Analisis Pendapat Masyarakat terhadap Berita Kesehatan Indonesia menggunakan Pemodelan Kalimat berbasis LSTM

(*Indonesian Stance Analysis of Healthcare News using Sentence Embedding Based on LSTM*)

Esther Irawati Setiawan[1,3], Adriel Ferdianto[3], Joan Santoso[1,3], Yosi Kristian[3], Gunawan[3], Surya Sumpeno[1,2], Mauridhi Hery Purnomo[1,2]

***Abstract*—The uncertainty of health news content, which is spread on social media, raises the need for validation of the truth. One validation approach is to consider the opinion or attitudes of most people, which is called a stance on a topic, whether they support, oppose, or being neutral. This paper proposes a stance analysis model to classify the relationship between sentences so that it can recognize the correlation of the opinion of the writer in the headline of the problem claim. The proposed model uses several Long Short-Term Memory (LSTM), which represent the interrelationship of news for analysis of the relationship between a claim with other news. The formation of word representation vectors is carried out in conjunction with LSTM-based stance classification training. Sentence embedding is done to get the vector representation of sentences with LSTM. Each word in a sentence occupies one time-step in LSTM and the output of the last word is taken as a sentence representation. Based on the results of trials with the Indonesian health-related dataset that was built for this study, the proposed stance classification model was able to achieve an average F1-score value of 71%, with the supporting value 69%, opposing as much as 70%, and neutral 74%.**

***Intisari*—Adanya ketidakpastian konten berita kesehatan yang tersebar di media sosial memunculkan kebutuhan untuk validasi kebenaran informasi. Salah satu pendekatan validasi dengan mempertimbangkan pendapat atau sikap sebagian besar masyarakat yang diistilahkan sebagai *stance* terhadap topik tersebut, yaitu mendukung, menentang, atau netral. Makalah ini membahas usulan model analisis *stance* untuk memahami hubungan antar kalimat sehingga dapat mengenali korelasi pendapat penulis berita terhadap klaim permasalahan dari judulnya. Usulan model menggunakan beberapa *Long Short-Term Memory* (LSTM) yang merepresentasikan keterkaitan antar berita untuk mengklasifikasikan relasi antara suatu judul berita kandidat validasi dengan berita-berita lain. Pembentukan vektor representasi kata-kata dilakukan bersamaan dengan pelatihan analisis pendapat melalui klasifikasi yang berbasis LSTM. Pemodelan kalimat dilakukan untuk mendapatkan vektor representasi kalimat dengan LSTM. Tiap kata dalam suatu kalimat menempati satu langkah waktu dalam LSTM dan *output* dari kata terakhir diambil sebagai representasi kalimat. Berdasarkan hasil uji coba dengan *dataset* bahasa Indonesia bertopik kesehatan yang telah dibangun untuk makalah ini, model analisis *stance* yang diusulkan mampu meraih rata-rata nilai *F*1 71%, dengan rincian label mendukung 69%, menentang 70%, dan netral 74%.**

***Kata Kunci*— Analisis Pendapat, *Stance Classification*, *Deep Learning*, LSTM, *Sentence Embedding*, Bahasa Indonesia.**

I. PENDAHULUAN

Berkembangnya teknologi membuat penyebaran informasi semakin mudah dan cepat melalui media sosial (Facebook, Twitter), blog, atau situs resmi suatu lembaga [1]. Mudahnya penyebaran informasi ini menyebabkan berbagai isu di web yang kredibilitas informasinya berkurang. Biasanya kebenaran suatu informasi (klaim) di web dapat diperiksa dengan mencari berita-berita terkait dari sumber-sumber lain. Berita-berita tersebut secara tidak langsung menunjukkan pendapat atau tanggapan penulis berita terhadap klaim yang diperiksa kebenarannya, baik berita-berita tersebut mendukung kebenaran isu/berita, menentang kebenaran isu/berita, maupun hanya mengulangi atau mempertanyakan klaim tersebut (netral) [2].

Analisis pendapat (*stance)* merupakan bagian dari penambangan opini yang bertujuan untuk mengklasifikasikan data berupa teks sebagai salah satu dari tiga pendapat, yakni mendukung, menentang, atau netral terhadap proposisi atau target [3]. Targetnya mungkin seseorang, organisasi, kebijakan pemerintah, gerakan, produk, atau lainnya. Analisis pendapat dapat digunakan untuk mengetahui pandangan masyarakat atau sumber berita lainnya, mendukung atau menentang suatu klaim, untuk membantu proses validasi fakta dalam dunia kesehatan.

Berbagai penelitian dalam bidang pengolahan teks telah dilakukan untuk secara otomatis melakukan analisis pendapat dalam pengambilan informasi, peringkasan teks, dan penelitian lainnya [4], [5]. Perbedaan dengan analisis sentimen adalah analisis pendapat menentukan sikap seseorang terhadap target atau topik tertentu, sedangkan analisis sentimen mengklasifikasikan suatu tulisan di media

[1] *Departemen Teknik Elektro, Fakultas Teknologi Elektro dan Informatika Cerdas, Institut Teknologi Sepuluh Nopember, Kampus ITS, Sukolilo, Surabaya 60111, Jawa Timur, Indonesia (tlp: 031-5947302; fax: 031-5931237; e-mail: esther16@mhs.ee.its.ac.id, joan.santoso13@mhs.ee.its.ac.id, surya@ee.its.ac.id, hery@ee.its.ac.id)*

[2] *Departemen Teknik Komputer, Fakultas Teknologi Elektro dan Informatika Cerdas, Institut Teknologi Sepuluh Nopember, Kampus ITS, Sukolilo, Surabaya 60111, Jawa Timur, Indonesia (tlp: 031-5922936; e-mail: surya@ee.its.ac.id, hery@ee.its.ac.id)*

[3] *Departemen Teknik Informatika, Fakultas Sains dan Teknologi, Institut Sains dan Teknologi Terpadu Surabaya, Jl. Ngagel Jaya Tengah 73-77, Surabaya, Jawa Timur, Indonesia 60284 (tlp:031-5027920; e-mail: esther@stts.edu, adriel1@mhs.stts.edu, joan@stts.edu, yosi@stts.edu, gunawan@stts.edu)*

sosial sebagai positif, negatif, atau netral. Analisis pendapat dapat dilihat sebagai bagian dari penambangan opini [6]. Untuk deteksi berita palsu, analisis pendapat adalah metode yang umum digunakan [7]. Terdapat berbagai penelitian analisis sentimen dalam bahasa Indonesia, tetapi analisis pendapat hanya telah dilakukan terhadap tokoh politik dengan *Naïve Bayes*, SVM, dan *Logistic Regression* [8].

Makalah ini mempelajari hubungan antar kata dalam suatu berita untuk memperoleh secara otomatis pendapat penulis antar berita. Berbagai algoritme dapat digunakan untuk analisis pendapat, seperti pohon keputusan, *Random Forest*, dan K-NN [9]. Pembelajaran mendalam atau yang dikenal dengan istilah *deep learning* juga dapat digunakan untuk mempelajari hubungan tersebut. *Deep learning* dimanfaatkan dalam berbagai riset karena kemampuannya untuk mendapatkan informasi yang tidak dapat diperoleh dengan algoritme pembelajaran mesin biasa [10], [11].

Fokus makalah ini adalah analisis pendapat pada berita berbahasa Indonesia. Algoritme yang digunakan adalah LSTM, yang kinerjanya baik untuk analisis pendapat melakukan klasifikasi otomatis. Hal ini berdasarkan hasil penelitian yang menggunakan LSTM dan berhasil memperoleh akurasi tertinggi untuk analisis pendapat pada SemEval 2016 dengan arsitektur pembelajaran mendalam [12]. Berdasarkan percobaan yang telah dilakukan pada penelitian ini, penerapan pemodelan kalimat yang diistilahkan *Sentence Embedding* dapat meningkatkan akurasi dari analisis pendapat dengan algoritme LSTM. *Dataset* bahasa Indonesia yang digunakan pada penelitian ini dibangun dengan nama *Indonesian News Stance Dataset* (INStaD). Penyebaran hoaks di Indonesia yang terbanyak adalah dalam bidang kesehatan, sehingga penelitian ini difokuskan pada analisis pendapat berita kesehatan [1], [13]. Rincian *dataset* dijelaskan lebih lanjut pada bagian ketiga dari makalah ini.

Fitur yang digunakan untuk analisis pendapat adalah pemodelan kalimat dengan *autoencoder* LSTM. Selain itu, terdapat fitur tambahan berupa keberadaan tanda tanya, kata negasi, dan kemiripan antar judul berita. Penelitian sebelumnya memanfaatkan algoritme *Branch* LSTM, tetapi belum menggunakan fitur *Sentence Embedding* [14]. Penerapan analisis pendapat juga dilakukan untuk validasi berita bahasa Indonesia sebagai dasar untuk penelitian melawan berita palsu yang banyak beredar di masyarakat saat ini.

Makalah ini terdiri atas lima bagian. Bagian pertama membahas pendahuluan dan latar belakang. Bagian kedua menjelaskan mengenai penelitian terkait analisis pendapat melalui klasifikasi. Bagian ketiga dan keempat menjelaskan mengenai pembentukan *dataset* dan analisis pendapat dengan LSTM, sedangkan bagian kelima menjabarkan hasil eksperimen dan kesimpulan.

II. Aplikasi-Aplikasi Analisis Pendapat

Tujuan pengolahan teks yang populer saat ini adalah analisis sentimen dan kategorisasi teks [15], [16]. Namun, penerapan analisis pendapat mulai dilakukan pada berbagai bidang. Salah satunya adalah analisis pendapat terhadap debat daring pada penelitian berbasis leksikon sebagai fitur targetnya [17]. Analisis pendapat juga dapat menjadi dasar untuk penelitian yang menangani berita palsu atau hoaks [18], [19]. Analisis pendapat juga telah diterapkan pada media sosial Twitter dengan pendekatan dua tahap [20].

Penelitian analisis pendapat dalam bahasa Inggris melalui klasifikasi kalimat telah menggunakan berbagai metode, seperti *Latent Dirichlet Allocation* untuk analisis argumen [21]. Pada penelitian lainnya, analisis pendapat diterapkan pada *dataset* Emergent yang dibangun untuk penanganan rumor dengan algoritme Regresi Logistik [22]. *Multi Layer Perceptron* telah diterapkan pula [23]. *Conditional* LSTM *encoding* merupakan algoritme yang dapat digunakan untuk pengolahan pendapat pada media sosial [24].

Pendekatan lainnya adalah LSTM dan *Branch* LSTM, yang berasal dari sebuah penelitian [25]. Paper ini pada awalnya mengajukan metode analisis pendapat untuk komentar Twitter (*tweet*). Model *Branch* LSTM menggunakan lapis-lapis LSTM unit untuk memproses seluruh cabang *tweet*, sehingga informasi struktural dari suatu perbincangan dapat disatukan [14]. Penelitian ini terinspirasi dari Branch LSTM, dengan usulan metode untuk mengambil pendapat berdasarkan relasi antar judul berita kesehatan dengan LSTM.

Berbagai fitur telah digunakan sebagai *input* untuk analisis pendapat, antara lain kata-kata *bag-of-words*, keberadaan *url*, *POS tag*, dan sentimen [9]. Namun, *bag-of-words* memiliki kelemahan, yaitu fitur yang tidak relevan ikut diolah sehingga dapat menyebabkan penurunan kinerja algoritme [26]. Oleh karena itu, makalah ini menggunakan *word embedding* yang dapat memberikan makna semantik [16]. Makalah ini juga mengolah *input* tiga fitur tambahan sesuai dengan karakteristik dataset yang dibangun dan berdasarkan penelitian sebelumnya [14].

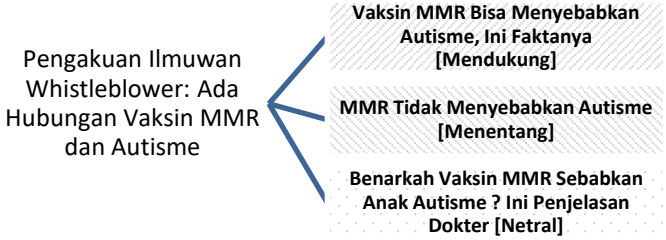
III. *Indonesian News Stance Dataset*

Dataset yang digunakan dalam makalah ini terdiri atas judul artikel-artikel berita berbahasa Indonesia yang terdiri atas klaim dan berita terkaitnya dalam bidang kesehatan. Bidang kesehatan dipilih karena banyaknya berita palsu yang beredar di bidang kesehatan [27]. *Dataset* ini dibangun dengan mengumpulkan judul-judul berita dari berbagai sumber web, seperti turnbackhoax.id, cekfakta.com, dan health.detik.com yang menampilkan berita terkini mengenai topik kesehatan yang populer bagi masyarakat.

Setiap klaim memiliki minimal lima berita terkait, dengan masing-masing berita terkait memiliki pendapat yang menanggapi klaimnya. Pendapat suatu berita terkait terhadap klaimnya dibagi menjadi tiga kategori, yakni mendukung, menentang, dan netral, sesuai dengan label pada penelitian sebelumnya [22]. Berita terkait merupakan berita yang judulnya berhubungan dengan klaim dan memiliki pendapat terhadap klaim. Klaim terhadap berita terkait memiliki hubungan *one-to-many*, dengan kata lain sebuah klaim dapat ditanggapi oleh lebih dari satu berita terkait, tapi sebuah berita terkait hanya dapat menanggapi sebuah klaim. Pelabelan dilakukan dengan melibatkan lebih dari satu pakar. Jumlah masing-masing label untuk dataset berita kesehatan bahasa Indonesia ditunjukkan pada Tabel I.

TABEL I
STATISTIK *DATASET* UNTUK ANALISIS PENDAPAT

Tipe	Total	Persentase
Mendukung	1.129	33,6%
Menentang	1.165	32,7%
Netral	1.160	33,7%



Gbr. 1 Contoh pendapat berita terhadap topik.

Dataset yang dikumpulkan memiliki label-label yang jumlahnya cukup seimbang satu sama lain, sehingga Jaringan Syaraf Tiruan (JST) dapat mempelajari fitur-fitur judul berita untuk tiap label secara seimbang. *Dataset* berisi total 500 klaim dan 3.454 berita terkait. Terdapat sebanyak 1.129 label menentang, 1.165 label netral, dan 1.160 label mendukung. Daftar klaim diperoleh dari berita kesehatan selama tahun 2014-2018 di Indonesia. Tahun 2014 dipilih karena merupakan tahun awal mula penyebaran berita hoaks di Indonesia [28]. Sedangkan artikel terkait diperoleh menggunakan bantuan mesin pencari otomatis berdasarkan kata-kata yang terkandung pada klaim.

IV. ANALISIS PENDAPAT DENGAN LSTM

Analisis pendapat bertujuan untuk mengklasifikasikan pendapat masyarakat terhadap target tertentu dalam teks. Dalam makalah ini telah dilakukan analisis pendapat-pendapat pada berita-berita kesehatan berbahasa Indonesia. Masing-masing berita diambil judulnya dan dilabeli pendapatnya, bahwa berita tersebut mendukung, menentang, atau netral terhadap suatu topik [5].

Judul-judul berita terkait dikategorikan pendapatnya masing-masing terhadap suatu judul berita yang memuat sebuah topik. Contoh kasus analisis pendapat adalah sebagai berikut: terdapat sebuah klaim ”Pengakuan Ilmuwan *Whistleblower*: Ada Hubungan Vaksin MMR dan Autisme” dengan judul berita yang mendukung adalah ”Vaksin MMR bisa Menyebabkan Autisme, Ini Faktanya”, yang menentang adalah ”MMR Tidak Menyebabkan Autisme” dan yang netral adalah ”Benarkah Vaksin MMR Sebabkan Anak Autisme? Ini Penjelasan Dokter.” Contoh struktur judul berita terdapat pada Gbr. 1.

Proses analisis pendapat dapat didefinisikan sebagai berikut. Terdapat sebuah judul berita yang merupakan klaim C dan sekumpulan judul berita kesehatan dengan topik terkait $R = \{ B_1, \dots, B_n \}$. Metode yang diusulkan akan menentukan sikap setiap berita berdasarkan judulnya, $Y = \{mendukung, menentang, netral\}$. Penentuan sikap akan menggunakan algoritme LSTM.

LSTM merupakan perkembangan dari JST yang terkenal dengan kemampuannya untuk mendapatkan informasi dari data berdasarkan urutan. LSTM memiliki kelebihan dibanding JST biasa dan arsitektur ini tepat digunakan untuk klasifikasi pendapat [29].

Gbr. 2 Arsitektur umum analisis pendapat.

LSTM dipilih untuk masalah analisis pendapat karena beberapa alasan. Alasan pertama yakni LSTM dapat mendapatkan informasi berdasarkan urutan pada kata-kata dalam tiap kalimat, sehingga masalah seperti negasi dapat diselesaikan. Kata-kata yang dianggap sebagai kata negasi antara lain tidak, bukan, jangan, tak, belum, mitos, hoaks, dan *hoax*. Alasan yang kedua yakni LSTM dapat menyimpan informasi dari kalimat-kalimat sebelumnya. Dibandingkan dengan JST biasa, LSTM sangat unggul, karena JST biasa tidak dapat memperhatikan urutan, sehingga banyak informasi yang hilang. Selain JST biasa, ada pula RNN yang juga dapat memperhatikan urutan. Namun, RNN sendiri memiliki masalah *vanishing gradient*, dengan RNN tidak dapat mengingat informasi urutan secara keseluruhan. Masalah ini diselesaikan dengan *gating* pada LSTM yang memberi LSTM kemampuan untuk mengingat informasi penting yang jauh. Arsitektur umum analisis pendapat ditunjukkan pada Gbr. 2.

Model yang diusulkan terdiri atas lima proses utama, yaitu praproses, pemetaan kata menjadi vektor, pemodelan kalimat, penggabungan fitur, dan klasifikasi dengan LSTM.

A. *Praproses*

Terdapat beberapa tahapan praproses setelah proses akuisisi data dilakukan. Tahap pertama dalam praproses adalah mendapatkan fitur tiap kalimat klaim atau berita terkait. Fitur-fitur yang diambil antara lain fitur ada tidaknya tanda tanya dan ada tidaknya kata negasi. Fitur-fitur tiap kalimat disimpan dalam bentuk vektor.

Setelah fitur-fitur tiap kalimat diperoleh, dilakukan pembersihan. Karakter-karakter yang bukan alfabet, seperti tanda tanya, titik dua, koma, dan sebagainya dihapus. Selain

itu, dilakukan proses perolehan kata dasar, yang disebut dengan *stemming*, terhadap tiap kata dalam kalimat-kalimat, sehingga kembali ke kata dasarnya. Perolehan kata dasar dilakukan dengan metode dari Sastrawi yang dikembangkan berdasarkan penelitian sebelumnya [30], [31].

Dari kata-kata yang telah melalui proses *stemming*, dibentuk daftar kata-kata yang memperoleh angka unik untuk tiap kata. Kemudian, proses pemodelan kalimat dilakukan dengan pemberian angka unik untuk tiap kalimat. Dalam daftar kalimat tersebut, tiap kalimatnya unik (tidak ada yang sama satu dengan yang lain).

B. Pemetaan Kata menjadi Vektor

Proses merepresentasikan kata dalam bentuk vektor dilakukan bersamaan dengan pelatihan model analisis dan disebut *on-trained word embedding* [32]. Nilai vektor-vektor kata diisi secara acak dan dilatih menggunakan metode JST, yang termasuk dalam arsitektur LSTM. *Word Embedding* merupakan sebuah teknik pemodelan bahasa dan pembelajaran fitur dalam bidang ilmu pengolahan bahasa alami. Setiap kata atau frasa dalam kosakata dipetakan ke vektor yang berisi bilangan-bilangan desimal. Pada lapis *embedding*, diberikan nilai vektor dengan tipe data bilangan real dengan panjang dan nilai-nilai tertentu untuk tiap *ID* dalam kamus kata. Panjang vektor merupakan salah satu parameter pelatihan yang akan diubah-ubah untuk tujuan percobaan. Parameter ini disebut *embedding size*. Nilai-nilai vektor untuk tiap *ID* kata dalam daftar kata yang didapat dari tabel *embedding*. Vektor untuk masing-masing *ID* kata pada awalnya diberi nilai acak. Nilai-nilai masing-masing vektor nantinya akan berubah seiring berjalannya proses pelatihan. Pelatihan dilakukan dengan menggunakan JST. Setelah pelatihan, kata-kata yang maknanya mirip memiliki vektor-vektor yang nilai-nilainya berdekatan.

C. Pemodelan Kalimat

Pemodelan kalimat digunakan untuk merepresentasikan kalimat dalam bentuk vektor yang merupakan kelanjutan dari proses pemetaan kata-kata menjadi vektor [33]. Jika *Word embedding* merupakan proses merepresentasikan tiap kata dengan sebuah vektor, maka pemodelan kalimat merupakan proses merepresentasikan tiap kalimat dengan sebuah vektor, berdasarkan kata-kata yang terkandung di dalamnya. Dengan kata lain, pemodelan kalimat bertujuan mengurangi dimensi data yang awalnya dua dimensi dengan ukuran (jumlah kata) * (banyaknya *embedding)* menjadi data satu dimensi dengan ukuran banyaknya *embedding*.

Hal ini perlu dilakukan karena normalnya LSTM hanya menerima vektor satu dimensi sebagai *input* untuk tiap langkah waktu. Selain itu, dengan pemodelan kalimat, relasi antar kata dalam sebuah kalimat, seperti konteks, akan ikut dipertimbangkan, sehingga cocok untuk tugas-tugas yang perlu memperhitungkan kemiripan semantik antar teks. Pemodelan kalimat didapatkan dengan menggunakan metode LSTM, dengan tiap vektor tiap kata dalam sebuah kalimat diberikan sebagai *input* ke LSTM pada sebuah langkah waktu. *Output* dari langkah waktu kata terakhir diambil sebagai vektor representasi kalimat.

Pemodelan kalimat pada makalah ini didasarkan pada *encoder* LSTM dari model *autoencoder* LSTM bersifat pembelajaran tanpa pengawasan yang ditujukan untuk representasi video [34]. Model *autoencoder* tersebut terdiri atas dua RNN, yakni *encoder* LSTM dan *decoder* LSTM. Pemanfaatan LSTM untuk pemodelan kalimat diperlihatkan pada Gbr. 3.

Input pada model ini berupa deretan vektor (potongan-potongan kecil gambar atau fitur). *Encoder* LSTM kemudian membaca *input* tersebut secara berurutan. Setelah *input* terakhir dibaca, *decoder* LSTM mengambil alih dan menghasilkan prediksi untuk deretan target. Deretan target sama dengan deretan *input*, tapi dalam urutan yang dibalik.

Dari model *autoencoder*, metode *encoder* LSTM dari model tersebut digunakan untuk pemodelan kalimat. Tiap kata dimasukkan ke sebuah LSTM pada langkah waktu sesuai urutan kata tersebut. LSTM pada tiap langkah waktu akan menghasilkan *output* vektor dengan panjang tertentu. Pada contoh di bawah, panjang *output* adalah tiga. Panjang *output* ini disamakan dengan parameter *embedding size*. Karena terdapat *padding* pada *sentence list*, LSTM pada langkah waktu yang merupakan *padding* tidak akan dihitung. *Sentence list* memiliki kalimat-kalimat yang memiliki panjang maksimum enam kata. Jika ada kalimat yang hanya memiliki empat kata, maka dua sisanya akan diberi angka *encoding* 0 sebagai *padding*. Hasil perhitungan untuk langkah waktu 5 dan 6 diabaikan, sehingga *output*-nya pun 0. *Output* LSTM untuk kata terakhir yang bukan *padding* diambil dan digunakan sebagai representasi kalimat tersebut.

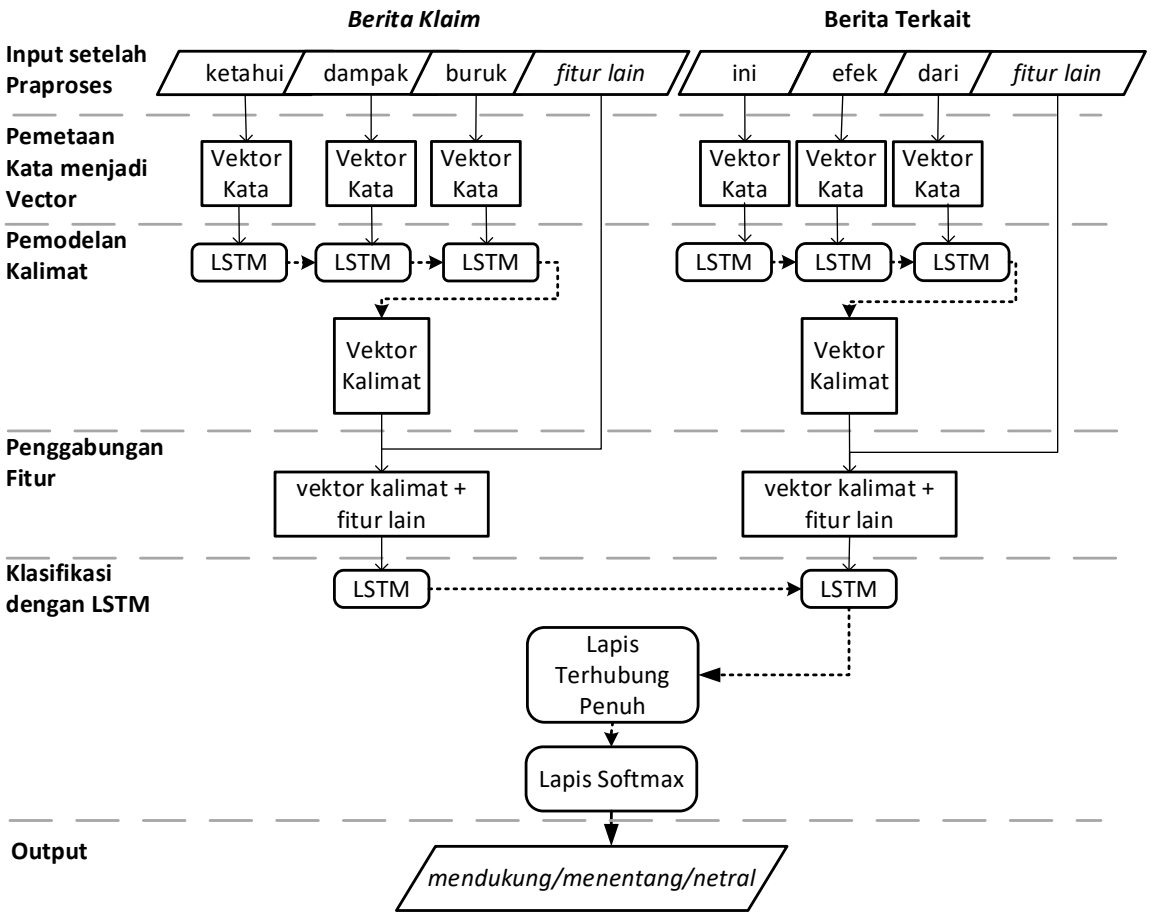
D. Penambahan Fitur pada Pemodelan Kalimat

Fitur tambahan digunakan untuk meningkatkan akurasi, sehingga JST yang digunakan tidak hanya melihat kalimat-kalimat yang diberikan sebagai faktor penentu *output*. Pada bagian praproses, tersimpan fitur-fitur tambahan berupa vektor dengan panjang jumlah fitur untuk tiap berita (kalimat).

Fitur-fitur tiap berita tersimpan dalam vektor tersendiri. Terdapat tiga fitur tambahan, yaitu keberadaan tanda tanya, keberadaan kata negasi, dan kemiripan judul berita klaim dengan judul berita terkaitnya. Vektor-vektor tersebut digabungkan (*concat*) pada vektor-vektor kalimat yang diperoleh dari pemodelan kalimat.

Setelah didapatkan vektor representasi untuk *input*, fitur tambahan dapat diberikan. Fitur tambahan berupa vektor dengan panjang jumlah fitur, yakni tiga. Setiap kalimat memiliki fitur tambahan sendiri, sehingga jika ada lima pasang kalimat, maka akan ada sepuluh vektor fitur (satu pasang kalimat berisi dua kalimat, yaitu klaim dan berita terkaitnya). Fitur yang digunakan adalah ada tidaknya tanda tanya, ada tidaknya kata negasi atau negatif, dan kemiripan sebuah judul berita terhadap klaimnya.

Fitur ada tidaknya tanda tanya direpresentasikan sebagai angka 1.0 atau 0.0. Jika dalam suatu kalimat terdapat tanda tanya, maka fitur ini direpresentasikan sebagai 1.0. Sebaliknya, jika dalam suatu kalimat tidak terdapat tanda tanya, maka fitur ini direpresentasikan sebagai 0.0. Fitur ini hanya diaplikasikan pada berita terkait. Pada kalimat klaim, fitur ini langsung



Gbr. 3 LSTM untuk pemodelan kalimat dan klasifkasi pendapat.

direpresentasikan 0.0, karena pada dasarnya klaim tidak akan memiliki tanda tanya.

Seperti fitur ada tidaknya tanda tanya, fitur ada tidaknya kata negasi juga direpresentasikan sebagai angka 1.0 atau 0.0, bergantung pada ada tidaknya kata negasi dalam suatu kalimat. Kata-kata yang dianggap sebagai kata negasi antara lain 'tidak', 'bukan', 'jangan', 'tak', 'belum', 'mitos', 'hoaks', dan 'hoax'.

Fitur kemiripan sebuah judul berita terhadap klaim dihitung menggunakan rumus *cosine similarity* pada (1). *Cosine similarity* merupakan sebuah pengukuran untuk kemiripan antar dua vektor bukan 0 dari sebuah *inner product space* yang mengukur *cosine* dari sudut di antaranya.

$$cosine\ similarity = \frac{\sum_1^n A_i\, B_i}{\sqrt{\sum_1^n A_i^2}\sqrt{\sum_1^n B_i^2}}. \quad (1)$$

A_i dan B_i merupakan komponen dari vektor *A* dan *B* secara berurutan. Dalam kasus ini, *A* merupakan vektor kalimat dari sebuah klaim dan *B* merupakan vektor kalimat lain. Karena semua kalimat akan diberi fitur ini, termasuk kalimat berita klaim, maka untuk *cosine similarity* untuk kalimat klaim, vektor kalimat klaim dihitung *cosine similarity*-nya dengan dirinya sendiri. Sedangkan untuk berita terkait, *cosine similarity* dihitung antara vektor dirinya sendiri dengan vektor kalimat klaim yang ditanggapinya.

Angka-angka representasi fitur untuk tiap kalimat tersebut digabung menjadi satu vektor. Vektor tersebut kemudian digabungkan dengan masing-masing vektor kalimat. Semua fitur tersebut dihasilkan pada praproses, kecuali fitur kemiripan kalimat. Karena vektor kalimat dihasilkan oleh pemodelan kalimat yang dilakukan bersamaan dengan proses pelatihan, maka fitur kemiripan kalimat juga selalu dihitung ulang dalam tiap iterasi pelatihan.

E. Klasifikasi dengan LSTM

RNN merupakan sebuah jenis JST dengan *output* yang dihasilkan dapat dijadikan *input* untuk perhitungan selanjutnya. Hal ini membuat RNN memperhatikan urutan sebagai salah satu fitur pembelajaran.

RNN memiliki sebuah *hidden state*, yang merupakan memori dari JST, yang dihitung dari *hidden state* sebelumnya. LSTM merupakan sebuah jenis RNN yang memiliki empat lapis JST yang saling berinteraksi pada masing-masing *hidden state*-nya. LSTM memiliki empat macam *gate* yang saling berinteraksi di dalamnya dan sebuah *cell state* yang berfungsi mengalirkan informasi dari satu *hidden state* ke selanjutnya tanpa melakukan perubahan yang banyak atau bahkan tanpa melakukan perubahan apa pun [6].

Di dalam arsitektur RNN, terdapat *input x*, *hidden state A*, dan *output t*, masing-masing berada pada waktu *t*. *Hidden state* A_t pada (2) merupakan memori dari JST, yang dihitung

dari *hidden state* sebelumnya. *U, V,* dan *W* merupakan bobot dari beberapa lapis RNN. Berbeda dari JST biasa yang memiliki bobot yang berbeda-beda untuk tiap lapis, RNN memiliki bobot yang sama untuk semua lapis.

$$A_t = f(x_t.U + A_{t-1}.W). \quad (2)$$

Untuk fungsi f, dapat digunakan fungsi tanh atau ReLu. Hipotesis *h* dituliskan pada (3).

$$h_t = softmax(A_t.V). \quad (3)$$

Langkah pertama dalam langkah kerja jaringan LSTM adalah membuang informasi dari *cell state* menggunakan *forget gate f* yang dituliskan pada (4). Vektor h_{t-1} pertama disalin dilewatkan terlebih dahulu ke *forget gate* dengan *b* adalah nilai *bias.*

$$f_t = \sigma\big(W_f.[h_{t-1}, x_t] + b_f\big). \quad (4)$$

Selanjutnya, jaingan LSTM memperbarui informasi baru ke *cell state* menggunakan *input gate i* yang tampak pada (5). *Gate* ini bertugas memutuskan informasi yang akan diperbarui pada *cell state*.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i). \quad (5)$$

Pada saat yang bersamaan, vektor yang sama juga dilewatkan ke *lapis* tanh untuk mengubah bentang nilai dalam vektor menjadi nilai real antara -1 hingga 1, menghasilkan *candidate cell state* $\tilde{C}_t$ yang tampak pada (6).

$$\tilde{C}_t = tanh(W_C.[h_{t-1}, x_t] + b_C). \quad (6)$$

Vektor yang melalui *input gate* dan *lapis* tanh selanjutnya melalui *pointwise operation* perkalian, sehingga hanya *output* yang diinginkan yang masuk ke *cell state*. Dengan begitu, operasi vektor-vektor setelah melalui ketiga *gate* LSTM dan lapis tanh adalah menghasilkan nilai *cell state* sesuai (7).

$$C_t = f_t.C_{t-1} + i_t.\tilde{C}_t. \quad (7)$$

Langkah yang terakhir adalah menentukan *output gate* pada (8). Vektor h_{t-1} disalin dan dilewatkan ke *output gate*.

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o). \quad (8)$$

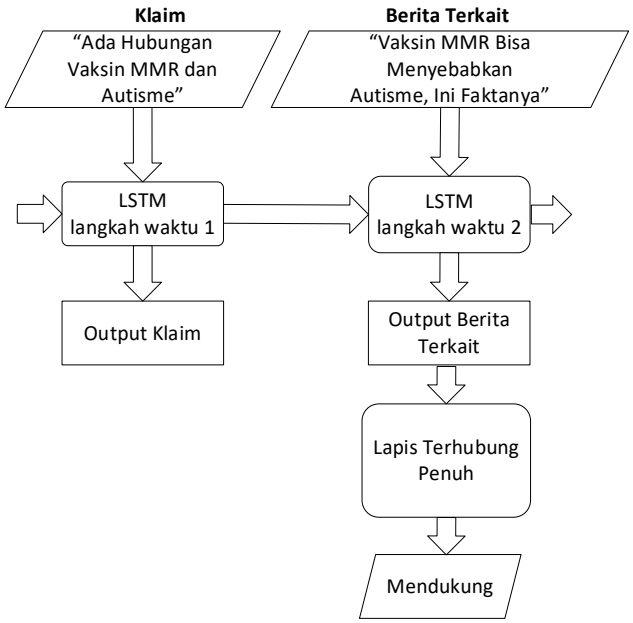
Pada saat yang sama, C_t disalin dan dilewatkan ke *lapis* tanh. Setelah melalui *lapis* tanh, C_t bertemu dengan o_t pada *pointwise operation* perkalian, menghasilkan nilai *output* dari *cell* sesuai (9).

$$h_t = o_t * tan\,h(C_t). \quad (9)$$

LSTM memiliki variasi lain yang menggunakan *forget gate* dan *input gate* yang digandengkan, bernama *coupled-input-forget-gate* LSTM. Variasi ini tidak menentukan besar *forget gate* dan *input gate* secara terpisah, tetapi secara bersamaan. Pada variasi ini, besar *input gate* bergantung pada besar *forget gate* yang tampak pada (10).

$$C_t = f_t * C_{t-1} + (1 - f_t) * \tilde{C}_t. \quad (10)$$

Pada variasi ini, besar *input gate* bergantung pada besar *forget gate*. Rumus-rumus di atas merupakan rumus fase propagasi maju LSTM yang berfungsi untuk menerima *input* dan menghasilkan *output*. Setelah proses fase propagasi maju, ada juga fase propagasi mundur yang berfungsi untuk menghitung besar-besar perubahan yang diperlukan untuk memperbarui bobot-bobot yang ada. Berkebalikan dengan fase propagasi maju, proses fase propagasi mundur dilakukan mulai dari langkah waktu terakhir hingga langkah waktu pertama.



Gbr. 4 Ilustrasi klasifikasi pendapat dengan LSTM.

Analisis pendapat dengan LSTM menggunakan LSTM pada tiap langkah waktu untuk memproses sebuah kalimat [14]. Dalam tugas analisis pendapat suatu judul artikel berita terhadap klaim ini, vektor representasi kalimat klaim diberikan sebagai *input* ke LSTM pada langkah waktu pertama dan vektor representasi kalimat berita terkait diberikan sebagai *input* ke LSTM pada langkah waktu kedua. Ilustrasi klasifikasi pendapat dengan LSTM beserta langkah waktunya ditunjukkan pada Gbr. 4.

Khusus untuk analisis, digunakan variasi LSTM, yakni *coupled-input-forget-gate* LSTM. Hasil *output* dari langkah waktu kedua merupakan representasi pendapat berita klaim terhadap berita terkait. Hasil *output* tersebut kemudian dilewatkan ke sebuah lapis terhubung penuh untuk mendapatkan probabilitas label. Probabilitas label kemudian dilewatkan sebuah lapis *softmax* untuk mendapatkan probabilitas yang lebih baik.

V. HASIL EKSPERIMEN DAN DISKUSI

Untuk uji coba, model JST diuji coba dengan berbagai metode dan parameter untuk menemukan arsitektur yang optimal. Semua uji coba dilakukan dengan data uji coba sebesar 20% dari total berita terkait sebanyak 500 *epoch* dengan *learning rate* 0,01. Akurasi diukur menggunakan *F1-score*. Nilai *F1-score* dihitung untuk masing-masing label dan *macro-average* dari *F1-score* semua label.

TABEL II
HASIL UJI COBA *WORD EMBEDDING PRE-TRAINED* DAN *ON-TRAINED* DENGAN DIMENSI *EMBEDDING* 50 DAN 100

Dimensi	Metode	*F*1 menentang	*F*1 netral	*F*1 mendukung	*Macro-averaged F*1
50	*Pre-trained*	46,983%	71,253%	48,141%	54,459%
50	***On-trained***	**67,105%**	**76%**	**67,227%**	**70,043%**
100	*Pre-trained*	49,785%	71,253%	49,902%	56,98%
100	*On-trained*	63,47%	77,462%	68,743%	63,47%

TABEL III
HASIL UJI COBA DIMENSI *WORD EMBEDING* DENGAN METODE *ON-TRAINED*

Dimensi	*Macro-averaged F*1
50	55,459%
100	56,98%
150	**69,899%**

TABEL IV
HASIL UJI COBA METODE PEMODELAN KALIMAT DENGAN DIMENSI *EMBEDDING* 100

Metode	*F*1 menentang	*F*1 netral	*F1* mendukung	*Macro-averaged F*1
Rata-rata	51,019%	71,253%	59,691%	60,655%
LSTM	67,105%	76%	67,227%	70,111%
***Coupled-input-forget-gate* LSTM**	**70%**	**74.312%**	**69,136%**	**71,149%**

TABEL V
HASIL UJI COBA METODE *CLASSIFIER* DENGAN UKURAN *OUTPUT* LSTM 75 DAN 100

Ukuran	Metode	*F*1 menentang	*F*1 netral	*F*1 mendukung	*Macro-averaged F*1
75	**LSTM**	**70%**	**74,312%**	**69,136%**	**71,149%**
75	*Coupled-input-forget-gate* LSTM	67,917%	77,232%	66,96%	70,703%
100	LSTM	65,517%	75,803%	66,074%	69,132%
100	*Coupled-input-forget-gate* LSTM	65,116%	74,678%	63,205%	67,667%

TABEL VI
HASIL UJI COBA PROBABILITAS *OUTPUT* DISIMPAN (*KEEP PROBABILITY*) UNTUK *DROPOUT*

Keep probability	*F*1 menentang	*F*1 netral	*F*1 mendukung	*Macro-averaged F*1
0,5	67,273%	75,803%	68,211%	70,429%
0,75	68,211%	76,991%	65,495%	70,232%
1	**70%**	**74,312%**	**69,136%**	**71,149%**

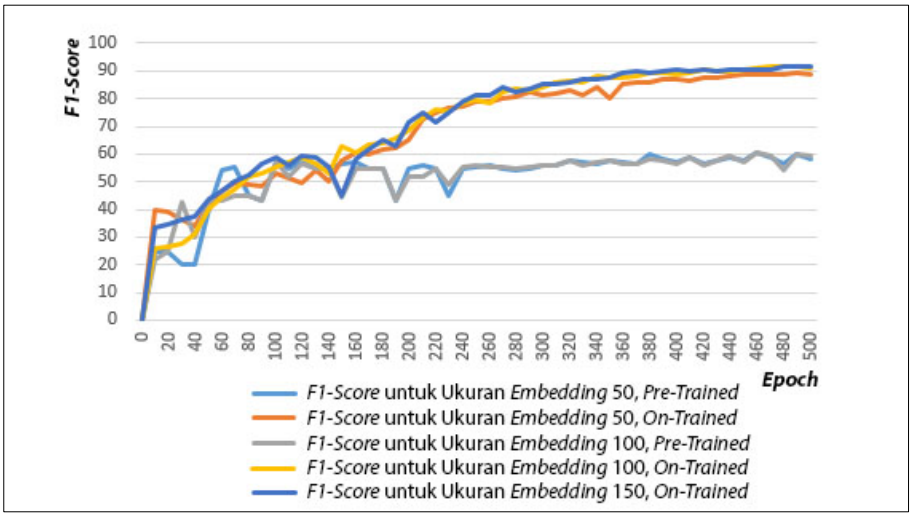
Uji coba telah dilakukan pada *dataset* bahasa Indonesia yang bertopik kesehatan. Uji coba ini meliputi uji coba pengaruh metode *word embedding*, dimensi *embedding* (*embedding size*), metode pemodelan kalimat, metode *classifier*, ukuran *output* LSTM (*RNN size*), *dropout*, fitur-fitur tambahan, dan perbandingan LSTM dengan *Convolutional Neural Network* (CNN).

Uji coba yang pertama melakukan analisis terhadap pengaruh *Word Embedding*, yaitu perbedaan hasil dari penggunaan *pre-trained word embedding* dan *on-trained word embedding*. *Pre-trained word embedding* adalah *word embedding* yang sudah dilatih terlebih dahulu dan siap dipakai untuk berbagai tugas *machine learning* teks secara umum. *Pre-trained word embedding* yang digunakan dibentuk dengan bantuan JST [16]. Teknik ini dinamakan Word2Vec [35]. *On-trained word embedding* adalah *word embedding* yang dilatih bersamaan dengan tugas *machine learning* yang dilakukan, dalam kasus ini adalah analisis *pendapat*. Artinya, nilai *on-trained word embedding* akan berubah tiap *epoch* pelatihan. *On-trained word embedding* dilatih menggunakan JST.

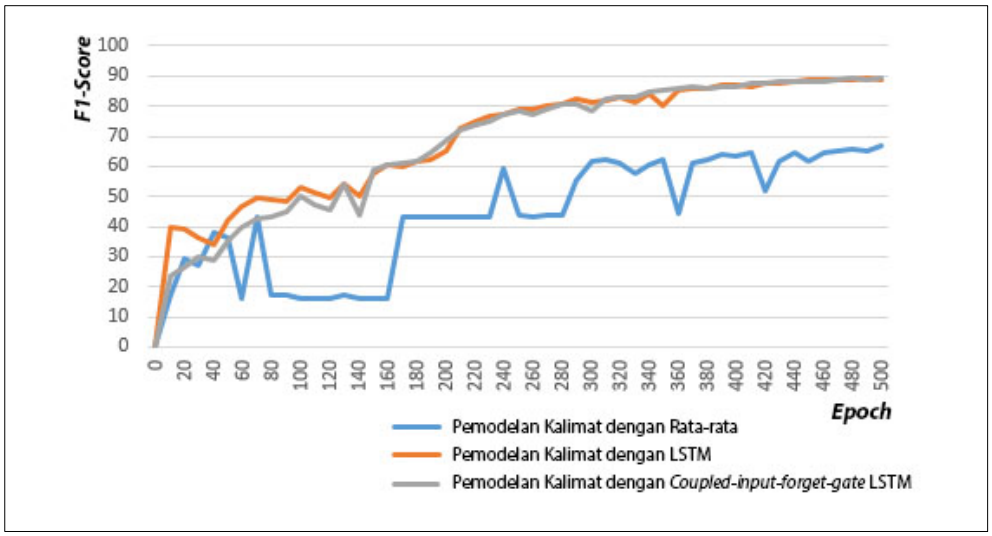
Uji coba terhadap *word embedding* ini dilakukan dengan acuan penelitian sebelumnya, yang mengajukan metode *pre-trained word embedding* dengan Word2Vec untuk analisis pendapat komentar Twitter [14]. Grafik *F1-Score* dari uji coba *word embeding* pada data pelatihan ditampilkan pada Gbr. 5. Hasil dari uji coba metode *word embedding* setelah *training* berakhir dengan *pemodelan kalimat* menggunakan LSTM, *classifier* menggunakan *coupled-input-forget-gate* LSTM, jumlah LSTM unit 75, dan *dropout keep prob* 1,0., disajikan pada Tabel II.

Untuk representasi kalimat, penelitian sebelumnya mengajukan metode menggunakan rata-rata [14]. Semua nilai *word embedding* untuk tiap dimensi dalam suatu kalimat diambil nilai rata-ratanya. Penerapan metode pemodelan kalimat yang lain yakni *autoencoder* menggunakan LSTM dapat meningkatkan akurasi. Variasi LSTM yang diuji coba untuk tahap ini ada dua, yakni LSTM biasa dan *coupled-input-forget-gate* LSTM. Sedangkan hasil dari uji coba metode *word embedding*, dengan *pemodelan kalimat* menggunakan LSTM, klasifikasi menggunakan *coupled-input-forget-gate* LSTM, jumlah LSTM unit 75, dan *dropout keep prob* 1,0, ditunjukkan pada Tabel III.

Kriteria uji coba yang lain adalah metode *classifier*. Penelitian sebelumnya mengajukan metode LSTM untuk



Gbr. 5 Grafik nilai *F1-Score* pelatihan uji coba *word embedding* bahasa Indonesia.



Gbr. 6 Hasil uji coba perbandingan pemodelan kalimat.

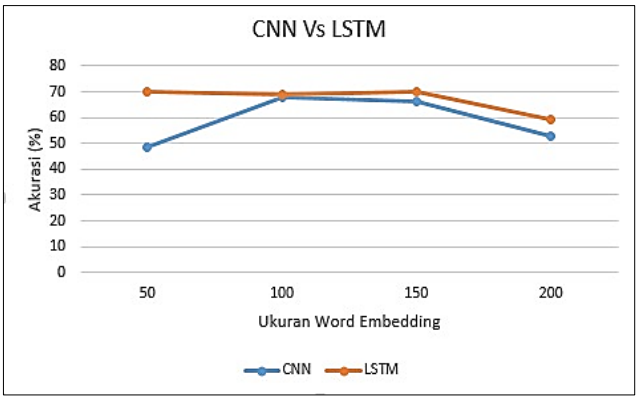
menganalisis pendapat suatu kalimat [14]. LSTM memiliki variasi lain, yakni *coupled-input-forget-gate* LSTM. Metode analisis diuji coba dengan LSTM biasa dan *coupled-input-forget-gate* LSTM. Uji coba untuk *classifier* ini juga diuji coba menggunakan jumlah unit LSTM yang berbeda-beda. Jumlah unit LSTM menentukan jumlah *output* yang dikeluarkan oleh *cell* LSTM untuk tiap langkah waktu. Jumlah unit LSTM yang digunakan untuk uji coba *classifier* ini adalah sebesar 50, 75, dan 100. Grafik *F1-Score* dari uji coba pemodelan kalimat pada data pelatihan ditampilkan pada Gbr. 6. Hasil terakhir pelatihan uji coba pemodelan kalimat, dengan ukuran *embedding* 100, *word embedding* menggunakan metode *on-trained*, klasifikasi menggunakan *coupled-input-forget-gate* LSTM, jumlah LSTM unit 75, dan *dropout keep prob* 1,0, diperlihatkan pada Tabel IV.

Sedangkan Tabel V menggambarkan hasil uji coba klasifikasi dengan ukuran *word embedding* 100, *word embedding* menggunakan metode *on-trained*, pemodelan kalimat menggunakan LSTM, klasifikasi menggunakan *coupled-input-forget-gate* LSTM, dan *dropout keep prob* 1.0.

Penelitian acuan juga mengajukan penggunaan *dropout* pada klasifikasi [14]. Untuk menguji kegunaan *dropout*, telah diuji coba model menggunakan *dropout*, dengan probabilitas menyimpan *input* dan *output (dropout keep prob)* sebesar 0.5, 0.75, dan 1.0. Tabel VI berisi hasil uji coba klasifikasi, dengan ukuran *word embedding* 100, *word embedding* menggunakan metode *on-trained*, pemodelan kalimat menggunakan LSTM, *classifier* menggunakan *coupled-input-forget-gate* LSTM, dan jumlah LSTM unit 75.

Uji coba telah dilakukan untuk melakukan perbandingan dengan penelitian sebelumnya, yaitu arsitektur CNN [36] dengan LSTM dalam melakukan analisis pendapat terhadap judul-judul berita. Terlihat pada grafik pada Gbr. 7, LSTM lebih unggul dibandingkan CNN. Hal ini dikarenakan LSTM dapat melihat urutan kata dalam merepresentasikan tiap kalimat dan menganalisis pendapat berita terkait terhadap klaimnya.

Arsitektur CNN memiliki hasil yang optimal dengan ukuran vektor kata sebesar 100, sedangkan LSTM memiliki hasil yang optimal dengan ukuran vektor kata sebesar 50.



Gbr. 7 Hasil uji coba perbandingan dengan penelitian sebelumnya.

TABEL VII
PERBANDINGAN *F1-SCORE* PADA *DATASET* BAHASA INGGRIS DAN INDONESIA

Percobaan	***F1-Score* Bahasa Indonesia**	***F1-Score* Bahasa Inggris**
Pre-trained word embedding ukuran 50	55,459	54,007
On-trained word embedding ukuran 50	70,043	68,133
Pre-trained word embedding ukuran 100	56,98	17,013
On-trained word embedding ukuran 100	68,741	65,182
On-trained word embedding	69,899	65,024
Representasi dengan rata-rata	60,655	43,726
Representasi dengan LSTM	70,111	68,133
Representasi dengan *coupled-input-gate* LSTM	71,149	67,827
Vanilla LSTM 50 unit	69,757	69,704
Coupled-input-gate LSTM 50 unit	69,079	65,657
Vanilla LSTM 75 unit	71,149	68,133
Coupled-input-gate LSTM 75 unit	70,703	66,013
Vanilla LSTM 100 unit	69,132	68,133
Coupled-input-gate LSTM 100 unit	67,667	66,166
Probabilitas *dropout* 0,5	70,429	71,251
Probabilitas *dropout* 0,75	70,232	67,511
Probabilitas *dropout* 1,0	71,149	68,133

Meskipun CNN dapat mengungguli LSTM pada dimensi tinggi, baik CNN maupun LSTM tetap mengalami penurunan akurasi dengan semakin tingginya dimensi vektor kata, yang disebabkan oleh persebaran data.

Percobaan juga dilakukan terhadap *dataset* bahasa Inggris Emergent [22] sebagai perbandingan dan hasilnya disajikan pada Tabel VII. *Dataset* ini digunakan untuk kompetisi penanganan berita palsu. Terdapat 300 buah klaim dan 2.595 berita terkait.

VI. KESIMPULAN

Pemodelan kalimat menggunakan LSTM pada berita kesehatan bahasa Indonesia memberikan nilai rata-rata *F1-Score* terbaik sebesar 71% dengan jumlah unit 75 dan dimensi *embedding* 50. Pemodelan kalimat dengan LSTM selalu memiliki hasil yang lebih baik dibanding metode rata-rata. Hal ini disebabkan tidak mampunya metode rata-rata mengenali urutan kata, sehingga gagal mengenali negasi.

Pada *dataset* kesehatan bahasa Indonesia, *word embedding* yang dilatih secara *on-train* memiliki hasil yang lebih baik dibanding *word embedding* yang dilatih secara terpisah (*pre-trained*), karena *on-train word embedding* memiliki representasi-representasi vektor yang lebih disesuaikan terhadap arsitektur tugas analisis *pendapat* yang digunakan. Perolehan *F1-Score on-train* sebesar 70% untuk dimensi *embedding* 50.

Berdasarkan hasil uji coba terhadap *dataset* bahasa Indonesia, LSTM biasa dan LSTM *coupled* tidak benar-benar memiliki perbedaan yang cukup signifikan. Selisih rata-rata *F1-Score* yang diperoleh sebesar 1%. Namun, LSTM biasa menghasilkan nilai yang lebih baik.

Untuk pengolahan analisis *pendapat* ini, *dropout* tidak menghasilkan perbedaan yang cukup signifikan dalam akurasi dan justru mengurangi akurasi. Model memiliki akurasi 71,149% tanpa *dropout*.

Masing-masing dari ketiga fitur tambahan yang digunakan turut menambah akurasi (sebesar 2% hingga 4%). Di antara ketiga fitur yang digunakan, fitur tanda tanya yang paling berpengaruh terhadap akurasi dan memberikan *F1-Score* sebesar 69,806%.

Penelitian mengenai analisis *pendapat* ini memiliki potensi yang besar dalam pengembangannya di Indonesia. Bagian-bagian dari arsitektur LSTM ini masih dapat dimodifikasi dengan berbagai metode pembelajaran mendalam lainnya. Upaya lainnya adalah menggunakan model representasi kata lainnya seperti *FastText* untuk analisis.

UCAPAN TERIMA KASIH

Terima kasih disampaikan kepada Kementerian Riset, Teknologi, dan Pendidikan Tinggi Republik Indonesia dan Kementerian Keuangan Republik Indonesia, serta Lembaga Pengelola Dana Pendidikan (LPDP) yang telah mendukung pendanaan penelitian ini melalui skema beasiswa BUDI DN.

REFERENSI

[1] (2017 "Hasil Survey Wabah Hoax Nasional 2017," [Online], https://mastel.id/hasil-survey-wabah-hoax-nasional-2017/, tanggal akses: 1-Nov-2019.

[2] W. Ferreira dan S.A. Vlachos, "For or Against? Assessing the Evidence for News Headline Claims," University College London, London, UK, M.Sc. Project, 2015.

[3] S.M. Mohammad, P. Sobhani, dan S. Kiritchenko, "Stance and Sentiment in Tweets," *ACM Trans. Internet Technol.*, Vol. 17, No. 3, hal 1–23, 2017.

[4] J. Ebrahimi, D. Dou, dan D. Lowd, "A Joint Sentiment-Target-Stance Model for Stance Classification in Tweets," *Proc. of COLING 2016, the 6th International Conference on Computational Lingustics*, 2016, hal 2656–2665.

[5] P. Sobhani, S.M. Mohammad, dan S. Kiritchenko, "Detecting Stance in Tweets and Analyzing Its Interaction with Sentiment," *Proc. Fifth Jt. Conf. Lex. Comput. Semant.*, 2016, hal 159–169.

[6] P. Krejzl, B. Hourová, dan J. Steinberger, "Stance Detection in Online Discussions," *arXiv Prepr.*, arXiv1701.00504, 2017.

[7] A. Hanselowski, Avinesh PVS, B. Schiller, F. Caspelherr, D. Chaudhuri, C.M. Meyer, dan I. Gurevych, "A Retrospective Analysis of the Fake

News Challenge Stance Detection Task," *Proc. of the 27th International Conference on Computational Linguistics*, 2018, hal. 1859–1874.

[8] M.H. Purnomo, S. Sumpeno, E.I. Setiawan, dan D. Purwitasari, "Keynote Speaker II: Biomedical Engineering Research in the Social Network Analysis Era: Stance Classification for Analysis of Hoax Medical News in Social Media," *Procedia Computer Science*, 2017, Vol. 116, hal. 3-9.

[9] A. Aker, L. Derczynski, dan K. Bontcheva, "Simple Open Stance Classification for Rumour Analysis," *Proc. of Recent Advances in Natural Language Processing*, 2017, hal 31–39.

[10] G. Rajendran, P. Poornachandran, dan B. Chitturi, "Deep Learning Model on Stance Classification," *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI 2017)*, 2017, hal 2407–2409.

[11] W.-F. Chen dan L.-W. Ku, "UTCNN: a Deep Learning Model of Stance Classification on Social Media Text," *arXiv Prepr.*, arXiv1611.03599, hal 1635–1645, 2016.

[12] G. Zarrella dan A. Marsh, "Mitre at Semeval-2016 task 6: Transfer Learning for Stance Detection," *Proc. of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, 2016, hal. 458–463.

[13] A.M. Hasan (2018) "Info Hoax Soal Kesehatan Paling Banyak Beredar di Masyarakat," [Online]. https://tirto.id/info-hoax-soal-kesehatan-paling-banyak-beredar-di-masyarakat-cnQZ, tanggal akses: 5-Nov-2019.

[14] E. Kochkina, M. Liakata, dan I. Augenstein, "Turing at SemEval-2017 Task 8: Sequential Approach to Rumour Stance Classification with Branch-LSTM," *Proc. of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, hal. 475–480.

[15] A. Zaini, M.A. Muslim, dan W. Wijono, "Pengelompokan Artikel Berbahasa Indonesia Berdasarkan Struktur Laten Menggunakan Pendekatan Self Organizing Map," *J. Nas. Tek. Elektro dan Teknol. Inf.*, Vol. 6, No. 3, hal 259–267, 2017.

[16] J. Santoso, A.D.B. Soetiono, E. Setyati, dan E.M. Yuniarno, "Self-Training Naive Bayes Berbasis Word2Vec untuk Kategorisasi Berita Bahasa Indonesia," *J. Nas. Tek. Elektro dan Teknol. Inf.*, Vol. 7, No. 2, hal 158–166, 2018.

[17] S. Somasundaran dan J. Wiebe, "Recognizing Stances in Ideological On-Line Debates," *Proc. of the NAACL HLT 2010 Workshop on Computational Approches to Analysis and Generation of Emotion in Text*, 2010, hal 116–124.

[18] S. Dungs, A. Aker, N. Fuhr, dan K. Bontcheva, "Can Rumour Stance Alone Predict Veracity?," *Proc. of the 27th International Conference on Computational Linguistics*, 2018, hal 3360–3370.

[19] D. Mrowca, E. Wang, dan A. Kosson, "Stance Detection for Fake News Identification," Standford University, Standford, CA, Project Report, 2017.

[20] K. Dey, R. Shrivastava, dan S. Kaushik, "Twitter Stance Detection - A Subjectivity and Sentiment Polarity Inspired Two-Phase Approach," *2017 IEEE International Conference on Data Mining Workshops*, 2017, hal 365–372.

[21] P. Sobhani, D. Inkpen, dan S. Matwin, "From Argumentation Mining to Stance Classification," *2nd Workshop on Argumentation Mining*, 2015, hal 67–77.

[22] W. Ferreira dan A. Vlachos, "Emergent: A Novel Data-set for Stance Classification," *Proc. of NAACL-HLT 2016*, 2016, hal 1163–1168.

[23] B. Riedel, I. Augenstein, G.P. Spithourakis, dan S. Riedel, "A Simple but Tough-to-beat Baseline for the Fake News Challenge Stance Detection Task," *arXiv Prepr.*, arXiv1707.03264, hal 1–6, 2017.

[24] I. Augenstein, T. Rocktäschel, A. Vlachos, dan K. Bontcheva, "Stance Detection with Bidirectional Conditional Encoding," *Proc. of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP-16)*, 2016, hal 876–885.

[25] A. Zubiaga, E. Kochkina, M. Liakata, R. Procter, dan M. Lukasik, "Stance Classification in Rumours as a Sequential Task Exploiting the Tree Structure of Social Media Conversations," *Proc. of COLING 2016, the 26th International Conference on Computational Linguistics*, 2016, hal 2438–2448.

[26] O. Somantri dan M. Khambali, "Feature Selection Klasifikasi Kategori Cerita Pendek Menggunakan Naïve Bayes dan Algoritme Genetika," *J. Nas. Tek. Elektro dan Teknol. Inf.*, Vol. 6, No. 3, hal 301–306, 2017.

[27] M. Novita (2019) "Studi: Berita Kesehatan di Media Sosial Sebagian Besar Hoax," [Online], https://gaya.tempo.co/read/1172696/studi-berita-kesehatan-di-media-sosial-sebagian-besar-hoax/full&view=ok, tanggal akses: 27-Nov-2019.

[28] C. Juditha, "Interaksi Komunikasi Hoax di Media Sosial serta Antisipasinya," *J. Pekommas*, Vol. 3, No. 1, hal. 31-44, 2018.

[29] C. Guggilla, T. Miller, dan I. Gurevych, "CNN- and LSTM-based Claim Classification in Online User Comments," *Proc. of COLING 2016, the 26th Int. Conf. Comput. Linguist*, hal 2740–2751, 2016.

[30] A.D. Tahitoe dan D. Purwitasari, "Implementasi Modifikasi Enhanced Confix Stripping Stemmer untuk Bahasa Indonesia dengan Metode Corpus Based Stemming," Skripsi, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, 2010.

[31] D. Wahyudi, T. Susyanto, dan D. Nugroho, "Implementasi dan Analisis Algoritma Stemming Nazief & Adriani dan Porter pada Dokumen Berbahasa Indonesia," *J. Ilm. SINUS*, Vol. 15, No. 2, hal 49–56, 2017.

[32] R. Johnson dan T. Zhang, "Semi-supervised Convolutional Neural Networks for Text Categorization via Region Embedding," *Advances in Neural Information Processing Systems (NIPS 2015)*, 2015, hal 1–9.

[33] H. Palangi, L. Deng, Y. Shen, J. Gao, X. He, J. Chen, X. Song, dan R. Ward, "Deep Sentence Embedding Using Long Short-Term Memory Networks: Analysis and Application to Information Retrieval," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, Vol. 24, No. 4, hal 694–707, 2016.

[34] N. Srivastava, E. Mansimov, dan R. Salakhutdinov, "Unsupervised Learning of Video Representations using LSTMs," *Proc. of the 32nd International Conference on International Conference on Machine Learning*, 2015, hal 843–852.

[35] T. Mikolov, K. Chen, G. Corrado, dan J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," *Advances in Neural Information Processing Systems (NIPS 2013)*, 2013, hal 1–9.

[36] Y.-C. Chen, Z.-Y. Liu, dan H.-Y. Kao, "IKM at SemEval-2017 Task 8: Convolutional Neural Networks for stance detection and rumor verification," *11th International Workshop on Semantic Evaluations*, 2017, hal 465–469.