SENTIMEN ANALISIS MULTI-LABEL PADA UJARAN KEBENCIAN DAN UMPATAN DI TWITTER INDONESIA MENGGUNAKAN PENDEKATAN *DEEP LEARNING*

Skripsi

diajukan untuk memenuhi salah satu syarat untuk memperoleh gelar
Sarjana Komputer

Oleh:

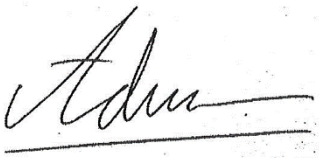
Nala Adina
4611416039

JURUSAN ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS NEGERI SEMARANG
2020

PERNYATAAN

Saya menyatakan bahwa skripsi saya yang berjudul "Sentimen Analisis Multi-Label pada Ujaran Kebencian dan Umpatan Di Twitter Indonesia Menggunakan Pendekatan *Deep Learning*" disusun atas dasar penelitian saya dengan arahan dosen pembimbing. Sumber informasi atau kutipan yang berasal dari karya yang diterbitkan telah disebutkan dalam teks dan dicantumkan dalam daftar pustaka di bagian akhir skripsi ini. Dan saya menyatakan bahwa skripsi ini bebas plagiat dan apabila di kemudian hari terbukti terdapat plagiat dalam skripsi ini, maka saya bersedia menerima sanksi sesuai ketentuan perundang – undangan.

Semarang, 24 September 2020



Nala Adina
NIM 46114160396

PERSETUJUAN PEMBIMBING

Nama : Nala Adina

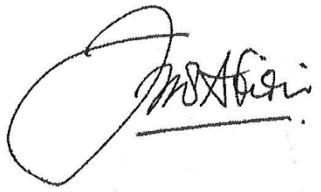
NIM : 4611416039

Program Studi : Teknik Informatika S1

Judul Skripsi : Sentimen Analisis Multi-Label pada Ujaran Kebencian dan Umpatan di Twitter Indonesia Menggunakan Pendekatan *Deep Learning*

Skripsi ini telah disetujui oleh pembimbing untuk diajukan ke sidang ujian skripsi Program Studi Teknik Informatika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Semarang.

Semarang, 24 September 2020



Zaenal Abidin S.Si., M.Cs., Ph.D.
NIP 198205042005011001

PENGESAHAN

Skripsi dengan judul:

Sentimen Analisis Multi-Label pada Ujaran Kebencian dan Umpatan di Twitter Indonesia Menggunakan Pendekatan *Deep Learning*

disusun oleh:

Nala Adina

4611416039

Telah dipertahankan di hadapan sidang Panitia Ujian Skripsi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Semarang pada tanggal 2 Oktober 2020.

Panitia:

Ketua

Dr. Sugianto, M.Si.
NIP 196102191993031001

Sekretaris

Dr. Alamsyah, S.Si., M.Kom.
NIP 197405172006041001

Penguji I

Endang Sugiharti, S.Si., M.Kom.
NIP 197401071999032001

Penguji II

Aji Purwinarko S.Si., M.Cs.
NIP 198509102015041001

Anggota Penguji/Pembimbing

Zaenal Abidin S.Si., M.Cs., Ph.D.
NIP 198205042005011001

MOTTO DAN PERSEMBAHAN

MOTTO

1. Sesungguhnya Allah tidak merubah keadaan sesuatu kaum sehingga mereka merubah keadaan yang ada pada diri mereka sendiri (QS. Ar Ra'd : 11)
2. It's the job that's never started as taken longest to finish (J.R.R. Tolkien)
3. Life is not daijobu (Anonim)

PERSEMBAHAN

Skripsi ini saya persembahkan kepada:

- Kedua Orang Tua saya Bapak Suhartoyo dan Ibu Eni Purwanti yang saya sayangi, kasihi, cintai dan selalu saya doakan, semoga ini bisa menghadirkan senyum di bibir dan hati kalian.
- Teman-teman saya di jurusan Ilmu Komputer, Fakultas MIPA, serta teman-teman di Universitas Negeri Semarang.
- Semua pihak yang tidak dapat disebutkan satu persatu yang telah membantu hingga terselesaikannya penulisan skripsi ini.
- Almamater, Universitas Negeri Semarang.

PRAKATA

Puji syukur penulis panjatkan kepada *Rabb* seluruh alam semesta Allah *Subhanahu wa ta'ala*, atas berkat rahmat, pertolongan, petunjuk, bimbingan serta kebaikan-Nya, penulis dapat menyelesaikan skripsi yang berjudul "Sentimen Analisis Multi-Label pada Ujaran Kebencian dan Umpatan di Twitter Indonesia Menggunakan Pendekatan *Deep Learning*". Shalawat serta salam tak lupa selalu terpanjatkan kepada junjungan kita makhluk paling mulia di langit dan di bumi, suri tauladan bagi kita semua, manusia yang membawa peradaban kepada zaman yang terang-benderang, *Rasulullah Muhammad Shallallahu alaihi wassalam*, keluarganya, sahabatnya dan para pengikutnya.

Penulis menyadari bahwa penulisan skripsi ini tidak akan selesai tanpa adanya dukungan serta bantuan dari berbagai pihak. Oleh karena itu, penulis ingin menyampaikan ucapan terima kasih kepada:

1. Prof. Dr. Fathur Rokhman, M.Hum., Rektor Universitas Negeri Semarang.
2. Dr. Sugianto, M.Si., Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Semarang.
3. Dr. Alamsyah, S.Si., M.Kom., Ketua Jurusan Ilmu Komputer FMIPA Universitas Negeri Semarang yang telah meluangkan waktu, membantu, membimbing, mengarahkan dan memberikan saran sehingga penulis dapat menyelesaikan skripsi ini.
4. Zaenal Abidin S.Si., M.Cs., Ph.D., Dosen Pembimbing saya, guru saya yang dengan penuh kesabaran telah meluangkan waktu, membantu, membimbing, mengarahkan, memberikan saran, dan membagi ilmunya sehingga penulis dapat menyelesaikan skripsi ini.
5. Bapak dan Ibu Dosen Jurusan Ilmu Komputer yang telah memberikan bekal kepada penulis dalam penyusunan skripsi ini.
6. Kedua Orang Tua saya Bapak Suhartoyo dan Ibu Eni Purwanti yang selalu tanpa lelah mencurahkan kasih sayang, doa, dan dukungannnya.

7. Teman-teman saya di jurusan Ilmu Komputer, terutama teman-teman ilkom angkatan 2016, yang telah memberikan semangat dan dukungannya.
8. Semua pihak yang telah membantu terselesaikannya skripsi ini yang tidak dapat penulis sebutkan satu persatu, terimakasih atas bantuannya.

Semoga skripsi ini dapat memberikan manfaat bagi pembaca di masa yang akan datang.

Semarang, 24 September 2020

Nala Adina
NIM 4611416039

ABSTRAK

Nala Adina. 2020. Sentimen Analisis Multi-Label pada Ujaran Kebencian dan Umpatan di Twitter Indonesia Menggunakan Pendekatan *Deep Learning.* Skripsi, Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Semarang. Pembimbing Zaenal Abidin S.Si., M.Cs., Ph.D.

Kata kunci: Analisi sentimen, *deep learning,* multi-label, ujaran kebencian, umpatan.

Ujaran kebencian dan umpatan sangat banyak ditemui di dalam media sosial dan dapat diakses oleh hampir semua orang di Indonesia. Penggunaan umpatan dengan ujaran kebencian mampu menimbulkan emosi bagi lawan bicaranya dan berdampak mempercepat terjadinya konflik individual maupun sosial. Pada penelitian ini, akan dilakukan analisis sentimen berdasarkan *dataset* multi-label yang memiliki data sejumlah 13.169 data dan 12 kelas tentang ujaran kebencian dan umpatan di Twitter beserta target, kategori, dan level ujaran kebencian. Metode klasfikasi yang digunakan adalah metode *Recurrent Neural Network Long Short Term Memory* (RNN-LSTM) dan metode *Convolutional Neural Network* (CNN) dan penggunaan metode *Word2Vec* sebagai *word embedding* untuk peningkatan akurasi algoritma klasifikasi. Setelah dihitung nilai akurasinya menggunakan *10-cross validation*, akurasi model RNN-LSTM mendapatkan rata – rata akurasi sebesar 67,22 %, sedangkan model CNN mendapatkan rata – rata akurasi sebesar 64,07 %. Akurasi yang didapat pada *baseline* model masih bisa ditingkatkan lagi apabila ditambah metode *Word2Vec.* Hasil akhir akurasi RNN-LSTM dengan *Word2Vec* mendapatkan rata – rata akurasi sebesar 73,94 % dan CNN dengan *Word2Vec* mendapatkan rata – rata akurasi sebesar 71,19 %. Dengan demikian penerapan *Word2Vec* sebagai *word embedding* dapat meningkatkan akurasi model RNN-LSTM sebesar 6,72% dan model CNN sebesar 7,12 %. Berdasarkan hasil akurasi yang didapat, metode RNN-LSTM dengan *Word2Vec* adalah yang terbaik karena mendapatkan hasil akurasi sebesar 73,94% dan penggunaan *Word2Vec* terbukti dapat meningkatkan akurasi pada kedua model.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

1.1. Latar Belakang

Ujaran kebencian adalah segala bentuk ekspresi yang menyebarkan, menghasut, mempromosikan atau membenarkan kebencian dalam hal rasis, agama keyakinan, entitas, kecacatan, jenis kelamin atau bentuk kebencian lainnya berdasarkan intoleransi, termasuk yang diungkapkan melalui nasionalisme dan etnosentrisme yang agresif, diskriminasi dan permusuhan terhadap minoritas, migran dan orang-orang dari luar negeri maupun dalam negeri (Johnson et al., 2019). Ujaran kebencian pula dapat menimbulkan efek yang sangat besar mulai dari tingkat individual sampai dengan skala yang luas, seperti konflik social ataupun sampai dengan genosida manusia (Ibrohim & Budi, 2019).

Ujaran kebencian juga sering ditemui pula bersama umpatan dalam penyampaiannya (Davidson et al., 2017). Umpatan adalah ucapan yang mengandung kata-kata kasar atau frasa yang disampaikan kepada lawan bicaranya (individu atau kelompok), baik secara lisan maupun tulisan, penggunaan umpatan dalam ujaran kebencian dapat menimbulkan emosi bagi lawan bicaranya dan berdampak mempercepat pada konflik individual maupun sosial (Ibrohim & Budi, 2019).

Ujaran kebencian sangat banyak ditemui di dalam media sosial yang dapat diakses oleh hampir semua orang di Indonesia, salah satu sosial media yang banyak menyebarkan ujaran kebencian adalah Twitter (Hakiem et al., 2019), karena penggunanya yang masif dan kemudahan pesan singkat (*tweet*) pada Twitter yang hanya maksimal 140 karakter, memungkinkan pengguna mampu menyampaikan opini ataupun maksud pesan dengan singkat, padat dan jelas.

Pada laporan finansial Twitter kuartal ke-3 tahun 2019, pengguna aktif harian di platform Twitter dicatat 145 juta pengguna (Twitter, 2020), dan Indonesia menjadi salah satu negara dengan penyumbang terbesar dalam pertumbuhan pengguna aktif Twitter (Clinten, 2019)

Seiring bertambahnya pengguna, metode analisis sentimen diperlukan untuk mengurangi tersebarnya *tweet* tentang ujaran kebencian. Analisis sentimen merupakan bidang studi yang menganalisis pendapat, sentimen, penilaian, evaluasi, sikap, dan emosi seseorang terkait suatu topik, layanan, produk, individu, organisasi, atau kegiatan tertentu (Liu, 2012). Opini maupun komentar dapat dinyatakan memiliki kecenderungan positif maupun negatif dengan analisis sentimen, dengan demikian analisis sentiment dapat dijadikan acuan dalam pengambilan keputusan suatu organisasi, meningkatkan suatu pelayanan, maupun sebagai *review* suatu produk.

Penentuan sentimen suatu opini dapat dilakukan secara manual, namun semakian banyak opini maka membutuhkan waktu dan usaha yang diperlukan untuk pengklasifikasian polaritas opini. Oleh karena itu, diajukan penerapan metode pembelajaran mesin untuk mengklasifikasi polaritas opini dari sumber data yang sangat banyak tersebut.

Analisis sentimen tentang ujaran kebencian telah banyak diangkat menjadi topik oleh beberapa peneliti sebelumnya, salah satunya pada permasalahan sentimen telah dilakukan penelitian untuk mengumpulkan data dalam Bahasa Indonesia (Alfina et al., 2017). Pada penelitian Alfina et al. (2017) yang mengumpulkan data dari Twitter dengan *tweet* yang berhubungan dengan Pemilihan Kepala Daerah (Pilkada) DKI Jakarta 2017 tetapi hanya diberikan 2 kelas (*binary* klasifikasi) saja. Alfina et al. menggunakan beberapa metode antara lain: *Naïve Bayes*, *Bayesian Logistic Regression*, *Random Forest Decision Tree*, dan *Support Vector Machine* dengan fitur *n-gram* dan *negative sentiment*. Hasil dari penelitian tersebut menunjukkan bahwa metode *Random Forest Decision Tree* menggunakan fitur *n-gram* mendapatkan *f1 score* tertinggi dibanding metode lainnya dengan 93.5%.

Penelitian tentang lain yang menggunakan metode *machine learning* juga dilakukan oleh Ibrohim dan Budi (2019), mereka melakukan klasifikasi multi-label dengan berbagai algoritma klasifikasi, yang ditambahkan dengan metode transformasi data dan juga berbagai metode ekstraksi fitur. Metode terbaik yang didapat adalah menggunakan metode *Random Forest Decision Tree* yang

menggunakan data transformasi *Label Powerset* dengan ekstraksi fitur *unigram feature*, yang menghasilkan akurasi terbaik sebesar 66.12% untuk *dataset* Twitter ujaran kebencian dan kata – kata kasar berbahasa Indonesia.

Tidak hanya metode *binary* klasifikasi untuk mengidentifikasi sentimen, ada beberapa permasalahan klasifikasi yang memiliki sifat multi-kelas multi-label yang sering ditemui. Salah satu contoh menggunakan metode *Logistic Regression with L2 Regularization* dalam mendeteksi sentimen dan masalahnya terhadap *offensive language* yang berbahasa inggris, penelitian tersebut menghasilkan akurasi *precision* 91%, *recall* 90%, sedangkan 40% dalam hasil klasifikasi (Davidson et al., 2017). Metode lainnya menggunakan *Convolutional Neural Network* (CNN) dalam mengklasifikasikan sentimen multi-kelas yang memiliki banyak model fitur tetapi *f1 score*-nya hanya 78,3% (Gambäck & Sikdar, 2017).

Kemudian pada penelitian van Aken et al. (2018) yang melakukan analisis sentimen tentang "Toxic Comment" dengan permasalahan multi-kelas multi-label berbahasa Inggris pada Twitter yang memiliki kelas yang tidak seimbang menggunakan metode *deep learning*. Penelitian tersebut mendapatkan hasil *f1 score* untuk metode CNN + Glove 76.9 % , dan hasil *f1 score* metode *Long Short Term Memory* (LSTM) + Glove adalah 78.1 % untuk hasil terbaik yaitu metode Essemble yang diajukan oleh peneliti dengan *f1 score* 79.3 % (van Aken et al., 2018).

Berdasarkan beberapa permasalahan di atas pada penelitian ini diberi judul "Sentimen Analisis Multi-Label Pada Ujaran Kebencian dan Umpatan di Twitter Indonesia Menggunakan Pendekatan *Deep Learning*". Diharapkan penelitian ini mampu mengklasifikasikan ujaran kebencian dan umpatan menggunakan metode *deep learning*. Penelitian ini dilaksanakan karena langkanya penelitian tentang klasifikasi multi-label pada dataset bahasa Indonesia yang menggunakan metode *deep learning,* algoritma *deep learning* yang dipilih oleh peneliti adalah algoritma CNN dan RNN-LSTM bersamaan dengan penerapan *Word2Vec* yang sering digunakan pada penelitian tentang sentimen analisis dan kemudian membandingkan hasil dari algoritma tersebut.

1.2. Rumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana cara kerja RNN-LSTM dan CNN dalam proses penilaian suatu sentimen?
2. Bagaimana penerapan *Word2Vec* untuk proses *word embedding* pada *data training*?
3. Bagaimana tingkat akurasi hasil yang diperoleh terhadap prediksi sentimen berdasarkan RNN-LSTM dan CNN dengan penerapan *Word2Vec* sebagai *word embedding* pada *data training*?

1.3. Batasan Masalah

Beberapa batasan dalam penelitian ini antara lain:

1. Data *training* untuk penelitian ini diambil dari penelitian Ibrohim dan Budi, (2019) tentang *Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter* yang tersedia secara publik.
2. Penelitian ini akan menghasilkan *model pre-trained* menggunakan RNN-LSTM dan CNN dengan *Word2Vec* yang berguna sebagai *word embedding*.

1.4. Tujuan Penelitian

Penelitian ini memiliki beberapa tujuan, antara lain:

1. Mengetahui cara kerja RNN-LSTM dan CNN dalam proses prediksi suatu sentimen.
2. Mengetahui penerapan *Word2Vec* untuk proses *word embedding* pada *data training*.
3. Mengetahui tingkat akurasi dari hasil yang diperoleh terhadap prediksi sentimen berdasarkan RNN-LSTM dan CNN dengan penerapan *Word2Vec* pada *dataset* berbahasa Indonesia.

1.5. Manfaat Penelitian

Manfaat penelitian ini adalah sebagai berikut:

1. Memahami penerapan algoritma RNN-LSTM dan CNN untuk klasifikasi multi-label.
2. Memahami penerapan *Word2Vec* pada algoritma RNN-LSTM dan algoritma CNN untuk klasifikasi multi-label.
3. Mengetahui hasil peningkatan akurasi antara algoritma RNN-LSTM dan algoritma CNN serta penggunaan *Word2Vec* pada kedua algoritma tersebut dalam klasifikasi multi-label pada *dataset* berbahasa Indonesia.
4. Dengan penerapan algoritma RNN-LSTM dan algoritma CNN yang menggunakan *Word2Vec* dapat menambah wawasan penulis dalam membantu Direktorat Tindak Pidana Siber Bareskrim dalam mencegah ujaran kebencian yang beredar di media sosial.

1.6. Sistematika Penulisan

Sistematika penulisan berguna untuk memudahkan dalam memahami jalan pemikiran secara keseluruhan skripsi. Penulisan skripsi ini secara garis besar dibagi menjadi tiga bagian, yaitu sebagai berikut.

1.6.1. Bagian Awal Skripsi

Bagian awal skripsi terdiri dari halaman judul, halaman pengesahan, halaman pernyataan, halaman motto dan persembahan, abstrak, kata pengantar, daftar isi, daftar gambar, daftar tabel dan daftar lampiran.

1.6.2. Bagian Isi Skripsi

Bagian isi skripsi terdiri dari lima bab, yaitu sebagai berikut.

1. BAB 1: PENDAHULUAN

   Bab ini berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat penelitian serta sistematika penulisan skripsi.
2. BAB 2: LANDASAN TEORI

Bab ini berisi penjelasan mengenai definisi maupun pemikiran-pemikiran yang dijadikan kerangka teoritis yang menyangkut dan mendasari pemecahan masalah dalam skripsi ini.

3. BAB 3: METODE PENELITIAN

   Bab ini berisi penjelasan mengenai studi pendahuluan, tahap pengumpulan data, tahap analisis data, model yang digunakan dalam penelitian dan analisis kebutuhan serta perancangan sistem.

4. BAB 4: HASIL DAN PEMBAHASAN

   Bab ini berisi hasil penelitian berserta pembahasannya.

5. BAB 5: PENUTUP

   Bab ini berisi simpulan dari penulisan skripsi dan saran yang diberikan penulis untuk mengembangkan skripsi ini.

1.6.3. Bagian Akhir Skripsi

Bagian akhir skripsi ini berisi daftar pustaka yang merupakan informasi mengenai buku-buku, sumber-sumber dan referensi yang digunakan penulis serta lampiran-lampiran yang mendukung dalam penulisan skripsi ini.

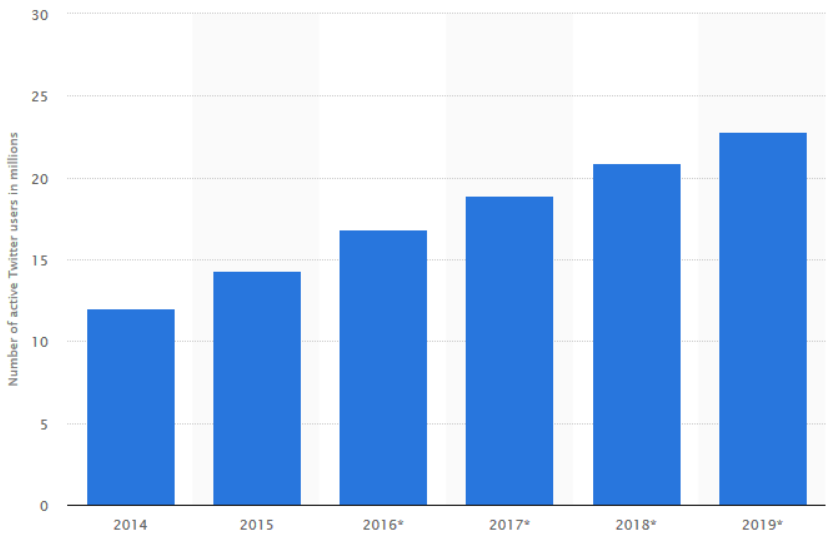
BAB 2

LANDASAN TEORI

2.1. Twitter

Twitter adalah sebuah situs jejaring sosial yang sedang berkembang pesat saat ini karena pengguna dapat berinteraksi dengan pengguna lainnya dari komputer ataupun perangkat *mobile* mereka dari manapun dan kapanpun. Setelah diluncurkan pada Juli 2006, jumlah pengguna harian Twitter pula meningkat untuk tahun 2019, rata-rata pengguna aktif harian skala internasional adalah 121 juta untuk Kuartal 4, dibandingkan dengan 99 juta pengguna pada periode yang sama tahun sebelumnya dan dibandingkan dengan 115 juta pengguna pada kuartal sebelumnya (Twitter, 2020).

Dapat dilihat pada Gambar 2.1, tahun 2014 sampai dengan 2019 akun pengguna Twitter di Indonesia mengalami kenaikan yang signifikan, yaitu diperkirakan sekitar 12 juta pengguna di Indonesia (Statista, 2019).



Gambar 2.1 Jumlah Pengguna Twitter di Indonesia dari 2014 – 2019 (Statista, 2019)

Pengguna Twitter sendiri bisa terdiri dari berbagai macam kalangan yang para penggunanya ini dapat berinteraksi dengan teman, keluarga hingga rekan

kerja. Twitter sebagai sebuah situs jejaring sosial memberikan akses kepada penggunanya untuk mengirimkan sebuah pesan singkat yang terdiri dari maksimal 140 karakter (disebut *tweet*). *Tweet* sendiri bisa terdiri dari pesan teks, foto maupun video.

Melalui *tweet* inilah pengguna Twitter dapat berinteraksi lebih dekat dengan pengguna Twitter lainnya dengan mengirimkan tentang apa yang sedang mereka pikirkan, apa yang sedang dilakukan, atau tentang berita terkini serta hal lainnya (Pournaki et al., 2020).

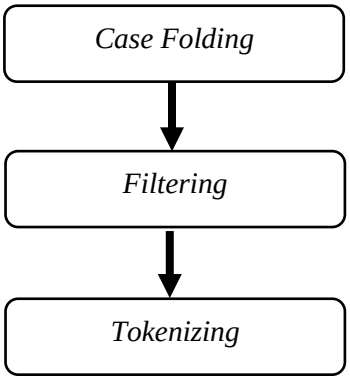
2.2. *Text Mining*

Text mining juga dikenal sebagai *data mining text* atau penemuan pengetahuan dari database tekstual. Sesuai dengan buku *Data Mining* (Aggarwal, 2015), *text mining* dapat didefinisikan sebagai suatu proses menggali informasi dimana seorang *user* berinteraksi dengan sekumpulan dokumen menggunakan *tools* analisis yang merupakan komponen-komponen dalam *data mining*. Tujuan dari *text mining* adalah untuk mendapatkan informasi yang berguna dari sekumpulan dokumen. Jadi, sumber data yang digunakan dalam *text mining* adalah sekumpulan teks yang memiliki format yang tidak terstruktur atau minimal semi terstruktur. Adapun tugas khusus dari *text mining* antara lain yaitu pengkategorisasian teks dan pengelompokkan teks (Allahyari et al., 2017).

2.3. *Word Embedding*

Word embedding adalah jenis representasi kata yang memungkinkan kata-kata dengan makna yang sama memiliki dan representasi yang serupa (Brownlee, 2017). Teks yang akan dilakukan proses *word embedding* yang telah dilakukan proses *text mining*, pada umumnya memiliki karakteristik diantaranya memiliki dimensi yang tinggi, terdapat *noise* pada data, dan terdapat struktur teks yang tidak baik. Dalam penelitian ini menggunakan data yang berasal dari Twitter. Data yang bersasal dari Twitter mempunyai kerumitan yang cukup tinggi. Hal ini di karenakan karakteristik dari Twitter adalah penggunaan bahasa yang tidak sesuai bahasa baku dan banyaknya kesalahan ejaan pada penulisan *tweet* (Zhou et al., 2014). Cara yang digunakan dalam mempelajari suatu data teks, adalah dengan terlebih dahulu menentukan fitur-fitur

yang mewakili setiap kata untuk setiap fitur yang ada pada dokumen. Menurut Patodkar dan I.R (2016), sebelum menentukan fitur-fitur yang mewakili, diperlukan tahap *preprocessing* yang dilakukan secara umum dalam *text mining* pada dokumen, yaitu *case folding*, *filtering*, dan *tokenizing* seperti yang ditunjukkan pada Gambar 2.2.



Gambar 2.2 Proses normalisasi kalimat (Patodkar dan I.R, 2016)

2.4. Analisis Sentimen

Informasi tekstual secara umum dapat dibagi menjadi informasi fakta dan opini (Liu, 2012). Fakta adalah ekspresi obyektif terhadap suatu benda, kejadian dan kepunyaan benda tersebut sedangkan opini biasanya berupa ekspresi subyektif yang menggambarkan sentimen, penilaian, atau perasaan seseorang terhadap suatu benda, kejadian atau kepunyaan dari benda tersebut dan analisis sentimen adalah bagian dari pekerjaan yang meninjau segala sesuatu berhubungan dengan pendapat komputasi, sentimen dan subjektivitas teks (Qiu et al., 2018). Tugas dasar dalam analisis sentimen adalah mengelompokkan polaritas dari teks yang ada dalam dokumen, apakah pendapat yang dikemukakan dalam dokumen bersifat positif, negatif atau netral

2.5. *Word2Vec*

Fitur pembobotan yang umum digunakan dalam klasifikasi teks adalah *bag-of-words*, tetapi saat ini *word vector* diusulkan sebagai cara baru untuk merepresentasikan kata di bidang *Natural Language Processing* (NLP). *Word*

vector dibentuk dengan bantuan jaringan saraf tiruan. Teknik ini diusulkan dengan nama *Word2Vec* (Mikolov et al., 2013).

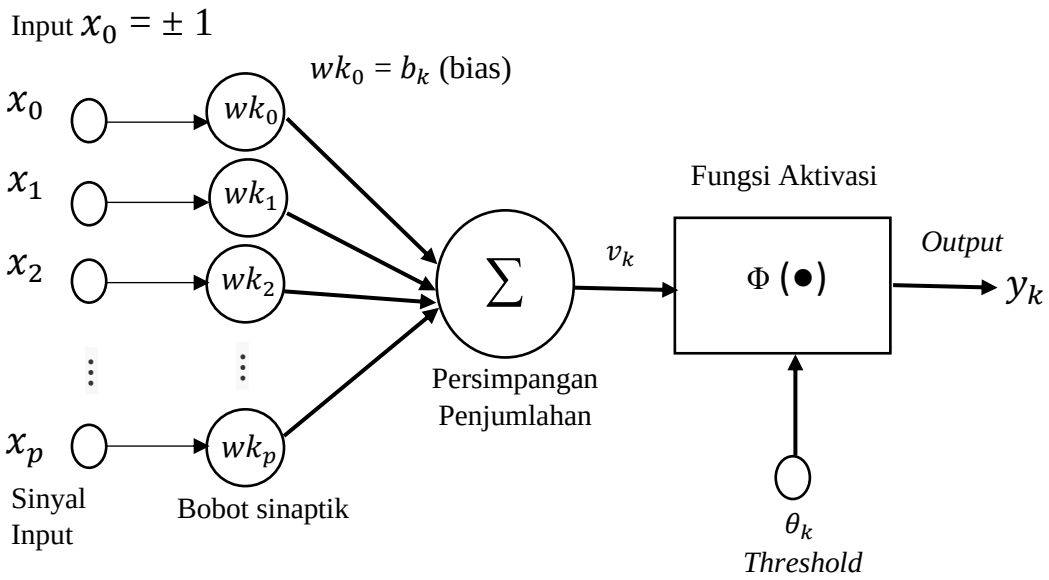
Pemanfaatan *Word2Vec* sebagai *word embedding* dalam klasifikasi dokumen di bahasa Inggris telah diusulkan oleh dua peneliti yaitu Lilleberg, Zhu, & Zhang, (2015) dan Xing, Wang, Zhang, & Liu, (2014).

Hasil penggunaan *Word2Vec* sebagai *word embedding* di penelitian yang lalu memiliki kinerja lebih baik jika dibandingkan dengan teknik representasi yang lain (Lilleberg et al., 2015). Penggunaan *Word2Vec* untuk klasifikasi dokumen berita di bahasa Indonesia juga telah diusulkan oleh Rahmawati & Khodra, (2016), hasil percobaan peneliti tersebut pada klasifikasi di Bahasa Indonesia memberikan angka *f1 score* terbaik, yaitu 81,10% dengan model yang digunakan adalah *Skip-Gram* dengan dimensi 200.

2.6. Jaringan Saraf Tiruan (JST)

Jaringan syaraf tiruan (*artificial neural network*) adalah sistem komputasi yang arsitektur dan operasinya diilhami dari pengetahuan tentang sel syaraf biologis di dalam otak. Jaringan syaraf tiruan merupakan salah satu representasi buatan dari otak manusia yang selalu mencoba menstimulasi proses pembelajaran pada otak manusia tersebut. Jaringan syaraf tiruan dapat digambarkan sebagai model matematis dan komputasi untuk fungsi aproksimasi non-linear, klasifikasi data *cluster* dan regresi non-parametrik atau sebuah simulasi dari koleksi model jaringan syaraf biologi. Perspektif saraf *deep learning* dimotivasi oleh dua ide utama. Gagasan pertama adalah bahwa otak memberikan bukti nyata melalui contoh yang diberikan, bahwa perilaku kecerdasan dapat dimungkinkan, dan caranya adalah merekayasa balik prinsip-prinsip komputasi di belakang otak dan menggandakan fungsinya. Perspektif lain adalah memahami otak dan prinsip-prinsip yang mendasari kecerdasan manusia, sehingga model pembelajaran mesin yang menjelaskan pertanyaan-pertanyaan ilmiah dasar ini akan berguna, selain dari kemampuan *deep learning* untuk menyelesaikan tugas – tugas tertentu (Goodfellow et al., 2016).

Permodelan pada Gambar 2.3 mewakili aktivitas aktual dari sel neuron. Semua *input* dijumlahkan seluruhnya dan dimodifikasi oleh bobot. Akhirnya, fungsi aktivasi mengontrol amplitudo *output*. Untuk contoh, rentang *output* yang dapat diterima biasanya antara 0 dan 1, atau bisa jadi -1 dan 1. Secara matematis, proses ini dijelaskan dalam Gambar 2.3.



Gambar 2.3 Model JST (Haykin, 2009)

Keterangan :

- $x_0, x_1, x_2 \ldots x_p$ adalah sinyal *input*
- Bobot (*weight*): wk_0, wk_1, $wk_2 \ldots wk_p$ adalah faktor bobot yang berhubungan dengan masing - masing *node*. Setiap *input* akan dikalikan dengan bobot dari *node*-nya masing – masing.
- *Threshold*: nilai ambang dari node θ adalah besarnya *offset* yang mempengaruhi aktivasi dari output node (lihat persamaan 1).

$$y_o : y_k = \sum_{j=1}^{p} X_j W_j - \theta, \quad (1)$$

Dari model ini, aktivitas interval jaringan saraf dapat ditunjukkan pada persamaan 2.

$$v_k = \sum_{j=1}^{p} w_{kj}\, x_j \,. \quad (2)$$

Output jaringan saraf y_k akan menjadi beberapa fungsi aktivasi pada nilai v_k.

Kemudian menghitung *total error* (*e*) untuk output y_k yang diperoleh dari node *k*, dan y_o, lalu dibandingkan dengan nilai yang sebernarnya yang disediakan oleh *dataset,* maka diperlakukan persamaan 3.

$$e = \frac{1}{n} \sum_{k=1}^{2} (y_o - y_k)^2 \quad (3)$$

2.6.1. Jenis pelatihan Jaringan Saraf Tiruan

Algoritma pembelajaran mesin dapat dikategorikan secara luas sebagai *supervised learning* dan *unsupervised learning* tergantung dari *dataset* seperti apa yang diberikan selama proses pembelajaran (Goodfellow et al., 2016), berikut penjelasannya menurut Goodfellow et al. (2016):

a. *Supervised learning*

Supervised learning secara garis besar, adalah algoritma pembelajaran yang belajar untuk mengasosiasikan beberapa *input* dengan beberapa *output*, diberikan satu set pelatihan untuk *input* x dan *output* y. Dalam banyak kasus, *output* y mungkin sulit untuk dikumpulkan secara otomatis dan harus disediakan oleh manusia, tetapi istilah tersebut tetap berlaku bahkan ketika pelatihan menetapkan target yang dikumpulkan secara otomatis. Jenis pelatihan ini menganalisis data pelatihan dan menghasilkan model yang akan disimpulkan oleh mesin, model tersebut akan dapat digunakan untuk memetakan contoh-contoh baru. Sebuah skenario pembelajaran yang optimal akan memungkinkan algoritma menentukan label kelas dengan benar untuk contoh yang belum pernah terlihat.

b. *Unsupervised learning*

Unsupervised learning adalah tugas pembelajaran mesin untuk menentukan suatu fungsi dari data yang tidak berlabel. Secara khusus, karena data tidak berlabel, tidak ada kesalahan atau penghargaan untuk memberi tahu algoritma jika dekat atau jauh dari solusi yang tepat. *Unsupervised learning* sangat penting ketika menggunakan pembelajaran mesin pada masalah di mana jawabannya tidak diketahui.

Perbedaan antara algoritma yang diawasi dan yang tidak diawasi tidak didefinisikan

secara formal dan kaku, karena tidak ada tes objektif untuk membedakan apakah suatu nilai adalah fitur atau target yang disediakan oleh *dataset*. Secara informal, *unsupervised learning* mengacu pada sebagian besar upaya untuk mengekstraksi informasi dari distribusi yang tidak memerlukan tenaga manusia untuk memberi anotasi contoh. Istilah ini biasanya dikaitkan dengan estimasi kerapatan, belajar mengambil sampel dari suatu distribusi, pembelajaran untuk mengurangi *noise* pada data dari beberapa distribusi, klasterisasi data, atau mengelompokkan data ke dalam kelompok-kelompok berdasarkan contoh terkait.

2.6.2. Fungsi Aktivasi

Fungsi Aktivasi merupakan operasi matematik yang dikenakan pada sinyal output *y*. Ada beberapa fungsi aktivasi yang biasa dipakai dalam JST tergantung dari masalah yang akan diselesaikan.

Ada beberapa pilihan fungsi aktivasi yang digunakan dalam metode JST, seperti fungsi *sigmoid* biner, dan *sigmoid* bipolar. Karakteristik yang harus dimiliki fungsi fungsi aktivasi tersebut adalah kontinue, diferensiabel, dan tidak menurun secara monoton. Fungsi aktivasi diharapkan dapat mendekati nilai-nilai maksimum dan minimum secara baik (Aggarwal, 2018)**.**

Fungsi aktivasi bertindak sebagai fungsi pemerataan, sehingga *output* dari *neuron* dalam jaringan saraf antara nilai-nilai tertentu yaitu 0 dan 1, atau -1 dan 1. Secara umum, ada tiga jenis fungsi aktivasi, dilambangkan dengan Φ (●).

Pertama, ada fungsi ambang yang mengambil nilai 0 jika *input* yang dijumlahkan kurang dari nilai ambang tertentu (*v*), dan nilai 1 jika *input* yang dijumlahkan lebih besar dari atau sama dengan nilai ambang (lihat persamaan 4).

$$\Phi(v) = \begin{cases} 1 \; if \; v \geq 0 \\ 0 \; if \; v < 0 \end{cases} \tag{4}$$

Kedua, ada fungsi *Piecewise – Linear* (lihat persamaan 5). Fungsi ini lagi dapat mengambil nilai 0 atau 1, tetapi juga dapat mengambil nilai di antara itu tergantung pada faktor amplifikasi di wilayah operasi linear tertentu.

$$\Phi(v) = \begin{cases} 1, & v \geq \frac{1}{2} \\ v, -\frac{1}{2} > v > \frac{1}{2} \\ 0, & v \leq -\frac{1}{2} \end{cases} \quad (5)$$

Ketiga, ada fungsi *sigmoid*. Fungsi ini dapat berkisar antara 0 dan 1, tetapi terkadang juga berguna untuk menggunakan rentang -1 hingga 1. Contoh dari fungsi *sigmoid* adalah fungsi tangen hiperbolik (lihat persamaan 6).

$$\Phi(v) = \tanh\left(\frac{v}{2}\right) = \frac{1-\exp(-v)}{1+\exp(-v)} \quad (6)$$

2.6.3. Jaringan Saraf Tiruan *Backpropagation*

Dalam *machine learning*, *backpropagation* adalah algoritma yang banyak digunakan dalam pelatihan jaringan saraf tiruan untuk *supervised learning*. Dalam pelatihan jaringan saraf, *backpropagation* menghitung gradien dari *loss function* (fungsi kerugian) sehubungan dengan bobot jaringan untuk satu contoh *input-output* yang disediakan oleh *dataset*, kemudian melakukan penghitungan fungsi kerugian secara efisien, tidak seperti perhitungan langsung secara naif dari gradien sehubungan dengan masing-masing berat secara individual (Goodfellow et al., 2016).

Efisiensi ini membuat metode *backpropagation* layak untuk menggunakan metode gradien untuk melatih jaringan *multi-layer*, kemudian memperbarui bobot untuk meminimalkan kerugian, biasanya *backpropagation* menggunakan optimasi *gradient descent*, atau varian seperti *stochastic gradient descent*. Algoritma *backpropagation* bekerja dengan menghitung gradien dari fungsi kerugian dengan memperhatikan setiap bobot menurut aturan rantai kalkulus, kemudian menghitung gradien satu lapisan pada satu waktu, lalu merunut mundur dari lapisan terakhir untuk menyesuaikan setiap bobot untuk tiap lapisan berdasarkan aturan rantai kalkulus, metode ini adalah contoh dari pemrograman dinamis (Goodfellow et al., 2016).

Backpropagation digunakan digunakan seteleah penghitungan *total error* menggunakan persamaan 3, dikarenakan *error* akan sangat besar, sehingga *backpropagation* dapat digunakan untuk menyesuaikan bobot untuk mengurangi kesalahan antara output jaringan dan nilai yang diinginkan.

Backpropagation dapat menyesuaikan bobot jaringan menggunakan metode optimisasi *stochastic gradient descent* (Goodfellow et al., 2016) ditunjukan pada persamaan 7.

$$w_k^{i+1} = w_k^i - \eta \frac{\partial e}{\partial w_i^k} \tag{7}$$

Dimana *i* adalah iterasi, dan η adalah *learning rate* dan $\frac{\partial e}{\partial w_i^k}$ adalah turunan dari total kesalahan sehubungan dengan bobot yang disesuaikan.

Setelah menghitung turunan untuk semua bobot dalam jaringan (turunan gradien yang sama), secara bersamaan dapat memperbarui semua bobot di jaringan dengan rumus *gradient descent*, seperti yang ditunjukkan persamaan 8.

$$\begin{bmatrix} w_1^{i+1} \\ w_2^{i+1} \\ \vdots \\ w_n^{i+1} \end{bmatrix} = \begin{bmatrix} w_1^i \\ w_2^i \\ \vdots \\ w_n^i \end{bmatrix} - \eta \begin{bmatrix} \frac{\partial e}{\partial w_1^k} \\ \frac{\partial e}{\partial w_2^k} \\ \vdots \\ \frac{\partial e}{\partial w_n^k} \end{bmatrix} \tag{8}$$

2.6.4. *Optimizer*

Selama proses pelatihan, parameter bobot (*weight*) pada model dilakukan *tweaking* dan memperbaharui bobot untuk mencoba dan meminimalkan fungsi kerugian (*loss function*) tersebut, dan menjadikan prediksi model seakurat mungkin (Kingma & Ba, 2014b). Di sinilah penggunaan *optimizer* dibutuhkan. *Optimizer* mengikat *loss function* dan parameter model dengan memperbarui bobot model sebagai respons terhadap *output* dari *loss function*. Dalam istilah yang lebih sederhana, pengoptimalisasi membentuk model ke dalam bentuknya yang paling

akurat dengan memanfaatkan bobotnya (Goodfellow et al., 2016). Adapun beberapa optimizer yang sering digunakan yaitu:

1. *Gradient descent*

Gradient descent adalah algoritma optimisasi pembelajaran mesin iteratif untuk mengurangi fungsi biaya (*loss function*). *Gradient descent* akan membantu model untuk membuat prediksi yang akurat.

Gradien akan menunjukkan arah kenaikan setelah melewati titik terrendah. Karena akurasi model yang baik ada pada titik minimum di lembah, model harus pergi ke arah yang berlawanan dari gradien. *Gradient descent* memperbarui parameter dalam arah gradien negatif untuk meminimalkan kerugian (*loss function*). Berikut persamaan untuk *Gradient descent* dapat dilihat pada persamaan 9:

$$\theta = \theta - \eta \nabla J(\theta; x, y) \tag{9}$$

Keterangan:

θ = bobot parameter (*weigth*)

η = *learning rate*

$\nabla J(\theta; x, y)$ = bobot gradien dari parameter θ

2. *RMSprop*

RMSprop adalah algoritma optimasi yang tidak dipublikasikan yang dirancang untuk jaringan saraf, *RMSprop* terletak pada bidang metode tingkat pembelajaran adaptif, yang telah meningkat popularitasnya dalam beberapa tahun terakhir, tetapi juga mendapatkan beberapa kritik (Wilson et al., 2017).

Gagasan utama RMSprop adalah untuk menjaga rata-rata gradien kuadrat untuk setiap bobot. Dan kemudian membagi gradien dengan akar kuadrat dari rata – rata kuadrat (*mean square*). Itulah mengapa disebut RMSprop (*root mean square*). RMSprop ditunjukan pada persamaan 10 dengan aturan pembaruan sebagai berikut.

$$E[g^2]_t = \beta E[g^2]_{t-1} + (1-\beta)(\frac{\delta C}{\delta w})^2 \quad (10)$$

Kemudian untuk memperbaharui bobot dilakukan persamaan 11, sebagai berikut.

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{E[g^2]_t}} \frac{\delta C}{\delta w} \quad (11)$$

Keterangan:

E[g] = rata-rata dinamis dari gradien kuadrat

$\frac{\delta C}{\delta w}$ = gradien dari fungsi biaya (*loss function*) sehubungan dengan bobot

η = *learning rate*

β = *hyperparameter*, biasanya diberikan nilai 0.9

3. *Adam*

Adam adalah algoritma optimisasi tingkat pembelajaran adaptif yang dirancang khusus untuk melatih *deep learning* (Kingma & Ba, 2014a). Adam dapat dilihat sebagai kombinasi dari *RMSprop* dan *Stochastic Gradient Descent* dengan momentum. *Adam* menggunakan gradien kuadrat untuk mengukur *learning rate* seperti *RMSprop* dan mengambil keuntungan dari momentum dengan menggunakan rata-rata dinamis dari gradien dan bukannya gradien itu sendiri seperti *SGD* dengan momentum (Goodfellow et al., 2016).

Adam memiliki dua moment dinamis yang akan dihitung, pertama *mean* (*m*) yang diberikan persamaan 12 sebagai berikut.

$$m_t = (1-\beta_1) \sum_{i=0}^{t} \beta_1^{t-i} g_i \quad (12)$$

Moment kedua adalah varians (*v*) yang tidak terpusat (artinya tidak mengurangi *mean* selama perhitungan varians). Yang ditunjukan dengan persamaan 13.

$$v_t = (1-\beta_2) \sum_{i=0}^{t} \beta_2^{t-i} g_t^2 \quad (13)$$

Keterangan:

m, v = estimator

g = gradien pada *mini-batch* saat ini

β_1 = *hyperparameter,* dengan nilai 0.9

β_2 = *hyperparameter,* dengan nilai 0.999

Nilai yang diharapkan dari estimator harus sama dengan parameter yang diestimasi agar tidak terjadi bias, kemudian berikan persamaan 14 dan 15 untuk mengoreksi estimator, sehingga nilai yang diharapkan adalah sama seperti yang dinginkan. Langkah ini biasanya disebut koreksi bias. Berikut persamaan 14 dan 15.

$$\widehat{m_t} = \frac{m_t}{1-\beta_1^t} \tag{14}$$

$$\widehat{v_t} = \frac{m_t}{1-\beta_2^t} \tag{15}$$

Kemudian menggunakan rata-rata dinamis tersebut untuk mengukur *learning rate* secara individual untuk setiap parameter, maka digunakan persamaan 16.

$$w_t = w_{t-1} - \eta \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}+\epsilon} \tag{16}$$

2.6.5. *Loss Function*

Dalam konteks algoritma optimisasi, fungsi yang digunakan untuk mengevaluasi solusi kandidat disebut sebagai fungsi objektif. Fungsi yang ingin diperkecil atau maksimalkan disebut pula fungsi objektif. Ketika meminimalkan fungsi objektif tersebut, secara tidak langsung dapat menyebutnya fungsi biaya (*cost function*), fungsi kerugian (*loss function*), atau fungsi kesalahan (*error function*) (Goodfellow et al., 2016)

Adapun beberapa *loss function* yang sering digunakan dalam pelatihan *deep learning* pada buku Goodfellow et al., (2016) sebagai berikut.

1. MSE (*Mean Square Error*)

MSE adalah *loss function*, berdasarkan nilai yang diharapkan dari hilangnya kesalahan kuadrat, MSE sering digunakan dalam tugas regresi dalam pelatihan, tetapi secara berangsur-angsur digantikan oleh *cross-entrophy* dan MLE (*maximum likehood estimator*) untuk perhitungan statistik maupun pembelajaran mesin (Goodfellow et al., 2016). Rumus MSE dapat dilihat pada persamaan 17.

$$L = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{17}$$

2. *Binary Crossentrophy*

Binary crossentropy adalah *loss function* yang digunakan dalam tugas klasifikasi biner. Klasifikasi biner adalah tugas yang menghasilkan nilai dengan hanya dua pilihan semisal 0 dan 1. Rumus *binary crossentropy* dapat dilihat pada persamaan 18.

$$\mathrm{L} = -\frac{1}{N} \sum_{i=1}^{N} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i) \tag{18}$$

3. *Categorical Crossentrophy*

Categorical crossentropy adalah fungsi kerugian yang digunakan dalam tugas klasifikasi multi-kelas. Klasifikasi multi-kelas adalah tugas-tugas di mana contoh hanya dapat dimiliki salah satu dari banyak kategori yang mungkin, dan model harus memutuskan yang mana. Dengan kata lain data yang digunakan haruslah memiliki sifat *mutual exclusive* terhadap label.

Rumus *categorical crossentropy* dapat dilihat pada persamaan 19.

$$L = -\sum_{i=1}^{N} y_i \cdot \log \hat{y}_i \tag{19}$$

Keterangan:

y_i = nilai label yang sebenarnya

$\hat{y}_i$ = nilai prediksi

N = jumlah kelas

2.6.6. *Early Stopping*

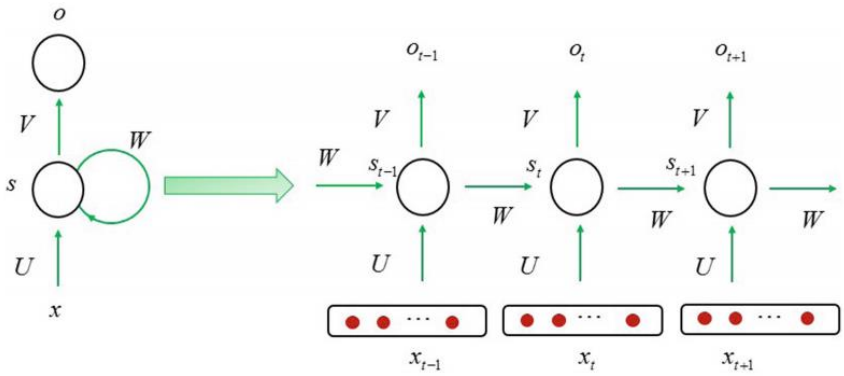
Early stopping adalah fungsi untuk menghentikan pelatihan sebelum pelatihan mengalami *overfitting.* Menurut pendapat Prechelt (2012) pelatihan dalam JST MLP dibagi menjadi tiga fase, fase pertama adalah dimana pelatihan dimulai dimana *complexity error* masih sedikit dan *approximation error* masih besar, fase kedua adalah dimana pelatihan pada titik optimalnya dimana *approximation error* mulai mengecil dan *complexity error* masih sedikit, dan fase ketiga adalah dimana pelatihan mulai menunjukkan berkembangnya *complexity error* selama pelatihan setelah fase kedua berlalu.

Kemudian dijelaskan pula pada awal pelatihan (fase I), kesalahannya didominasi oleh apa yang sebut kesalahan aproksimasi (*approximation error*), jaringan hampir tidak mempelajari apa pun dan masih memiliki bias yang besar. Selama pelatihan, bagian *approximation error* semakin berkurang. Namun, pada saat yang sama, komponen kesalahan yang lain meningkat yaitu kesalahan kompleksitas (*complexity error*) yang disebabkan oleh meningkatnya varians model jaringan seiring besarnya kemungkinan dan keragaman bobot yang tumbuh. Jika algoritma dilatih cukup lama, kesalahan akan didominasi oleh *complexity error* (fase III). Oleh karena itu, ada fase selama pelatihan, ketika perkiraan dan kompleksitas komponen dari kesalahan bersaing tetapi tidak satupun dari mereka yang mendominasi (fase II). Tugas *early stopping* seperti yang dijelaskan dalam pada penelitian Prechelt (2012) adalah untuk mendeteksi kapan fase II berakhir dan dominasi bagian varians dimulai.

2.7. *Recurrent Neural Networks* (RNN)

Manusia tidak mengulang informasi baru yang masuk setiap saat. Seseorang yang membaca suatu novel, akan memahami setiap kata berdasarkan pada pemahaman mereka tentang kata-kata sebelumnya. Hal ini diupayakan agar tidak membuang segala informasi yang didapat dan mulai berpikir dari awal lagi. Jaringan saraf tradisional tidak dapat melakukan ini. Misalnya, seseorang ingin mengklasifikasikan adegan seperti apa yang terjadi di setiap bab dalam novel. Tidak

jelas bagaimana jaringan saraf tradisional dapat menggunakan pemahaman informasi yang didapat tentang peristiwa pada bab sebelumnya untuk diinformasikan maupun diprediksi. RNN dapat mengatasi masalah ini, RNN adalah jaringan berulang, hal ini memungkinkan informasi untuk bertahan untuk bisa digunakan kembali (Mikolov et al., 2014). Arsitektur RNN berdasarkan (Li & Xu, 2018) dapat dilihat pada Gambar 2.4.



Gambar 2.4 Arsitektur RNN (Li & Xu, 2018)

Berikut keterangan simbol untuk Gambar 2.4.

- x_t adalah input pada setiap langkah atau *time step*
- s_t adalah *hidden state* pada setiap *time step* t.
- o_t adalah *output* untuk setiap *step* t
- U, V, W adalah matriks parameter pada RNN. RNN tidak menggunakan parameter yang berbeda untuk disetiap *layer,* hal ini memungkinkan untuk menghemat parameter yang harus dipelajari karena RNN menjalankan tugas yang sama untuk tiap langkahnya tetapi berbeda input saja.

Hidden state bisa disebut sebagai "*memory*" pada sebuah *network* atau jaringan yang berfungsi menyimpan hasil kalkulasi dan rekaman yang telah dilakukan. s_t dihitung sesuai dengan *hidden state* sebelumnya dengan *input* saat ini. Maka diberlakukan persamaan 20.

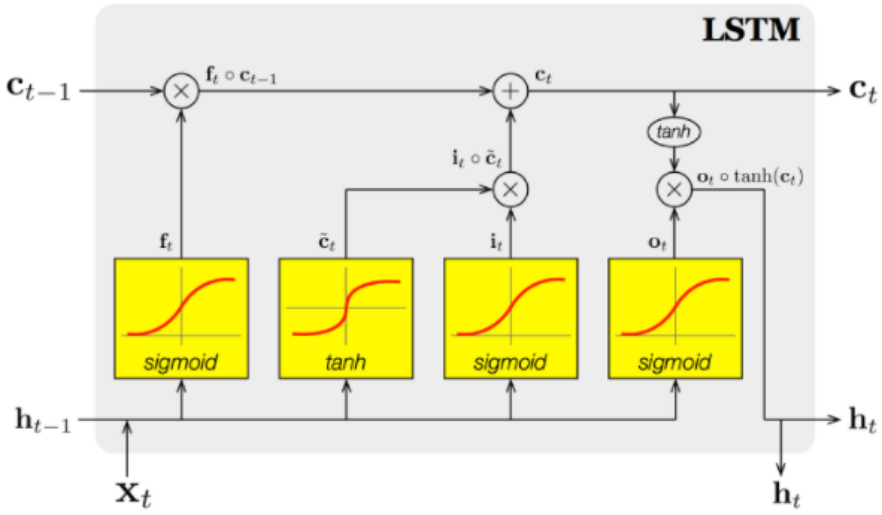
$$s_t = f(Ux_t + Ws_{t-1}) \tag{20}$$

Pada fungsi f biasanya adalah fungsi non-linearitas seperti *tanh* pada persamaan 20 s_{t-1} digunakan untuk menghitung *hidden state* yang pertama.

Tetapi s_t tidak bisa merekam informasi terlalu banyak untuk tiap *time stepnya* (Mikolov et al., 2014)

2.8. *Recurrent Neural Networks Long - Short Term Memory* (RNN-LSTM)

RNN-LSTM merupakan salah satu pengembangan *neural network* yang mampu mempelajari *long-term dependency*. LSTM diperkenalkan oleh Hochreiter dan Schmidhuber pada tahun 1997, yang kemudian semakin disempurnakan dan dipopulerkan hingga saat ini (Baytas et al., 2017). Penjelasan yang umum digunakan untuk LSTM terdiri dari *forget gate, input gate, output gate*. Gambaran LSTM secara umum dapat dilihat pada Gambar 2.5.



Gambar 2.5 Arsitektur LSTM (Eugine Kang, 2017)

Pada arsitektur RNN-LSTM terdapat nilai *cell state* yang dapat disimpan dalam periode lama maupun singkat pada *cell*. Secara teori RNN dapat menangani *long-term dependency*, tetapi RNN tidak dapat memilih parameter untuk memecahkan suatu masalah, yang berakibat RNN tidak dapat mempelajari masalah tersebut (Seo et al., 2020). Fungsi aktivasi *sigmoid* pertama adalah *forget gate*, informasi mana yang harus dilupakan dari keadaan sel sebelumnya $(c_t - 1)$. *Sigmoid* kedua dan pertama tanpa fungsi aktivasi adalah gerbang *input gate*.

Sigmoid terakhir adalah *output gate* dan menyoroti informasi mana yang harus menuju ke *hidden status***.**

Berikut adalah penjelasan gerbang – gerbang yang ada pada LSTM (Goodfellow et al., 2016):

1. *Forget gate* (f_t)

Gerbang pertama, *forget gate* (f_t). Gerbang ini memutuskan informasi apa yang harus dibuang atau disimpan. Informasi dari keadaan *hidden state* sebelumnya dan informasi dari *input* saat ini dilewatkan melalui fungsi *sigmoid*. Nilai *output* antara 0 dan 1. Semakin dekat ke 0 berarti dibuang, dan semakin dekat ke 1 berarti disimpan. Rumus dari f_t dapat dilihat pada persamaan 21.

$$f_t = \sigma\,(W_f S_{t-1} + W_f X_t) \tag{21}$$

Keterangan W_f = bobot dari *forget gate*, S_{t-1} = *state* sebelumnya, X_t = *input* pada waktu t, dan σ = fungsi aktivasi *sigmoid*

2. *Input gate* (i_t)

Untuk memperbarui *cell state*, LSTM memiliki *input gate* (i_t) . Pertama, nilai melewati *hidden state* sebelumnya dan *input* saat ini ke dalam fungsi *sigmoid*. Bertujuan untuk memutuskan nilai mana yang akan diperbarui dengan mengubah nilai menjadi antara 0 dan 1. 0 berarti tidak penting, dan 1 berarti penting. Nilai juga melewati *hidden state* dan *input* fungsi *tanh* untuk menekan nilai antara -1 dan 1 untuk membantu mengatur jaringan. Kemudian nilai akan mengandakan *output tanh* dengan *output sigmoid*. *Output sigmoid* akan memutuskan informasi mana yang penting untuk disimpan dari *output tanh*. Rumus *Input gate* ditunjukkan pada persamaan 22.

$$i_t = \sigma\,(W_i S_{t-1} + W_i X_t) \tag{22}$$

Keterangan W_i = bobot dari *input gate*, S_{t-1} = *state* sebelumnya, dan X_t = *input* pada waktu *t*, dan σ = fungsi aktivasi *sigmoid*. Kemudian nilai gerbang *input* dikalikan dengan *output* dari lapisan kandidat ($\tilde{C}$). Persamaan 23 merupakan rumus dari $\tilde{C}$.

$$\tilde{C} = tanh\,(W_c S_{t-1} + W_c X_t) \tag{23}$$

Lapisan ini menerapkan tangen hiperbolik (tanh) ke campuran *input* dan *output* sebelumnya, mengembalikan vektor kandidat yang akan ditambahkan ke *state*. *State* diperbarui dengan rumus persamaan 24.

$$c_t = (i_t * \tilde{c}_t + f_t * c_{t-1}) \quad (24)$$

Keterangan $\tilde{C}$ = *intermediate cell state,* W_c = bobot dari *cell state,* S_{t-1} = *state* sebelumnya X_t = *input* pada waktu *t, state* sebelumnya dikalikan dengan *forget gate* dan kemudian ditambahkan ke fungsi kandidat baru yang diizinkan oleh *output gate*.

3. *Output gate* (o_t)

Terakhir adalah *output gate*. *Output gate* memutuskan *hidden state* berikutnya. Dimana *hidden state* berisi informasi tentang *input* sebelumnya. Pertama, nilai melewati *hidden state* sebelumnya dan *input* saat ini ke fungsi *sigmoid*. Kemudian nilai melewati *state cell* yang baru dimodifikasi ke fungsi *tanh*. Kemudian melipat gandakan *output tanh* dengan *output sigmoid* untuk memutuskan informasi apa yang harus dibawa oleh *hidden state*. *Output*-nya adalah *hidden state*. Keadaan *cell* dan *hidden* baru, kemudian dibawa ke langkah berikutnya. Rumus *output gate* (o_t) ditunjukkan pada persamaan 25.

$$o_t = \sigma (W_o S_{t-1} + W_o X_t) \quad (25)$$

Keterangan W_o = bobot dari *output gate,* iS_{t-1} = *state* sebelumnya, X_t = *input* pada waktu *t,* dan σ = fungsi aktivasi *sigmoid.* Rumus dari *cell state* (h_t) baru ditunjukkan pada persamaan 26.

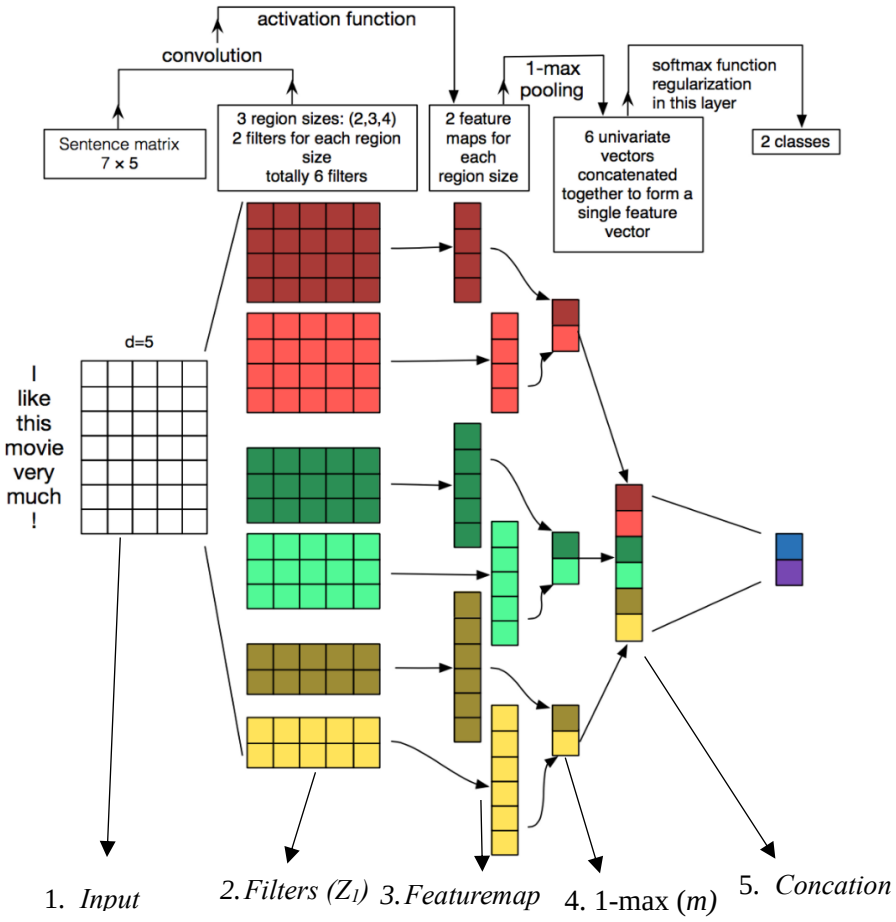
$$h_t = o_t * \tanh (c_t) \quad (26)$$

Ketiga *gate* di atas memiliki bobot dan bias yang independen, oleh karena itu jaringan akan belajar berapa banyak dari *output* terakhir yang harus disimpan, berapa banyak dari *input* saat ini untuk disimpan, dan berapa banyak dari *state* untuk dikirim ke *output*.

2.9. *Convolutional Neural Network* (CNN)

Dalam *Deep Learning*, CNN adalah kelas jaringan saraf yang dalam, paling umum diterapkan untuk menganalisis citra visual (Goodfellow et al., 2016). CNN juga dikenal sebagai *shift invariant* atau *space invariant artificial neural networks*, berdasarkan arsitektur *shared-weight* dan karakteristik terjemahan invariant (Yaseen, 2018).

Tetapi beberapa tahun terakhir ini, CNN mendapat perhatian yang luas dikarenakan tidak hanya baik dalam menyelesaikan tugas – tugas dalam *Image Processing* tetapi juga dalam *Natural Language Processing* (NLP), dikarenakan CNN pula telah berhasil dalam berbagai tugas klasifikasi teks (Brownlee, 2017). Dalam penelitian Kim, (2014), menunjukkan bahwa CNN sederhana dengan sedikit penyetelan *hyperparameter* dan vektor statis mencapai hasil yang sangat baik pada beberapa tolok ukur peningkatkan *state-of-the-art* pada 4 dari 7 tugas. Dapat dilihat pada Gambar 2.6, bagaimana CNN melakukan klasifikasi pada teks menurut Y. Zhang dan Wallace, (2015)**:**



Gambar 2.6 Arsitektur CNN untuk klasifikasi kalimat (Y. Zhang dan Wallace, 2015)

Penjelasan:

1. Input (x)

Input (x) yaitu kalimat yang telah di-*encoding* pada proses sebelumnya menjadi matriks 7×5, dan tanda seru diperlakukan sebagai kata.

2. *Filters (f)*

Salah satu sifat yang diinginkan dari CNN adalah mempertahankan orientasi spasial 2D dalam visi komputer. Teks, seperti gambar, juga memiliki orientasi. Walaupun 2 dimensi, teks memiliki struktur satu dimensi di mana urutan kata penting. Ukuran wilayah mengacu pada jumlah baris yang mewakili kata dari matriks kalimat yang akan disaring. Di sini, Y. Zhang dan Wallace (2015) memilih

untuk menggunakan 6 filter – 2 filter komplementer untuk mempertimbangkan 2, 3, dan 4 kata. Sehingga diberlakukan rumus seperti pada persamaan 27.

$$Z_1 = x * f_n \tag{27}$$

Kemudian, filter bergerak ke bawah 1 kata dan menindih pada vektor kata dibawahnya dan melakukan operasi yang sama. Oleh karena itu *ouput* maktriks (*o*) akan memiliki bentuk $s - h + 1 * 1$, dimana *s* adalah jumlah kolom matrik input dan *h* adalah jumlah kolom matrik *filter*.

3. *Featuremap* (*c*)

Setelah dilakukan perkalian, maka didapat *output (o)* sementara, untuk mendapatkan *featuremap* (*c*), peneliti menambahkan istilah bias (skalar, misalnya. bentuk 1*1) dan menerapkan fungsi aktivasi (A_1), semisal fungsi aktivasi ReLU.

$$ReLu = R(x) = \max(0, x).$$

maka diperlakukan fungsi dengan persamaan 28.

$$A_1 = ReLu(Z_1) \tag{28}$$

Hal ini menghasilkan *c*, dengan bentuk yang sama dengan *o* ($s - h + 1 * 1$).

4. 1-max (m)

Setelah berhasil melakukan penghitungan fitur, maka dilakukan $1 - max$ (mengambil nilai terbesar untuk setiap fitur), perhatikan bahwa dimensi *c* bergantung pada *s* dan *h*, dengan kata lain, panjang kolom *c* bervariasi di antara kalimat dengan panjang yang berbeda dan filter dari ukuran wilayah yang berbeda. Untuk mengatasi masalah ini, peneliti menggunakan fungsi penyatuan *1-max* dan mengekstrak angka terbesar dari setiap vektor *c*. Maka digunakanpersamaan 29.

$$m_n = max(c_n) \tag{29}$$

5. Pengabungan *1-max (concatenate 1-max)*

Setelah "*1-max pooling* (*m*)", didapat vektor dengan panjang yang tetap yakni 6 elemen. Vektor dengan panjang tetap ini kemudian dapat dimasukkan ke dalam fungsi *softargmax* agar terhubung penuh.

maka diberlakukan persamaan 30.

$$softargmax = fi(x) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad (30)$$

untuk melakukan klasifikasi maka vektor *m* diberlakukan persamaan 31.

$$ouput = softargmax(m) \quad (31)$$

Kesalahan dari klasifikasi kemudian diperbanyak kembali ke parameter berikut sebagai bagian dari pembelajaran:

a. Matriks yang diproduksi *o*
b. Istilah bias yang ditambahkan ke *o* untuk menghasilkan *c*
c. Vektor kata (opsional, menggunakan kinerja data validasi)

2.10. Penelitian Terkait

Penelitian ini dikembangkan dari beberapa referensi yang mempunyai keterkaitan dengan metode dan objek penelitian. Penggunaan referensi ini ditujukan untuk memberikan batasan-batasan terhadap metode dan sistem yang nantinya akan dikembangkan lebih lanjut. Berikut adalah hasil dari penelitian sebelumnya:

1. Ibrahim et al., (2018) melakukan penelitian pada klasifikasi multi-label pada dataset *Multilabel Wikipedia's Talk Page Edits* yang disediakan oleh Kaggle yang berbahasa Inggris, peneliti menggunakan strategi data augmentasi untuk menangani masalah ketidak seimbangan data. Jumlah data terdiri dari 159.571 buah dan terdiri dari enam kelas, peneliti menggunakan pembagian data sebesar 80% *data training,* 10 % *data testing*, dan 10 % data validasi. Penelitian tersebut membandingkan terhadap beberapa jenis metode dengan data augmentasi untuk klasifikasi yang bertipe kasar (*toxic*), yaitu NB-SVM dengan hasil *f1 score* sebesar 85.60 %, CNN sebesar 86.75 %, CNN Esemble 87.11 %, Bidirectional LSTM 85.98 %, Bidirectional GRU 86.43 %, dan *proposed method* dari peneliti

mendapatkan hasil *f1 score* sebesar 87.24 %. Peneliti memperkenalkan metode data augmentasi untuk menangani kelas yang tidak seimbang pada dataset.

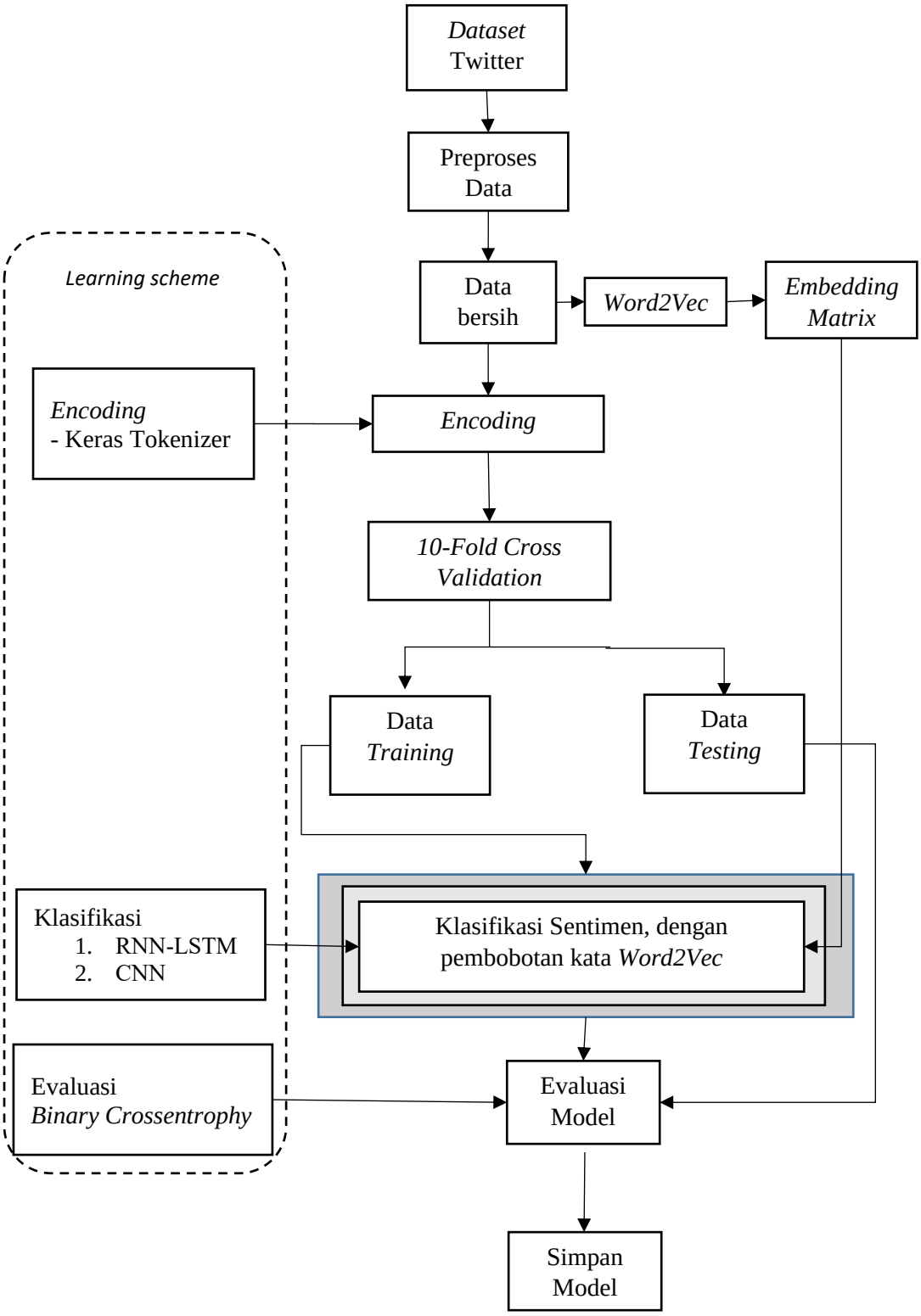
2. Jang, Kim, dan Kim, (2019) melakukan penelitian *tweet classification* pada artikel berita dan *tweet* pada Twitter. Penelitian mereka menggunakan *dataset* dari API Twitter dan NAVER yang berbahasa Korea dengan 35 kata kunci yang spesifik sebanyak 291.309 untuk data *tweet* dan 122.258 untuk artikel berita, variabel yang digunakan untuk penelitian ini adalah CNN yang dikombinasikan dengan *word embedding* seperti *Skip-gram* atau *Continous Bag-Of-Word* (CBOW). Skenario yang diberikan pada penelitian tersebut adalah membagi *dataset* menjadi dua bagian yang kemudian dilakukan *unsupervided learning* untuk membuat *Word2Vec* CBOW dan *Word2Vec Skip-gram,* untuk pengujiannya peneliti melakukan perbandingan antara data artikel berita dan artike *tweet* dengan klasifikasi CNN. Hasil dari penelitian tersebut CNN + CBOW untuk artikel berita mendapatkan hasil *f1 score* sebesar 93.51 % sedangkan untuk data *tweet,* CNN + *Skip-gram* unggul dengan *f1 score* sebesar 90.97%. Peneliti mengamati bahwa penggunaan *Word2Vec* yang mempelajari hubungan semantik antara kata - kata secara signifikan dapat meningkatkan kinerja model klasifikasi. Dari penelitian tersebut menunjukan bahwa artikel berita memiliki kinerja yang baik apabila menggunakan CBOW, sedangkan untuk data *tweet* menghasilkan kinerja yang baik pula apabila menggunakan *Skip-gram.* Hal ini menyiratkan bahwa penggunaan algoritma tertentu berdasarkan jenis data yang akan dianalisis dapat menghasilkan kinerjan yang lebih baik. Data artikel berita memiliki *f1 score* yang tinggi dikarenakan model CNN dapat mengekstraksi fitur dan melakukan klasifikasi lebih akurat dengan format kata yang lebih seragam dibandingkan data *twee*t.

3. Heikal, Torki, & El-Makky, (2018) melakukan penelitian tentang analisis sentimen berbahasa Arab menggunakan metode *Deep Learning CNN-LSTM.* Penelitian mereka menggunakan *ASTD (Arabic Sentiment Tweets Dataset) dataset* yang terdiri dari 10.000 *tweet* yang didistribusikan menjadi 4 kelas positif, negatif, netral, dan objektif. Peneliti menggunakan prosedur yang sama seperti penelitian Baly et al., (2017)**,** tetapi menghapus tweet kelas objektif, karena peneliti memfokuskan pada klasifikasi sentimen opini daripada subjektifitas klasifikasi.

Dengan demikian *dataset* yang digunakan hanya 3.315 *tweet* dengan distribusi kelas 24% kelas positif, 51% kelas negatif, dan 25% kelas netral. Kemudian membagi *dataset* menjadi 3 bagian yaitu 70% pelatihan, 10% validasi, dan 20% pengujian. Peneliti juga mengunakan presentase pembagian *dataset* sama seperti penelitian Baly et al., (2017) untuk perbandingan yang adil terdahap penelitian Baly et al., (2017)**.** Hasil dari penelitian tersebut menggunakan *pre-trained word vector* yang menghasilkan akurasi dari metode CNN sebesar 64.30%, akurasi metode LSTM sebesar 64.75%, dan akurasi dari usulan metode dari peneliti yaitu CNN-LSTM sebesar 65.05%. Hasil yang dicapai dalam penelitian ini cukup signifikan walaupun hanya menggunakan sistem yang sederhana dan pula meninjau Bahasa Arab yang memiliki banyak tantangan, seperti strukturnya yang kompleks, berbagai macam dialek, serta kurangnya sumber daya yang ada.

2.11. Kerangka Berfikir

Model kerangka berpikir yang diterapkan adalah menerapkan algoritma RNN-LSTM dan CNN untuk klasifikasi sentimen pada *tweet* yang berorientasi pada ujaran kebencian dan umpatan. Setelah *preprocessing* selesai, dilakukan *word embedding* menggunakan *Word2Vec, Word2Vec* melatih kata-kata terhadap kata-kata lain yang bertetangga dalam corpus *data input*. Kemudian data yang telah melewati tahap *preprocessing* dilakukan *encoding data* yaitu menyandikan setiap kata dalam vektor. Kemudian membagi data menjadi dua bagian menggunakan metode *10 Fold cross validation,* setelah itu melaksanakan tugas klasifikasi yang dilanjutkan dengan evaluasi. Kerangka berpikir dapat dilihat pada Gambar 2.7.



Gambar 2.7 Kerangka Berpikir

BAB 5

PENUTUP

5.1. Kesimpulan

Berdasarkan hasil penelitian dan pembahasan, maka dapat ditarik kesimpulan sebagai berikut.

1. Cara kerja algoritma RNN-LSTM dan CNN untuk klasifikasi multi-label hampir sama untuk melakukan klasifikasi multi-kelas, perbedaan yang mendasar dalam klasifikasi multi-label adalah penggunaan fungsi aktivasi sigmoid pada layer *output*, fungsi *sigmoid* menghasilkan hasil probabilitas yang independen, dan tidak dibatasi menjadi satu bagian kelas, oleh karena itu fungsi aktivasi *sigmoid* melihat setiap nilai *output* mentah secara terpisah.
CNN adalah salah satu JST *feed-forward* dalam *deep learning* dimana koneksi antar neuron tidak diulang. CNN umumnya digunakan dalam *computer vision*, namun CNN menunjukkan hasil yang menjanjikan ketika diterapkan pada berbagai tugas NLP juga. Sedangkan RNN-LSTM adalah salah satu JST dimana koneksi antar neuron membentuk grafik yang diarahkan sepanjang *sequence* dalam pelatihan. RNN-LSTM pada dasarnya adalah urutan blok JST yang terhubung satu dengan yang lain seperti rantai. Pada algoritma RNN-LSTM memungkinkan menyimpan informasi pelatihan dan menangkap data sekuensial yang menjadikan pendekatan yang lebih 'alami' ketika berhadapan dengan data tekstual karena teks memang telah berurutan secara alami. Kemudian untuk algoritma RNN-LSTM, melakukan penilaian sentiment lebih baik ketika klasifikasi ditentukan oleh ketergantungan semantik jarak jauh daripada beberapa frase kunci lokal. Sedangkan CNN lebih baik dalam mengekstraksi fitur lokal dalam posisi-invarian. Untuk tugas di mana deteksi fitur dalam teks lebih penting, misalnya, mencari istilah marah, kesedihan, penyalahgunaan, entitas yang ditentukan. CNN bekerja dengan baik sedangkan untuk tugas di mana pemodelan sekuensial lebih penting, RNN bekerja lebih baik seperti menterjemahkan kalimat (Yin et al., 2017).
2. Penerapan *Word2Vec* dalam pelatihan *deep learning* adalah untuk

menyesuaikan bobot guna mengurangi *loss function*. Dikarenakan *Word2Vec* mampu menangkap berbagai tingkat kesamaan antar kata, sehingga pola semantik dan sintaksis dapat direproduksi menggunakan aritmatika vektor. Namun *Word2Vec* tidak akan digunakan untuk klasifikasi dalam pelatihan, sebagai gantinya algoritma klasifikasi hanya mengambil bobot yang tersembunyi dalam *Word2Vec*, dan menggunakan bobot tersebut sebagai *word embedding*, dan kemudian algoritma klasifikasi yang melakukan sisanya. Oleh karena itu, *Word2Vec* mampu meningkatkan performa akurasi klasifikasi model.

3. Pada penelitian ini hasil akurasi dari klasifikasi analisis sentimen multi-label untuk ujaran kebencian dan umpatan pada Twitter Indonesia menunjukkan bahwa algoritma RNN-LSTM dengan menerapkan *Word2Vec* mendapatkan hasil akurasi sebesar 73.94 %, berarti algoritma RNN-LSTM menghasilkan akurasi yang lebih tinggi dibandingkan algoritma CNN dengan menerapkan *Word2Vec* yang mendapatkan hasil akurasi sebesar 71.19 %.

Berdasarkan hasil tersebut, klasifikasi multi-label pada dataset dapat di klasifikasikan dengan akurasi yang lebih baik.

5.2. Saran

Adapun saran dari penelitian ini adalah sebagai berikut.

1. Melakukan penelitian dengan menerapkan gabungan dari CNN-LSTM, penambahan algoritma *Attention based*, untuk melihat kemungkinan bagaimana tingkat kinerja dari klasifikasi analisis sentimen.
2. Perlu dilakukan *fine tuning* pada *hyperparameter* dalam pelatihan agar mendapatkan performa akurasi yang maksimal, dikarenakan *hyperparameter* yang digunakan dalam penelitian ini adalah *hyperparameter* penelitian terdahulu.
3. Perlu dilakukan pembobotan yang seimbang untuk penelitian selanjutnya. Proses penyeimbangan *dataset* dapat dilakukan dengan mengumpulkan data baru dan melakukan proses anotasi dengan fokus pada data yang berlabel kecil. Atau menggunakan augmentasi data seperti penelitian Kobayashi, (2018) dan Wang & Yang, (2015).

DAFTAR PUSTAKA

Adriani, M., Asian, J., Nazief, B., Tahaghoghi, S. M. M., & Williams, H. E. (2007). Stemming Indonesian. *ACM Transactions on Asian Language Information Processing*, *6*(4), 1–33. https://doi.org/10.1145/1316457.1316459

Aggarwal, C. C. (2015). *Data Mining*. Springer International Publishing. https://doi.org/10.1007/978-3-319-14142-8

Aggarwal, C. C. (2018). *Neural Networks and Deep Learning*. Springer International Publishing. https://doi.org/10.1007/978-3-319-94463-0

Aghaebrahimian, A., & Cieliebak, M. (2019). Hyperparameter tuning for deep learning in natural language processing. *CEUR Workshop Proceedings*, *2458*. https://doi.org/10.21256/zhaw-18993

Alfina, I., Mulia, R., Fanany, M. I., & Ekanata, Y. (2017). Hate speech detection in the Indonesian language: A dataset and preliminary study. *2017 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, *2018-Janua*(October), 233–238. https://doi.org/10.1109/ICACSIS.2017.8355039

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). *A Brief Survey of Text Mining: Classification, Clustering and Extraction Techniques*. http://arxiv.org/abs/1707.02919

Baly, R., Badaro, G., El-Khoury, G., Moukalled, R., Aoun, R., Hajj, H., El-Hajj, W., Habash, N., & Shaban, K. (2017). A Characterization Study of Arabic Twitter Data with a Benchmarking for State-of-the-Art Opinion Mining Models. *Proceedings of the Third Arabic Natural Language Processing Workshop*, 110–118. https://doi.org/10.18653/v1/W17-1314

Baytas, I. M., Xiao, C., Zhang, X., Wang, F., Jain, A. K., & Zhou, J. (2017). Patient Subtyping via Time-Aware LSTM Networks. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '17*, 65–74. https://doi.org/10.1145/3097983.3097997

Bernard, J. (2016). Python Data Analysis with pandas. In *Python Recipes Handbook* (pp. 37–48). Apress. https://doi.org/10.1007/978-1-4842-0241-8_5

Brownlee, J. (2017). *Deep Learning for Natural Language Processing*. https://github.com/abhiramkadali/Machine_Learning-_Resources

Chawla, N. V. (2009). Data Mining for Imbalanced Datasets: An Overview. In *Data Mining and Knowledge Discovery Handbook* (pp. 875–886). Springer US. https://doi.org/10.1007/978-0-387-09823-4_45

Claesen, M., & De Moor, B. (2015). *Hyperparameter Search in Machine Learning*. http://arxiv.org/abs/1502.02127

Clinten, B. (2019). Pengguna Aktif Harian Twitter Indonesia Diklaim Terbanyak.

Kompas.Com. https://tekno.kompas.com/read/2019/10/30/16062477/pengguna-aktif-harian-twitter-indonesia-diklaim-terbanyak.

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). *Automated Hate Speech Detection and the Problem of Offensive Language*. http://arxiv.org/abs/1703.04009

El-Amir, H., & Hamdy, M. (2020a). *Deep Learning Pipeline*. Apress. https://doi.org/10.1007/978-1-4842-5349-6

El-Amir, H., & Hamdy, M. (2020b). *Deep Learning Pipeline*. Apress. https://doi.org/10.1007/978-1-4842-5349-6

ElKafrawy, P., Mausad, A., & Esmail, H. (2015). Experimental Comparison of Methods for Multi-label Classification in different Application Domains. *International Journal of Computer Applications*, *114*(19), 1–9. https://doi.org/10.5120/20083-1666

Eugine Kang. (2017). *Long Short-Term Memory (LSTM): Concept*. https://medium.com/@kangeugine/long-short-term-memory-lstm-concept-cb3283934359

Gambäck, B., & Sikdar, U. K. (2017). Using Convolutional Neural Networks to Classify Hate-Speech. *Proceedings of the First Workshop on Abusive Language Online*, *7491*, 85–90. https://doi.org/10.18653/v1/W17-3013

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org/

Grinberg, M. (2014). *Flask Web Development: Developing Web Applications with Python*. O'Reilly Media. https://coddyschool.com/upload/Flask_Web_Development_Developing.pdf

Hakiem, M., Fauzi, M. A., & Indriati. (2019). Klasifikasi Ujaran Kebencian pada Twitter Menggunakan Metode Naïve Bayes Berbasis N-Gram Dengan Seleksi Fitur Information Gain. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *3*, 2443–2451. http://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/4682

Haykin, S. (2009). *Neural Networks and Learning Machines (3rd Edition)* (3rd ed.). Pearson Education, Inc. http://dai.fmph.uniba.sk/courses/NN/haykin.neural-networks.3ed.2009.pdf

Heikal, M., Torki, M., & El-Makky, N. (2018). Sentiment Analysis of Arabic Tweets using Deep Learning. *Procedia Computer Science*, *142*, 114–122. https://doi.org/10.1016/j.procs.2018.10.466

Ibrahim, M., Torki, M., & El-Makky, N. (2018). Imbalanced Toxic Comments Classification Using Data Augmentation and Deep Learning. *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*,

875–878. https://doi.org/10.1109/ICMLA.2018.0014

Ibrohim, M. O., & Budi, I. (2018). A Dataset and Preliminaries Study for Abusive Language Detection in Indonesian Social Media. *Procedia Computer Science*, *135*, 222–229. https://doi.org/10.1016/j.procs.2018.08.169

Ibrohim, M. O., & Budi, I. (2019). Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter. *Proceedings of the Third Workshop on Abusive Language Online*, 46–57. https://doi.org/10.18653/v1/W19-3506

Jang, B., Kim, I., & Kim, J. W. (2019). Word2vec convolutional neural networks for classification of news articles and tweets. *PLOS ONE*, *14*(8), e0220976. https://doi.org/10.1371/journal.pone.0220976

Johnson, N. F., Leahy, R., Restrepo, N. J., Velasquez, N., Zheng, M., Manrique, P., Devkota, P., & Wuchty, S. (2019). Hidden resilience and adaptive dynamics of the global online hate ecology. *Nature*, *573*(7773), 261–265. https://doi.org/10.1038/s41586-019-1494-7

Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1746–1751. https://doi.org/10.3115/v1/D14-1181

Kingma, D. P., & Ba, J. (2014a). *Adam: A Method for Stochastic Optimization*.

Kingma, D. P., & Ba, J. (2014b). *Adam: A Method for Stochastic Optimization*. http://arxiv.org/abs/1412.6980

Kobayashi, S. (2018). Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 452–457. https://doi.org/10.18653/v1/N18-2072

Komnas HAM. (2015). *Buku Saku Penanganan Ujaran Kebencian (Hate Speech)*. Komisi Nasional Hak Asasi Manusia.

Li, S., & Xu, J. (2018). *A Recurrent Neural Network Language Model Based on Word Embedding* (pp. 368–377). Springer, Cham. https://doi.org/10.1007/978-3-030-01298-4_30

Lilleberg, J., Zhu, Y., & Zhang, Y. (2015). Support vector machines and Word2vec for text classification with semantic features. *Proceedings of 2015 IEEE 14th International Conference on Cognitive Informatics and Cognitive Computing, ICCI*CC 2015*, 136–140. https://doi.org/10.1109/ICCI-CC.2015.7259377

Liu, B. (2012). Sentiment Analysis: A Fascinating Problem. In *Sentiment Analysis and Opinion Mining*.

Mikolov, T., Joulin, A., Chopra, S., Mathieu, M., & Ranzato, M. (2014). *Learning Longer Memory in Recurrent Neural Networks*. 1–9.

http://arxiv.org/abs/1412.7753

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). *Distributed Representations of Words and Phrases and their Compositionality*. 1–9. http://arxiv.org/abs/1310.4546

Moolayil, J. (2019a). *Learn Keras for Deep Neural Networks*. Apress. https://doi.org/10.1007/978-1-4842-4240-7

Moolayil, J. (2019b). *Learn Keras for Deep Neural Networks*. Apress. https://doi.org/10.1007/978-1-4842-4240-7

Patodkar, V. N., & Sheikh, I. . (2016). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. *IJARCCE*, *5*(12), 320–322. https://doi.org/10.17148/IJARCCE.2016.51274

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Müller, A., Nothman, J., Louppe, G., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2012). *Scikit-learn: Machine Learning in Python*. http://arxiv.org/abs/1201.0490

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. https://doi.org/10.3115/v1/D14-1162

Pournaki, A., Gaisbauer, F., Banisch, S., & Olbrich, E. (2020). *The twitter explorer: a framework for observing Twitter through interactive networks*. http://arxiv.org/abs/2003.03599

Prechelt, L. (2012). *Early Stopping — But When?* (pp. 53–67). https://doi.org/10.1007/978-3-642-35289-8_5

Pressel, D., Ray Choudhury, S., Lester, B., Zhao, Y., & Barta, M. (2018a). Baseline: A Library for Rapid Modeling, Experimentation and Development of Deep Learning Algorithms targeting NLP. *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, 34–40. https://doi.org/10.18653/v1/W18-2506

Pressel, D., Ray Choudhury, S., Lester, B., Zhao, Y., & Barta, M. (2018b). Baseline: A Library for Rapid Modeling, Experimentation and Development of Deep Learning Algorithms targeting NLP. *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, 34–40. https://doi.org/10.18653/v1/W18-2506

Qiu, W., Chen, M., Li, L., & Si, L. (2018). NLP_HZ at SemEval-2018 Task 9: a Nearest Neighbor Approach. *Proceedings of The 12th International Workshop on Semantic Evaluation*, 909–913. https://doi.org/10.18653/v1/S18-1148

Rahmawati, D., & Khodra, M. L. (2016). *Word2vec Semantic Representation in Multilabel Classification for Indonesian News Article*. 0–5.

Sechidis, K., Tsoumakas, G., & Vlahavas, I. (2011). *On the Stratification of Multi-label Data* (pp. 145–158). https://doi.org/10.1007/978-3-642-23808-6_10

Seo, S., Kim, C., Kim, H., Mo, K., & Kang, P. (2020). Comparative Study of Deep Learning-Based Sentiment Classification. *IEEE Access*, *8*, 6861–6875. https://doi.org/10.1109/ACCESS.2019.2963426

Sharami, J. P. R., Sarabestani, P. A., & Mirroshandel, S. A. (2020). *DeepSentiPers: Novel Deep Learning Models Trained Over Proposed Augmented Persian Sentiment Corpus*. http://arxiv.org/abs/2004.05328

Sorower, M. (2010). A literature survey on algorithms for multi-label learning. *Oregon State University, Corvallis*, *March*, 1–25. http://people.oregonstate.edu/~sorowerm/pdf/Qual-Multilabel-Shahed-CompleteVersion.pdf

Statista. (2019). *Number of monthly active Twitter users worldwide from 1st quarter 2010 to 1st quarter 2019 (in millions)*. https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/

Twitter, I. (2020, February). *Twitter Announces Fourth Quarter and Fiscal Year 2019 Results*. https://s22.q4cdn.com/826641620/files/doc_financials/2019/q4/Q4-2019-Earnings-Press-Release.pdf

van Aken, B., Risch, J., Krestel, R., & Löser, A. (2018). *Challenges for Toxic Comment Classification: An In-Depth Error Analysis*. http://arxiv.org/abs/1809.07572

van der Walt, S., Colbert, S. C., & Varoquaux, G. (2011a). The NumPy Array: A Structure for Efficient Numerical Computation. *Computing in Science & Engineering*, *13*(2), 22–30. https://doi.org/10.1109/MCSE.2011.37

van der Walt, S., Colbert, S. C., & Varoquaux, G. (2011b). The NumPy Array: A Structure for Efficient Numerical Computation. *Computing in Science & Engineering*, *13*(2), 22–30. https://doi.org/10.1109/MCSE.2011.37

Wang, W. Y., & Yang, D. (2015). That's So Annoying!!!: A Lexical and Frame-Semantic Embedding Based Data Augmentation Approach to Automatic Categorization of Annoying Behaviors using #petpeeve Tweets. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2557–2563. https://doi.org/10.18653/v1/D15-1306

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., & Recht, B. (2017). *The Marginal Value of Adaptive Gradient Methods in Machine Learning*. http://arxiv.org/abs/1705.08292

Xing, C., Wang, D., Zhang, X., & Liu, C. (2014). Document classification with distributions of word vectors. *2014 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2014*.

https://doi.org/10.1109/APSIPA.2014.7041633

Yaseen, A. F. (2018). A Survey on the Layers of Convolutional Neural Networks. *International Journal of Computer Science and Mobile Computing*, *7*(12), 191–196.

Yin, W., Kann, K., Yu, M., & Schütze, H. (2017). *Comparative Study of CNN and RNN for Natural Language Processing*. http://arxiv.org/abs/1702.01923

Zhang, M.-L., & Zhou, Z.-H. (2014). A Review on Multi-Label Learning Algorithms. *IEEE Transactions on Knowledge and Data Engineering*, *26*(8), 1819–1837. https://doi.org/10.1109/TKDE.2013.39

Zhang, Y., & Wallace, B. (2015). *A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification*. http://arxiv.org/abs/1510.03820

Zhou, Z., Zhang, X., & Sanderson, M. (2014). *Sentiment Analysis on Twitter through Topic-Based Lexicon Expansion* (pp. 98–109). https://doi.org/10.1007/978-3-319-08608-8_9