Alih Bentuk Kalimat Non-Formal Menjadi Kalimat Formal Menggunakan Pendekatan Machine Translation

Nathanael Rezaputra [1)] Yulius Denny Prabowo [2)]

Informatika, Fakultas Industri Kreatif, Institut Teknologi dan Bisnis Kalbis
Jalan Pulomas Selatan Kav. 22, Jakarta Timur, 13210
[1)]Email: rezaputrawinata09@gmail.com
[2)]Email: yulius.prabowo@kalbis.ac.id

Abstract: *This research aims to apply the Long Short-Term Memory algorithm to the conversion of informal sentences into formal sentences using Indonesian sentences. Development of LSTM model application software to convert informal sentences to formal sentences in this study uses the incremental method. The BLEU evaluation determines whether or not the predicted sentence is accepted by measuring the closeness of the context of the formal sentence from the dataset. This research produces a website-based application. The conclusion obtained is that the training model that is built produces accuracy and loss values with a value of 0.7011 and 3.7401. Based on these results, an assessment of the training model is not good enough. This is due to several factors that cause the model training to experience overfitting.*
Keywords: *Natural Language Programming, LSTM, over form of formal sentences, BLEU*

Abstrak: *Penelitian ini bertujuan untuk menerapkan algoritma Long Short-Term Memory pada alih bentuk kalimat tidak formal menjadi kalimat formal menggunakan kalimat dalam bahasa Indonesia.Pembangunan perangkat lunak penerapan model LSTM untuk alih bentuk kalimat tidak formal menjadi kaliomat formal dalam penelitian ini menggunakan metode inkremental. Evaluasi BLEU menentukan diterima atau tidaknya kalimat hasil prediksi dengan mengukur kedekatan konteks kalimat formal dari dataset. Penelitian ini menghasilkan aplikasi berbasis website. Simpulan yang didapatkan adalah pelatihan model yang dibangun menghasilkan nilai akurasi dan loss dengan nilai 0,7011 dan 3,7401. Berdasarkan hasil tersebut memberikan penilaian terhadap model training belum cukup baik. Hal ini disebabkan beberapa faktor yang menyebabkan pelatihan model mengalami overfitting.*
Kata Kunci: *Pemrograman Bahasa Alami, LSTM, alih bentuk kalimat formal, BLEU*

I. PENDAHULUAN

Salah satu indikator keberhasilan seseorang dalam berkomunikasi adalah merangkai suatu kata dengan menerapkan struktur kalimat yang baik dan benar, baik itu komunikasi secara langsung maupun tidak langsung. Menurut KBBI, kalimat diartikan sebagai kesatuan ujar yang mengungkapkan suatu konsep pikiran dan perasaan. Selain itu, Kamus Besar Bahasa Indonesia juga mengartikan kalimat sebagai suatu satuan bahasa yang dapat berdiri sendiri, berintonasi, dan terdiri atas klausa dalam bahasa Indonesia. Kalimat sedniri mempunyai sejumlah jenis, dimana jenis-jenis kalimat tersebut digolongkan menjadi jenis-jenis kalimat berdasarkan fungsinya, jenis- jenis kalimat berdasarkan unsurnya, jenis-jenis kalimat berdasarkan

subjeknya, dan jenis-jenis kalimat berdasarkan strukturnya.

Kalimat tidak formal atau kalimat nonformal adalah kalimat yang menyimpang dari kaidah tata bahasa dan cenderung menggunakan bahasa gaul. Kalimat ini sering sekali digunakan didalam percakapan sehari-hari antar teman sebaya. Sedangkan kalimat formal merupakan kalimat yang ditulis berdasarkan kaidah bahasa Indoneisa yang baik dan benar. Secara sederhana, kalimat formal adalah kalimat yang ditulis berdasarkan aturn kaidah bahasa yang berlaku, sedangkan kalimat tidak formal atau nonformal adalah kalimat yang menyimpang dari kaidah bahasa.

Dalam komunikasi secara tidak langsung, struktur kalimat sangat diperlukan untuk membantu komunikan dapat dengan mudah mengerti pesan-pesan yang diberikan oleh komunikator. Suatu kalimat dapat terbentuk karena dipengaruhi oleh beberapa faktor, seperti dimensi ruang, waktu, serta suasana komunikator dalam penyampaian pesan kepada komunikan[1].

Frekuensi penggunaan kalimat formal yang cukup banyak tidak menjamin masyarakat Indonesia fasih dalam penggunaan kalimat formal, khususnya dalam menentukan kata-kata baku yang akan digunakan. Banyaknya bahasa asing yang ingin dikuasai, ketidakseimbangan penggunaan kalimat baku dan tidak baku, mengabaikan definisi suatu kalimat, serta penguasaan struktur bahasa yang rendah menjadi faktor sulitnya menentukan kalimat yang baku dalam penulisan. Penulisan kalimat tidak formal diteliti untuk dapat diterjemahkan kedalam kalimat formal dengan memperhatikan urutan kata pada suatu kalimat.

Evaluasi BLEU menentukan diterima atau tidaknya kalimat hasil prediksi dengan mengukur kedekatan konteks kalimat. Penelitian dengan tujuan mengembangkan model dengan menerapkan algoritma *Long Short Term Memory*(LSTM) dalam pemrosesan bahasa alami untuk alih bentuk kalimat tidak formal menjadi kalimat formal berbasis website serta menentukan faktor penunjang dan penghambat dari nilai akurasi yang dihasilkan dalam proses pembelajaran mesin.

II. METODE PENELITIAN

Proses alih bentuk dilakukan melalui proses pembelajaran mesin dengan algoritma *encoder-decoder* oleh arsitektur *Long Short Term Memory*. Untuk melakukan proses pembelajaran mesin, maka penelitian ini memerlukan data berupa kumpulan kalimat. Setiap kalimat memiliki dua jenis, yaitu kalimat formal dan kalimat non-formal. Dataset perlu dibangun karena peneliti tidak menemukan kumpulan korpus kalimat formal dan kalimat non-formal. Data kalimat formal dikumpulkan dari portal berita Kompas.id. Secara umum, penulisan paragraf berita menggunakan kata-kata baku. Sehingga, peneliti mengumpulkan kalimat-kalimat berita untuk dijadikan kumpulan kalimat formal pada dataset. Proses pertama dilakukan *web scrapper* untuk mendapatkan link berita dan *web crawler* untuk mendapatkan paragraf berita. Data yang berupa kumpulan berita diakuisisi dengan mengubahnya menjadi susunan kalimat dengan melakukan tokenisasi kalimat dari setiap *file* paragraf. Pra-pemrosesan kembali dilakukan oleh ekspresi reguler(*regular expression*) dan pengubahan alfabet menjadi huruf kecil (*lowercase*). Hasil dari proses akuisisi disimpan dan diterjemahkan oleh peneliti kedalam kalimat non-formal. Dengan

meninjau terjemahan kata berdasarkan kata-kata yang tidak sesuai dengan ejaan dalam KBBI (Kamus Besar Bahasa Indonesia) dan kalimat yang menyimpang dari kaidah tata bahasa serta cenderung menggunakan bahasa gaul.

Penelitian dilakukan dengan melakukan pra-pemrosesan dataset melalui metode *word embedding*, yaitu mengubah kata menjadi susunan *vector*. Melalui susunan *vector* tersebut, data siap menjadi *input* pembelejaran mesin. Pembelajaran mesin menggunakan algoritma LSTM untuk mendapatkan probabilitas kata per kata yang menjadi hasil prediksi terjemahan kalimat. Algoritma LSTM digunakan untuk metode encoder-decoder. Penelitian dalam pembangunan model dilakukan dengan membedakan matriks pembentuk *embedding*. Bobot matriks embedding yang digunakan sebanyak 3 matriks, yaitu dengan bobot 100, 200, dan 300. Masing-masing model dengan ketiga bobot matriks tersebut melalui proses testing dan evaluasi BLEU. Proses selanjutnya adalah membangun sistem dengan menggunakan model terbaik hasil evaluasi BLEU menggunakan kalimat masukkan diluar dari dataset.

Hasil dari penelitian merupakan aplikasi berbasis *website* yang berguna untuk memprediksi kalimat formal dari masukkan kalimat tidak formal.

Bahasa digunakan sebagai suatu komponen dalam berkomunikasi. Dalam berkomunikasi, bahasa dibagi menjadi 2 katagori, yaitu bahasa baku dan bahasa tidak baku. Komunikasi yang digunakan sehari-hari merupakan penggunaan bahasa yang tidak baku. Sehingga, bahasa yang tidak baku menjadi bahasa yang sangat sering digunakan. Dalam KBBI (Kamus Besar Bahasa Indonesia), satuan bahasa terkecil dan terlengkap maknanya disebut kalimat.

Kalimat formal adalah kalimat yang disusun berdasarkan bahasa Indonesia yang baik dan benar. Karena dikomposisikan dalam pemahaman dengan aturan-aturan kaidah bahasa Indonesia, maka kalimat formal memiliki kata-kata baku di dalamnya. Selain itu, kalimat formal memiliki ciri-ciri sebagai berikut: mengandung unsur kalimat dalam bahasa Indonesia, serta mempunyai proposisi di dalamnya, adanya penggunaan kata fungsi atau kata penghubung dan pelengkap yang digunakan secara efektif.

Kalimat tidak formal atau yang disebut kalimat non-formal merupakan kalimat yang bertentangan dari aturan struktur bahasa. Kalimat non-formal digunakan dalam diskusi hari demi hari di antara teman sebaya. Kalimat non-formal memiliki ciri-ciri sebagai berikut: adanya tambahan imbuhan, terjadi pengubahan kata, terjadi interferensi dalam kalimat yang digunakan, terjadi penyingkatan kata.

Tabel 1 Bentukan Kata

Bentukan Kata Formal	**Bentukan Kata Non-formal**
Berkata	Bilang
Hanya	Cuma, Cuman
Lepas	Copot
Tidak	Nggak
Kamu	Lu, Elu
Sampai	Sampe, Sampek

Pemrosesan Bahasa Alami atau Natural Language Processing(NLP) merupakan salah satu dari cabang ilmu kecerdasan buatan untuk mengolah bahasa yang secara umum digunakan oleh manusia untuk melakukan komunikasi[2]. Bahasa tersebut juga disebut sebagai bahasa natural. Pemrosesan bahasa alami muncul

untuk memudahkan pekerjaan pengguna dan untuk memenuhi keinginan berkomunikasi dengan komputer dalam bahasa alami. Pemrosesan bahasa alami memungkinkan komputer untuk berlatih bahasa. Masalah terbesar melatih komputer dalam pembelajaran mesin adalah konversi data yang ada menjadi deretan angka biner untuk dimengerti oleh komputer. Dalam proses pelatihan, pemrograman bahasa mesin memiliki beberapa tahapan proses pra-pemrosesan untuk mengolah suatu bahasa agar dapat diproses oleh komputer.

A. Tokenisasi

Tokenisasi merupakan salah satu pra-pemrosesan dalam pemrosesan bahasa alami untuk memproses suatu struktur kalimat. Melalui tokenisasi, susunan kalimat diuraikan menjadi suatu rangkaian kata/token(satuan terkecil) yang disimpan dalam bentuk list. Tokenisasi juga dapat menghilangkan beberapa karakter tertentu, seperti tanda baca. Token adalah turunan dari urutan karakter dalam beberapa dokumen tertentu yang dikelompokkan bersama sebagai unit semantik yang berguna untuk diproses[3]. Kata-kata dalam bahasa Indonesia sering dipisahkan dengan tanda spasi. Akan tetapi, tidak semua kata dalam bahasa Indonesia dengan satu kata memiliki makna. Terdapat frasa dalam bahasa Indonesia yang memungkinkan terdapat dua kata untuk satu makna.

Proses tokenisasi pada penelitian ini menggunakan kelas tokenizer dari pustaka keras.preprocessing.text. Kelas tokenizer melakukan dua tugas, yaitu membagi kalimat mejadi daftar kata yang sesuai, kemudian mengubah kata menjadi bilangan bulat[4]. Mengubah susunan kata menjadi suatu bilangan sangat penting karena pembelajaran yang dalam dan algoritma pembelajaran mesin bekerja dengan angka. Selain tokenisasi dan konversi integer, atribut word_index dari kelas tokenizer mengembalikan kamus kata-ke-indeks di mana kata-kata adalah kunci dan bilangan bulat yang sesuai adalah nilainya. Selain itu, proses akuisisi data pada penlitian ini diawali dengan melakukan tokenisasi kalimat dari setiap *file* paragraf berita dengan fungsi sent_token() dari NLTK. Tokenisasi kalimat yang menggunakan algoritma tanpa pengawasan untuk membangun model kata singkatan, gabungan kata, dan kata-kata yang memulai kalimat; dan kemudian menggunakan model itu untuk menemukan batasan kalimat[5]. Proses pelatihan model pada pustaka NLTK menggunakan *Punkt sentence tokenizer* untuk memisahkan token kata dari beberapa tanda baca. Punctuation tersebut diantaranya *';', ':', ',', '.', '!', '?'*[5].

B. Vector Semantic

Vector Semantic yang membuat hipotesis semantik vektor linguistik dengan mempelajari representasi makna kata[6]. Representasi ini digunakan dalam setiap aplikasi pemrosesan bahasa alami yang memanfaatkan makna, dan mendasari representasi kata kontekstual. Dengan *Vector Semantic* dapat membangun model komputasi yang berhasil menangani berbagai aspek makna kata[6]. Dalam penggunaan beberapa bahasa logis untuk mendefinisikan setiap kata, maka harus mendefinisikan kata dengan beberapa representasi tentang bagaimana kata itu digunakan oleh orang-orang yang sebenarnya dalam berbicara dan memahami.

Salah satu cara dalam *Vector Semantic* adalah dengan menentukan kata berdasarkan lingkungan atau distribusinya dalam penggunaan bahasa. Distribusi kata adalah serangkaian konteks di mana kata

itu muncul, kata-kata tetangga atau lingkungan tata bahasa. Idenya adalah bahwa dua kata yang muncul dalam distribusi yang sangat mirip (yang muncul bersamaan dengan kata-kata yang sangat mirip) cenderung memiliki arti yang sama[6].

Penelitian ini menggunakan *vector semantic* untuk menghitung sebuah kata dari distribusi kata-kata di sekitarnya. Sistem melakukan perhitungan nilai kesamaan semantik hasil dari vektor semantik word embedding. Secara singkat, *vector semantic* memberikan pengertian kepada sistem tentang kata-kata yang digunakan pada dataset serta kedekatan konteks kata tersebut.

1. Word2Vec

Word2vec adalah sekelompok model terkait yang digunakan untuk menghasilkan apa yang disebut *embeddings* kata. Model ini merupakan jaringan saraf dua lapis, yang dilatih untuk merekonstruksi konteks kata-kata linguistik [6]. Setelah pelatihan, model word2vec dapat digunakan untuk memetakan setiap kata ke vektor yang terdiri dari beberapa ratus elemen, yang mewakili hubungan kata itu dengan kata-kata lain. Vektor ini adalah lapisan tersembunyi jaringan saraf.

Untuk menggunakan seluruh vektor kata yang telah dibuat, diperlukan pembuatan *embedding layer* sebagai salah satu arsitektur model. Pembuatan *embedding layer* dapat menggunakan suatu model ruang vektor, seperti Gensim Model. Gensim (*Generate Similar*) Model merupakan model ruang vektor dan *toolkit topic modeling* yang paling kuat, efisien, dan tanpa kerumitan untuk mewujudkan pemodelan semantik tanpa pengawasan dari teks biasa. Gensim mengimplementasikan tf-idf, latent semantic analysis (LSA), Latent Dirichlet Analysis (LDA), dan lain-lain[7].

2. Embedding

Gagasan semantik vektor merepresentasikan sebuah kata sebagai suatu titik dalam ruang semantik multidimensi. Vektor untuk mewakili kata umumnya disebut *embeddings*, karena kata tersebut tertanam dalam ruang vektor tertentu. Pendekatan pembelajaran mesin terhadap pemrosesan bahasa alami membutuhkan kata-kata yang diekspresikan dalam bentuk vektor.

```
print(embeddings_dictionary["pembayaran"])

[ 2.07003876e-02 -3.05215079e-02 -7.02933073e-02 -1.17460785e-02
  3.94088349e-02 -1.52572533e-02 -9.20030922e-02  1.91521775e-02
 -4.59346697e-02  8.29700439e-04  3.41154896e-02  1.40415058e-02
 -2.67137106e-02 -1.08325580e-01 -2.16602430e-01 -5.02851980e-02
 -8.19856599e-02 -6.57655001e-02 -3.31080388e-02 -1.70048416e-01
 -1.26776733e-02  7.40104619e-02 -9.63080273e-03  2.56504755e-02
 -3.58062088e-02 -7.12843686e-02  8.70716199e-02  2.83000179e-02
  1.27654066e-02  3.35415341e-02  5.63987158e-02 -3.20734307e-02
 -1.01786681e-01 -5.18115045e-02  1.20849685e-01  2.69385248e-01
 -9.19702053e-02  4.15440723e-02 -3.56114528e-04  1.84271481e-02
 -3.55797634e-02  8.33225995e-02  3.77166718e-02 -1.65776089e-01
  1.11568563e-01  1.01661816e-01  2.36394300e-01 -1.05519101e-01
 -4.99906929e-02 -4.95282449e-02  7.26641808e-03 -7.34527335e-02
```

Gambar 1 Embedding Dictionary

Embeddings adalah teknik rekayasa fitur di mana kata-kata yang direpresentasikan sebagai vektor[8]. Penerapan *Embedding* memberikan masukkan kepada komputasi pembelajaran mesin melalui rangkaian angka biner yang merepresentasikan kata dalam data. Pada matriks *embedding*, susunan baris pada matriks akan mewakili nilai integer untuk kata dan kolom disesuaikan dengan panjang dimensi kata. Matriks ini berisi *word embeddings* untuk kata-kata dalam kalimat *input*. Matriks *embedding* kata ini digunakan untuk membuat layer *embedding* pada model LSTM.

C. Encoder-Decoder

Setiap masukan untuk komputasi bahasa direpresentasikan ke dalam *vector*. Sehingga *encoder* menanamkan input ke *vector*, dan *dekoder* menghasilkan beberapa keluaran dari vektor[9]. Perhitungan pada proses encoder-decoder membuat vektor kata dengan proses *embedding*, menghasilkan vektor *encoder*, menghasilkan vektor *decoder*, dan mengubah vektor *decoder* menjadi suatu kata dari *embedding dictionary*. Model yang dibangun merupakan struktur model *encoder-decoder*. Input ke *encoder* akan menjadi kalimat non-formal dan *output* akan menjadi *hidden state* dan *cell state* LSTM. Dalam mendefinisikan *decoder*, *decoder* akan memiliki dua input, yaitu *hidden state* dan *cell state* dari *encoder* dan kalimat input, yang sebenarnya akan menjadi kalimat keluaran dengan token yang ditambahkan di awal. Output dari LSTM *decoder* dilewatkan melalui *dense layer* untuk memprediksi *output decoder*.

Contoh alur proses encoder-decoder pada kalimat masukkan sebagai berikut:

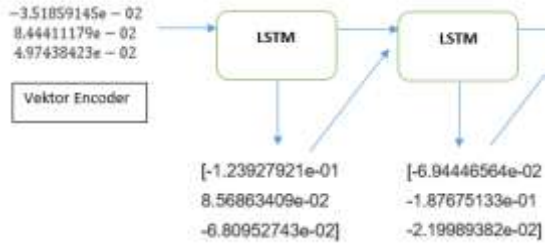
Tabel 2 Embedding Encoder

Kata	***Indeks Kata***	***Vektor Embedding***
"hal"	*[107]*	*[2.40940764e-03 -1.54547110e-01 -4.82013971e-02]*
"itu"	*[22]*	*[2.27030478e-02 -3.60983051e-02 3.03871557e-02]*

Vektor embedding digunakan untuk masukkan proses encoder. Masukkan pada node LSTM merupakan vector embedding dari setiap kata pada kalimat masukkan.

Gambar 2 Proses Encoder

Proses encoder menghasilkan vektor *encoder*. Melalui vektor *encoder* tersebut menjadi masukkan proses *decoder*.



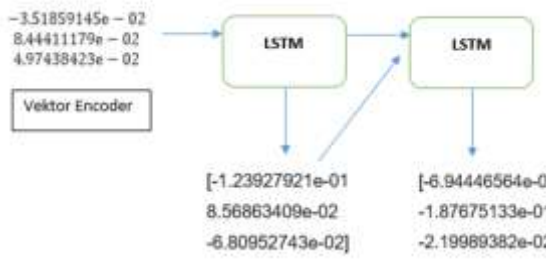
Gambar 3 Proses Decoder

Proses *decoder* menghasilkan vektor kata berupa *embedding dictionary*. Vektor kata yang telah dihasilkan menjadi masukkan kembali untuk node selanjutnya. Proses decoder dilakukan sebanyak kata yang menjadi input proses *encoder*.

Tabel 3 Embedding Decoder

Vektor Embedding	***Indeks Kata***	***Kata***
[-1.23927921e-01 8.56863409e-02 -6.8095274-02]	*[7]*	***"hal"***
[-6.94446564e-02 7.33305588e-02 1.59179881e-01]	*[92]*	***"tersebut"***

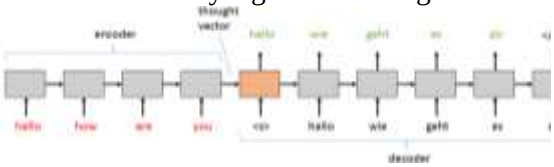
Proses tersebut mengubah *embedding* matriks menjadi indeks kata. Melalui indeks kata tersebut diubah menjadi kata dari bahasa asli.



Gambar 4 Proses Decoder

D. Machine Translation

Machine Translation (Terjemahan Mesin) merupakan penggunaan komputer untuk mengotomatisasi beberapa atau semua proses penerjemahan dari satu bahasa ke bahasa lainnya. Terjemahan, secara umum adalah upaya manusia yang sulit, menarik, dan intens, sekaya bidang kreativitas manusia lainnya[8]. Pembelajaran mesin tersebut menerjemahkan teks "sumber" dari satu bahasa ke bahasa yang berbeda "target".



Gambar 5 Terjemahan Mesin dengan LSTM

Dalam terjemahan mesin dibutuhkan dataset berupa kumpulan kalimat terjemahan dari bahasa "sumber" ke bahasa "target". Dalam terjemahan mesin menghasilkan suatu proses Many to Many. Yang dimaksud many to many adalah data masukan untuk pembelajaran mesin dengan jumlah yang banyak, serta keluaran dari hasil pembelajaran mesin merupakan probabilitas dari setiap kata terjemahan bahasa "sumber".

Penelitian menggunakan *Statistical Machine Translator*(SMT) dimana metode tersebut menggunakan hitung-hitungan statistik untuk menerjemahkan suatu kalimat bahasa tertentu ke bahasa yang lain. Teknologi SMT mengandalkan korpora bilingual seperti korpus terjemahan dan glosarium untuk sistem mempelajari pola bahasa. Serta data korpora monolingual yang digunakan untuk meningkatkan kelancaran *output*.

Penelitian menggunakan SMT karena bahasa Indonesia terus berkembang seiring berjalannya waktu, sehingga model *machine translat*ion juga dapat terus dikembangkan. Selain itu, model SMT dapat dibangun dalam jangka waktu singkat dan tidak memerlukan ahli linguistik untuk menerapkan atran bahasa ke sistem[10]. Akan tetapi, Model SMT memerlukan data yang banyak utnuk menghasilkan sistem terjemahan yang cerdas.

E. Long Short Term memory

LSTM adalah versi modifikasi dari jaringan saraf berulang, yang membuatnya lebih mudah untuk mengingat data masa lalu dalam memori. Masukkan pada node LSTM merupakan inputan data yang disimbolkan dengan Xt. Masing-masing gate pada LSTM dijelaskan sebagai berikut:

1. *Input Gate* digunakan untuk menemukan nilai dari input mana yang harus digunakan untuk memodifikasi memori.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C) \quad \text{... (1)}$$

   Dimana W dan b merupakan bobot dan bias dari setiap node LSTM, ht-1 merupakan hidden state dari node sebelumnya, dan x merupakan inputan pada node LSTM tersebut. Pada input gate menggunakan fungsi aktivasi sigmoid dan tanh. *Hyperbolic Tangent Activation(*tanh*) Function* merupakan fungsi aktivasi yang menghasilkan outpur dengan kisaran 1 dan -1[11].

2. *Forget Gate* digunakan untuk menemukan detail apa yang harus dibuang dari blok. Diputuskan oleh fungsi sigmoid. Fungsi aktivasi Sigmoid berguna dalam prediksi probabilitas yang membatasi output ke kisaran antara 0 hingga 1[11].

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right) \quad \text{... (2)}$$

Dimana W dan b merupakan bobot dan bias dari setiap node LSTM, ht-1 merupakan hidden state dari node sebelumnya, dan x merupakan inputan pada node LSTM tersebut. Fungsi aktivasi yang digunakan adalah sigmoid.

3. *Output Gate* menciptakan input dan memori blok yang digunakan untuk memutuskan output.

$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right) \quad \text{... (3)}$$

Node LSTM menghasilkan nilai hidden state yang disimbolkan ht. LSTM digunakan pada penelitian ini karena LSTM memiliki memori, sehingga LSTM dapat membaca data input secara keseluruhan dan mengingat konteks data. LSTM dapat memproses data secara sekuensial[12]. Pada penelitian ini, data berupa kalimat dapat diproses secara sekuensial, dimana kata per kata menjadi inputan pada proses LSTM. Selain itu, LSTM dapat memproses data sekuensial dengan panjang data yang tidak menentu. Dataset kalimat yang digunakan pada penelitian ini tidak memiliki panjang kalimat yang sama. LSTM dapat memproses data tersebut dengan baik.

F. Evaluasi BLEU

Cara mengukur kinerja terjemahan dengan menilai kedekatan terjemahan mesin dengan terjemahan manusia. Semakin dekat terjemahannya, maka semakin baik. Untuk menilai kualitas terjemahan mesin, seseorang mengukur kedekatannya dengan satu atau lebih terjemahan manusia referensi berdasarkan metrik numerik. Dengan demikian, sistem evaluasi *Machine Translation* kami membutuhkan metrik "kedekatan terjemahan" numerik dan kumpulan terjemahan referensi manusia yang berkualitas baik[13].

Proses evaluasi BLEU pada penelitian ini menggunakan dua pendekatan, yaitu evaluasi berdasarkan korpus dan kalimat. Dengan pendekatan korpus menggunakan corpus_bleu dan pendekatan kalimat menggunakan sentence_bleu. NLTK menyediakan fungsi sentence_bleu() untuk mengevaluasi kalimat kandidat terhadap satu atau lebih kalimat referensi[14]. NLTK juga menyediakan fungsi yang disebut corpus_bleu() untuk menghitung skor BLEU untuk beberapa kalimat seperti paragraf atau dokumen.

BLEU mengukur skor presisi berbasis statistik yang dimodifikasi antara hasil terjemahan otomatis dan terjemahan yang dirujuk menggunakan konstan bernama *brevity penalty* (BP).[15]

$$\mathrm{BP}_{\mathrm{BLEU}} = \begin{cases} 1 & \text{if } c > r \\ e^{\left(1-\frac{r}{c}\right)} & \text{if } x > r \end{cases} \quad \text{... (4)}$$

Simbol BP adalah *brevity penalty*, c adalah jumlah kata dari terjemahan otomatis, r adalah panjang referensi terjemahan yang efektif, dan pn adalah skor presisi yang dimodifikasi.

$$BLEU = BP.\exp\left(\sum_{n=1}^{N} w_n log\, p_n\right) \quad \ldots (5)$$

Nilai wn = 1 / N. Standar nilai N ke BLEU adalah 4 karena nilai presisi BLEU pada umumnya diukur hingga 4 gram[15]. Simbol pn merupakan jumlah berbasis statistik dalam hasil terjemahan yang sesuai dengan kalimat referensi dibagi dengan jumlah berbasis statistik dalam hasil kalimat terjemahan atau prediksi[15].

III. HASIL DAN PEMBAHASAN

Pada tahap pra-penelitian, peneliti mengidentifikasi masalah dengan mencari beberapa data penggunaan kalimat formal dalam Bahasa Indonesia berlanjut pada pengumpulan data. Data yang dikumpulkan merupakan kumpulan berita dari portal berita Kompas.id. Dari data tersebut, peneliti mengakuisisi data menjadi kumpulan kalimat yang disusuan dalam bentuk *Spreadsheet*. Dataset tersebut kemudian dilakukan penerjemahan ke dalam kalimat non-formal secara manual. Model yang diterapkan merupakan model deep learning, sehingga model tersebut bekerja dengan angka. Maka dari itu, hasil dari proses tokenisasi adalah mengubah kata-kata menjadi representasi vektor numerik.

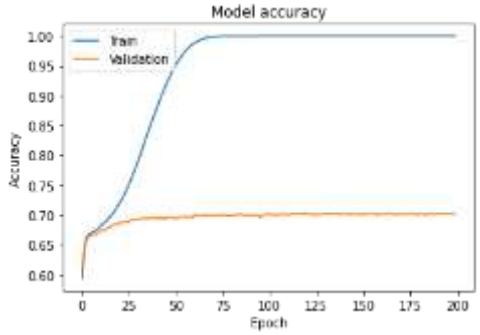
Proses penelitian dilakukan dengan menggunakan model inkremental.

A. Inkremental 1

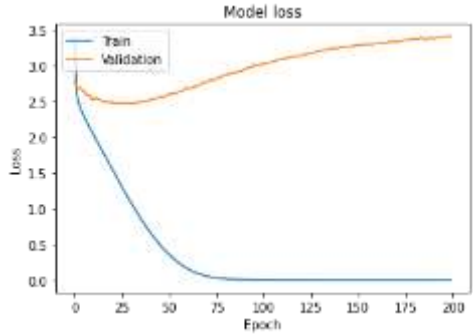
Hasil *training* dan validasi keseluruhan model. Angka tersebut merupakan hasil dari nilai akurasi dan validasi pada *epochs* terakhir.

Model	**Training**		**Validasi**	
	Akurasi	Loss	Akurasi	Loss
Model 100	0.9998	0,0011	0.7044	3.5888
Model 200	0.9999	0.0010	0.7046	3.5497
Model 300	0.9999	0.0010	0.7019	3.5861

Hasil proses *training* dan validasi menunjukkan nilai yang kurang baik untuk hasil validasi dan terlalu baik untuk hasil *training*. Hasil tersebut dikatakan *Overfitting* pada proses *training* data. Berikut hasil akurasi dan loss dari model 200.



Gambar 6 Hasil Akurasi Model 200



Gambar 7 Hasil Loss Model 200

Seluruh nilai akurasi dan *loss* pada *training* maupun validasi memiliki nilai yang tidak jauh berbeda. Penggunaan *corpus* khusus domain abstrak ilmiah, diamati bahwa kata-kata yang hanya muncul dalam konteks yang sama cenderung memiliki vektor yang lebih panjang daripada kata-kata dengan

frekuensi yang sama yang muncul dalam berbagai konteks[16].

B. Inkremental 2

Untuk memberikan penilaian pada masing-masing data, proses evaluasi BLEU dilakukan untuk mengukur kedekatan konteks kalimat formal dari dataset. Kedekatan konteks tersebut dinilai berdasarkan kalimat prediksi dengan kalimat aktual pada dataset.

Data	Model Encoder-Decoder	Nilai Skor BLEU
Training	Encoder-Decoder 100	0.7280457305781349
	Encoder-Decoder 200	0.7280473997795752
	Encoder-Decoder 300	0.7280823633362032
Testing	Encoder-Decoder 100	0.6992517960648524
	Encoder-Decoder 200	0.7112693303294064
	Encoder-Decoder 300	0.7018157593464326

Hasil dari nilai skor sentence_bleu baik data training maupun data testing.

Data	Model Encoder-Decoder	N-Gram	Nilai Skor BLEU
Training	Encoder-Decoder 100	BLEU-1	0.250243
		BLEU-2	0.500242
		BLEU-3	0.659946
		BLEU-4	0.707278
	Encoder-Decoder 200	BLEU-1	0.250193
		BLEU-2	0.500193
		BLEU-3	0.659907
		BLEU-4	0.707243
	Encoder-Decoder 300	BLEU-1	0.250253
		BLEU-2	0.500253
		BLEU-3	0.659955
		BLEU-4	0.707286
Testing	Encoder-Decoder 100	BLEU-1	0.221488
		BLEU-2	0.470625
		BLEU-3	0.636217
		BLEU-4	0.686021
	Encoder-Decoder 200	BLEU-1	0.235957
		BLEU-2	0.485754
		BLEU-3	0.648410
		BLEU-4	0.696960
	Encoder-Decoder 300	BLEU-1	0.223749
		BLEU-2	0.473021
		BLEU-3	0.638158
		BLEU-4	0.687765

Hasil dari nilai skor corpus_bleu baik data training maupun data testing. Nilai tertinggi didapatkan pada model encoder-decoder dengan bobot matriks embedding 200.

Pada nilai skor sentence_bleu, didapatkan nilai 0.7280473997795752 untuk data *training* dan 0.7112693303294064 untuk data *testing*.

Sedangkan pada nilai skor corpus_bleu() untuk data *training*, didapatkan nilai:

- BLEU-1: 0.250193,
- BLEU-2: 0.500193,
- BLEU-3: 0.659907;
- BLEU-4: 0.707243

untuk data *testing* dengan nilai:

- BLEU-1: 0.235957,
- BLEU-2: 0.485754,
- BLEU-3: 0.648410;
- BLEU-4: 0.696960

C. Inkremental 3

Pada pengujian inkremental tiga memiliki satu proses, yaitu pengembangan perangkat lunak serta dilakukan penamaan label pada langkah proses prediksi kalimat.

Penamaan Label	Proses
Debug 1	Pengiriman Data Kalimat request.script_root() (HTML)
	Penerimaan Data Kalimat request.args.get() (Python)
Debug 2	Preproces() Encoder_input_sequences (Python)
Debug 3	Translate() Ouput_sentences (Python)
Debug 4	Pengiriman kalimat hasil prediksi Jsonify() (Python)
	Penerimaan kalimat hasil prediksi Function(data).text() (HTML)

Hal ini dilakukan untuk memberikan informasi proses yang sedang dijalankan oleh aplikasi. Setiap label yang berhasil ditampilkan menandakan setiap proses dari satu fungsi ke fungsi lainnya berjalan dengan baik.

IV. SIMPULAN

Melalui penelitian yang telah dilakukan, dapat ditarik simpulan bahwa menggunakan pendekatan *machine translation* dapat menghasilkan model prediksi untuk sistem alih bentuk kalimat non-formal menjadi kalimat formal. Algoritma *long short term memory*(LSTM) dapat diterapkan untuk membuat model alih bentuk kalimat non-formal menjadi kalimat formal.

Pelatihan model yang dibangun menghasilkan nilai akurasi dan *loss* dengan nilai 0,7011 dan 3,7401. Berdasarkan hasil tersebut memberikan penilaian terhadap model *training* belum cukup baik. Hal ini disebabkan beberapa faktor yang menyebabkan pelatihan model mengalami *overfitting*. Diantaranya adalah:

- Dataset alih bentuk kalimat non-formal menjadi kalimat formal tidak menunjang kemampuan pelatihan model untuk mencapai akurasi generalisasi yang baik.
- Kata pada alih bentuk kalimat formal ke bentuk kalimat non-formal dalam dataset tidak konsisten. Upaya anotasi untuk dataset besar terpecah di antara beberapa kata dari latar belakang budaya yang berbeda.
- Prediksi kata yang tidak terdapat pada *dictionary word embedding*, secara sistem diberi indeks dari token <unk> pada dataset.

Penelitian menggunakan model dengan matriks vektor kata pada *word embedding* 100, 200, dan 300. Melalui hasil yang didapatkan, skor perhitungan pada pelatihan model tidak menunjukkan perbedaan yang signifikan. Kata-kata yang hanya muncul dalam konteks yang sama cenderung memiliki vektor yang lebih

panjang daripada kata-kata dengan frekuensi yang sama yang muncul dalam berbagai konteks. Sedangkan pada penelitian ini, dataset diambil dari portal berita dengan topik bisnis dan ekonomi.

Kemungkinan yang dapat dilakukan untuk pengembangan selanjutnya adalah Pengumpulan dataset yang lebih besar dengan konteks kata yang lebih luas, Membuat strategi *fine-tuning* untuk terjemahan mesin saraf lebih kuat dengan menggunakan beberapa teknik regularisasi.

DAFTAR RUJUKAN

[1] N. R. Sarfika, E. A. Maisa, and Windy Freska, *Keperawatan Dasar Dasar Komunikasi Terapeutik Dalam Keperawatan*. 2018.

[2] D. Khurana, A. Koli, K. Khatter, and S. Singh, "Natural Language Processing: State of The Art, Current Trends and Challenges," no. Figure 1, 2017.

[3] "Tokenization." [Online]. Available: https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html. [Accessed: 20-Jan-2020].

[4] "Backend - Keras Documentation." [Online]. Available: https://keras.io/backend/. [Accessed: 21-Jan-2020].

[5] "nltk.tokenize package — NLTK 3.5 documentation." [Online]. Available: https://www.nltk.org/api/nltk.tokenize.html#nltk.tokenize.punkt.PunktSentenceTokenizer. [Accessed: 14-Aug-2020].

[6] R. Klabunde, "Daniel Jurafsky/James H. Martin, Speech and Language Processing," *Zeitschrift für Sprachwiss.*, vol. 21, no. 1, Jan. 2002.

[7] K. Ade Sekarwati, L. Yuniar Banowosari, I. Made Wiryana, and D. Kerami, *Pengukuran Kemiripan Dokumen dengan Menggunakan Tools Gensim*, vol. 1, no. 1. 2015.

[8] R. Lebret and R. Collobert, "Word embeddings through Hellinger PCA," in *14th Conference of the European Chapter of the Association for Computational Linguistics 2014, EACL 2014*, 2014, pp. 482–490.

[9] A. J-P Tixier, "Notes on Deep Learning for NLP."

[10] Sreelekha S, "Statistical Vs Rule Based Machine Translation; A Case Study on Indian Language Perspective."

[11] J. A. Naser, "Neural networks - a brief introduction," *Proc. Am. Power Conf.*, vol. 53, no. pt 2, 1991.

[12] "Pengenalan Recurrent Neural Network (RNN) – Bagian 1 – Belajar Pembelajaran Mesin Indonesia." [Online]. Available: https://indoml.com/2018/04/04/pengenalan-rnn-bag-1/. [Accessed: 15-Aug-2020].

[13] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a Method for Automatic Evaluation of Machine Translation LK - https://rug.on.worldcat.org/oclc/204431620," *Annu. Meet. Assoc. Comput. Linguist. Ta - Tt* -, vol. 40, no. July, pp. 311–318, 2002.

[14] "nltk.translate.bleu_score — NLTK 3.5 documentation." [Online]. Available: https://www.nltk.org/_modules/nltk/translate/bleu_score.html. [Accessed: 15-Aug-2020].

[15] A. Hermanto, T. B. Adji, and N. A. Setiawan, "Recurrent neural network language model for English-Indonesian Machine Translation: Experimental study," *Proc. - 2015 Int. Conf. Sci. Inf. Technol. Big Data Spectr. Futur. Inf. Econ. ICSITech 2015*, 2016.

[16] B. Wilson and A. M. J. Schakel, "Controlled Experiments for Word Embeddings," 2015.