperlukan kemajuan penelitian lebih lanjut dalam memberikan jaminan ketahanan untuk model ML.

4. KESIMPULAN

4.1. Kesimpulan

Berdasarkan hasil praktek yang telah diterapkan pada pengujian perangkat lunak pada *Machine learning*. Ditarik kesimpulan sebagai berikut.

1. Hasil eksperimen yang paling optimal antara penggunaan metode *machine learning* adalah penggunaan metode machine learning dengan metode *missing oracle*.
2. Contoh missal untuk menyelesaikan masalah deteksi berita palsu pada berita *online* berbahasa Indonesia, cukup menggunakan metode *machine learning missing oracle* atau uji prioritas data.

4.2. Saran

Pengujian perangkat lunak ML menghadapi berbagai tantangan penelitian terbuka, dan penelitian lebih lanjut yang berfokus pada mengatasi tantangan ini diperlukan. Kami membayangkan pekerjaan lebih lanjut seperti itu berkembang ke arah berikut.

1. Oracle uji otomatis. Test oracle sering hilang dalam pengujian sistem ML, yang membuat pemeriksaan kebenaran keluarannya menjadi sangat menantang. Pengujian metamorf dapat membantu mengatasi tantangan ini, dan pekerjaan lebih lanjut diperlukan menggunakan ML untuk mengotomatiskan pembuatan hubungan metamorfik. Metrik cakupan untuk model ML. Metrik cakupan yang ada tidak memadai dalam beberapa konteks. Kriteria cakupan struktural dapat menyesatkan, yaitu terlalu kasar untuk input musuh dan terlalu halus untuk input alami yang salah klasifikasi (Li et al. 2019). Cakupan neuron yang tinggi tidak berarti kebal terhadap contoh musuh (Sun et al. 2019). Selain itu, cakupan neuron dapat menyebabkan ledakan ruang masukan. Adaptasi teknik pengujian kombinatorial adalah pendekatan yang menjanjikan untuk tantangan ini, mengingat kemajuan yang dicapai dalam meningkatkan skalabilitasnya untuk model ML dunia nyata.
2. Kualitas set data uji untuk model ML. Evaluasi kualitas kumpulan data untuk model ML masih dalam tahap awal. Adaptasi pengujian mutasi dapat meringankan tantangan ini. Operator mutasi umum tidak cukup untuk pengujian mutasi DNN. Sebaliknya, operator khusus domain diperlukan.
3. Efektivitas biaya dari contoh permusuhan. Strategi pembangkitan untuk contoh permusuhan perlu lebih maju untuk mengurangi kompleksitas komputasi dan meningkatkan efektivitas untuk pengklasifikasi yang berbeda.

REFERENCES

Cisco Systems. (n.d.). https://www.netacad.com/. Retrieved from Cisco Networking Academy: https://www.netacad.com/

Cisco Systems. (n.d.). https://www.youtube.com/c/TechAcad?app=desktop. Retrieved from Tech Acad-Youtube: https://www.youtube.com/c/TechAcad?app=desktop

Cisco Systems, Inc. (1984, December 10). https://www.cisco.com/. Retrieved from Cisco Systems: https://www.cisco.com/

Edward E. Ogheneovo1 and Ibiba S. Kio1. (2014). Modeling Network Router, Switches and Security Using Cisco and OPNET Simulation Software. Modeling Network Router, Switches and Security Using Cisco and OPNET Simulation Software, IOSR Journal of Engineering (IOSRJEN) www.iosrjen.org ISSN (e):.

Indria, N. &. (2017). Perancangan Jaringan Nirkabel Sebagai Redundancy Link Pada Infrastruktur Wan Yayasan Kesehatan (yakes) Telkom Bandung Menggunakan Metodologi Network Development Life Cycle (NDLC). eProceedings of Engineering, 4(2).

Javid, S. R. (2014). “Role of Packet Tracer in learning Computer Networks”. “Role of Packet Tracer in learning Computer Networks”,, , International Journal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 5,.

JRIIN : Jurnal Riset Informatika dan Inovasi
Volume 1, No. 2, Juli 2023
ISSN 9999-9999 (media online)
Hal 409-413

Kawuka, G. R. (2018). . Perancangan Local Area Network Di Smk Negeri 1 Sinonsayang. Engineering Education Journal-E2J, 6.

ManDerser. (n.d.). https://www.youtube.com/watch?v=LE_ay_WbZsg. Retrieved from Youtube: https://www.youtube.com/watch?v=LE_ay_WbZsg

Nathaniel S. Tarkaa, P. I. (2017). Design and Simulation of Local Area Network Using Cisco Packet Tracer The International Journal of Engineering and Science (IJES),. Design and Simulation of Local Area Network Using Cisco Packet Tracer The International Journal of Engineering and Science (IJES),, Volume 6, Issue 10, PP 63- 77.

Noor Maizura Mohamad Noor, N. Y. (2018). n, "Effectiveness of Using Cisco Packet Tracer as a Learning Tool: A Case Study of Routing Protocol". n, "Effectiveness of Using Cisco Packet Tracer as a Learning Tool: A Case Study of Routing Protocol", , International Journal of Information and Education Technology, Vol. 8, No.1.

Sucipto, S. A. (2019). Perancangan Jaringan Hotspot untuk Peningkatan Layanan Teknologi Informasi. Antivirus: Jurnal Ilmiah Teknik Informatika, 13, 72-79.

Suwandi, E. L. (2019). Analisis Dan Perancangan Jaringan Komputer Di Dinas Komunikasi Dan Informatika Kabupaten Minahasa. Engineering Education Journal-E2J, 6(1).

Hein, M., and Andriushchenko, M. 2017. Formal guarantees on the robustness of a classifier against adversarial manipulation. In Annual Conf. on Neural Inf. Processing Systems.

Helle, P., and Schamai, W. 2016. Testing of autonomous systems – challenges and current state-of-the-art. In INCOSE Int. Symposium (IS 2016).

Huang, X.; Kwiatkowska, M.; Wang, S.; and Wu, M. 2017. Safety verification of deep neural networks. In Computer Aided Verifica- tion, 3–29.

Hutchison, C.; Zizyte, M.; Lanigan, P. E.; Guttendorf, D.; Wagner, M.; Le Goues, C.; and Koopman, P. 2018. Robustness testing of autonomy software. In IEEE/ACM Int. Conf. on Soft. Eng., 276– 285.

Jia, R., and Liang, P. 2017. Adversarial examples for evaluating reading comprehension systems. In Conf. on Emp. Methods in Nat- ural Lang. Process.