Implementasi *Bidirectional Long Short-Term Memory* untuk Identifikasi Entitas Saham

http://dx.doi.org/10.28932/jutisi.v11i1.9775

Riwayat Artikel
Received: 17 Agustus 2024 | *Final Revision*: 06 Maret 2025 | *Accepted*: 12 Maret 2025

Akmalia Fatimah✉ #1, Badieah#2, Sam Farisa Chaerul Haviana#3

Program studi Teknik Informatika, Universitas Islam Sultan Agung Semarang
Jalan Kaligawe Raya No. Km.4, Kota Semarang, 50112, Indonesia
[1]akmaliafatimah31@gmail.com
[2]badieah.assegaf@unissula.ac.id
[3]sam@unissula.ac.id
✉Corresponding author: akmaliafatimah31@gmail.com

Abstrak — **Salah satu produk keuangan di pasar modal yang banyak diminati adalah saham. Saham adalah bukti kepemilikan suatu perusahaan yang berfluktuasi dan cenderung memiliki tingkat risiko yang tinggi serta perubahan harga yang bersifat nonliniear. Untuk mendapat keputusan investasi yang tepat, investor dituntut untuk dapat melakukan analisis informasi saham yang sangat melimpah dengan teliti dan cepat. Dalam menghadapi tantangan ini, *Named Entity Recognition* (NER) dapat menjadi solusi yang berpotensi dalam analisis informasi saham dengan mengenali entitas saham serta mengelompokkannya ke label tertentu. Pada penelitian ini NER dikembangkan dengan algoritma *Bidirectional Long Short-Term Memory*, yang digunakan untuk mengidentifikasi lima entitas saham yaitu Nama Perusahaan, Kode Saham, Indeks Saham, Sektor Industri dan Sub Sektor. Dengan hasil *accuracy* sebesar 99.81% pada data pengujian, algoritma Bi-LSTM dapat mengidentifikasi entitas dengan baik dan mengelompokkan masing-masing token ke lima entitas tersebut.**

Kata kunci— ***Bidirectional Long Short-Term Memory*; *Named Entity Recognition*; Saham.**

Implementation of Bidirectional Long Short-Term Memory for Stock Entity Identification

Abstract — ***One of the financial products in the capital market that is in great demand is stock. Shares are proof of company ownership that fluctuates and tends to have a high level of risk and nonlinear price changes. To make the right investment decision, investors must be able to analyze the abundant stock information carefully and quickly. In facing this challenge, Named Entity Recognition (NER) can be a potential solution in analyzing stock information by recognizing stock entities and grouping them into certain labels. In this research, NER is developed with the Bidirectional Long Short-Term Memory algorithm, which is used to identify five stock entities: company name, stock code, stock index, industry sector, and sub-sector. With an accuracy of 99.81% on the test data, the Bi-LSTM algorithm can identify the entities well and group each token into five entities.***

Keywords— ***Bidirectional Long Short-Term Memory*; *Named Entity Recognition*; *Stock*.**

I. PENDAHULUAN

Pasar modal memegang peranan penting dalam perekonomian suatu negara. Salah satu produk keuangan yang diperdagangkan adalah saham. Saham adalah bukti kepemilikan suatu perusahaan yang berfluktuasi dan cenderung memiliki tingkat risiko yang tinggi [1]. Kenaikan dan penurunan harga seringkali berubah secara nonlinear. Namun, hal tersebut tidak menghalangi pertumbuhan investor saham yang terus meningkat. Bursa Efek Indonesia mencatat bahwa terjadi kenaikan jumlah investor saham dari 8,11 juta investor menjadi 15,11 juta investor pada Desember 2023. Hal tersebut mempengaruhi penyebaran informasi yang sangat banyak di internet.

Perubahan harga yang cepat dan jumlah informasi yang melimpah, menuntut investor untuk dapat melakukan analisis informasi saham secara teliti dan cepat, agar dapat mengambil keputusan investasi yang tepat. Dalam menghadapi tantangan ini, *Named Entity Recognition* (NER) dapat menjadi solusi yang berpotensi dalam analisis informasi saham. *Named Entity Recognition* (NER) adalah teknik untuk mengekstraksi dan mengklasifikasikan entitas ke dalam kategori semantik yang telah ditentukan sebelumnya, seperti nama orang, tempat dan organisasi [2]. Entitas yang telah diekstraksi dapat digunakan untuk menemukan informasi yang relevan dengan lebih cepat dan mengklasifikasikan informasi ke dalam kategori tertentu. Sehingga, dapat membantu proses analisis informasi saham dengan lebih efektif.

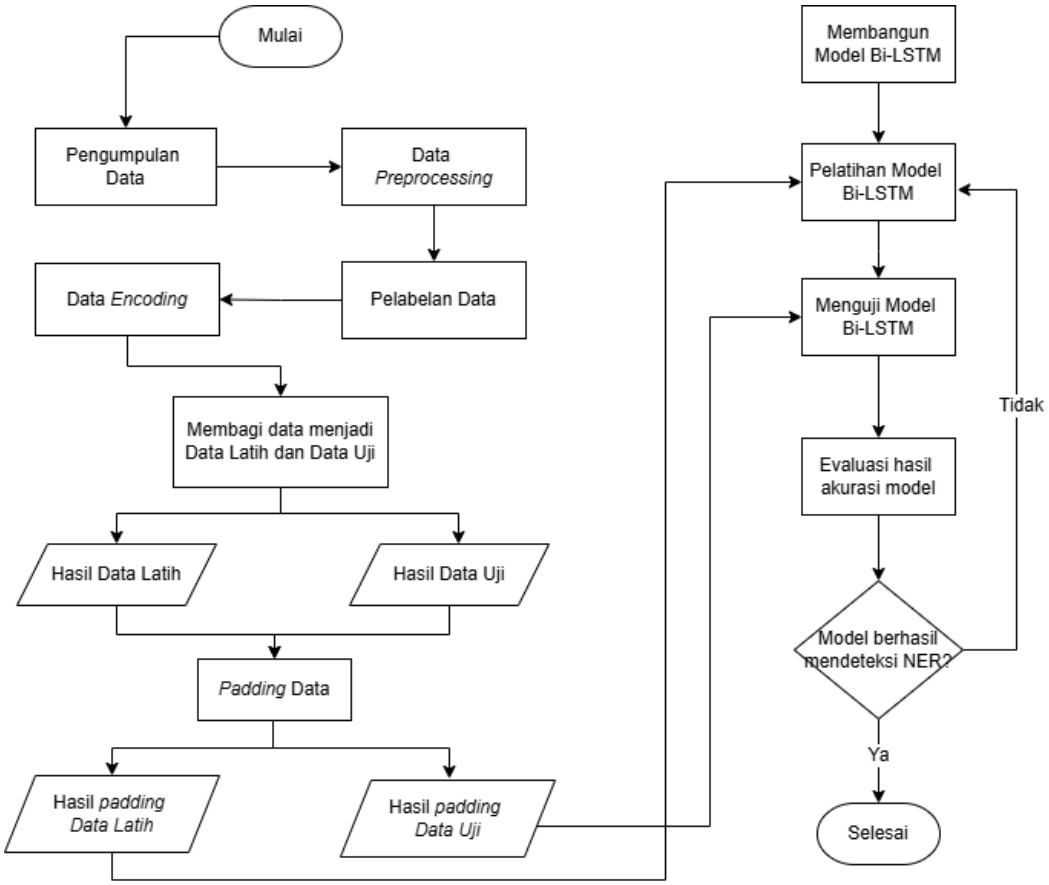
Saat ini, NER telah banyak dikembangkan untuk mengekstraksi teks di berbagai bidang dan mencapai hasil yang baik [3]. Namun, penting untuk diingat bahwa pengembangan NER tidak dapat secara langsung diterapkan dari satu bidang ke bidang lain, karena setiap bidang memiliki entitas yang unik. Sebagai contoh, dalam bidang biomedis, Dang et al. [4] mengembangkan NER untuk pengenalan entitas *Traditional Chinese Medicine* (TCM), dengan tujuan untuk mengenali entitas penting dalam TCM (meliputi nama ramuan, nama penyakit, gejala, dan efek samping) secara otomatis. Dalam penelitian lain yang dilakukan oleh Fudholi et al. [5], juga melakukan pengembangan NER dalam bidang biomedis untuk pengenalan nama obat Indonesia dengan entitas nama obat, kandungan dan komposisi. Dari contoh tersebut, dapat dilihat bahwa meskipun NER telah berhasil diterapkan dalam bidang biomedis, namun penting untuk menyesuaikannya dalam konteks tertentu, seperti yang perlu dilakukan dalam pengenalan entitas pada saham.

Pengembangan NER pada makanan juga pernah dilakukan oleh Cenikj et al. [6]. Penelitiannya tersebut bertujuan untuk mengidentifikasi entitas makanan dari data "*FoodBase*" yang telah dianotasi sebelumnya berdasarkan Hansard *taxonomy*. Penelitian lain oleh Fanny et al. [7], melakukan pengembangan NER untuk entitas seni tradisional di Indonesia. Penelitian ini bertujuan untuk membuat NER yang dapat mengidentifikasi entitas pada seni tradisional Indonesia, yang terdiri dari tujuh entitas yaitu *artistic entities, dance, performing arts, music, artistic figures,* dan *musical instruments.* Penelitian lain oleh Luthfi et al. [2], bertujuan untuk mengidentifikasi dan mengotentikasi perawi teks Hadits. Penelitian ini bertujuan untuk mengidentifikasi entitas Narrator pada hadits. Data yang digunakan adalah kumpulan teks hadits dari buku Hadits Bukhari, sebanyak seratus dua hadits. Dengan mengidentifikasi satu entitas yaitu narrator.

NER memiliki peran penting dalam *Natural Language Processing* (NLP), seperti pada klasifikasi teks, peringkasan teks, mengoptimalkan algoritma pencarian dan sistem rekomendasi berbasis teks [8]. Terdapat empat teknik dalam pengembangan NER yang umum digunakan yaitu melalui *rule based*, *unsupervised learning based*, *supervised learning based* dan *deep learning based*. Pendekatan *deep learning* merupakan salah satu pendekatan yang menghasilkan akurasi yang paling baik. NER berbasis *deep learning* memahami hubungan *semantic* dan sintaks antar kata, sehingga memberikan identifikasi yang lebih akurat [9]. Untuk melakukan pengembangan NER, *deep learning* menjadi model pendekatan yang sangat dominan dalam beberapa tahun terakhir. Karena NER yang dibangun menggunakan *deep learning* memiliki hasil akurasi yang lebih tinggi *Bidirectional Long Short-Term Memory* (Bi-LSTM) merupakan algoritma *deep learning* yang dikembangkan dari *Long Short-Term Memory* (LSTM). Bi-LSTM menerima *input* dari dua arah dan mampu memanfaatkan informasi dari kedua sisi. Algoritma ini, merupakan alat yang ampuh untuk memodelkan ketergantungan sekuensial antara kata dan frasa di kedua arah kalimat [10]. Bi-LSTM telah banyak digunakan pada penelitian sebelumnya dan menghasilkan hasil yang cukup baik.

Berdasarkan permasalahan dan penelitian sebelumnya, penelitian ini dilakukan untuk mengembangkan model NER untuk mengidentifikasi dan mengklasifikasikan kata secara otomatis pada kalimat yang memuat informasi tentang saham di Indonesia. Tujuan penelitian ini adalah untuk mengembangkan model NER menggunakan metode Bi-LSTM yang dapat mengidentifikasi lima entitas saham yaitu nama perusahaan, kode saham, indeks saham, sektor industri dan sub sektor pada kalimat berbahasa Indonesia yang mengandung informasi tentang saham. Algoritma Bi-LSTM dipilih karena dapat menangkap informasi dari dua arah pada data teks dan dapat menangkap informasi dengan komputasi yang lebih cepat. Dataset yang digunakan dalam penelitian ini merupakan data perusahaan saham yang termasuk dalam indeks LQ45 dan IDX30. Kontribusi utama dalam penelitian ini adalah pembuatan model Bi-LSTM untuk mengidentifikasi entitas penting pada data saham di Indonesia. Selain itu, diharapkan dapat memberikan gambaran secara komprehensif mengenai evaluasi model agar dapat digunakan sebagai landasan ilmiah untuk kasus serupa.

II. METODE PENELITIAN



Gambar 1. Metode penelitian

Gambar 1 menunjukkan metode penelitian yang dimulai dari pengumpulan data, *preprocessing*, pembuatan model Bi-LSTM hingga pengujian model.

A. Deskripsi Data

Teknik pengumpulan data yang digunakan adalah *scraping*, yaitu mengekstraksi informasi dari halaman website. Pada penelitian ini, *scraping* dilakukan menggunakan bahasa pemrograman Python dengan *library* BeautifulSoup. Data yang digunakan berupa judul berita yang diambil dari Google Search, menggunakan *keyword* nama perusahaan, kode saham, indeks saham, sektor industri dan sub sektor yang telah ditetapkan sebelumnya. Tabel 1 merupakan contoh *keyword* yang digunakan dan hasil judul berita yang diperoleh.

TABEL 1
CONTOH *KEYWORD* PENCARIAN DATASET

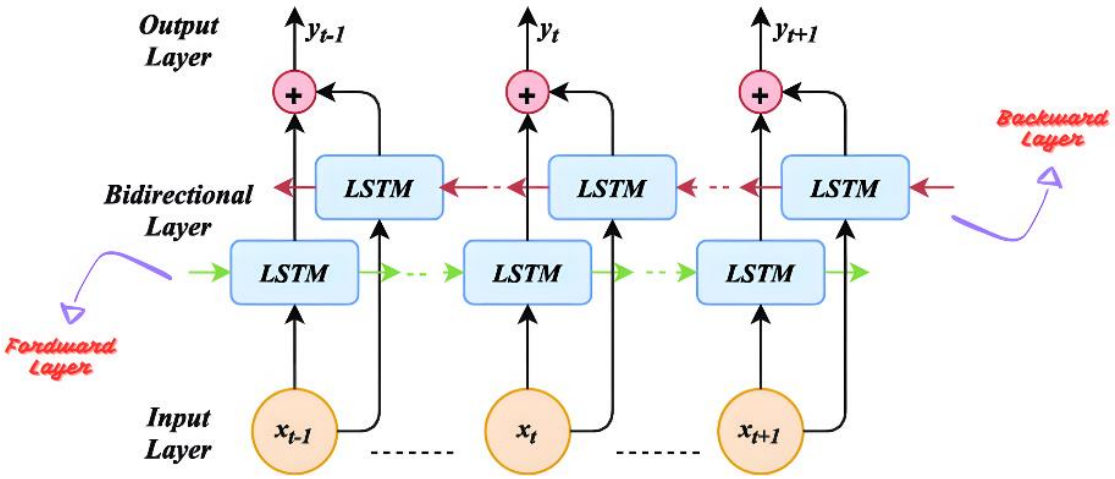
No	*Keyword*	Judul Berita
1	Saham TGRA	TGRA Mau Terbitkan Green Bond Rp1 Triliun Untuk Tambahan Modal Kerja
2	Saham PT Bank Artha Graha Internasional Tbk	PT Bank Artha Graha Internasional Tbk Hasilkan Peningkatan dan Komitmen Tingkatkan Keamanan Digital
3	Saham Sektor Batu Bara	Daftar 86 Saham Energi, Batu Bara, Minyak dan Gas di Pasar Modal 2024
4	Indeks Saham IDX80	Intip Daftar Saham IDX80 Periode Februari-Juli 2024 yang Berlaku Hari Ini
5	Saham TGRA	TGRA Mau Terbitkan Green Bond Rp1 Triliun Untuk Tambahan Modal Kerja

Data yang telah terkumpul akan disimpan dalam satu file dengan kolom “Judul Berita” dengan tipe data string. Seluruh data yang terkumpul merupakan kalimat yang mengandung entitas yang dibutuhkan dalam penelitian ini yaitu nama perusahaan, kode saham, sektor industri, sub sektor dan indeks saham. Berikut adalah penjelasan untuk masing-masing entitas yang digunakan dalam dataset:

1. Nama perusahaan, yaitu nama emiten atau perusahaan publik yang termasuk dalam indeks LQ45 dan IDX30. Terdapat 45 nama perusahaan yang ditetapkan oleh Bursa Efek Indonesia sebagai bagian dari indeks tersebut.
2. Kode saham yaitu kode unik yang mewakili perusahaan di pasar modal. Kode saham terdiri dari empat huruf kapital unik yang mewakili sebuah perusahaan. Contohnya adalah KLBF, BBRI, BBNI, GOTO dan lain sebagainya.
3. Sektor Industri yaitu sektor dari sebuah perusahaan yang menggambarkan perusahaan tersebut bergerak pada bidang tertentu, misalnya kesehatan, perindustrian, teknologi dan lain sebagainya. Terdapat 9 jenis sektor industri yang digunakan.
4. Sub Sektor menggambarkan fokus perusahaan bergerak pada bidang tertentu yang lebih spesifik daripada sektor industri, misalnya bidang kesehatan terdapat beberapa sub sektor diantaranya sub sektor farmasi, penyedia jasa kesehatan, peralatan kesehatan dan lain sebagainya. Terdapat 18 jenis sub sektor yang digunakan.
5. Indeks Saham yaitu sebuah pengukuran untuk sejumlah saham yang dihitung untuk mengkategorikan saham oleh Bursa Efek Indonesia. Misalnya IHSG, IDX30, IDX Technology dan lain sebagainya. Pada penelitian ini, hanya menggunakan indeks LQ45, IHSG dan IDX30.

B. Bidirectional Long Short-Term Memory (Bi-LSTM)

Bidirectional Long-Short Term Memory (Bi-LSTM) merupakan pengembangan dari algoritma *Long-Short Term Memory* (LSTM), yang diusulkan oleh Schuster dan Paliwal. Bi-LSTM menjalankan unit LSTM untuk berjalan pada dua arah yaitu maju (*forward)* dan mundur (*backward).* Kombinasi arah LSTM tersebut memungkinkan model untuk dapat menangkap konteks informasi dari masa lalu dan masa depan [10].



Gambar 2. Arsitektur Bi-LSTM [11]

Arsitektur Bi-LSTM dengan dua lapisan LSTM ditunjukkan pada Gambar 2. Dalam Bi-LSTM terdapat tiga *layer* utama yaitu *input layer*, *bidirectional layer* (*forward layer* dan *backward layer*) dan *output layer*. Setiap lapisan memiliki peran penting dalam pemrosesan data secara berurutan dan mendukung kemampuan Bi-LSTM dalam memahami hubungan kontekstual dalam data sekuensial.

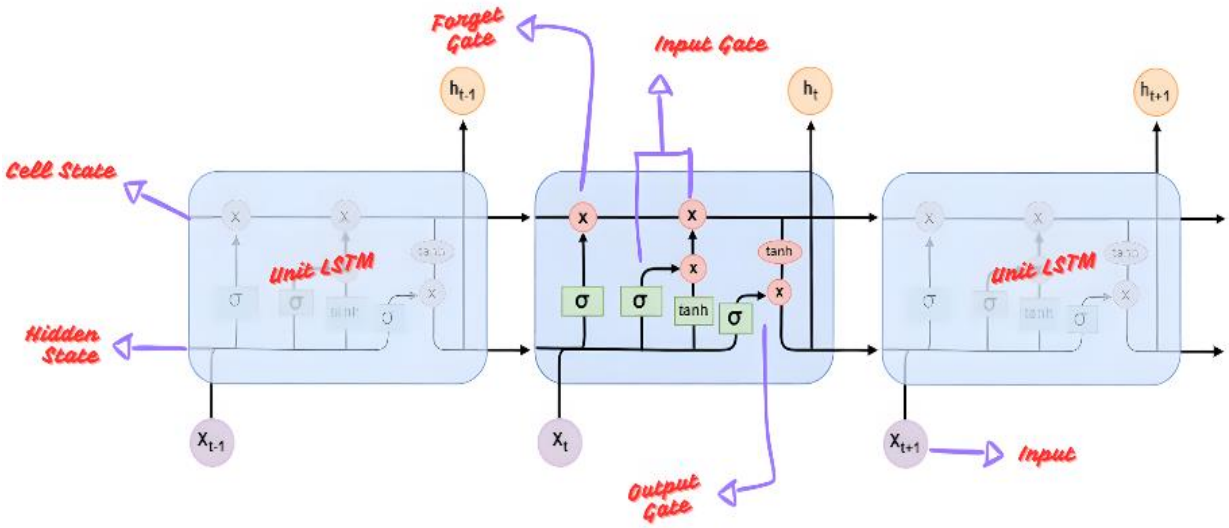
1) *Input Layer:*

Input layer merupakan lapisan pertama dalam arsitektur Bi-LSTM. Lapisan ini berfungsi untuk menerima data masukan dalam bentuk angka, yang umumnya berupa representasi vektor dari teks atau data sekuensial lainnya. Data tersebut kemudian diteruskan ke *bidirectional layer* untuk diproses lebih lanjut, sehingga model dapat menangkap informasi dari kedua arah secara simultan [10].

2) *Bidirectional Layer:*

Bidirectional layer merupakan lapisan ke-dua dalam arsitektur Bi-LSTM yang terdiri dari *forward* dan *backward layer* yang memproses *input* dari dua arah. *Forward layer* memproses data dari awal ke akhir urutan, sementara *backward layer* memprosesnya dari akhir ke awal. Kombinasi kedua lapisan ini memungkinkan model untuk menangkap konteks dari kedua sisi, meningkatkan pemahaman terhadap dependensi dalam data sekuensial.

p-ISSN : 2443-2210
e-ISSN : 2443-2229



Gambar 3. Arsitektur LSTM [11]

Masing-masing *layer* dalam *bidirectional layer* terdiri dari rangkaian unit LSTM yang memproses data secara sekuensial. Gambar 3 menunjukkan arsitektur dalam setiap unit LSTM, yang terdiri dari tiga *gate* atau gerbang utama yaitu *forget gate, input gate,* dan *output gate.*

a. *Forget Gate*

Forget gate adalah jalur pertama yang dilewati dalam pemprosesan di unit LSTM. *Forget gate* berfungsi untuk memutuskan informasi apa yang akan dibuang dari *cell state.* Terdapat satu fungsi aktivasi *sigmoid* (σ) di dalamnya. Fungsi *sigmoid* tersebut bekerja sebagai penyaring informasi yang akan dimasukkan ke *cell state. Sigmoid* mengubah setiap *input*an data menjadi angka antara 0 dan 1. Angka 1 menunjukkan informasi penting untuk disimpan semua. Sedangkan *output* 0 menunjukkan informasi tersebut tidak penting untuk disimpan. Informasi yang teridentifikasi penting akan disimpan di *cell state* dan disimpan sampai proses berikutnya. Perhitungan pada *forget gate* dapat dilihat pada persamaan (1).

$$f_t = \sigma(W_f . [h_{t-1}, x_t] + b_f) \quad (1)$$

Forget gate dihitung dengan mengevaluasi nilai *input* (X_t) dan nilai *hidden input* dari proses sebelumnya (h_{t-1}), dikalikan dengan matriks bobot (W) dan ditambahkan dengan nilai bias (b). Kemudian dikali dengan fungsi *sigmoid* (σ) yang akan menghasilkan *output* antara 0 dan 1.

b. *Input Gate*

Gate ke-dua pada unit LSTM adalah *input gate*. *Gate* ini berfungsi untuk memutuskan informasi baru yang akan disimpan ke dalam *cell state.* Sama seperti *forget gate*, *input gate* juga mengambil dua jenis *input* yaitu *input* saat ini (x_t) dan nilai *hidden state* (h_{t-1}) dari proses sebelumnya. *Input* ini kemudian diolah melalui fungsi *sigmoid* (σ) dan *hyperbolic tangent* ($tanh$). Kombinasi kedua fungsi ini memungkinkan LSTM untuk secara selektif memperbarui sel memori dengan mempertimbangkan informasi baru yang masuk serta konteks dari *timestep* sebelumnya [12].

Fungsi *sigmoid* digunakan untuk mengontrol seberapa banyak informasi baru harus disaring, fungsi ini akan menghasilkan keluaran berupa angka 0 hingga 1. Perhitungan untuk fungsi *sigmoid* dapat dilihat pada persamaan (2).

$$i_t = \sigma(W_i . [h_{t-1}, x_t] + b_i) \quad (2)$$

Persamaan tersebut menunjukkan persamaan untuk fungsi *sigmoid* pada *input gate.* Di mana i_t adalah nilai dari *input gate*, σ adalah fungsi *sigmoid*, W_i merupakan bobot untuk nilai *input*, h_{t-1} adalah nilai dari *hidden layer* dari proses sebelumnya, x_t adalah nilai *input* pada waktu ke t dan b_i merupakan bias pada *input gate.*

Sementara, fungsi *tanh* digunakan untuk menghasilkan kandidat nilai baru yang mungkin akan diperbarui dalam sel memori, dengan keluaran -1 hingga 1. Perhitungan dari fungsi *tanh* ini dihitung dengan persamaan (3)

$$\tilde{C}_t = tanh\,(W_c . [h_{t-1}, x_t] + b_c) \quad (3)$$

Persamaan tersebut menunjukkan persamaan untuk fungsi *tanh* pada *input gate.* Di mana C_t adalah nilai kandidat *cell* state, $tanh$ adalah fungsi *hyperbolic tangent*, W_c merupakan bobot untuk nilai *input*, h_{t-1} adalah nilai dari *hidden layer* dari proses sebelumnya, x_t adalah nilai *input* pada waktu ke t dan b_t merupakan bias pada *input gate.*

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (4)$$

Untuk memperbarui nilai *cell state*, hasil fungsi *sigmoid* dan *tanh* dihitung dengan menggunakan persamaan (4). Persamaan tersebut menunjukkan persamaan untuk memperbarui *cell state*. Di mana C_t adalah nilai baru *cell state*, f_t adalah nilai *forget gate*, C_{t1} merupakan nilai *cell state* saat ini atau nilai yang disimpan dari proses sebelumnya, i_t adalah nilai *input gate* dan $\tilde{C}_t$ adalah nilai kandidat *memory cell state*.

c. *Output gate*

Output gate berfungsi untuk memutuskan informasi yang akan diteruskan menjadi *hidden state* pada *short-term memory*. *Output gate* dalam LSTM bekerja untuk mengontrol seberapa banyak informasi dari memori jangka panjang (*cell state*) yang akan dijadikan bagian dari memori jangka pendek (*hidden state*) pada suatu *timestep*. Sama seperti *gate* sebelumnya, *output gate* mempunyai dua *input* dan dua fungsi aktivasi. Masing-masing fungsi aktivasi mempunyai fungsi yang sama seperti pada *gate* sebelumnya. Perhitungan pada *output gate* dapat dilihat pada persamaan (5) dan (6).

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_t \quad (5)$$

$$h_t = o_t * tanh\,(C_t) \quad (6)$$

Persamaan (5) tersebut menunjukkan persamaan pada *output gate*. Di mana o_t adalah nilai *output gate*, σ adalah fungsi *sigmoid*, h_{t-1} adalah nilai dari *hidden layer* dari proses sebelumnya, x_t adalah nilai *input* pada waktu ke t dan b_t merupakan bias pada *output gate*. Persamaan (6) menunjukkan h_t adalah *output hidden state*, o_t adalah nilai *output gate*, $tanh$ adalah fungsi *hyperbolic tangent* dan C_t nilai *memory cell state* yang baru.

Data akan diproses melalui setiap unit LSTM menggunakan masing-masing gerbang (*gate*) di dalamnya. Hasil akhir dari output *gate* tersebut kemudian diteruskan dan diproses lebih lanjut pada *output layer*.

3) *Output Layer:*

Hasil dari *forward* dan *backward layer* dalam Bi-LSTM digabungkan dan diproses untuk menghasilkan *output* akhir. Pada tahap ini, model mengintegrasikan informasi menggunakan *direct sum* dari kedua arah untuk menghasilkan prediksi akhir. Perhitungan untuk *output layer* Bi-LSTM dapat dilihat pada persamaan (7).

$$h_t = h_t^{\,forward} \oplus h_t^{\,backward} \quad (7)$$

Persamaan (7) menunjukkan persamaan pada *final output* Bi-LSTM. Di mana h_t adalah nilai *final output*, $h_t^{\,forward}$ adalah nilai dari *forward layer* dan $h_t^{\,backward}$ adalah nilai dari *backward layer*. Untuk menghasilkan h_t maka nilai dari *forward layer* dijumlahkan dengan *backward layer* menggunakan $\oplus$ atau *direct sum*. *Direct sum* menggabungkan elemen dengan mempertahankan identitas asalnya. Setiap elemen dari masing-masing *layer* dipasangkan secara terpisah, menghasilkan pasangan yang merepresentasikan hubungan antara elemen dari *layer* [11].

C. Evaluasi Model

Pada tahap ini, model akan dievaluasi dengan menggunakan 3 *metric* evaluasi yaitu *precision*, *recall* dan F1-*score*. Berikut tiga evaluasi model yang akan digunakan untuk mengukur kinerja model NER [13]:

1. *Precision* adalah matrik evaluasi yang bertujuan untuk mengukur prediksi positif yang dihasilkan oleh model. Perhitungan untuk menentukan *precision* dapat dilihat pada persamaan (8).

$$Precision = \frac{Tp}{Tp+Fp} \quad (8)$$

Persamaan (8) tersebut menunjukkan persamaan untuk menghitung *precision*, di mana *True Positive* atau Tp dibagi dengan Tp yang telah ditambahkan dengan *False Positive* atau Fp. Dengan mengukur *precision*, kinerja model dalam mengidentifikasi entitas yang benar dapat dievaluasi, sekaligus memastikan bahwa model tidak salah mengidentifikasi token yang tidak relevan.

2. *Recall* adalah matrik evaluasi yang bertujuan untuk mengukur proporsi prediksi positif yang benar dari seluruh kejadian positif sebenarnya. Perhitungan untuk menentukan *recall* dapat dilihat pada persamaan (9).

$$Recall = \frac{Tp}{Tp+Fn} \quad (9)$$

Persamaan (9) tersebut menunjukkan persamaan untuk menghitung *recall*, di mana Tp dibagi dengan Tp yang telah ditambahkan dengan *False Negative* atau Fn. Dengan mengukur *recall*, kemampuan model dalam mengidentifikasi semua entitas yang ada dapat dievaluasi.

3. *F1-score* adalah matrik evaluasi yang bertujuan untuk mengukur keseimbangan antara *precision* dan *recall*. Perhitungan untuk menentukan *precision* dapat dilihat pada persamaan (10).

$$\text{F1-}score = 2 * \frac{Precision*Recall}{Precision+Recall} \quad (10)$$

Persamaan (10) tersebut menunjukkan persamaan untuk menghitung F1-*score*, yaitu dengan mengalikan *precision* dan *recall*, kemudian dibagi dengan hasil penjumlahannya, dan dikalikan dua. Dengan persamaan tersebut, F1-*score* dapat memberikan evaluasi tentang kemampuan model dalam mendeteksi entitas secara keseluruhan.

III. HASIL DAN PEMBAHASAN

A. Pengumpulan Data

Data yang berhasil dikumpulkan berjumlah 4048 data dan dijadikan dalam satu file pada satu kolom yang diberi nama "Judul Berita" dengan tipe data string. Tabel 2 menunjukkan contoh hasil dataset yang telah berhasil dikumpulkan dan akan digunakan untuk proses selanjutnya.

TABEL 2
CONTOH HASIL DATASET

No	Judul Berita
1	Saham BBTN Dibuka Melesat, Antrian Beli Terbanyak di Harga Segini
2	Apa Itu Indeks LQ45? Simak Penjelasan Lengkap!
3	Daftar 46 Saham Teknologi yang Tercatat di Pasar Modal Indonesia 2024
4	Bedah Saham PT Aspirasi Hidup Indonesia Tbk yang Punya Prospek Baik di 2024
5	Saham Sub Sektor Media Hiburan Terdaftar di BEI 2024, Cek Saham Incaranmu

B. Data Preprocessing

Adapun tahap dalam *Preprocessing* adalah *remove punctuation* atau menghapus tanda baca yang tidak diperlukan pada data, seperti tanda baca "(), ., / \ < >". *Remove Stopwords* atau menghapus kata-kata hubung yang tidak dibutuhkan pada data, seperti "Perusahaan ini didirikan pada tahun 1990" menjadi "Perusahaan didirikan tahun 1990". Kemudian, tokenisasi yaitu proses memecah kalimat menjadi token-token tunggal, seperti pada kalimat "Saham BRI termasuk indeks LQ45" menjadi "Saham, BRI, termasuk, indkes, LQ45".

Setelah proses tokenisasi, data akan dipecah menjadi token per token independen dan disimpan menjadi dataframe baru. Kemudian ditambahkan kolom "*sentence*" sebagai penanda indeks kalimat untuk masing-masing token. Pemecahan kata dan penambahan kolom ini dilakukan untuk mempermudah proses pelabelan dan *encoding*. Hasil dari data tokenisasi yang telah dipecah dan dibuat menjadi dataframe baru yang terdiri dari kolom *index*, *sentence* dan token ditunjukkan pada Tabel 3. Setelah proses pemecahan ini data berjumlah sebanyak 33,205 token.

TABEL 3
HASIL DATAFRAME BARU

No	Sentence	Token
1	sentence: 1	ACES
2	sentence: 1	Catat
3	sentence: 1	Rekor
4	sentence: 1	Penjualan
5	sentence: 1	Tertinggi
6	sentence: 2	Indeks
7	sentence: 2	LQ45
8	sentence: 2	Makin
9	sentence: 2	Naik

C. Pelabelan Data

Pelabelan data merupakan tahap memberi label pada masing-masing token. Terdapat lima jenis label yang sudah dijelaskan sebelumnya dan ditambahkan dengan label 'O' atau *outside*, yaitu untuk menandai token yang tidak termasuk dari salah satu entitas. Proses pelabelan dilakukan pada masing-masing token yang telah dipecah pada tahap sebelumnya. Teknik pelabelan dilakukan secara manual menggunakan NER *annotations tools* dan excel. Hasil pelabelan dapat dilihat pada Tabel 4.

TABEL 4
HASIL DATAFRAME BARU

No	Token	Tag
1	ACES	Kode Saham
2	Catat	O
3	Rekor	O
4	Penjualan	O
5	Indeks	O

No	Token	Tag
6	LQ45	Indeks Saham
7	PT GoTo Gojek Tbk	Perusahaan

Setelah proses pelabelan, dapat dilihat distribusi masing-masing entitas pada dataset. Tabel 5 menunjukkan jumlah masing-masing entitas pada dataset. Di mana token dengan label O adalah token yang paling mendominasi dalam data yaitu berjumlah 29064. Hal tersebut karena sebagian besar kalimat tersusun dari kata yang menjelaskan peristiwa atau kejadian yang terkait dengan entitas. Sedangkan, token dengan jumlah paling rendah adalah Perusahaan dengan jumlah 789.

TABEL 5
DISTRIBUSI ENTITAS

No	Entitas	Jumlah
1	Sub Sektor	793
2	Perusahaan	789
3	Sektor Industri	838
4	Indeks Saham	815
5	Kode Saham	906
6	O	29064

D. Data Encoding

Encoding adalah proses mengubah kata menjadi representasi numerik. *Encoding* dilakukan agar data dapat diproses oleh model sebelum diubah ke representasi vektor pada proses *embedding*. Dengan mengubah kata menjadi numerik akan mempermudah proses *embedding*. Pada penelitian ini, teknik *encoding* yang digunakan adalah pengkodean yang paling sederhana yang disebut dengan *index-based method*. *Index-based method encoding* adalah teknik *encoding* dengan mengubah setiap kata atau token menjadi indeks numerik yang mewakili token atau kata unik [14].

Teknik *index-based method* dilakukan dengan memetakan setiap kata atau elemen unik pada data menjadi angka atau indeks yang unik. Proses *encoding* ini diterapkan dengan membuat kamus untuk pemetaan yang terdiri dari dua jenis yaitu kamus yang berisi indeks numerik menjadi token atau kata dan token menjadi indeks numerik. Tahap *encoding* ini, diterapkan pada setiap token dan label yang ada di dalam data. Label *encoding* yang digunakan dapat dilihat pada tabel 6.

TABEL 6
DESKRIPSI LABEL *ENCODING*

No	Entitas	Label_Enode
1	Indeks Saham	5
2	Sub Sektor	4
3	Kode Saham	3
4	Perusahaan	2
5	Sektor Industri	1
6	O	0

Hasil dari *encoding* ini yang nantinya akan digunakan untuk proses selanjutnya. Tahap ini diperlukan karena model Bi-LSTM yang akan dibangun hanya dapat memproses data dalam representasi numerik, sebelum diubah menjadi vektor.

Gambar 4 merupakan hasil data yang telah dipetakan menjadi representasi numerik. Hasil *encoding* disimpan pada kolom baru yaitu “token_encode” yang berisi hasil *encode* untuk masing-masing token dan “label_encode” yang berisi hasil representasi numerik dari label.

p-ISSN : 2443-2210
e-ISSN : 2443-2229

	sentence	token	label	token_encode	label_encode
0	sentence: 1	ACES	Kode Saham	1052	0
1	sentence: 1	Catat	O	2828	5
2	sentence: 1	Rekor	O	3605	5
3	sentence: 1	Penjualan	O	1818	5
4	sentence: 1	Tertinggi	O	1666	5
5	sentence: 1	Tahun	O	266	5
6	sentence: 1	2024	O	316	5
7	sentence: 2	Saham	O	1165	5
8	sentence: 2	ACES	Kode Saham	1052	0
9	sentence: 2	Naik	O	2199	5

Gambar 4. Hasil *Encoding*

E. Pembagian Data

Setelah proses *encoding*, data akan dibagi menjadi dua yaitu data latih dan data uji. Namun sebelumnya, data tersebut akan disatukan kembali dari token-token menjadi sebuah kalimat seperti semula, berdasarkan nomor kalimat pada kolom "*sentence*". Proses ini diperlukan untuk menyusun kembali data agar model dapat mengetahui konteks antar entitasnya. Tabel 7 merupakan format data yang telah digabungkan.

TABEL 7
HASIL DATA YANG TELAH DIGABUNGKAN

No	Sentence	Token	Label	Token Encode	Label Encode
1	sentence: 0	[ACES, Catat, Rekor, Penjualan, Tertinggi, Tahun, 2024]	[Kode Saham, O, O, O, O, O, O]	[1052, 2828, 3605, 1818, 1666, 266, 316]	[0, 5, 5, 5, 5, 5, 5]
2	sentence: 10	[Program, Loyalitas, Pelanggan, ACES, Tingkatkan, Keuntungan, Perusahaan]	[O, O, O, Kode Saham, O, O, O]	[1245, 693, 1087, 1052, 532, 145, 248]	[5, 5, 5, 0, 5, 5, 5]
3	sentence: 100	[ANTM, Donasikan, Sebagian, Laba, Pembangunan, Teknologi]	[Kode Saham, O, O, O, O, O]	[1514, 1451, 2721, 461, 1511, 131]	[0, 5, 5, 5, 5, 5]
4	sentence: 1000	[Daftar, Emiten, Bagi, Dividen, Bulan, Juni, 2024, Termasuk, PT Aneka Tambang Tbk]	[O, O, O, O, O, O, O, O, Perusahaan]	[4104, 2911, 1649, 2893, 2886, 2777, 316, 2990, 853]	[5, 5, 5, 5, 5, 5, 5, 5, 4]
5	sentence: 1001	[PT Bank Jago Tbk, Menebar, Dividen, Rp, 132, Miliar, Intip]	[Perusahaan, O, O, O, O, O, O]	[853, 370, 2893, 2886, 1169, 1679, 2981, 281]	[4, 5, 5, 5, 5, 5, 5, 5]

Data yang telah digabungkan kemudian dibagi menggunakan fungsi *split* data dari Scikit-Learn, dengan pembagian 80% untuk *training* dan 20% untuk *testing*. Kolom yang digunakan adalah "token_encode" dan "label_encode". Hasil dari pembagian terdapat 3267 data untuk data *training* dengan jumlah token berjumlah 26584. Sedangkan untuk data *testing* berjumlah 817 dan 2481 token. Hasil dari pembagian data dapat dilihat pada Tabel 8.

TABEL 8
PEMBAGIAN DATA

Jenis Data	Jumlah Data	Jumlah Token
Data Latih	3267	26584
Data Uji	817	2481

F. *Padding*

Padding adalah proses menyamakan panjang data dengan menambah ataupun mengurangi panjang token. Dalam LSTM, "*cell state*" bekerja dengan terus memperbarui *sequence* setiap waktu. Sehingga, LSTM memerlukan *sequence* atau urutan panjang yang sama. Oleh karena itu, *padding* diperlukan pada model Bi-LSTM yang merupakan pengembangan dari LSTM, yang membutuhkan urutan data dengan jumlah yang konsisten. Dalam tahap *padding*, data dapat dipotong atau ditambahkan sesuai dengan jumlah token yang diperlukan [15].

Nilai *default* penyisipan untuk penambahan token pada proses *padding* adalah 0. Namun, nilai penambahan *padding* juga bisa disesuaikan dengan nilai yang dibutuhkan. Terdapat 2 teknik dalam *padding*, diantaranya sebagai berikut [16]:

1. *Pre-Padding*, yaitu menambahkan elemen di awal data berdasarkan barisan terpanjang atau panjang yang sudah ditentukan.
2. *Post-Padding*, yaitu menambahkan elemen di akhir data berdasarkan barisan terpanjang atau panjang yang sudah ditentukan.

Pada tahap ini masing-masing data *encoding* akan disamakan panjangnya menjadi 50 token agar dapat diproses pada tahap selanjutnya. Data yang kurang dari 50 maka akan ditambah dengan token unik, sebaliknya data yang lebih dari Panjang tersebut akan dipotong. Teknik *padding* yang digunakan adalah *post padding*, yaitu menambahkan atau memotong token pada akhir kalimat.

Gambar 5. Ilustrasi *Padding* [16]

Gambar 5 merupakan ilustrasi cara kerja *padding.* Panjang *padding* ditetapkan menjadi 50, berdasarkan panjang maksimal token pada dataset. Penyisipan *padding* ditambahkan dengan nilai 0. Sehingga data yang panjang kalimat atau jumlah token dalam kalimatnya kurang dari 50, akan ditambahkan dengan nilai 0 di akhir token. Sedangkan data yang panjang kalimatnya lebih dari 50 token akan dipotong.

G. *Perancangan Struktur Model*

Model: "functional_8"

Layer (type)	Output Shape	Param #
input_layer_13 (InputLayer)	(None, 50)	0
embedding_14 (Embedding)	(None, 50, 100)	425,900
bidirectional_15 (Bidirectional)	(None, 50, 200)	160,800
time_distributed_11 (TimeDistributed)	(None, 50, 6)	1,206

Gambar 6. Struktur model Bi-LSTM

Gambar 6 adalah struktur model Bi-LSTM dibuat dengan empat *layer* utama yaitu *input layer*, *embedding layer*, *bidirectional layer* dan *timeDistributed layer*. Berikut penjelasan untuk masing-masing *layer* yang digunakan, yaitu:

1. Lapisan pertama adalah *input layer*, *layer* ini berfungsi untuk menetapkan dimensi *input* yang akan diterima model. Pada perancangan model Bi-LSTM ini, dimensi *input* ditetapkan menjadi 1 dimensi dengan panjang data 50.
2. *Embedding layer*, yaitu lapisan yang berfungsi untuk mengubah data *input* menjadi dense vectors yang berguna untuk memudahkan model dalam memproses data. Lapisan ini membantu model untuk merepresentasikan hubungan semantik antar data. *Output* dari lapisan ini berupa *dense vectors* dengan panjang 50 data, di mana setiap vektor mewakili satu elemen dari data *input* dan 100 dimensi.
3. *Bidirectional layer*, yaitu lapisan dua arah LSTM yang berfungsi untuk memproses *input* data dari dua arah *forward* (maju) dan *backward* (mundur). Dalam setiap lapisan terdapat unit-unit LSTM yang berjumlah 100 unit. Pada arah *forward*, model memproses data dari urutan *input* pertama menuju ke unit-unit LSTM dan menghasilkan *output* yang disimpan di *cell state*. Pada arah *backward*, model memproses dari urutan *input* terakhir dari data kemudian menuju ke unit LSTM dan disimpan juga ke *cell state*. Masing-masing hasil dari *layer* ini disimpan dan menghasilkan *output* berupa vektor berdimensi 100.
4. *TimeDistributed layer*, yaitu lapisan yang berfungsi untuk menerapkan *Dense layer* atau lapisan penghubung padat pada setiap data *input*. Pada lapisan ini, akan menerapkan *dense layer* pada setiap *vector* yang dihasilkan dari *output layer* sebelumnya untuk menghasilkan prediksi.

H. Training dan Testing Model Bi-LSTM

Proses *training* dilakukan menggunakan data latih untuk melatih model Bi-LSTM mengenali entitas yang sudah diubah dalam representasi vektor. Setelah pelatihan selesai evaluasi model menggunakan data uji, hasilnya akan membantu untuk memahami seberapa baik kinerja model. Jika model tidak berhasil mengidentifikasi entitas, maka akan diulang ke proses *training*. Namun, jika berhasil akan dilanjutkan ke proses selanjutnya yaitu evaluasi.

Proses pelatihan model dilakukan menggunakan data yang telah disiapkan sebelumnya sebesar 80% dengan jumlah data sebanyak 3267. Distribusi masing-masing entitas pada data latih dapat dilihat pada Tabel 9 berikut.

TABEL 9
DISTRIBUSI ENTITAS PADA DATA LATIH

Entitas	Jumlah
O	23271
Kode Saham	748
Perusahaan	619
Indeks Saham	646
Sektor Industri	663
Sub Sektor	637

Dalam data latih tersebut token dengan label "O" paling mendominasi pada data latih, sedangkan paling sedikit adalah token dengan label Perusahaan dengan jumlah 619 token. Dataset tersebut kemudian digunakan untuk melatih model Bi-LSTM yang telah dirancang sebelumnya. Proses pelatihan model dilakukan dalam 5 *epoch* dengan ukuran *batch* sebanyak 32 data. Pada setiap *epoch*, model akan melakukan pelatihan sebanyak jumlah data yaitu 3267 dan memperbarui bobot setiap mencapai 32 data.

Tabel 10 menunjukkan hasil *log training* model Bi-LSTM. Terdapat 2 kolom yang menunjukkan hasil *accuracy* dan *loss* selama proses *training* sebanyak 5 kali perulangan. Nilai *accuracy* menunjukkan seberapa besar model dapat mengenali data. Sedangkan nilai *loss* merupakan nilai kesalahan model dalam mengenali data. Hasil akhir pada *epoch* ke-5 dari proses *training* yaitu *accuracy* 0.992929 dengan *loss* 0.025016.

TABEL 10
HASIL *ACCURACY* DAN *LOSS TRAINING*

Epoch	*Accuracy*	*Loss*
1	0.960789	0.291353
2	0.979718	0.088985
3	0.979718	0.078621
4	0.981353	0.054649
5	0.992929	0.025016

Setelah model dilatih pada proses *training*, proses selanjutnya adalah pengujian model. Model diuji menggunakan data yang telah disiapkan sebelumnya sebesar 20% dari keseluruhan data. Dengan distribusi masing-masing entitas yang dapat dilihat pada Tabel 11.

TABEL 11
DISTRIBUSI ENTITAS PADA DATA UJI

Entitas	Jumlah
O	5793
Kode Saham	158
Perusahaan	170
Indeks Saham	169
Sektor Industri	175
Sub Sektor	156

I. Hasil Evaluasi Model

Evaluasi model dilakukan dengan membandingkan hasil *training* dan *testing* data dengan menggunakan nilai *precision*, *recall* dan F1-*score* yang dihasilkan. Nilai *accuracy* pada data latih mencapai 99.29% dan 99.81 % pada data uji. Sedangkan F1-*score* pada data latih mencapai 99.90% dan 99.81% pada data uji. Tabel 12 menunjukkan hasil evaluasi pada proses *training* dan *testing*.

TABEL 12
HASIL EVALUASI

	Accuracy	***Precision***	***Recall***	**F1-*score***
***Training* Model**	0.99292	0.99905	0.99905	0.99903
***Testing* Model**	0.99818	0.99817	0.99818	0.99810

Dapat disimpulkan bahwa model Bi-LSTM dapat mengidentifikasi entitas dengan sangat baik pada kedua data *training* maupun *testing*. Nilai F1-*score* yang tinggi menunjukkan bahwa model memiliki performa yang baik, hasil *precision* tersebut menunjukkan bahwa model dapat mengidentifikasi entitas dengan baik meskipun dengan sedikit kesalahan dan hasil *recall* yang tinggi menunjukkan model dapat mengidentifikasi entitas dengan baik.

Selain itu, evaluasi untuk setiap entitas juga dilakukan untuk mengetahui kinerja model pada masing-masing entitas. Tabel 13 menunjukkan hasil *classification report* untuk masing-masing entitas. Di mana hasil *precision*, *recall* dan F1-*score* entitas Indeks Saham pada proses *training* dan *testing*, mencapai 100%. Hal ini dikarenakan jumlah variasi pada Indeks Saham hanya ada 3 dengan data berjumlah 815, sehingga model dapat mengenali dan mengidentifikasi entitas Indeks Saham dengan sangat baik. Pada entitas Perusahaan, terdapat 45 variasi entitas dengan 789 data. Di mana terdapat 619 untuk *training* dan 170 data untuk *testing*. Dalam proses *training* menghasilkan nilai *precision*, *recall* dan F1-*score* dengan akurasi yang baik. Namun, pada proses *testing* entitas yang berjumlah 170, sangat mempengaruhi nilai masing-masing matriks evaluasi. Sama halnya dengan entitas kode saham, terdapat 45 jenis entitas dengan jumlah data sebanyak 906. Pada proses *testing* entitas Kode Saham berjumlah 158, yang menyebabkan nilai *precision*, *recall* dan F1-*score* juga cenderung rendah. Oleh karena itu, dapat disimpulkan bahwa jumlah data pada masing-masing entitas dan variasi entitas sangat berpengaruh dalam pengembangan model NER. Jumlah data yang sedikit dan distribusi entitas yang tidak seimbang sangat mempengaruhi hasil evaluasi model, seperti pada entitas Indeks Saham dan Kode Saham yang memperoleh nilai rendah pada proses *testing*.

TABEL 13
HASIL EVALUASI MASING-MASING ENTITAS

	Entitas	***Precision***	***Recall***	**F1-*score***
***Training* Model**	Kode Saham	1.00000	0.90890	0.95278
	Perusahaan	0.99280	0.92210	0.95209
	Indeks Saham	1.00000	1.00000	1.00000
	Sektor Industri	1.00000	1.00000	1.00000
	Sub Sektor	1.00000	0.96279	0.98581
***Testing* Model**	Kode Saham	0.99652	0.87922	0.92841
	Perusahaan	0.98413	0.76223	0.86392
	Indeks Saham	1.00000	1.00000	1.00000
	Sektor Industri	1.00000	0.99782	1.00000

Entitas	*Precision*	*Recall*	F1-*score*
Sub Sektor	1.00000	0.92928	0.96980

IV. SIMPULAN

Berdasarkan hasil penelitian tentang pengembangan *Named Entity Recognition* untuk entitas saham, dapat disimpulkan bahwa *Bidirectional Long Short-Term Memory* mampu mengidentifikasi entitas dengan baik. Dengan hasil evaluasi yang menunjukkan *accuracy* sebesar 99.81% pada data uji. Model dapat mengidentifikasi lima entitas saham yaitu nama perusahaan, kode saham, indeks saham, sektor industri dan sub sektor pada teks tentang saham dalam Bahasa Indonesia dengan tepat. Perancangan data sangat berpengaruh pada hasil akhir model. Jumlah data yang seimbang sangat penting untuk pengembangan model NER menggunakan Bi-LSTM. Selain itu, jumlah variasi token merupakan salah satu faktor yang perlu diperhatikan. Semakin banyak jumlah data yang dikumpulkan dengan struktur kalimat yang beragam akan membantu model untuk memahami lebih banyak token dan konteksnya.

Pada penelitian ini, objek yang digunakan hanya fokus pada perusahaan atau emiten saham yang termasuk dalam Indeks LQ45 dan IDX30 pada Bursa Efek Indonesia. Oleh karena itu, untuk penelitian selanjutnya dapat memperluas perusahaan saham pada indeks lain. Selain itu, dapat dibandingkan dengan percobaan Teknik tokenisasi seperti BIO tag atau semacamnya. Algoritma yang digunakan dalam penelitian ini adalah *Bidirectional Long Short-Term Memory*. Untuk melakukan penelitian selanjutnya dapat ditambahkan algoritma lain, seperti *Conditional Random Field* (CRF) agar mendapatkan hasil yang lebih baik.

UCAPAN TERIMA KASIH

Penulis menyampaikan penghargaan yang sebesar-besarnya kepada para pembimbing dan rekan-rekan di Universitas Islam Sultan Agung atas bimbingan dan masukan berharga selama proses penelitian ini.

DAFTAR PUSTAKA

[1] B. Widagdo, M. Jihadi, Y. Bachitar, O. E. Safitri and S. K. Singh, "Financial Ratio, Macro Economy, and Investment Risk on Sharia Stock Return," *Journal of Asian Finance, Economics and Business,* vol. 7, no. 12, pp. 919-926, 2020.

[2] E. T. Luthfi, E. T. Luthfi and E. T. Luthfi, "BERT based Named Entity Recognition for Automated Hadith Narrator Identification," *(IJACSA) International Journal of Advanced Computer Science and Applications,* vol. 13, no. 1, pp. 1-8, 2022.

[3] W. Huang, D. Hu, Z. Deng and J. Nie, "Named entity recognition for Chinese judgment documents based on BiLSTM and CRF," *Eurasip Journal on Image and Video Processing,* vol. 2020, no. 1, pp. 1-14, 2020.

[4] N. Deng, H. Fu and X. Chen, "Named Entity Recognition of Traditional Chinese Medicine Patents Based on BiLSTM-CRF," *Wireless Communications and Mobile Computing,* vol. 2021, pp. 1-12, 2021.

[5] D. H. Fudholi, R. A. N. Nayoan, A. F. Hidayatullah and D. B. Arianto, "A Hybrid CNN-BIlLSTM Model for Drug Named Entity Recognition," *Journal of Engineering Science and Technology,* vol. 17, no. 1, pp. 730-744, 2022.

[6] G. Cenikj, G. Popovski, R. Stojanov, B. K. Seljak and B. K. Seljak, "BuTTER: BidirecTional LSTM for Food Named-Entity Recognition," *Proceedings - 2020 IEEE International Conference on Big Data,* pp. 3550-3556, 2020.

[7] C. Fanny, C. Fanny and J. C. Young, "Implementation of Conditional Random Field for Named Entity Recognition in Indonesian Traditional Arts Digital Article," *International Journal of Multidisciplinary Research and Publications (IJMRAP),* vol. 5, no. 2, pp. 51-55, 2022.

[8] G. I. G. Situmeang, "Impact of Text Preprocessing on Named Entity Recognition Based on Conditional Random Field in Indonesian Text," *Jurnal Mantik,* vol. 6, no. 1, pp. 423-430, 2021.

[9] J. Li, A. Sun, J. Han and C. Li, "A Survey on Deep Learning for Named Entity Recognition," *IEEE Transactions on Knowledge and Data Engineering,* pp. 1-20, 2020.

[10] A. F. Hanif, T. B. Sasongko and T. B. Sasongko, "Perbandingan Kinerja LSTM, Bi-LSTM, dan GRU pada Klasifikasi Judul Berita Clickbait," *Indonesian Journal of Computer Science,* vol. 12, no. 4, pp. 2136-2159, 2023.

[11] I. Ihianle, A. Nwajana, S. Ebenuwa, R. Otuka, K. Owa and M. Orisatoki, "A Deep Learning Approach for Human Activities Recognition From Multimodal Sensing Devices," *IEEE Access,* vol. 8, pp. 179028-179038, 2020.

[12] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Physica D: Nonlinear Phenomena,* vol. 403, pp. 1-43, 2020.

[13] M. Joseph, "Comparative study of NER using Bi-LSTM-CRF with different word vectorization techniques on DNB documents," *Norwegian University of Life Sciences,* pp. 1-114, 2021.

[14] R. Anderson and N. Papernot, "Bad Characters: Imperceptible NLP Attacks," *IEEE Symposium on Security and Privacy (SP),* 2022.

[15] A. Lopez-del Rio, M. Martin, A. Perera-Lluna and R. Saidi, "Effect of sequence padding on the performance of deep learning models in archaeal protein functional prediction," *Scientific Reports,* vol. 1, p. 10, 2020.

[16] M. Dwarampudi and N. V. S. Reddy, "Effects of padding on LSTMs and CNNs," 2020. [Online]. Available: https://arxiv.org/abs/1903.07288.