Peningkatan Performansi Deteksi Pesan Spam Melalui Optimasi LSTM Berbasis Word2Vec dan Grid Search

Hilman Singgih Wicaksana[1,*], Hargokendar Suhud[2]

[1,2] Universitas Karya Husada Semarang
singgih.hilman@gmail.com[*)]

Abstract— *The growth of short message services and instant messaging applications in today's digital era has significantly improved public access to communication. However, this convenience is accompanied by new challenges in the form of increasing spam messages that can interfere with user comfort and potentially threaten user data security. This study proposes a Long Short-Term Memory (LSTM) approach with Word2Vec and grid search. In this architecture, Word2Vec functions as a word embedding technique to create informative word vector representations, LSTM is used as a Deep Learning model to study long-term dependency patterns in text, and grid search is applied as a hyperparameter tuning technique to find the best parameter combination to produce an optimal model. The purpose of this study is to compare the performance of the LSTM model before and after optimization using grid search on a dataset consisting of 1,143 data points. The results show that the optimized LSTM model experienced an increase in performance compared to the pre-optimization model, with an increase in precision of 0.0426, f1-score of 0.0228, and accuracy of 0.0261. Thus, the application of the grid search optimization technique proved to be effective in improving the ability of the LSTM model to detect spam messages more precisely and accurately.*

Keyword— ***spam detection, long short-term memory, word2vec, grid search, hyperparameter tuning***

Abstrak— Pertumbuhan layanan pesan singkat dan aplikasi perpesanan instan pada era digital saat ini telah meningkatkan akses komunikasi masyarakat secara signifikan. Namun, kemudahan tersebut diiringi dengan tantangan baru berupa meningkatnya pesan spam yang dapat mengganggu kenyamanan serta berpotensi mengancam keamanan data pengguna. Penelitian ini mengusulkan pendekatan Long Short-Term Memory (LSTM) dengan Word2Vec dan grid search. Dalam arsitektur ini, Word2Vec berfungsi sebagai teknik word embedding untuk menciptakan representasi vektor kata yang informatif, LSTM digunakan sebagai model Deep Learning untuk mempelajari pola ketergantungan jangka panjang pada teks, dan grid search diterapkan sebagai teknik hyperparameter tuning untuk mencari kombinasi parameter terbaik guna menghasilkan model yang optimal. Tujuan dari penelitian ini adalah melakukan komparasi kinerja antara model LSTM sebelum dan sesudah dioptimasi menggunakan grid search pada dataset yang terdiri dari 1.143 data. Hasil penelitian menunjukkan bahwa model LSTM yang telah dioptimasi mengalami peningkatan kinerja dibandingkan model sebelum optimasi, dengan kenaikan precision sebesar 0,0426, f1-score sebesar 0,0228, dan accuracy sebesar 0,0261. Dengan demikian, penerapan teknik optimasi grid search terbukti efektif dalam meningkatkan kemampuan model LSTM untuk mendeteksi pesan spam dengan lebih tepat dan akurat.

Kata Kunci— ***deteksi spam, long short-term memory, word2vec, grid search, hyperparameter tuning***

1. Pendahuluan

Pertumbuhan layanan pesan singkat seperti *short message system* (SMS) dan aplikasi *chat* instan telah meningkatkan akses komunikasi digital untuk masyarakat [1]. Namun, kemudahan ini juga menimbulkan tantangan baru, yaitu meningkatnya jumlah pesan yang tidak diinginkan yang masuk ke perangkat pengguna atau yang dapat disebut sebagai pesan *spam*. Pesan tersebut justru mengganggu aktivitas pengguna sehingga menciptakan ketidaknyamanan pengguna dan dapat berpotensi menimbulkan risiko dalam keamanan data pengguna [2], [3]. Untuk mengatasi isu tersebut, maka diperlukan pendekatan *supervised learning* untuk mendeteksi pesan yang tidak diinginkan dengan menganalisis pola teks yang ada di dalamnya.

Supervised learning merupakan sebuah pendekatan yang digunakan oleh model *machine learning* atau *deep learning* dengan mempelajari data berlabel yang dilatih, sehingga model tersebut mampu memprediksi data uji [4]. Pendekatan tersebut dapat digunakan untuk mendeteksi pesan *spam*, karena *dataset* yang digunakan memiliki dua kategori sebagai labelisasinya meliputi *spam* dan *ham*. Oleh

karena itu, *dataset* tersebut dapat digunakan lebih lanjut dengan menggunakan algoritma yang mendukung pendekatan *supervised learning*.

Penelitian ini menggunakan algoritma *deep learning* untuk dapat mempelajari data berlabel menggunakan model *Long Short-Term Memory* (LSTM). Model tersebut memiliki keunggulan dalam mengingat informasi dalam jangka waktu yang panjang sehingga sangat berguna dalam menghadapi ketergantungan jangka panjang dalam data yang berurutan [5]. Oleh karena itu, diperlukan langkah-langkah untuk mengubah teks menjadi bentuk numerik yang sesuai sebelum bisa diproses dengan baik oleh model.

Representasi numerik tersebut didapatkan melalui teknik *word embedding*, yaitu sebuah metode yang mengubah kata-kata dalam teks menjadi vektor dengan nilai kontinu sehingga struktur semantik dan hubungan antar kata dapat dipertahankan [6]. Teknik tersebut memungkinkan model untuk mendapatkan *input* dalam bentuk teks yang lebih terorganisir dan memiliki makna yang jelas. Dalam studi ini, pendekatan tersebut diterapkan dengan memanfaatkan Word2Vec, yang mengkonversi setiap kata ke dalam berbagai dimensi vektor tetap berdasarkan pola kemunculan yang terdapat dalam korpus [7]. Dengan teknik tersebut, Word2Vec menawarkan representasi kata yang lebih informatif sehingga model LSTM mampu menginterpretasikan konteks dari pesan dengan cara yang lebih efisien.

Penelitian yang telah dilakukan oleh [8] menggunakan *dataset* yang bersumber dari Kaggle yang terdiri dari kelas *spam* dan *ham* dengan total data sebanyak 5.572 baris data. Penelitian tersebut melakukan perbandingan dari performansi model antara LSTM dan CNN dalam mendeteksi pesan *spam*. Hasil dari penelitian tersebut menunjukkan bahwa model LSTM lebih unggul dalam memprediksi pesan *spam* dibandingkan dengan CNN dengan *accuracy* sebesar 98,72% dengan *loss* sebesar 0,0377 pada model LSTM, sedangkan pada model CNN memiliki *accuracy* sebesar 87,78% dengan *loss* sebesar 0,3659. Namun, penelitian tersebut menggunakan *embedding* secara *default*, bukan berbasis *word embedding*. Selain itu, penelitian tersebut juga belum menerapkan teknik *hyperparameter tuning* untuk membangun model yang lebih optimal dalam menjalankan tugas klasifikasi.

Di samping itu, studi yang telah dilakukan oleh [9] menggunakan *dataset* SpamAssassin yang berisi pesan e-mail yang terdiri dari 2551 pesan *ham* dan 501 pesan *spam*. Dalam penelitian tersebut, model LSTM digunakan untuk memprediksi pesan *spam* pada e-mail. Model yang dibangun dalam penelitian tersebut menghasilkan *accuracy* sebesar 99,35%. Namun, penelitian tersebut juga memiliki keterbatasan dengan tidak menerapkan *word embedding* pada arsitektur model LSTM yang dibangun dan tidak menerapkan teknik *hyperparameter tuning*.

Berdasarkan beberapa penelitian terdahulu, penelitian ini mengatasi keterbatasan tersebut dengan menggunakan pendekatan *word embedding* untuk menciptakan representasi vektor kata per kata dari *input* kalimat tertentu dan meningkatkan performansi model LSTM dengan *hyperparameter tuning*. Peningkatan performansi model ini pernah dilakukan oleh [10] dengan menunjukkan keakuratan model dalam memprediksi serangan jantung menggunakan model *support vector machine* (SVM) dengan metode *grid search* sebagai teknik *hyperparameter tuning* yang digunakan. Pendekatan tersebut sangat terstruktur dan memastikan bahwa setiap kombinasi diujicobakan untuk mengidentifikasi kombinasi parameter terbaik yang optimal berdasarkan ukuran evaluasi seperti *accuracy* atau *loss* [11]. Dengan demikian, penelitian ini menggunakan *grid search* sebagai teknik *hyperparameter tuning* untuk meningkatkan performansi model LSTM.

Meskipun telah ada beberapa penelitian yang telah dilakukan sebelumnya, masih ada beberapa kelemahan yang perlu diteliti lebih lanjut. Penelitian sebelumnya cenderung tidak menggunakan pendekatan *word embedding*. Selain itu, pada penelitian sebelumnya juga tidak menerapkan teknik *hyperparameter tuning* untuk meningkatkan performansi modelnya. Oleh karena itu, penelitian ini akan bertujuan untuk mengatasi keterbatasan tersebut dengan melakukan perbandingan antara model LSTM sebelum dan sesudah dioptimasi menggunakan *grid search*.

Penelitian ini mengusulkan sebuah pendekatan untuk meningkatkan performansi model LSTM agar mampu mendeteksi pesan *spam* dengan akurat dengan menggunakan Word2Vec sebagai *word embedding* dan *grid search* sebagai *hyperparameter tuning* yang digunakan dalam model LSTM.

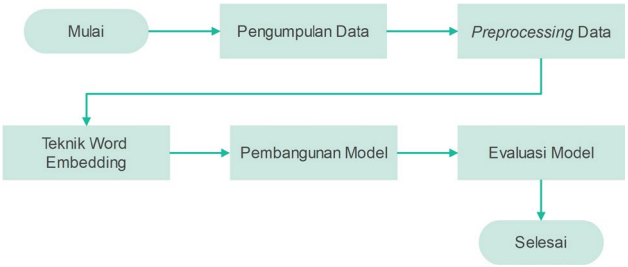
Dengan demikian, penelitian ini bertujuan untuk mengevaluasi dan meningkatkan

performansi model LSTM dalam mendeteksi pesan *spam* dengan menggunakan Word2Vec sebagai teknik *word embedding* dan proses *hyperparameter tuning* menggunakan metode *grid search*. Penelitian ini menggunakan *dataset* dengan total data sebanyak 1.143 baris data yang terdiri dari dua kelas meliputi kelas *spam* dan *ham*. Oleh karena itu, penelitian ini diharapkan dapat memberikan kontribusi yang berarti dalam mendukung upaya pencegahan bagi pengguna dalam membedakan antara pesan *spam* dan *ham*, serta berfungsi sebagai panduan dalam evaluasi kinerja model deteksi pesan *spam* di masa depan.

2. Metode

Pada penelitian ini terdiri dari beberapa tahapan meliputi pengumpulan data, *preprocessing* data, teknik *word embedding*, dan pembangunan model seperti yang dapat ditunjukkan pada Gambar 1.

Pada tahap pengumpulan data, penelitian ini menggunakan *dataset* dari Kaggle yang berjumlah 1.143 baris yang terdiri dari dua kelas yaitu kelas *spam* dan *ham* dimana kelas *spam* terdiri dari 574 sampel data, sedangkan kelas *ham* terdiri dari 569 sampel data. *Dataset* dalam penelitian ini dapat dibagi menjadi tiga bagian: 80% data *training*, 10% data *validation*, dan 10% data *testing*.



Gambar. 1 Alur Penelitian

Langkah selanjutnya yang dapat dilakukan adalah *preprocessing*. *Preprocessing* data merupakan tahap penting dalam penambangan data, karena data yang diperoleh sering kali belum siap digunakan, misalnya mengandung *missing value*, *outlier*, redundansi, atau format yang tidak sesuai, sehingga diperlukan langkah *preprocessing* ini untuk memperbaiki masalah tersebut sebelum analisis dilakukan lebih lanjut [12]. Dalam tahap *preprocessing*, terdapat beberapa langkah yang perlu dilakukan antara lain *casefolding*, *tokenization*, *stemming*, *filtering*, dan *cleaning*.

Casefolding merupakan langkah mengubah seluruh huruf pada suatu kata menjadi bentuk lowercase untuk menyatukan variasi penulisan, sehingga teks yang diolah memiliki format yang konsisten dan seragam [13]. Penerapan dari *casefolding* dilakukan terhadap teks awal dari setiap kalimat pesan. Tabel 1 menunjukkan contoh hasil dari penerapan *casefolding* yang telah dilakukan.

Tabel 1. Contoh Hasil dari *Casefolding*

Teks Awal	*Casefolding*
Plg Yth: Simcard anda mendptkan bonus poin plus-plus 555 dr:PT.INDOSAT pin anda:277fg49 u/info klik di www.indosat-555.blogspot.com atau Hub:021-3338-0074.	plg yth: simcard anda mendptkan bonus poin plus-plus 555 dr:pt.indosat pin anda:277fg49 u/info klik di www.indosat-555.blogspot.com atau hub:021-3338-0074.

Tokenizing merupakan tahapan pemecahan teks menjadi unit-unit kata penyusunnya untuk memudahkan langkah pemrosesan berikutnya, seperti perhitungan frekuensi kata, pemberian bobot, hingga pengubahan data menjadi vektor berdimensi tinggi [14]. Penerapan *tokenization* ini dilakukan dengan memecah kata per kata dari suatu kalimat pada kalimat yang telah dilakukan *casefolding*. Tabel 2 menjelaskan contoh hasil *tokenization* yang telah dilakukan.

Tabel 2. Contoh Hasil *Tokenization*

Casefolding	*Tokenization*
plg yth: simcard anda mendptkan bonus poin plus-plus 555 dr:pt.indosat pin anda:277fg49 u/info klik di www.indosat-555.blogspot.com atau hub:021-3338-0074.	['plg', 'yth', 'simcard', 'anda', 'mendptkan', 'bonus', 'poin', 'plus', 'plus', 'dr', 'pt', 'indosat', 'pin', 'anda', 'fg', 'u', 'info', 'klik', 'di', 'www', 'indosat', 'blogspot', 'com', 'atau', 'hub']

Stemming dapat digunakan untuk mengubah kata yang memiliki imbuhan menjadi kata dasar [15]. Dalam penelitian ini menggunakan *library* Sastrawi untuk melakukan *stemming* yang dikhususkan dalam bahasa Indonesia. Tabel 3 menunjukkan contoh hasil *stemming* yang telah dilakukan.

Tabel 3. Contoh Hasil *Stemming*

Tokenization	*Stemming*
['plg', 'yth', 'simcard', 'anda', 'mendptkan', 'bonus', 'poin', 'plus', 'plus', 'dr', 'pt', 'indosat',	['plg', 'yth', 'simcard', 'anda', 'mendptkan', 'bonus', 'poin', 'plus', 'plus', 'dr', 'pt', 'indosat',

'pin', 'anda', 'fg', 'u', 'info', 'klik', 'di', 'www', 'indosat', 'blogspot', 'com', 'atau', 'hub']	'pin', 'anda', 'fg', 'u', 'info', 'klik', 'di', 'www', 'indosat', 'blogspot', 'com', 'atau', 'hub']

Filtering merupakan proses untuk membuang data yang tidak lengkap atau mengandung kesalahan sekaligus melakukan stopword removal dengan menghapus kata-kata umum yang sering muncul tetapi tidak memberikan kontribusi berarti terhadap makna kalimat [16]. Proses tersebut dilakukan pada kalimat yang sudah dilakukan *stemming* sebelumnya. Tabel 4 menunjukkan hasil dari *filtering* yang telah dilakukan.

Tabel 4. Contoh Hasil *Filtering*

Stemming	***Filtering***
['plg', 'yth', 'simcard', 'anda', 'mendptkan', 'bonus', 'poin', 'plus', 'plus', 'dr', 'pt', 'indosat', 'pin', 'anda', 'fg', 'u', 'info', 'klik', 'di', 'www', 'indosat', 'blogspot', 'com', 'atau', 'hub']	['plg', 'yth', 'simcard', 'mendptkan', 'bonus', 'poin', 'plus', 'plus', 'dr', 'pt', 'indosat', 'pin', 'fg', 'u', 'info', 'klik', 'www', 'indosat', 'blogspot', 'com', 'hub']

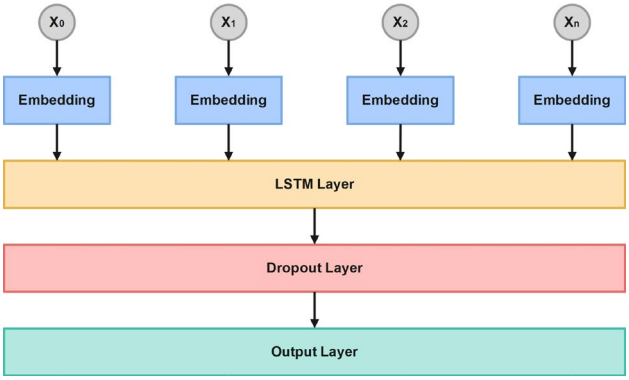
Kemudian, langkah terakhir dalam melakukan *preprocessing data* adalah *cleaning*. *Cleaning* dapat dilakukan dengan menghapus hasil *filtering* yang telah dilakukan sebelumnya sehingga dapat dilanjutkan ke tahap selanjutnya dengan teknik *word embedding*. Tabel 5 menunjukkan hasil *cleaning* yang telah dilakukan.

Tabel 5. Contoh Hasil *Cleaning*

Filtering	***Cleaning***
['plg', 'yth', 'simcard', 'mendptkan', 'bonus', 'poin', 'plus', 'plus', 'dr', 'pt', 'indosat', 'pin', 'fg', 'u', 'info', 'klik', 'www', 'indosat', 'blogspot', 'com', 'hub']	plg yth simcard mendptkan bonus poin plus plus dr pt indosat pin fg u info klik www indosat blogspot com hub

Pada penelitian ini, teknik *word embedding* yang digunakan adalah Word2Vec. Word2Vec merupakan teknik ekstraksi fitur yang merepresentasikan kata dalam bentuk vektor berdimensi rendah sehingga mampu menangkap kemiripan dan hubungan semantik antarkata, termasuk memahami makna, mengenali keterkaitan berdasarkan pola kemunculan, serta mengidentifikasi relasi seperti sinonim dan antonim [17]. Word2Vec akan menghasilkan representasi vektor kata dari suatu kalimat sehingga dapat dijadikan *input* dalam model tertentu.

Arsitektur model yang dibangun dalam penelitian ini yaitu LSTM. Dalam arsitektur tersebut terdapat *input layer*, *embedding layer*, *dropout layer*, dan *output layer*. *Input layer* berfungsi menerima dan mengonversi data teks ke dalam bentuk urutan indeks numerik. *Embedding layer* digunakan untuk memetakan indeks tersebut menjadi representasi vektor yang lebih bermakna secara semantik [18]. *Dropout layer* berperan mencegah *overfitting* dengan menonaktifkan sebagian *neuron* secara acak selama *training* [19]. Sedangkan, *output layer* menghasilkan prediksi akhir berupa probabilitas kategori antara kelas *spam* dan *ham*. Penelitian ini menggunakan *activation function* berupa *sigmoid* karena fungsi tersebut akan memberikan nilai *output* dalam rentang antara 0 sampai 1 [20]. Dengan demikian, *loss function* yang dapat digunakan adalah *binary crossentropy* dimana fungsi tersebut dapat digunakan dalam klasifikasi biner [21]. Gambar 2 menunjukkan arsitektur model yang dibangun dalam penelitian ini.



Gambar. 2 Arsitektur Model yang Diusulkan

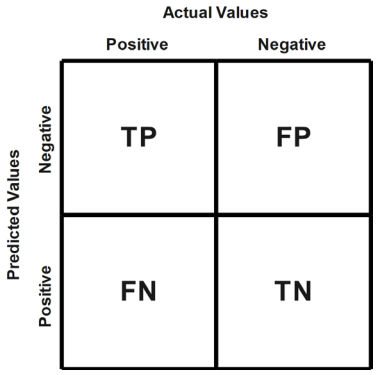
Pada Gambar 2 terdapat beberapa parameter yang akan dilakukan *hyperparameter tuning* pada arsitektur model yang diusulkan dalam penelitian ini antara lain *hidden units* pada model LSTM, *dropout*, dan *learning rate*. Parameter-parameter tersebut kemudian didaftarkan ke dalam teknik *grid search* agar mampu menemukan kombinasi parameter terbaik dalam arsitektur model tersebut. Tabel 6 menunjukkan nilai-nilai yang terdapat pada setiap parameternya.

Tabel 6. Nilai pada Setiap *Hyperparameter*

Parameter	**Nilai**
Hidden units LSTM	64, 128
Dropout	0.3, 0.5

Learning rate	0.0001, 0.001

Setelah melakukan pembangunan model, maka diperlukan evaluasi performansi model berdasarkan *confusion matrix*. *Confusion matrix* merupakan sebuah teknik penilaian untuk model klasifikasi yang disajikan dalam format tabel matriks, digunakan untuk membandingkan hasil prediksi model dengan data yang sebenarnya [22]. Di dalam matriks tersebut terdapat nilai *true positive* (TP), *false positive* (FP), *true negative* (TN), dan *false negative* (FN). Bentuk *confusion matrix* ini dapat ditunjukkan pada Gambar 3.



Gambar. 3 *Confusion Matrix*

Pada penelitian ini menggunakan metrik evaluasi yang meliputi *accuracy*, *precision*, *recall*, dan *f1-score*. *Precision* merupakan rasio antara jumlah prediksi benar yang positif dibandingkan dengan total prediksi positif yang dilakukan [23]. *Recall* merupakan proporsi antara jumlah prediksi positif yang akurat dengan total jumlah data yang seharusnya positif [24]. *F1-score* suatu metode yang mengukur rata-rata yang ditimbang antara *precision* dan *recall* [25]. Sedangkan, *accuracy* dapat digunakan utuk mengukur proporsi total prediksi yang benar dari seluruh prediksi yang dihasilkan [26]. Formula (1) hingga (4) digunakan dalam penelitian ini untuk menghitung nilai *accuracy*, *precision*, *recall*, dan *f1-score*.

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

$$F1 - score = 2 \times \frac{Recall \times Precision}{Recall + Precision} \quad (3)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

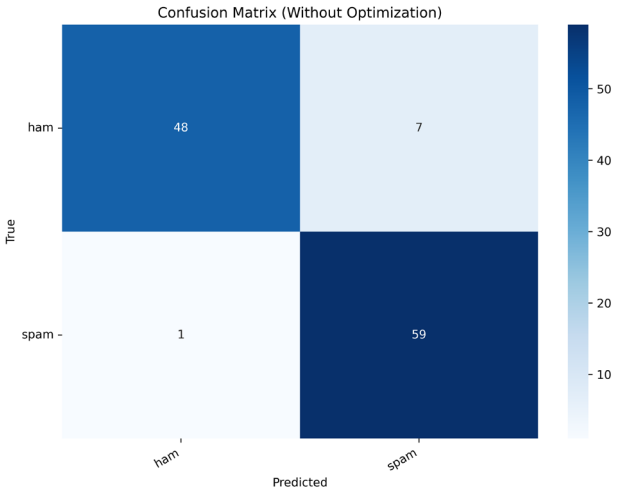
3. Pembahasan

Pada bagian ini akan dibahas menjadi tiga bagian, yakni evaluasi model sebelum dioptimasi, evaluasi model setelah dioptimasi, dan seleksi model terbaik. *Hyperparameter tuning* digunakan untuk menemukan kombinasi parameter optimal secara otomatis selama proses *training* model. Penilaian kinerja model dilakukan untuk mengungkapkan hasil dari setiap model yang dapat diukur melalui tingkat *precision*, *recall*, *f1-score*, serta *accuracy*. Uji coba model ini dilakukan untuk mengukur seberapa efektif model dalam memprediksi data dari suatu tautan yang diidentifikasi sebagai *spam* atau *ham*.

3.1 Evaluasi Model Sebelum Dioptimasi

Pada tahapan ini dilakukan evaluasi sebelum dilakukan optimasi dengan *hyperparameter tuning* menggunakan *grid search*. Pada arsitektur model yang dibangun dalam penelitian ini telah diatur parameter *default* pada LSTM *layer* dengan *hidden units* LSTM sebesar 128, *dropout layer* sebesar 0,5, dan *learning rate* sebesar 0,0001.

Berdasarkan parameter *default* tersebut telah menghasilkan nilai di tingkat *precision* sebesar 0,8939 atau sekitar 89,39%, *recall* sebesar 0,9833 atau sekitar 98,33%, *f1-score* sebesar 0.9365 atau sekitar 93,65%, dan *accuracy* sebesar 0.9304 atau sekitar 93,04%. Nilai pada setiap metrik tersebut didapatkan berdasarkan *confusion matrix* pada model LSTM sebelum dilakukan optimasi seperti yang ditunjukkan pada Gambar 4.



Gambar. 4 *Confusion Matrix* pada Model LSTM Sebelum Dilakukan Optimasi

Pada Gambar 4 terdapat *confusion matrix* pada model LSTM sebelum dioptimasi dalam mendeteksi pesan *spam*. Sampel data sebanyak 48 data terprediksi sebagai pesan *ham*, sementara terdapat 7 sampel data yang terprediksi sebagai *spam*. Selain itu, model LSTM juga mampu memprediksi data sebanyak 59 sampel data sebagai *spam*, sementara hanya terdapat 1 sampel data yang terprediksi sebagai *ham*.

3.2 Evaluasi Model Setelah Dioptimasi

Pada tahapan ini dilakukan evaluasi setelah dilakukan optimasi dengan *hyperparameter tuning* menggunakan *grid search*. Pada arsitektur model yang dibangun dalam penelitian ini telah didaftarkan beberapa parameter yang meliputi *hidden units* LSTM, *dropout layer*, dan *learning rate* seperti yang terdapat pada Tabel 6.

Selama proses *training*, model LSTM menghasilkan kombinasi parameter terbaik yang meliputi *hidden units* LSTM sebesar 128, *dropout* sebesar 0,3, dan *learning rate* sebesar 0,0001. Proses tersebut dilakukan dengan menggunakan *cross-validation* sebanyak 5 *folds* pada saat menerapkan *grid search*. Untuk mengetahui proses *cross-validation* yang terjadi pada saat *training* dapat ditunjukkan pada Tabel 7.

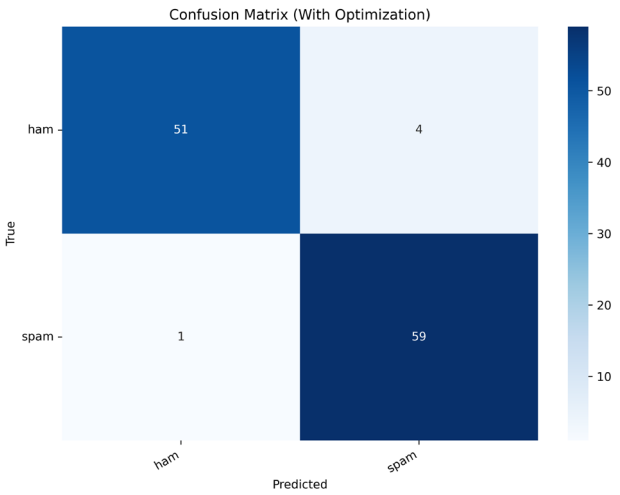
Tabel 7. Proses Grid Search yang Terjadi

Folds	Accuracy
Fold ke-1	0,9727
Fold ke-2	0,9727
Fold ke-3	0,9454
Fold ke-4	0,9617
Fold ke-5	0,9560
Mean Accuracy	0,9617
Standard Deviation (Std.)	0,0104

Berdasarkan Tabel 7 menyajikan hasil evaluasi performansi model menggunakan teknik 5-*fold cross-validation* untuk mengukur konsistensi akurasi. Selama lima kali pengujian pada subset data yang berbeda, model mencatatkan skor *accuracy* yang berkisar antara 0,9454 hingga 0,9727. Secara keseluruhan, model menunjukkan performa yang sangat baik dengan *mean accuracy* mencapai 0,9617. Selain itu, nilai *standard deviation* yang sangat kecil yaitu 0,0104, mengindikasikan bahwa model tersebut sangat stabil dan mampu memberikan prediksi yang konsisten pada berbagai variasi data validasi.

Setelah proses *grid search* dilakukan, langkah selanjutnya yaitu mengetahui sejauh mana performansi model LSTM dalam memprediksi pesan *spam* menggunakan metrik evaluasi yang digunakan dalam penelitian ini meliputi tingkat *precision* sebesar 0,9365 atau sekitar 93,65%, *recall* sebesar 0,9833 atau sekitar 98,33%, *f1-score* sebesar 0,9593 atau sekitar 95,93%, dan *accuracy* sebesar 0,9565 atau sekitar 95,65%. Nilai pada setiap metrik tersebut didapatkan berdasarkan *confusion matrix* pada model LSTM setelah dilakukan optimasi seperti yang ditunjukkan pada Gambar 5.

Pada Gambar 5 terdapat *confusion matrix* pada model LSTM setelah dioptimasi dalam mendeteksi pesan *spam*. Sampel data sebanyak 51 data terprediksi sebagai pesan *ham*, sementara terdapat 4 sampel data yang terprediksi sebagai *spam*. Selain itu, model LSTM juga mampu memprediksi data sebanyak 59 sampel data sebagai *spam*, sementara hanya terdapat 1 sampel data yang terprediksi sebagai *ham*.



Gambar. 5 *Confusion Matrix* pada Model LSTM Setelah Dilakukan Optimasi

3.3 Seleksi Model Terbaik

Tahap ini bertujuan menyeleksi model terbaik berdasarkan evaluasi performansi model LSTM sebelum dan sesudah dioptimasi. Hasil evaluasi menunjukkan bahwa model LSTM yang telah dioptimasi menggunakan teknik *grid search* mengalami peningkatan pada metrik *precision*, *f1-score*, dan *accuracy*. Peningkatan tersebut meliputi kenaikan *precision* sebesar 0,0426, *f1-score* sebesar 0,0228, dan *accuracy* sebesar 0,0261. Sementara itu, nilai *recall* cenderung

stabil. Perbandingan kinerja model sebelum dan sesudah optimasi disajikan pada Tabel 8.

Tabel 8. Perbandingan Evaluasi Performansi Model Sebelum dan Setelah Dioptimasi

Metrik Evaluasi	Sebelum Dioptimasi	Setelah Dioptimasi
Precision	0,8939	0,9365
Recall	0,9833	0,9833
F1-score	0,9365	0,9593
Accuracy	0,9304	0,9565

Merujuk pada rincian nilai yang tercantum dalam Tabel 8, model LSTM setelah dioptimasi memperlihatkan capaian perfomansi yang lebih unggul dibandingkan model sebelum dioptimasi. Berdasarkan perbandingan data tersebut, model LSTM setelah dioptimasi dipilih sebagai model terbaik dalam penelitian ini. Superioritas performansi yang terlihat secara menyeluruh pada berbagai metrik evaluasi menjadikan model ini kandidat yang paling tepat untuk digunakan dalam pengembangan sistem di masa mendatang.

4. Kesimpulan

Penelitian ini berkontribusi pada peningkatan kinerja model LSTM melalui penerapan teknik *hyperparameter tuning* menggunakan *grid search* dengan 5-*fold cross validation*. Hasil pengujian menunjukkan adanya peningkatan metrik evaluasi dibandingkan model sebelum optimasi, yaitu kenaikan *precision* sebesar 0,0426, *f1-score* sebesar 0,0228, dan *accuracy* sebesar 0,0261, dengan nilai *recall* yang cenderung stabil. Kombinasi parameter terbaik yang dihasilkan meliputi *hidden units* LSTM sebesar 128, *dropout* sebesar 0,3, dan *learning rate* sebesar 0,0001. Dengan demikian, metode optimasi *grid search* terbukti efektif dalam menghasilkan model LSTM yang lebih unggul.

Referensi

[1] G. Widjaja, “Perubahan Pola Komunikasi dalam Masyarakat Akibat Penggunaan Aplikasi Pesan Instan,” *Prosiding Seminar Nasional Indonesia*, vol. 3, pp. 10–16, 2025.

[2] E. Amaia, A. N. Izzah, A. Akram, and N. Risal, “Klasifikasi Penyalahgunaan Pesan Singkat Menggunakan Algoritma Naïve Bayes,” *Jurnal Ilmu Komputer dan Teknologi Informasi*, vol. 8, no. 1, 2023, [Online]. Available: http://tsel.me/tsel

[3] E. Sri Ainun, U. Inayah, and M. Ilmih, “Klasifikasi Email Spam Dan Ham Menggunakan Algoritma Support Vector Machine, Naive Bayes Dan Logistic Regression,” *Scientific: Journal of Computer Science and Informatics*, vol. 2, pp. 77–84, 2025, doi: 10.34304/scientific.v2.i2.399.

[4] M. A. Yulianto and R. Andrianto, “Analisa Kinerja Algoritma Supervised Learning pada Sentimen Ulasan Aplikasi Investasi Online Bibit,” *Journal of Artificial Intelligence and Digital Business (RIGGS)*, vol. 4, no. 3, pp. 1486–1496, Aug. 2025, doi: 10.31004/riggs.v4i3.2168.

[5] R. Z. N. Ahmad, N. S. Harahap, S. Agustian, I. Iskandar, and S. Sanjaya, “Perbandingan Performa Random Forest dan Long Short-Term Memory dalam Klasifikasi Teks Multilabel Terjemahan Hadits Bukhari,” *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 5, no. 3, pp. 862–874, Jun. 2025, doi: 10.57152/malcom.v5i3.2046.

[6] B. E. S. Dewi, “Pengukuran Kemiripan Kalimat Bahasa Indonesia Menggunakan Representasi Word Embedding FastText,” *Jurnal Teknologi Informasi dan Digital (TRIDI)*, vol. 2, no. 2, pp. 20–29, 2025.

[7] H. Utama and A. Masruro, “Analisis Sentimen pada Twitter menggunakan Word Embedding dengan Pendekatan Word2Vec,” *Jurnal Sistem Cerdas*, vol. 5, no. 2, pp. 128–134, 2022.

[8] M. Al Kautsar, G. G. Setiaji, and A. Rifa’i, “Analisis Komparasi Kinerja LSTM dan CNN dalam Deteksi Spam Email Berbasis Deep learning,” *Bulletin of Computer Science Research*, vol. 5, no. 4, pp. 584–593, Jun. 2025, doi: 10.47065/bulletincsr.v5i4.572.

[9] M. W. S. Utomo, H. W. Murti, A. W. I. Sujatmoko, and A. P. Sari, “Deteksi Spam Email Menggunakan Metode LSTM (Long Short Term Memory),” *Jurnal Mahasiswa Teknik Informatika (JATI)*, vol. 8, no. 6, pp. 11406–11411, 2024.

[10] Z. Maisat Eka Darmawan and A. Fauzan Dianta, “Implementasi Optimasi Hyperparameter GridSearchCV Pada Sistem Prediksi Serangan Jantung Menggunakan SVM,” *Teknologi*, vol. 13, no. 1, pp. 8–15, Jan. 2023, doi: 10.26594/teknologi.v13i1.3098.

[11] R. Zikri, A. Sunyoto, and A. Yaqin, “Optimasi Hyperparameter Grid Search Dan Random Search Pada Inception V3 Untuk Kematangan Buah Kelapa Sawit,” *Jurnal Ilmiah Penelitian dan Pembelajaran Informatika (JIPI)*, vol. 10, no. 4, pp. 3043–3053, 2025, doi: 10.29100/jipi.v10i4.6537.

[12] Herwinsyah and A. Witanti, “Analisis Sentimen Masyarakat Terhadap Vaksinasi Covid-19 Pada Media Sosial Twitter Menggunakan Algoritma Support Vector Machine (SVM),” *Jurnal Sistem Informasi dan Informatika (Simika) P-ISSN*, vol. 5, pp. 2622–6901, 2022.

[13] Andre Pratama, J. Nababan, and N. S. Salsabila, “Algoritma Smith-Waterman Untuk

Mengidentifikasi Kemiripan Judul Proyek Mahasiswa," *JIKTEKS : Jurnal Ilmu Komputer dan Teknologi Informasi*, vol. 3, no. 01, pp. 22–34, Dec. 2024, doi: 10.70404/jikteks.v3i01.125.

[14] A. Ariansyah and U. Indahyanti, "Fitur Ekstraksi pada Pemodelan Topik Menggunakan Metode Latent Dirichlet Allocation pada Peristiwa Kebocoran Data," *Indonesian Journal of Applied Technology*, vol. 1, no. 2, pp. 1–24, Jul. 2024, doi: 10.47134/ijat.v1i2.3041.

[15] L. Pertiwi, "Penerapan Algoritma Text Mining, Steaming Dan Texrank Dalam Peringkasan Bahasa Inggris," *Bulletin of Multi-Disciplinary Science and Applied Technology (BIMASAKTI)*, vol. 1, no. 3, pp. 100–104, 2022.

[16] B. Delvika, Apriana, N. Abror, and U. R. Gurning, "Perbandingan Algoritma NBC dan C4.5 Dalam Analisa Sentimen Pemilihan Presiden 2024 Pada Twitter," in *SENTIMAS: Seminar Nasional Penelitian dan Pengabdian Masyarakat*, 2023, pp. 41–48. [Online]. Available: https://journal.irpi.or.id/index.php/sentimas

[17] S. N. Cahyani and G. W. Saraswati, "Implementation of Support Vector Machine Method in Classifying School Library Books with Combination of TF-IDF and Word2Vec," *Jurnal Teknik Informatika (JUTIF)*, vol. 4, no. 6, pp. 1555–1566, 2023, doi: 10.52436/1.jutif.2023.4.6.1536.

[18] M. Susanty and S. Sukardi, "Perbandingan Pre-trained Word Embedding dan Embedding Layer untuk Named-Entity Recognition Bahasa Indonesia," *PETIR*, vol. 14, no. 2, pp. 247–257, Sep. 2021, doi: 10.33322/petir.v14i2.1164.

[19] S. Bahri, A. Sunyoto, and M. P. Kurniawan, "Klasifikasi Hama Pada Daun Sawi Menggunakan Convolutional Neural Network (CNN) Dengan Algoritma Xception dan Optimasi Adam," *Journal of Electrical Engineering and Computer (JEECOM)*, vol. 6, no. 2, pp. 359–370, 2024, doi: 10.33650/jeecom.v4i2.

[20] K. Kurniawan, B. Ceasaro, and S. #3, "Perbandingan Fungsi Aktivasi Untuk Meningkatkan Kinerja Model LSTM Dalam Prediksi Ketinggian Air Sungai," *Jurnal Edukasi dan Penelitian Informatika (JEPIN)*, vol. 10, no. 1, pp. 134–143, Apr. 2024.

[21] T. I. Z. M. Putra, Suprapto, and A. F. Bukhori, "Model Klasifikasi Berbasis Multiclass Classification dengan Kombinasi Indobert Embedding dan Long Short-Term Memory untuk Tweet Berbahasa Indonesia," *Jurnal Ilmu Siber dan Teknologi Digital (JISTED)*, vol. 1, no. 1, pp. 1–28, Nov. 2022, doi: 10.35912/jisted.v1i1.1509.

[22] F. R. Valerian, M. Syarief, and D. A. Fatah, "Klasifikasi Tingkat Obesitas Menggunakan Metode GBM dan Confusion Matrix," *Jurnal Mahasiswa Teknik Informatika (JATI)*, vol. 9, no. 2, pp. 2242–2249, 2025.

[23] A. A. Lubis, S. I. Al Idrus, Z. Indra, K. S. S, and Chairunisah, "Klasifikasi Akun Buzzer Menggunakan Algoritma K-Nearest Neighbor pada Tagar #STYTanpaDiasporaNol di Media Sosial X," *Blend Sains Jurnal Teknik*, vol. 4, no. 2, pp. 248–259, Oct. 2025, doi: 10.56211/blendsains.v4i2.1093.

[24] Y. Nabilawati Rumbia *et al.*, "Perbandingan Metode KNN dan Naive Bayes untuk Klasifikasi Kelulusan Mahasiswa Pada Mata Kuliah Probstat," *Jurnal PTI (Jurnal Pendidikan Teknologi Informasi)*, vol. 12, no. 1, pp. 7–13, Apr. 2025, doi: 10.35134/jpti.v12i1.228.

[25] F. A. Larasati, D. E. Ratnawati, and B. T. Hanggara, "Analisis Sentimen Ulasan Aplikasi Dana dengan Metode Random Forest," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 6, no. 9, pp. 4305–4313, 2022, [Online]. Available: http://j-ptiik.ub.ac.id

[26] R. E. Pambudi, H. Purnomo, and A. Aglasia, "Analisis Klasifikasi Sentimen Pengguna MyPertamina Menggunakan Metode Evaluasi Precision, Recall, dan F1-Score," *Aisyah Journal of Informatics and Electrical Engineering*, vol. 7, no. 2, pp. 17–22, Aug. 2025, [Online]. Available: https://jti.aisyahuniversity.ac.id/index.php/AJIEE