Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – KS 141501

ANALISIS SENTIMEN TEKS BAHASA INDONESIA PADA MEDIA SOSIAL MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (STUDI KASUS : E-COMMERCE)

SENTIMENT ANALYSIS INDONESIAN LANGUAGE IN SOCIAL MEDIA WITH CONVOLUTIONAL NEURAL NETWORK (CASE STUDY : E-COMMERCE)

ALDEN DELFIAN WATTIMENA
NRP 05211440007007

Dosen Pembimbing :
Renny Pradina K., S.T, M.T., SCJP

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – KS 141501

ANALISIS SENTIMEN TEKS BAHASA INDONESIA PADA MEDIA SOSIAL MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (STUDI KASUS : E-COMMERCE)

ALDEN DELFIAN WATTIMENA
NRP 05211440007007

Dosen Pembimbing :
Renny Pradina K., S.T, M.T., SCJP

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – KS 141501

SENTIMENT ANALYSIS INDONESIAN LANGUAGE IN SOCIAL MEDIA WITH CONVOLUTIONAL NEURAL NETWORK (CASE STUDY : E-COMMERCE)

ALDEN DELFIAN WATTIMENA
NRP 05211440007007

SUPERVISOR :
Renny Pradina K., S.T, M.T., SCJP

DEPARTMENT OF INFORMATION SYSTEMS
Faculty of Information Technology and Communication
Institut Teknologi Sepuluh Nopember
Surabaya 2018

LEMBAR PENGESAHAN

ANALISIS SENTIMEN TEKS BAHASA INDONESIA PADA MEDIA SOSIAL MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (STUDI KASUS : E-COMMERCE)

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:

ALDEN DELFIAN WATTIMENA
NRP. 05211440007007

Surabaya, 10 Juli 2018

KEPALA
DEPARTEMEN SISTEM INFORMASI

Dr. Ir. Aris Tjahyanto. M.Kom
NIP. 19650310 199102 1 001

LEMBAR PERSETUJUAN

ANALISIS SENTIMEN TEKS BAHASA INDONESIA PADA MEDIA SOSIAL MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (STUDI KASUS : E-COMMERCE)

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Oleh :

ALDEN DELFIAN WATTIMENA
NRP. 05211440007007

Disetujui Tim Penguji :Tanggal Ujian : 10 Juli 2018
Periode Wisuda : September 2018

Renny Pradina K., S.T, M.T., SCJP
(Pembimbing I)

Faizal Johan Atletiko, S.Kom., MT
(Penguji I)

Radityo Prasetianto Wibowo, S.Kom., M.Kom
(Penguji II)

ANALISIS SENTIMEN TEKS BAHASA INDONESIA PADA MEDIA SOSIAL MENGGUNAKAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK (STUDI KASUS : E-COMMERCE)

Nama Mahasiswa : Alden Delfian Wattimena
NRP : 05211440007007
Jurusan : Sistem Informasi FTIK-ITS
Pembimbing 1 : Renny Pradina K., S.T, M.T., SCJP

ABSTRAK

Penggunaan internet di Indonesia sebagian besar digunakan untuk mendapatkan pembaruan informasi khususnya dari media sosial. Media sosial saat ini banyak diminati oleh setiap kalangan, yang memungkinkan penggunanya dapat mengirimkan pesan secara bebas. E-commerce dapat menggunakan informasi yang didapat melalui media sosial untuk mengidentifikasi pola perilaku pelanggan melalui analisis sentimen. Mendeteksi sentimen terhadap suatu layanan dan mengidentifikasi indikator yang dapat mendorong inovasi untuk pertumbuhan e-commerce. Namun, analisis sentimen pada praktiknya dirasa memiliki beberapa tantangan antara lain, keberagaman bahasa yang digunakan oleh masyarakat Indonesia, yang mengakibatkan masalah sekaligus tantangan terhadap proses klasifikasi tulisan atau opini yang berada di media sosial yang menggunakan bahasa Indonesia.

Pada penelitian ini akan digunakan algoritma Convolutional Neural Network (CNN) untuk membangun sebuah model untuk mengklasifikasikan teks yang dapat melakukan analisis sentimen untuk menentukan pandangan publik mengenai layanan e-commerce melalui media sosial. Sehingga mampu memberikan rekomendasi terbaik terhadap pengambil keputusan. Pada penelitian ini digunakan data yang berhubungan dengan topik e-commerce. Data yang dikumpulkan berasal dari akun

media sosial twitter 5 e-commerce besar di Indonesia yaitu Bukalapak, Tokopedia, Blibli, Lazada dan Shopee.

Penggunaan algoritma Convolutional Neural Network digunakan dalam beberapa tahapan. Tahapan tersebut diawali dengan melihat hasil pengujian menggunakan output model word embedding yang paling baik. Pada penelitian didapatkan model Word2Vec dengan learning algorithm Skip-gram menghasilkan nilai lebih baik dibandingkan model yang lainnya. Selanjutnya dilakukan pengujian algoritma Convolutional Neural Network menggunakan dua model yaitu CNN-non-static dan CNN-static. Hasil evaluasi pengukuran pada 3 subtask juga menghasilkan nilai CNN-static lebih baik pada 2 subtask dibandingkan dengan model CNN-non-static. Namun, jika dilihat hasil evaluasi pengukuran memiliki nilai perbedaan yang sangat kecil.

Selain hal tersebut, dalam menentukan paramater dilakukan proses perubahan paramaterr pada ukuran filter size dan feature maps. Hasil evaluasi pengukuran mendapati rata-rata hasil evaluasi pengukuran menunjukkan proses meningkatkan filter size dari single menjadi multiple dapat meningkatkan hasil evaluasi pengukuran yang dilakukan pada model. Selain itu hasil dari tren perubahan nilai evaluasi pengukuran single filter region size menunjukkan tren yang semakin rendah seiring dengan pertambagan ukuran single filter region size. Proses pelatihan model pada 3 subtask berbeda didapatkan nilai filter region size paling baik adalah 1, 2, 3 dengan nilai rata-rata nilai feature maps adalah 100.

Kata kunci: Deep Learning, Analisis Sentimen, Media Sosial, Convolutional Neural Network, Klasifikasi Teks, Word Embedding, Word2Vec, FastText

ANALISIS SENTIMEN TEKS BAHASA INDONESIA PADA MEDIA SOSIAL MENGGUNAKAN *ALGORITMA CONVOLUTIONAL NEURAL NETWORK* (STUDI KASUS : E-COMMERCE)

Student Name : Alden Delfian Wattimena
NRP : 05211440007007
Department : Sistem Informasi FTIK-ITS
Supervisor 1 : Renny Pradina K., S.T, M.T., SCJP

ABSTRACT

Internet usage in Indonesia is mostly to get information from social media. Social media is currently in great demand by every group, allowing users to learn messages freely. E-commerce can use information obtained through social media to find the right patterns of analysis. Detects sentiments into services and indicators that can drive innovation for e-commerce growth. However, the analysis of sentiments in practice is felt to have several challenges, among others, the diversity of languages used by the people of Indonesia, which brings problems as well as to the process of understanding language in social media using the Indonesian language.

In this research, the Convolutional Neural Network (CNN) algorithm will be used to create a model for collecting text that can perform analysis to determine the public through e-commerce through social media. Only able to provide the best results for decision makers. In this study used data relating to the topic of e-commerce. The data collected comes from the social media accounts of twitter 5 major e-commerce in Indonesia namely Bukalapak, Tokopedia, Blibli, Lazada and Shopee.

Use of Convolutional Neural Network algorithm in several stages. The stages are started by looking at the test results using the best embedding word output model. In the research obtained by Word2Vec model with Skip-gram learning algorithm make

better value than other model. Then the convolutional neural network algorithm is tested using two models, namely CNN-non-static and CNN-static. The evaluation results in 3 subtasks also resulted in better CNN-static values in 2 subtasks with the CNN-non-static model. However, people will get very little results.

In addition to these things, in determining the parameters that exist on filter size and feature map parameters. The result of the average measurements of the highest average results used to measure the size of the filter from one to several can improve the measurement results performed on the model. In addition, the result of the study is the single-size filter area of increasing trend sizes along with the size of a single sieve size. The model training process in 3 different subtasks obtained the average filter size of the best area is 1, 2, 3 with the average value of the average feature map is 100.

Keywords: Sentiment Analysis, Politic, Social Media, Deep Learning, Convolutional Neural Network, Word Embedding, Word2Vec

KATA PENGANTAR

Puji dan syukur penulis tuturkan ke hadirat Allah SWT, Tuhan Semesta Alam yang telah memberikan karunia dan hidayah-Nya kepada penulis sehingga penulis mendapatkan kelancaran dalam menyelesaikan tugas akhir dengan judul:

ANALISIS SENTIMEN TEKS BAHASA INDONESIA PADA MEDIA SOSIAL MENGGUNAKAN ALGORITMA *CONVOLUTIONAL NEURAL NETWORK* (STUDI KASUS : E-COMMERCE)

Terima kasih penulis sampaikan kepada pihak-pihak yang telah mendukung, memberikan saran, motivasi, semangat, dan bantuan baik berupa material maupun moril demi tercapainya tujuan pembuatan tugas akhir ini. Tugas akhir ini tidak akan pernah terwujud tanpa bantuan dan dukungan dari berbagai pihak yang sudah meluangkan waktu, tenaga dan pikirannya. Secara khusus penulis akan menyampaikan ucapan terima kasih yang sebanyak-banyaknya kepada :

1. Bapak Niel dan Ibu Magdalena selaku kedua orang tua serta Felicia Hilary dan Godwin Halley selaku saudara kandung dari penulis yang tiada memberikan dukungan dan semangat.
2. Ibu Renny Pradina K., S.T., M.T., SCJP, selaku dosen pembimbing dan sebagai narasumber yang senantiasa meluangkan waktu, memberikan ilmu dan petunjuk, serta memotivasi untuk kelancaran tugas akhir.
3. Bapak Faizal Johan Atletiko, S.Kom., MT dan Bapak Radityo Prasentianto W., S.Kom., M.Kom., selaku dosen penguji yang telah memberikan saran dan kritik untuk perbaikan tugas akhir.
4. Seluruh dosen Jurusan Sistem Informasi ITS yang telah memberikan ilmu yang bermanfaat kepada penulis.
5. Rogue One yang telah memberikan informasi, motivasi dan canda tawa selama proses penulisan berlangsung.

6. Chrifan Oldry Purimahua selaku sahabat yang selalu mendukung selama pengerjaan Tugas Akhir ini
7. Warung Squad yang telah menemani penulis selama masa perkuliahan yang telah memberikan banyak pengalaman baru
8. Anjel, Lidia, Satria, Ira, Nia, Iqhbal, Nur, Agnes selaku teman-teman 6th Surabaya yang telah banyak membantu saat masa-masa SMA hingga perkuliahan
9. Rekan-rekan OSIRIS yang telah memberikan banyak kenangan manis dan pahit semasa kuliah
10. Berbagai pihak yang tidak bisa disebutkan satu persatu yang telah turut serta menyukseskan penulis dalam menyelesaikan tugas akhir.

Penyusunan laporan ini masih jauh dari kata sempurna sehingga penulis menerima adanya kritik maupun saran yang membangun untuk perbaikan di masa yang akan datang. Semoga buku tugas akhir ini dapat memberikan manfaat bagi pembaca.

Surabaya, 28 Juni 2018

Penulis

DAFTAR ISI

ABSTRAK vi
ABSTRACT viii
KATA PENGANTAR x
DAFTAR ISI xii
DAFTAR GAMBAR xvi
DAFTAR TABEL xviii
DAFTAR KODE xxii
BAB I PENDAHULUAN 1
1.1. Latar Belakang 1
1.2. Rumusan Masalah 4
1.3. Batasan Permasalahan 5
1.4. Tujuan Penelitian 5
1.5. Manfaat Penelitian 6
1.6. Relevansi 6
BAB II TINJAUAN PUSTAKA 7
2.1. Penelitian Sebelumnya 7
2.2. Landasan Teori 11
2.2.1. *Machine Learning* 11
2.2.2. *Deep Learning* 12
2.2.3. *Sentimen Analysis* 13
2.2.4. *Word Embedding* 13
2.2.5. *Convolutional Neural Network (CNN)* 19
2.2.6. *Text Mining* 21
2.2.7. *E-Commerce* 24
2.2.8. Media Sosial 25
2.2.9. *Crawling* 25
2.2.10. *Pytorch* 26

BAB III METODOLOGI PENELITIAN 27
3.1. Arsitektur 27
3.2. Tahapan Pelaksanaan Penelitian Tugas Akhir 28
3.2.1. Studi Literatur 29
3.2.2. Pengumpulan Data 30
3.2.3. Pra-Pemrosesan Data 31
3.2.4. Data Labeling 38
3.2.5. Pembuatan Model Word Embedding 40
3.2.6. Pembuatan Model CNN 40
3.2.7. Analisis dan Pengujian Hasil Model CNN 41
3.2.8. Dokumentasi 42
BAB IV PERANCANGAN 43
4.1. Akuisisi Data Media Sosial 43
4.2. Perancangan *Crawler* 43
4.3. Perancangan Read Data JSON 43
4.4. Desain *Database* 44
4.5. Desain Crawler 45
4.6. Perancangan Pengumpulan Kata 45
4.7. Perancangan Pra-Pemrosesan Data 45
4.7.1. Perancangan Penggabungan Dataset 46
4.7.2. Perancangan Penghapusan Data yang Terduplikasi 46
4.7.3. Perancangan Filtering Bahasa Indonesia 46
4.7.4. Perancangan Perubahan Emoticon menjadi Token 47
4.7.5. Perancangan Perubahan URL dan Mention menjadi Token 48

4.7.6. Perancangan Penghapusan Huruf Berulang 48
4.7.7. Perancangan Penghapusan Tanda Baca, Simbol dan Angka 48
4.7.8. Perancangan Penghapusan Data yang Tidak Mengandung Topik dalam Teks untuk Data CNN..... 49
4.8. Perancangan Pelabelan Data 49
4.9. Perancangan Pembuatan Model *Word Embedding* 50
4.9.1. Perancangan *Training* Word2Vec..... 50
4.9.2. Perancangan *Training* FastText 51
4.10. Perancangan *Convolutional Neural Network* 51
4.11. Perancangan Evaluasi Pengukuran 54
BAB V IMPLEMENTASI 57
5.1. Persiapan Implementasi 57
5.2. Pembuatan Filtering Bahasa 59
5.3. Pembuatan Crawler..... 60
5.4. Pra-Pemrosesan Dataset..... 63
5.4.1. Penggabungan Dataset 63
5.4.2. Penghapusan Kata yang Terduplikasi 64
5.4.3. Penghapusan Tanda Baca, Simbol dan Angka 65
5.4.4. Menghapus Huruf yang Berulang 66
5.4.5. Mengubah *Emoticon* menjadi Token String..... 66
5.4.6. Mengubah URL dan Mention menjadi Token String 68
5.5. Pembuatan Model Word2Vec..... 69
5.6. Pembuatan Model Convolutional Neural Network 73
BAB VI HASIL DAN PEMBAHASAN 89
6.1. Data *Crawling* 89

6.2. *Filtering* Bahasa 89
6.3. Hasil Pelabelan Data 90
6.4. Model *Word Embedding* 101
6.5. Hasil Pengujian Data 103
BAB VII KESIMPULAN DAN SARAN 149
7.1. Kesimpulan 149
7.2. Saran 151
DAFTAR PUSTAKA 152
BIODATA PENULIS 157
LAMPIRAN A 159
LAMPIRAN B 163
LAMPIRAN C 169

DAFTAR GAMBAR

Gambar 2.1 Arsitektur CBOW 15
Gambar 2.2 Arsitektur *Skip-gram* 16
Gambar 2.3 Arsitektur *Convolutional Neural Network* 20
Gambar 3.1 Arsitektur Model 27
Gambar 3.2 Tahapan Pelaksanaan Tugas Akhir 29
Gambar 3.3 Proses Crawling Twitter 31
Gambar 3.4 Proses Pra-Pemrosesan Data 32
Gambar 3.5. Proses Model Word Embedding 40
Gambar 3.6. Alur Proses Model CNN 41
Gambar 4.1 Query untuk Menghapus Data yang Terduplikasi ... 46
Gambar 5.1 Skema Basis Data Anotasi 69
Gambar 5.2 Desain Antar Muka Aplikasi Anotasi 70
Gambar 5.3 Bagian *Card* pada Aplikasi Anotasi 70
Gambar 6.1 Confusion Matrix antara pelabel 1 dan pelabel 2 98
Gambar 6.2 Confusion Matrix antara pelabel 1 dan pelabel 3 98
Gambar 6.3 Confusion Matrix antara pelabel 2 dan pelabel 3 99
Gambar 6.4 Jumlah Kata Terdeteksi dari Model *Word Embedding* 103
Gambar 6.5 Performa Model Terbaik Single Filter Subtask A . 105
Gambar 6.6 Perubahan *Feature Maps Multiple Filter Subtask A* 108
Gambar 6.7 Grafik Performa Model Terbaik *Multiple Region Size Subtask* A 109
Gambar 6.8 *Confusion Matrix* Model *Non-static* subtask A 110
Gambar 6.9 *Normalized Confusion Matrix* Model *Non-static* subtask A 111
Gambar 6.10 *Confusion Matrix* Model *Static subtask* A 113
Gambar 6.11 *Normalized Confusion Matrix* Model *Static subtask* A 113
Gambar 6.12 Performa Model Terbaik *Single Filter Subtask* B 116
Gambar 6.13 Grafik Performance Multiple Filter Subtask B 118
Gambar 6.14 Perubahan Feature Maps Multiple Filter Subtask B 119
Gambar 6.15. Grafik Performance Model Terbaik *Subtask* B .. 120

Gambar 6.16 *Confusion Matrix Model Non-static subtask B* ... 121
Gambar 6.17 *Normalized, Confusion Matrix Model Non-static subtask B* 122
Gambar 6.18 *Confession Matrix Model Static Subtask B* 124
Gambar 6.19 *Normalized, Confession Matrix Model Static Subtask B* 124
Gambar 6.20 Performa Model Terbaik *Single Filter Subtask* C (semakin kecil semakin baik) 128
Gambar 6.21 Perubahan Feature Maps pada Subtask C (semakin kecil semakin baik) 130
Gambar 6.22 Grafik Performa Model Terbaik Subtask C 131
Gambar 6.23 *Confusion Matrix Model Non-static subtask C* ... 132
Gambar 6.24 *Normalized, Confusion Matrix Model Non-static subtask C* 133
Gambar 6.25 *Confusion Matrix Model Static subtask C*..... 135
Gambar 6.26 *Normalized, Confusion Matrix Model Static subtask C*..... 136
Gambar 6.27 Pengaruh *single filter region size* CNN-*non-static* pada perubahan evaluasi pengukuran 139
Gambar 6.28 Pengaruh *single filter region size* CNN-*static* pada perubahan evaluasi pengukuran..... 140
Gambar 6.29 Pengaruh *feature maps* CNN-*non-static* pada perubahan evaluasi pengukuran..... 146
Gambar 6.30 Pengaruh *feature maps CNN-static* pada perubahan evaluasi pengukuran..... 146

DAFTAR TABEL

Tabel 2.1 Studi Sebelumnya 7
Tabel 3.1 Daftar Keyword Media Sosial Twitter 30
Tabel 3.2 Proses Pemilihan Teks yang Mengandung Topik 33
Tabel 3.3 Proses Mengubah menjadi Lowercase 34
Tabel 3.4 Proses Mengubah Emoticon menjadi String 35
Tabel 3.5 Proses Mengubah URL dan Mention menjadi Token String 36
Tabel 3.6 Proses Menghapus Tanda Baca, Simbol dan Angka ... 37
Tabel 3.7. Proses Tokenizing 37
Tabel 3.8 Proses Pelabelan Data 38
Tabel 4.1 Desain Database Crawler Post Twitter 44
Tabel 4.2 Daftar Mapping Emoticon 47
Tabel 4.3 Parameter yang digunakan dalam Model *Convolutinal neural Network* 52
Tabel 5.1 Spesifikasi Perangkat Keras Laptop 57
Tabel 5.2. Daftar Library 57
Tabel 6.1 Pembagian Data Berdasarkan Topik 90
Tabel 6.2 Penyeberan Data berdasarkan masing-masing pelabel 91
Tabel 6.3 Penyeberan Data Hasil Final Pelabelan 92
Tabel 6.4 Penyebaran Data Berdasarkan Topik pada Hasil Pelabelan Final 92
Tabel 6.5 Range Nilai Kappa berdasarkan Tingkat Kesepakatan 96
Tabel 6.6 Nilai Kappa Antar tiap Pelabelan 97
Tabel 6.7 Contoh label dari masing-masing pelabel 99
Tabel 6.8 Parameter Model Yon Kim 101
Tabel 6.9 Percobaan Algoritma *Word Embedding* pada Model *Non-Static* 102
Tabel 6.10 Percobaan Algoritma *Word Embedding* pada Model *Static* 102
Tabel 6.11 Perbandingan Kecepatan Penggunaan GPU dengan CPU 103
Tabel 6.12 Penyebaran Data pada *Subtask* A 104
Tabel 6.13 Percobaan *Single Filter Region Size Subtask A* Pada Model CNN-non-Static 104

Tabel 6.14 Percobaan *Single Filter Region Size Subtask A* Pada Model CNN-static 105
Tabel 6.15 Hasil Evaluasi Pengukuran *Multiple Filter Subtask A Non-Static* 107
Tabel 6.16 Hasil Evaluasi Pengukuran *Multiple Filter Subtask A Static* 107
Tabel 6.17 Pengukuran Evaluasi Model Terbaik dari *Subtask* A 108
Tabel 6.18 Perhitungan Recal per Label Model *Non-static Subtask* A 111
Tabel 6.19 Perbandingan Nilai Pengukuran Evaluasi *Single* dan *Multiple Filter Subtask A Non-static* 112
Tabel 6.20 Perhitungan Recal per Label Model *Static* Subtask A 114
Tabel 6.21 Perbandingan NiliaPengukuran Evaluasi Single dan Multiple Filter Subtask A Static 114
Tabel 6.22 Penyebaran Data pada Subtask B 115
Tabel 6.23 Percobaan *Single Filter Region Size Subtask B* Pada Model *CNN-non-Static* 115
Tabel 6.24 Percobaan *Single Filter Region Size Subtask* B Pada Model *CNN-static* 116
Tabel 6.25 Hasil Evaluasi Pengukuran *Multiple Filter Subtask B Non-Static* 117
Tabel 6.26 Hasil Evaluasi Pengukuran *Multiple Filter Subtask B Static* 118
Tabel 6.27 Pengukuran Evaluasi Model Terbaik dari Subtask B 120
Tabel 6.28 Perhitungan Recal per Label Model *Non-static Subtask* B 123
Tabel 6.29 Perbandingan Nilai Pengukuran Evaluasi *Single* dan *Multiple Filter Subtask B Non-static* 123
Tabel 6.30 Perhitungan Recal per Label Model Static Subtask B 125
Tabel 6.31 Perbandingan Nilai Pengukuran Evaluasi *Single* dan *Multiple Filter Subtask B Static* 126

Tabel 6.32 Penyebaran Data pada *Subtask* C.....126
Tabel 6.33 Percobaan *Single Filter Region Size Subtask* C Pada Model *CNN-non-static* 127
Tabel 6.34 Percobaan *Single Filter Region Size Subtask* C Pada Type Model *CNN-static* 127
Tabel 6.35 Hasil Evaluasi Pengukuran *Multiple Filter Subtask* C *Non-Static*.....129
Tabel 6.36 Hasil Evaluasi Pengukuran *Multiple Filter Subtask C Static* 129
Tabel 6.37 Pengukuran Evaluasi Model Terbaik dari Subtask C 131
Tabel 6.38 Perhitungan Recal per Label Model *Non-static Subtask C*.....134
Tabel 6.39 Perbandingan Nilai Pengukuran Evaluasi *Single* dan *Multiple Filter Subtask C Non-static*.....135
Tabel 6.40 Perhitungan Recal per Label *Model Static Subtask C* 137
Tabel 6.41 Perbandingan Nilai Pengukuran Evaluasi *Single* dan *Multiple Filter Subtask C Static* 138
Tabel 6.42 Model Terbaik Setiap Subtask Berdasarkan *Single Filter Size* 138
Tabel 6.43 Model Terbaik Setiap Subtask Berdasarkan *Multiple Filter Size* 140
Tabel 6.44 Datat Percobaan Uji Signifikansi Subtask B.....142
Tabel 6.45 Datat Percobaan Uji Signifikansi Subtask C.....144
Tabel 6.46 Perbandingan *Convolutional Neural Network* dengan Algoritma lain 147

Halaman ini sengaja dikosongkan

DAFTAR KODE

Kode 5.1 Proses Filtering Bahasa dengan Menggunakan Polyglot 59
Kode 5.2 Proses Membaca File Json dengan Library Ijson 60
Kode 5.3 Autentikasi Twitter 61
Kode 5.4 Pengambilan Data Twitter sesuai dengan Keyword 62
Kode 5.5 Menggabungkan File Twitter menjadi Satu File SQL . 63
Kode 5.6 Query Database untuk Menghapus Kata Duplikat 64
Kode 5.7 Proses Menghapus Tanda Baca, Simbol dan Angka ... 65
Kode 5.8 Proses Menghapus Huruf yang Berulang lebih dari 2 kali 66
Kode 5.9 Perubahan Emoticon Menjadi Token String 68
Kode 5.10 Perubahan URL dan Mention menjadi Token String 68
Kode 5.11 Inisialisasi Parameter Model Word2Vec 71
Kode 5.12 Training pada Word2Vec 72
Kode 5.13 *Method Tokenize Word* 73
Kode 5.14 Pembuatan Variasi Model CNN 74
Kode 5.15 Method Forward Pembelajaran Model CNN 75
Kode 5.16 Method untuk membaca model Word Embedding 77
Kode 5.17 Method untuk membaca dataset dan melakukan perubahan polarity label 78
Kode 5.18 Proses perulangan pembacaan data 79
Kode 5.19 Proses Inisialisasi Vektor Kata 80
Kode 5.20 Proses Pembagain Data Testing dan Data Training... 81
Kode 5.21 Proses Pembatan Vocabulary 82
Kode 5.22 Proses Pemanggilan Method Training Model CNN .. 83
Kode 5.23 Proses Training Model CNN 84
Kode 5.24 Proses Testing Model CNN 85
Kode 5.25 Proses Perhitungan Evaluasi Model 86
Kode 5.26 Evaluasi Pengukuran Setiap Epoch 87

Halaman ini sengaja dikosongkan

BAB I
PENDAHULUAN

Pada bab pendahuluan ini akan diuraikan proses identifikasi masalah penelitian yang meliputi latar belakang masalah, perumusan masalah, batasan masalah, tujuan tugas akhir, manfaat kegiatan tugas akhir, manfaat kegiatan tugas akhir dan relevansi terhadap pengerjaan tugas akhir. Berdasarkan uraian pada bab ini, harapannya gambaran umum permasalahan dan pemecahan masalah pada tugas akhir dapat dipahami.

1.1. Latar Belakang

Indonesia merupakan salah satu negara dengan angka penggunaan internet terbesar di dunia. Menurut hasil *survey* APJII (Asosiasi Penyelenggara Jasa Internet Indonesia) pada tahun 2017, penetrasi pengguna internet di Indonesia mencapai 143.26 juta [1]. Angka pengguna internet di Indonesia ini mengalami kenaikan dibandingkan dengan tahun 2016 yang mencapai 132.7 juta [1]. Angka ini menunjukkan penetrasi pengguna internet sebesar 54.68% dari total populasi penduduk Indonesia. Tingginya pengguna internet di Indonesia disebabkan oleh berbagai alasan yang sangat beragam, menurut hasil *survey* APJII (Asosiasi Penyelenggara Jasa Internet Indonesia) pada tahun 2016 pembaruan informasi menjadi alasan yang paling tinggi dalam mengakses internet. Untuk mendapatkan pembaruan informasi ini, pengguna internet Indonesia mengakses berbagai konten yang tersedia. Media sosial menjadi layanan kedua tertinggi sebesar 87.13% setelah *chatting* 89.35% ini menunjukkan bahwa tingkat penggunaan media sosial saat ini sangat berkembang seiring dengan jumlah pengguna yang terus meningkat [1]. Di Indonesia berbagai media sosial telah digunakan dan diakses oleh berbagai kalangan masyarakat, mulai dari Twitter, Facebook, dan Instagram yang merupakan media sosial yang paling populer di Indonesia. Dari informasi tersebut dapat disimpulkan bahwa masyarakat Indonesia banyak menggunakan media sosial untuk

saling berbagi informasi dan pembaruan informasi. Sehingga, media sosial saat ini memiliki informasi yang melimpah dan dapat dijadikan sebagai sumber informasi yang berharga.

Media sosial saat ini banyak diminati oleh setiap kalangan, yang memungkinkan penggunanya dapat mengirimkan pesan secara bebas. Sehingga membuat media sosial mempunyai peranan yang sangat penting dalam memberikan informasi yang sangat cepat. Informasi yang dihasilkan juga sangat beragam seperti opini, komentar, kritik baik yang bersifat positif maupun negatif. Banyaknya jumlah pengguna media sosial di Indonesia tentu saja memunculkan berbagai macam bentuk status maupun komentar terhadap topik tertentu tanpa adanya batasan waktu dalam berbagi informasi pada media sosial. Dengan beragamnya bentuk status maupun komentar terhadap topik tertentu, maka hal ini dapat dijadikan sebagai bahan untuk melakukan pemantauan terhadap media sosial di Indonesia. Informasi yang tersebar di dalamnya dapat dijadikan sebagai bahan untuk mencari *insight* agar dapat dijadikan sebagai sumber informasi yang berharga. Salah satu topik yang dapat memanfaatkan pencarian informasi pada sosial media ini adalah *e-commerce*.

E-commerce dapat menggunakan informasi yang didapat untuk mengidentifikasi pola perilaku pelanggan. Mendeteksi sentimen terhadap suatu layanan dan mengidentifikasi indikator yang dapat mendorong inovasi untuk pertumbuhan *e-commerce*. Sentimen terhadap layanan dapat dilihat salah satunya dari tingkat kepuasan pelanggan, di antaranya adalah mutu pelayanan yang diberikan. Oleh karena itu, para pelaku usaha *e-commerce* harus memperhatikan kepuasan pelanggan dengan cara memperhatikan faktor tersebut, dengan demikian para konsumen akan sukarela kembali karena puas dengan pelayanan yang telah diberikan. Keingintahuan pelaku usaha *e-commerce* terhadap sentimen layanan ini juga dipacu dengan tingkat persaingan yang semakin tinggi di antara para pelaku usaha *e-commerce*. Namun, untuk dapat mengetahui respon publik tersebut membutuhkan biaya

dan usaha yang tidak mudah. Sehingga perusahaan ataupun perseorangan harus mencari jawaban atas keingintahuan tersebut di berbagai tempat. Seiring dengan berjalannya waktu semakin banyak pelanggan *e-commerce* yang menggunakan media sosial dan mulai membagikan respon mereka terhadap layanan yang diberikan oleh *e-commerce* di media sosial yang mereka miliki. Banyak opini-opini yang dibagikan oleh orang-orang melalui media sosial. Maka, yang perlu dilakukan adalah usaha untuk mengumpulkan serta mengolah seluruh opini tersebut menjadi suatu informasi yang dapat menjawab pertanyaan-pertanyaan tentang sentimen layanan yang diberikan oleh para pelaku usaha *e-commerce*. Salah satunya *e-commerce* dapat menggunakan analisis sentimen sebagai teknik untuk mengetahui respon opini publik di media sosial.

Analisis Sentimen sangat berguna dalam pemantauan media sosial karena dapat memungkinkan untuk mendapatkan *insight* dan gambaran umum opini publik yang lebih luas dibalik topik tertentu. Sentimen cukup mudah untuk dimengerti. Itu merupakan wujud perasaan atau emosi, sikap atau opini. Di media sosial, sentimen sebuah postingan bisa dilihat dengan nada atau emosi yang disampaikan terhadap sebuah brand. Namun, analisis sentimen pada praktiknya dirasa memiliki beberapa tantangan antara lain, keberagaman bahasa yang digunakan oleh masyarakat Indonesia, yang mengakibatkan masalah sekaligus tantangan terhadap proses klasifikasi tulisan atau opini yang berada di media sosial yang menggunakan bahasa Indonesia. Analisis sentimen jika digunakan tepat pada sasaran, maka dapat digunakan terhadap bisnis dan lebih proaktif dalam melihat perubahan dinamika pasar dan pelanggan. Oleh karena itu, diperlukan alat yang dapat membantu proses analisis sentimen, sehingga analisisnya dapat dilakukan dapat lebih cepat dan efisien.

Dalam pengolahan analisis sentimen dapat menggunakan metode *Natural Languange Processing (NLP)* yang sebagian besar metode pembelajaran dilakukan dengan metode *Deep Learning* yang telah melibatkan representasi *word*

vector melalui model *neural language* [2] dan komposisi performa berdasarkan pembelajaran *word vector* untuk klasifikasi [3]. *Convolutional Neural Network (CNN)* merupakan merupakan salah satu algoritma pengembangan dari deep learning yang dapat digunakan untuk mengatasi permasalahan pada *Natural Languange Processing (NLP)*.

Penggunaan Algoritma *Convolutional Neural Network (CNN)* dipilih karena dianggap lebih baik dalam mengatasi permasalahan *Big Data* dan *Machine Learning* sehingga dapat dikembangkan untuk membangun sebuah model untuk mengklasifikasikan sebuah kata, kalimat atau paragraph. Dengan adanya tugas akhir ini diharapkan dapat melakukan analisis sentimen untuk menentukan pandangan publik mengenai layanan atau produk *e-commerce* melalui media sosial menggunakan Algoritma *Convolutional Neural Network (CNN).* Sehingga mampu memberikan rekomendasi terbaik terhadap pengambil keputusan.

1.2. Rumusan Masalah

Berdasarkan uraian latar belakang, rumusan masalah dari penelitian ini adalah :

1. Bagaimana mendapatkan data posting atau komentar dari Twitter dengan topik yang telah ditentukan ?
2. Bagaimana metode untuk melakukan pra-pemrosesan terhadap kumpulan data teks yang diperoleh ?
3. Bagaimana metode untuk membuat model *word2vec* untuk proses *word embeddings* pada kumpulan data teks ?
4. Bagaimana melakukan pra-pemrosesan data, pembuatan dan pengujian model *Convolutional Neural Network* ?
5. Bagaimana cara untuk mengukur dan mengevaluasi kinerja dari model *Convolutional Neural Networks* ?

1.3. Batasan Permasalahan

Berdasarkan permasalahan yang disebutkan di atas, maka batasan masalah pada penelitian tugas akhir ini adalah sebagai berikut:

1. Penelitian ini menggunakan data yang bersumber dari media sosial Twitter yang menggunakan Bahasa Indonesia
2. Dataset yang digunakan pada algoritma *Convolutional Neural Network* merupakan dataset yang berlabel yang merujuk pada topik *E-commerce*
3. Proses tagging akan membagi dataset menjadi 3 subtask yaitu :
    - Subtask A : Sentiment positif dan negatif
    - Subtask B : Sentimen positif, negatif dan netral
    - Subtask C : Sentimen sangat positif, positif, netral, negatif, sangat negatif

1.4. Tujuan Penelitian

Tujuan dari pengerjaan penelitian tugas akhir ini adalah sebagai berikut:

1. Menerapkan metode crawling untuk mendapatkan data yang diinginkan dari media sosial Twitter.
2. Menerapkan metode pra-pemrosesan data terhadap data yang didapatkan dari media sosial agar diperoleh data yang layak untuk diproses.
3. Membuat model *Convolutional Neural Networks* untuk klasifikasi teks.
4. Membuat model analisis sentimen terhadap postingan dan komentar yang ada pada media sosial dengan menggunakan algoritma *Convolutional Neural Network (CNN).*

5. Menampilkan hasil analisa sentimen dari percobaan menggunakan algoritma *Convolutional Neural Network (CNN)* dari data hasil crawler.

1.5. Manfaat Penelitian

Manfaat yang diharapkan dapat diperoleh dari tugas akhir ini adalah:

1. Bagi penulis, Mengetahui dan memahami metode *deep learning* pada NLP (*Natural Language Processing*). Serta untuk mengetahui analisis sentimen dari sekumpulan dataset posting dan komentar sosial media menggunakan algoritma *Convolutional Neural Network (CNN)* serta mengetahui nilai akurasinya.
2. Bagi masyarakat, sebagai bentuk penelitian awal yang memungkinkan untuk terdapat penelitian-penelitian selanjutnya dan dapat dikembangkan kedalam beberapa hal seperti bisnis maupun rancang bangun aplikasi yang memanfaatkan algoritma *Convolutional Neural Network (CNN)* ini.

1.6. Relevansi

Relevansi tugas akhir ini terhadap laboratorium Akuisisi Data dan Diseminasi Informasi (ADDI) adalah karena tugas akhir ini berkaitan dengan penerapan mata kuliah bidang keilmuan laboratorium ADDI. Mata kuliah tersebut antara lain Sistem Cerdas, Sistem Pendukung Keputusan, dan Penggalian Data dan Analitika Bisnis.

BAB II
TINJAUAN PUSTAKA

Pada bab ini akan membahas penelitian sebelumnya yang berhubungan dengan tugas akhir dan teori-teori yang berkaitan dengan permasalahan tugas akhir.

2.1. Penelitian Sebelumnya

Tabel 1 menampilkan daftar penelitian sebelumnya yang mendasari tugas akhir ini.

Tabel 2.1 Studi Sebelumnya

1. Convolutional Neural Networks for Sentence Classification
Penulis/Tahun/Sumber : Y. Kim;2014 [4]
Metode : - Convolutional Neural Network
Kesimpulan : Dalam penelitian ini telah dilakukan serangkaian eksperimen menggunakan *Convolutional Neural Network (CNN)* yang dibangun di atas Word2Vec. Penggunaan berbagai variasi tipe CNN dan algoritma pembanding lainya juga dilakukan dalam penelitian ini guna mendapatkan hasil dengan akurasi terbaik untuk setiap tipe data yang digunakan. Meskipun hasilnya sedikit *tuning* dari *hyperparameters,* CNN sederhana dengan satu lapisan konvolusi menghasilkan hasil yang sangat baik. Hasil ini menambah bukti bahwa *pre-training* dari vektor kata merupakan unsur yang penting dalam penggunaan *deep learning* untuk *Natural Languange Processing (NLP).*

2. A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification
Penulis/Tahun/Sumber : Zhang, Y., & Wallace, B.;2015 [5]
Metode : - Convolutional Neural Network
Kesimpulan : Dalam penelitian ini menjelaskan tentang kinerja yang pada dataset berbeda yang dicapai oleh model ada beberapa hal yang didapat antara lain : - Pada penelitian ini menjelaskan bahwa pilihan representasi vektor seperti Word2Vec dan GloVe memiliki dampak pada kinerja. Namun, representasi yang berbeda akan berkinerja lebih baik untuk tugas yang berbeda. - Ukuran *filter region*, dapat memiliki efek yang besar terhadap kinerja. - Jumlah dari *feature maps* juga dapat memainkan peran penting dalam kinerja dan meningkatkan jumlah *feature maps* akan meningkatkan waktu pelatihan model sedangkan regularisasi / normalisasi merupakan parameter yang memberikan pengaruh terkecil. - Pertimbangan fungsi aktivasi yang berbeda juga mungkin dapat mempengaruhi hasil dari model, ReLU dan tanh menjadi model yang paling baik untuk dapat dipilih. - Saat melakukan penilaian kinerja model sangat penting untuk dapat mempertimbangkan *variance* dan *range*.

Oleh karena itu, replikasi prosedur *cross-fold validation* harus dilakukan.
3. BB_twtr at SemEval-2017 Task 4 : Twitter Sentiment Analysis with CNNs and LSTMs
Penulis/Tahun/Sumber : C. Mathieu;2017 [6]
Metode : - Convolutional Neural Network - Long Short-Term Memory
Kesimpulan : Penelitian ini menggunakan 10 model berbeda yaitu 10 model CNN dan 10 model LSTM. Algoritma unsupervised yang digunakan hanya Word2Vec dan FastText, dikarenakan algoritma GloVe memiliki nilai yang terlalu kecil saat proses pemodelan. Hasil yang didapat pada penelitian membuktikan bahwa CNN dan LSTM baik pada proses *ensemble* . Namun, dirasa proses tersebut akan lebih baik lagi jika dilakukan proses penggabungan secara lebih organik.
4. Efficient Estimation of Word Representations in Vector Space
Penulis/Tahun/Sumber : T. Mikolov, G. Corrado, K. Chen, dan J. Dean;2013 [7]
Metode : - Skip Gram - Contiunous Bags of Word

Kesimpulan :

Penelitian ini berhasil menemukan cara untuk melakukan representasi vektor dari sebuah kata dengan waktu yang relatif cepat dan dengan dataset yang cukup besar. Kemudian, vektor kata tidak hanya ditemukan kesamaan sintaks saja namun juga kesamaan semantic dari kata tersebut. Penelitian ini juga membandingkan akurasi hasilnya dengan teknik neural networks yang mana memiliki hasil yang lebih baik

5. Distributed Representations of Words and Phrases and their Compositionality

Penulis/Tahun/Sumber :

T. Mikolov, K. Chen, G. Corrado, dan J. Dean;2013 [2]

Metode :

- Skip Gram
- Contiunous Bags of Word
- Negative Sampling

Kesimpulan :

Pada penelitian ini, penulis lebih memberikan tambahan agar model Continuous Skip Gram lebih memiliki representasi vektor yang lebih berkualitas dan meningkatkan kecepatan dari training dataset. Dalam penelitian ini juga membuktikan bahwa tambahan metode dapat diaplikasikan terhadap dataset yang cukup besar dengan waktu yang cukup singkat. Kemudian, pada penelitian ini menggunakan pendekatan hierarchical softmax dan negative sampling. Kemudian dibuat library yang bernama Word2Vec untuk mengimplementasikan metodenya.

6. Bag of Tricks for Efficient Text Classification
Penulis/Tahun/Sumber : A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov;2016 [8]
Metode : - FastText
Kesimpulan : Dalam penelitian ini diusulukan metode dalam klasifikasi teks. Tidak seperti *word vectors* dari Word2Vec, fitur kata dalam metode ini dapat dirata-ratakan bersama untuk membentuk representasi kalimat yang baik. Dalam beberapa tugas, FastText memperoleh kinerja yang lebih cepat.

2.2. Landasan Teori

Landasan teori berisi teori-teori yang digunakan dalam pengerjaan penelitian tugas akhir. Dalam landasan teori, acuan yang digunakan adalah berdasarkan penelitian dan buku.

2.2.1. *Machine Learning*

Algoritma *Machine Learning* merupakan algoritma yang mempunyai kemampuan untuk mampu belajar dari data dan pengalaman. Program komputer dikatakan belajar dari pengalaman, dengan memperhatikan beberapa hal seperti tugas dan ukuran kinerja, jika suatu tugas yang dapat diukur untuk dapat meningkatkan pengalaman, sehingga mesin tersebut dapat dikatakan belajar [9].

Tantangan utama dalam *Machine Learning* adalah bahwa algoritma harus dapat berkinerja dengan baik pada hal-hal yang sebelumnya belum diketahui, tidak hanya pada model yang digunakan. Kemampuan untuk berkinerja baik pada input yang sebelumnya tidak diketahui disebut dengan generalisasi.

Namun, masalah yang sering terjadi dalam *Machine Learning* adalah bahwa set pelatihan yang besar diperlukan untuk generalisasi yang baik, namun rangkaian pelatihan yang besar juga lebih mahal secara komputasi. Sehingga terdapat faktor-faktor yang dapat menentukan seberapa baik kemampuan suatu algoritma *Machine Learning* yaitu :

1. Membuat nilai kesalahan yang dihasilkan data pelatihan kecil.
2. Membuat jarak antara nilai kesalahan yang dihasilkan oleh data pelatihan dan data test kecil.

Terdapat dua faktor yang sesuai dengan tantangan di dalam *Machine Learning* ini yaitu *underfitting* dan *overfitting* [10]. *Underfitting* terjadi ketika suatu model tidak dapat memperoleh nilai kesalahan yang rendah pada set pelatihan. Sedangkan, *overfitting* terjadi ketika jarak antara nilai kesalahan yang dihasilkan data pelatihan dan uji terlalu besar.

2.2.2. *Deep Learning*

Deep Learning merupakan salah satu cabang *Machine Learning (ML)* yang menggunakan *Deep Neural Network* untuk menyelesaikan permasalahan pada masalah *Machine Learning (ML)* [10]. Kinerja dari model *Deep Learning* terinspirasi oleh sistem kerja otak yang dapat diekspresikan secara matematis, dan parameter yang menentukan model matematis yang dapat berurutan hingga lebih dari 100 juta, di mana hal tersebut dapat dipelajari secara otomatis dari data. Hampir semua algoritma *deep learning* dapat digambarkan dengan contoh resep sederhana yang cukup sederhana yaitu : menggabungkan spesifikasi kumpulan data, *cost function*, prosedur pengoptimalan dan model [10].

Hampir semua *deep learning* didukung oleh satu algoritma yang sangat penting *stochastic gradient descent* (SGD). Metode *gradient descent* pada umumnya sering dianggap lamban atau tidak dapat diandalkan. Namun seperti

diketahui bahwa model *machine learning* dapat bekerja sangat baik saat dilatih dengan gradien [10] . *Deep Learning* adalah kunci utama teknologi bertenaga *Artificial Intelligence (AI)* yang sedang dikembangkan di seluruh dunia.

2.2.3. *Sentimen Analysis*

Sentiment Analysis adalah proses komputasi untuk mengidentifikasi dan kategori pendapat yang diungkapkan dalam sebuah teks, terutama untuk menentukan apakah sikap terhadap topik tertentu, produk, dan lain-lain adalah positif, negatif, atau netral [11]. *Sentiment Analysis* biasa dikenal juga sebagai *opinion mining*. *Sentiment analysis* dapat memberikan gambaran jika terjadi perubahan opini publik terhadap suatu aspek bisnis. Kelemahan maupun kekuatan terhadap suatu produk untuk dapat melakukan perbaikan produk, atau untuk melakukan analisis terhadap pembuatan *new marketing campaign*.

Dengan kata lain, *Sentiment Analysis* adalah garis penelitian yang memanfaatkan opini dan sikap orang-orang dalam kaitannya dengan berbagai topik, produk, peristiwa dan atribut. Ini adalah perpanjangan *data mining* di domain NLP *(Natural Language Processing)*. Konsep ini melibatkan kemampuan untuk dapat mempelajari setiap kata atau frase yang terdapat dalam teks dan memberi label sebagai positif, negatif atau netral.

2.2.4. *Word Embedding*

Word Embedding adalah perubahan yang dilakukan terhadap teks menjadi angka yang mempunyai representasi numerik yang berbeda antar teks. *Word Embedding* adalah representasi vektor dari sebuah bentuk kata di mana kata-kata serupa diharapkan saling dekat di dalam suatu ruang vektor, contohnya seperti cheetah, jaguar, panther, dan macan di mana kata-kata tersebut sangat dekat dalam suatu ruang vektor [12].

Algoritma *Machine Learning* dan hampir semua arsitektur *Deep Learning* tidak dapat memproses *string* atau teks biasa sehingga dibutuhkan penomoran sebagai masukan dalam proses pembelajaran baik untuk klasifikasi, regresi, dan yang lainnya. Sehingga, dibutuhkan peran *Word Embedding* untuk melakukan tugas ini.

Word Embedding merupakan salah satu aplikasi pembelajaran *unsupervised* yang paling berhasil terutama karena kekuatan generalisasi mereka. Susunan dari *Word Embedding* sangat bervariasi, namun secara umum *neural language* model dilatih pada korpus besar dan *output* dari jaringan digunakan untuk mempelajari *vector* kata (misalnya Word2Vec) [7].

a. Word2Vec

Word2Vec merupakan sebuah proyek yang dikerjakan dengan mengimplementasikan *word embedding/word representation* yang dibuat oleh Tomas Mikolov, dkk [7]. Pada *Word2Vec* digunakan representasi distribusi dari sebuah kata. *Word2Vec* memperlakukan setiap kata sebagai unit terkecil yang representasi vektornya dapat ditemukan.

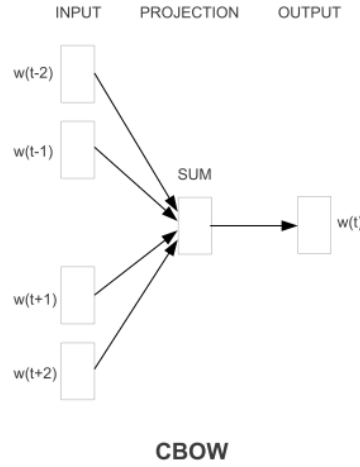
Saat ini, telah banyak model yang bertujuan untuk dapat merepresentasikan kata-kata secara kontinyu seperti LDA *(Latent Dirichlet Allocation)* dan LSA *(Latent Semantic Analysis)*. Tomas Mikolov kemudian menggunakan sebuah metode hasil dari modifikasi *Neural Network Languange Model* (NNLM) yang mana neural networks dapat bekerja dengan baik dalam menjaga keteraturan linear antara kata-kata dibandingkan dengan LSA dan lebih baik dalam menangani data besar dibandingkan dengan LDA [7]. Model yang digunakan ialah *Continuous Bag-of-Words* dan *Continuous Skip-gram*. Model dari *neural network language* dapat berhasil dalam dua tahap yaitu : pertama, vektor kata yang terus menerus dipelajari dengan menggunakan model sederhana, dan kemudian N-gram *Neural Network Languange Model* (NNLM) dilatih dengan menggunakan representasi kata-kata terdistribusi ini [7].

1) Arsitektur Word2Vec (*Log-Linear Models*)

Terdapat dua arsitektur model dalam *Word2Vec* untuk mempelajari representasi kata-kata terdistribusi yang mencoba meminimalkan kerumitan komputasi yang keduanya sama – sama menggunakan model *log-linier.*

a) Continuous Bag-of-Words Model (CBOW)

Arsitektur ini mirip dengan *Neural Network Languange Model* (NNLM) *feedforward*, di mana lapisan tersembunyi non-linier dilepaskan dan lapisan proyeksi dibagi untuk semua kata. Dengan demikian, semua kata dapat diproyeksikan ke posisi yang sama (dengan merata-ratakan nilai vektornya). Urutan kata tidak mempengaruhi nilai proyeksi. CBOW menggunakan representasi kontekstual terdistribusi secara berkelanjutan.



Gambar 2.1 Arsitektur CBOW

CBOW pada *word2vec* digunakan untuk memprediksi sebuah kata (t) berdasarkan konteks kata-kata yang ada di sektiarnya (c), tujuan utamanya adalah memperbesar peluang P(t|c) pada sekumpulan training dataset. Contohnya adalah

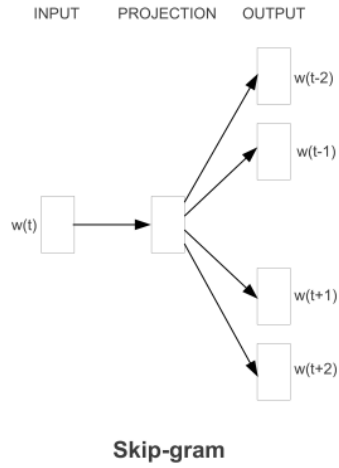
terdapat kalimat seperti “warga dari X telah melakukan demo hari ini”, maka hasil dari X yang mana merupakan kata target (t) kemungkinan adalah nama tempat, kota, ataupun negara yang berhubungan secara semantic. Mikolov et al. juga telah menggambarkan training complexity dari CBOW pada persamaan berikut [7].

$$Q = N \times D + D \times \log2(V)$$

Keuntungan menggunakan CBOW adalah proses training lebih cepat dibandingkan dengan Skip-gram dan juga memiliki akurasi yang cukup baik pada kata yang jarang muncul.

b) Continuous Skip-gram Model

Arsitektur Skip-gram Model memprediksi kata saat ini berdasarkan konteksnya, ia mencoba memaksimalkan klasifikasi kata berdasarkan kata lain dalam kalimat yang sama. Dapat digunakan setiap kata saat ini untuk dapat memprediksi kata-kata dalam rentang tertentu sebelum dan sesudah kata sekarang.



Gambar 2.2 Arsitektur *Skip-gram*

Mikolov juga menuliskan persamaan yang mana mencerminkan *training complexity* dari model *Skip-gram* ini yang dapat dilihat pada persamaan di bawah ini [7].

$$Q = C \times (D + D \times \log2\ (V))$$

Menurut Mikolov, Skip-gram memiliki keuntungan dapat bekerja dengan baik pada data yang tidak terlalu besar dan dapat merepresentasikan kata-kata yang jarang/frase dengan cukup baik.

2) Metode Training

Dalam proses *training methods* Word2Vec juga memiliki dua model, yaitu *Hierarchical Softmax* dan *Negative Sampling*.

a) Hierarchical Softmax

Hierarchical Softmax dapat melakukan perhitungan dengan lebih cepat dengan bantuan *binary tree structure*. *Hierarchical Softmax* melakukan *encode* model bahasa keluaran dari *softmax layer* ke dalam bentuk *tree hierarchy*, dimana setiap *node* adalah satu kata dan secara eksplisit mewakili probabilitas dari *children nodes.* Hal ini mendefinisikan pergerakan acak yang memberikan kemungkinan pada setiap kata.

Struktur tree yang digunakan pada hierarchical softmax memiliki dampak yang perlu dipertimbangkan dalam performa yang dimiliki. Mikolov menggunakan binary Huffman tree, yang memberikan kode pada kata yang sering muncul dari hasil training. Hasil pengamatan membuktikan bahwa dengan mengelompokkan kata secara bersamaan dengan kata kata yang memiliki frekuensi tinggi dapat berjalan dengan baik dan merupakan teknik yang mampu mempercepat pemrosesan neural network [2].

b) Negative Sampling

Negative Sampling adalah metode yang direkomendasikan oleh Mikolov dengan model Skip-gram. Metode ini muncul karena Metode ini muncul karena adanya modifikasi yang dilakukan pada paper kedua Mikolov untuk membuat proses training lebih mudah dan cepat [2].

Pada word2vec, untuk mendapatkan hasil kemiripan yang tinggi dari hasil dot product diantara banyak vektor kata yang muncul bersamaan pada sebuah teks dan meminimalkan kemiripan pada kondisi yang sebaliknya, denominator harus menghitung kemiripan target kata (w) dengan seluruh konteks pada setiap konteks kata(c) dan memastikan bahwa kata yang muncul bersamaan akan memiliki kemiripan yang lebih besar daripada yang tidak. Vektor kata dengan jumlah komponen dan vocabulary yang besar akan menyebabkan beban yang semakin besar pada pemrosesan neural network dengan hidden layer dan output layer yang dimilikinya. Menjalankan algoritma gradient descent pada sebuah neural network dengan data tersebut akan memakan waktu yang sangat lama sehingga dibutuhkan training data dalam jumlah yang sangat besar untuk memberikan hasil yang baik dan menghindari overfitting data. Dari permasalahan tersebut Mikolov membuat inovasi dengan memodifikasi dalam melakukan optimasi dengan menngunakan metode Negative Sampling yang hanya memilih pasangan kata dalam konteks c secara random sehingga memungkinkan proses word2vec akan lebih cepat. Setiap saat, sebuah kata akan semakin dekat dengan tetangganya, sedangkan sejumlah kecil kata lain (dipilih secara acak dari distribusi unigram di seluruh kata pada korpus) akan dijauhkan.

b. FastText

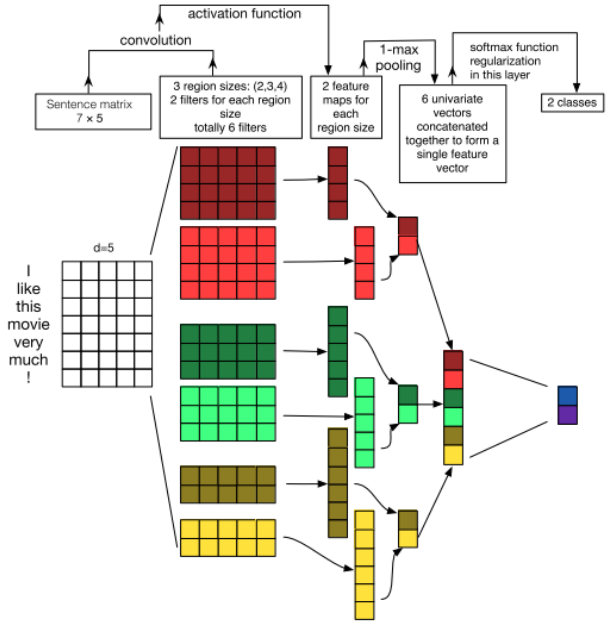
FastText merupakan *library open source* yang dikembangkan oleh tim peneliti Facebook untuk mempelajari representasi kata-kata dan klasifikasi kalimat secara efisien. *FastText* merupakan suatu metode cepat dan efektif untuk dapat mempelajari representasi dari sebuah kata dan melakukan klasifikasi teks [8].

Tujuan utama dari *embedding FastText* adalah mempertimbangkan struktur kata-kata internal daripada mempelajari representasi kata. Ini sangat berguna untuk bahasa yang memiliki morfologi yang kaya sehingga representasi morfologi bentuk kata yang berbeda akan dipelajari secara independen [13].

FastText dapat membantu dalam mengolah dataset dengan jumlah kategori yang sangat banyak, yang menggunakan klasifikasi *hierarchical* dibandingkan dengan pengkategorian yang disusun secara tree (binary tree). Hal ini mengurangi kompleksitas waktu pelatihan dan pengujian klasifikasi teks dari linier ke logaritmik yang berhubungan dengan jumlah kelas [14]. *FastText* mengasumsikan sebuah kata akan dibentuk oleh susunan karakter n-gram, misalnya kata *sunny* tersusun dari [sun, sunn, sunny], [sunny, unny, nny] dan lain-lain, dimana nilai n berkisar dari 1 sampai panjang kata tersebut.

2.2.5. *Convolutional Neural Network (CNN)*

Convolutional Neural Network (CNN) awalnya diciptakan untuk *computer vision* yang merupakan inti dari sebagian besar sistem *computer vision* saat ini, mulai dari penandaan foto otomatis pada Facebook hingga mobil yang dapat menggerakkan dirinya sendiri. *Convolutional Neural Network* merupakan terobosan besar dalam pengembangan *Image Classification*. Namun, ternyata model CNN kemudian terbukti efektif untuk *Natural Languange Processing (NLP)* dan telah mencapai hasil yang sangat baik [15].



Gambar 2.3 Arsitektur *Convolutional Neural Network*

Convolutional Neural Network merupakan model yang terdiri dari satu atau lebih lapisan konvolusi dan kemudian diikuti oleh satu atau lebih lapisan yang saling terhubung seperti pada jaringan syaraf multilayer standar [16]. Seperti pada gambar 2.2.3 terdapat kalimat *I like this movie very much !* yang terdiri dari 6 kata dan 1 tanda baca yang terwakili oleh 5 vektor dimensi. Kemudian 6 kata tersebut akan diberikan filter size dengan besar masing-masing 2, 3, dan 4 yang kemudian akan digunakan untuk menghasilkan 6 peta dari atribut. 6 peta atribut akan memiliki ukuran 4x1, 5x1, dan 6x1 dimana peta tersebut akan mengalami proses penyatuan dan digabungkan untuk membentuk vektor unik dengan ukuran 6x1. Kemudian, vektor unik akan masuk pada lapisan *Softmax*, yang akan memeras menjadi vektor 2x1 yang mengandung probabilitas 2 kelas (positif atau negatif). *Convolutional Neural Network*

memanfaatkan lapisan dengan filter konvolusi yang diterapkan pada fitur local [17].

Hingga saat ini banyak penerapan aplikasi *Convolutional Neural Network (CNN)* untuk *Natural Languange Processsing* diantaranya adalah *Sentiment Analysis*, *Spam Detection* or *Topic Categorization*.

2.2.6. *Text Mining*

Text Mining berhubungan dengan penemuan informasi baru secara otomatis atau semi otomatis yang sebelumnya tidak diketahui dari data berkualitas tinggi dari sejumlah besar teks yang tidak terstruktur [18]. Dalam melakukan proses *Text Mining* terdapat tiga teknik secara berurutan, yang masing-masing dapat dikaitkan dengan *Text Mining* [19]. Teknik tersebut merupakan penggabungan dari *information retrieval*, *natural language processing*, dan *data mining*. Pertama, *Information Retrieval* bertugas dalam mengumpulkan teks yang berpotensi dan relevan terhadap tugas yang diberikan. Kedua, *Natural Languange Processing* bertugas dalam menganalisis teks masukan untuk mengidentifikasi dan menyusun informasi yang relevan. Ketiga, *Data Mining* bertugas dalam menemukan pola dalam informasi terstruktur yang telah disimpulkan dari teks.

Aspek utama *Text Mining* sebenarnya sama dengan yang dipelajari dalam *empirical computational linguistic.* Meskipun fokus pada *Natural Languange Processing*, beberapa masalah yang terkait dengan linguistik komputasi juga dibahas dalam *information retrieval* dan *data mining.*

a. Information Retrieval

Information Retrieval adalah suatu metode untuk mencari dan mendapatkan teks-teks dari sekumpulan besar teks yang tidak terstruktur yang dapat memenuhi kebutuhan

informasi, berdasarkan *query* yang diberikan [20]. Ada beberapa konsep dalam *Information Retrieval* yang berkatian dengan analisis teks antara lain :

- *Vectors*, Untuk menentukan relevansi teks, banyak pendekatan memetakan semua teks dan kueri ke dalam model ruang vektor [20]. Model seperti itu mendefinisikan representasi vektor umum $x = (x_1, ..., x_k)$, $k \geq 1$, untuk semua masukan dimana masing-masing $x_i \in x$ merepresentasikan properti masukan. Masukan konkret seperti teks D kemudian ditunjukkan oleh satu nilai $x_i^{(D)}$ untuk masing-masing x_i.
- Similarity yaitu dengan memberikan representasi umum, kesamaan antara teks dan *query* yang dapat dihitung. Dengan adanya representasi umum, kesamaan antara teks dan *query* dapat dihitung. Sebagian besar frekuensi kata dari *query* pencarian akan sering menghasilkan nilai 0.
- Indexing yaitu semua teks dalam koleksi tertentu telah diindeks sebelumnya. Permintaan kemudian dicocokkan dengan indeks pencarian, sehingga menghindari memproses teks sebenarnya selama pencarian. Pendekatan pengindeksan yang sangat canggih ada dan digunakan di mesin pencari web saat ini [20]. Dalam bentuk dasarnya, indeks pencarian berisi satu entri untuk setiap properti terukur. Setiap entri menunjuk ke semua teks yang relevan sehubungan dengan properti.
- Filtering dibagi menjadi dua tingkat *Text Filtering* dan *Passage Retrieval*. *Text Filtering* digunakan untuk mengklasifikasikan teks lengkap sebagai data teks yang relevan digunakan atau tidak relevan untuk digunakan, sedangkan *Passage Retrieval* bertujuan untuk menentukan bagian teks yang relevan untuk menjawab *query* tertentu.

b. Natural Languange Processing

Natural Languange Processing (NLP) dapat didefinisikan sebagai sebuah pemrosesan bahasa manusia secara otomatis atau semi otomatis [21]. Proses tersebut dilakukan untuk mengkaji interaksi komputer dengan bahasa alami manusia yang seringkali digunakan dalam kehidupan sehari-hari. Pengembangan *Natural Languange Processing (NLP)* disebabkan karena adanya tantangan dimana mengharuskan manusia untuk dapat berbicara kepada komputer dalam bahasa pemrograman yang tepat, tidak ambigu dan sangat terstruktur, atau melalui sejumlah perintah suara yang jelas. Proses komputasi bahasa direpresentasikan sebagai suatu rangkaian simbol yang memenuhi aturan tertentu. Dalam proses natural language processing terdapat beberapa kesulitan diantaranya sering terjadi ambiguitas atau makna ganda dan jumlah kosa kata dalam bahasa alami yang semakin besar dan berkembang dari waktu ke waktu yang memungkinkan adanya interpretasi yang berbeda, sehingga algoritma analisis teks perlu untuk dapat menyelesaikan masalah ambiguitas. Bagaimanapun, seringkali perkataan manusia tidak selalu tepat, seringkali ambigu dan struktur tata bahasa dapat bergantung pada banyak variabel yang kompleks, termasuk bahasa gaul, dialek daerah, dan konteks sosial.

c. Data Mining

Data mining dapat juga disebut sebagai penemuan pengetahuan dalam database. Hal ini biasanya didefinisikan sebagai proses untuk menemukan pola atau pengetahuan yang berguna dari sumber data [22]. Pola harus valid, berpotensi bermanfaat, dan mudah dimengerti. Data mining bertujuan untuk menyimpulkan informasi baru dari jenis informasi yang telah ditentukan dari sejumlah besar data masukan yang biasanya besar, yang sudah diberikan dalam bentuk terstruktur [23]. Seringkali dalam data mining yang sering menjadi permasalah adalah prediksi. Untuk mengatasi masalah prediksi

tersebut, data pertama kali diubah menjadi contoh representasi yang didefinisikan dan kemudian dimasukkan ke algoritma *Machine Learning*. Algoritma mengenali pola statistik dalam kasus yang relevan untuk masalah prediksi tersebut, proses ini disebut *training*. Pola yang ditemukan kemudian digeneralisasi, sehingga bisa diterapkan untuk menyimpulkan informasi baru dari data yang tidak terlihat, yang umumnya disebut sebagai prediksi. Dalam hal ini, *Machine Learning* dapat dilihat sebagai dasar teknis aplikasi data mining [23].

Data Mining dan *Text Mining* terkait memiliki keterkaitan dalam dua hal. Pertama, informasi keluaran terstruktur dari analisis teks berfungsi sebagai masukan untuk *Machine Learning*, mis. untuk melatih pengklasifikasi teks. Kedua, banyak analisis teks itu sendiri bergantung pada algoritma *Machine Learning* untuk menghasilkan informasi keluaran.

2.2.7. *E-Commerce*

Electronic commerce atau *e-commerce* adalah istilah untuk semua jenis bisnis, atau transaksi komersil, yang melibatkan pengalihan informasi di Internet secara otomatis. Ini mencakup berbagai jenis bisnis, mulai dari situs ritel berbasi konsumen, melalui situs lelang atau music, hingga pertukaran bisnis yang memperdagangkan barang dan jasa antar perusahaan.

Ada beberapa jenis e-commerce, antara lain *E-commerce business to business (B2B), E-commerce business to consumer (B2C), E-commerce consumer to consumer (C2C), dan E-commerce consumer to business (C2B)*. Ada banyak keuntungan yang didapatkan dari e-commerce. Salah satunya adalah menjual produk atau jasa secara *online* tanpa harus mendirikan toko atau kantor besar seperti yang dilakukan oleh para pelaku bisnis *offline* sebagai tempat usaha. E-commerce juga memungkinkan konsumen untuk secara elektronik menukar barang dan jasa tanpa hambatan waktu atau jarak

2.2.8. Media Sosial

Media sosial menurut definisi Walter & Riviera pada tahun 2004 merupakan hubungan yang ada antar tiap orang di dalam sebuah jaringan. Media sosial merupakan kumpulan saluran komunikasi secara online dapat didedikasikan untuk dapat saling berinteraksi, saling berbagi konten, dan berkolaborasi di dunia virtual untuk dapat menyampaikan pendapat atau informasi kepada publik. Media sosial sendiri adalah struktur sosial yang terdiri dari elemen individual atau organisasi yang mempertemukan kelompok yang berhubungan karena kesamaan sosialitas, visi, ide, dan lain-lain. Media sosial mengajak siapa saja yang tertarik untuk saling berpartisipasi dengan memberi kontribusi dan feedback secara terbuka, memberi komentar, serta membagi informasi dalam waktu cepat dan tak terbatas. Salah satu media sosial terbesar adalah Twitter.

Twitter adalah suatu situs web layanan jaringan sosial dan mikroblog yang memberikan fasilitas bagi pengguna untuk mengirimkan "pembaharuan" berupa tulisan teks dengan panjang maksimum 280 karakter. Twitter didirikan pada Maret tahun 2006 oleh perusahaan rintisan Obvious Corp. Twitter dapat menjadi sumber yang sangat bermanfaat untuk mengumpulkan data yang digunakan dalam penelusuran informasi yang berkembang.

Pada September 2010, didapatkan 95 juta tweet per hari yang membuat Twitter mampu menjadi sumber informasi yang tepat untuk menganalisa informasi di media sosial. Namun di sisi lain, konten dari tweet yang ada di dalam Twitter seringkali memiliki tata bahasa yang kurang baik dikarenakan batas huruf atau karakter yang dibatasi untuk sekali tweet.

2.2.9. *Crawling*

Crawling merupakan sebuah proses untuk mendapatkan informasi konten atau keseluruhan isi halaman yang terdapat pada suatu halaman *website* dan menyimpannya secara *offline*

[24]. Dalam penelitian ini *crawling* dilakukan pada laman media sosial Twitter. *Crawling* bertujuan untuk mendapatkan data berupa tweet yang bersumber dari media sosial Twitter. Data-data yang diambil adalah data publik berupa tweet, waktu, pengguna, dan keterangan lain yang dibutuhkan. Semua informasi ini didapatkan melalui bantuan layanan dari Twitter API.

2.2.10. *Pytorch*

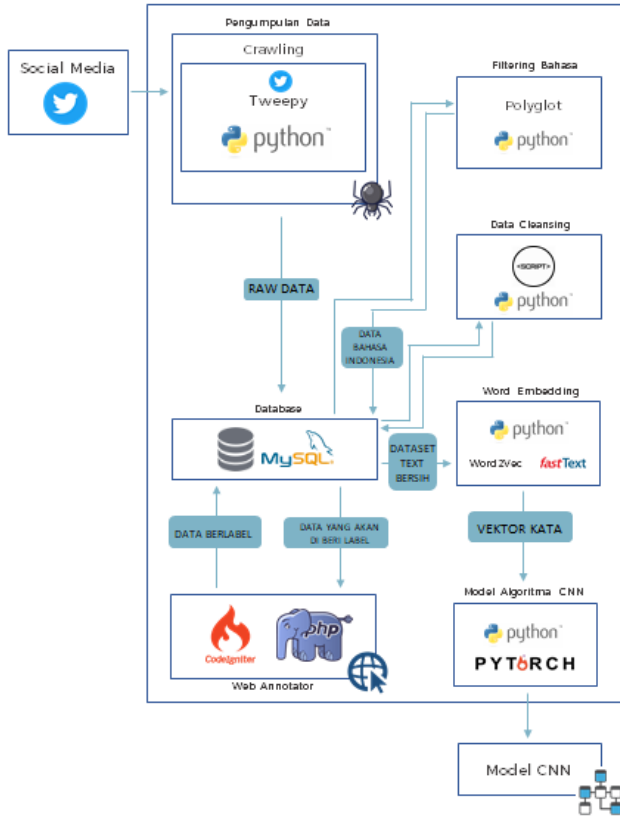
Pytorch merupakan paket komputasi ilmiah yang berbasis Python. *Pytorch* ditargetkan pada dua hal yaitu sebagai pengganti NumPy untuk dapat menggunakan kekuatan GPU dan merupakan sebuah platform riset yang memberikan fleksibilitas dan kecepatan maksimal. *Pytorch* juga menyediakan fitur tingkat tinggi yaitu perhitungan tensor (seperti NumPy) dengan akselerasi GPU yang kuat dan *Deep Neural Networks* yang dibangun di atas sistem berbasi tipe autograd. Autograd merupakan salah satu bagian terpenting pada PyTorch, Autograd secara otomatis akan menghitung *gradient* secara menyeluruh atas *computation graph* yang sudah kita buat.

BAB III
METODOLOGI PENELITIAN

Pada bab ini akan dijelaskan mengenai gambaran metode dan alur pengerjaan tugas akhir. Berikut ini merupakan alur pengerjaan tugas akhir.

3.1. Arsitektur

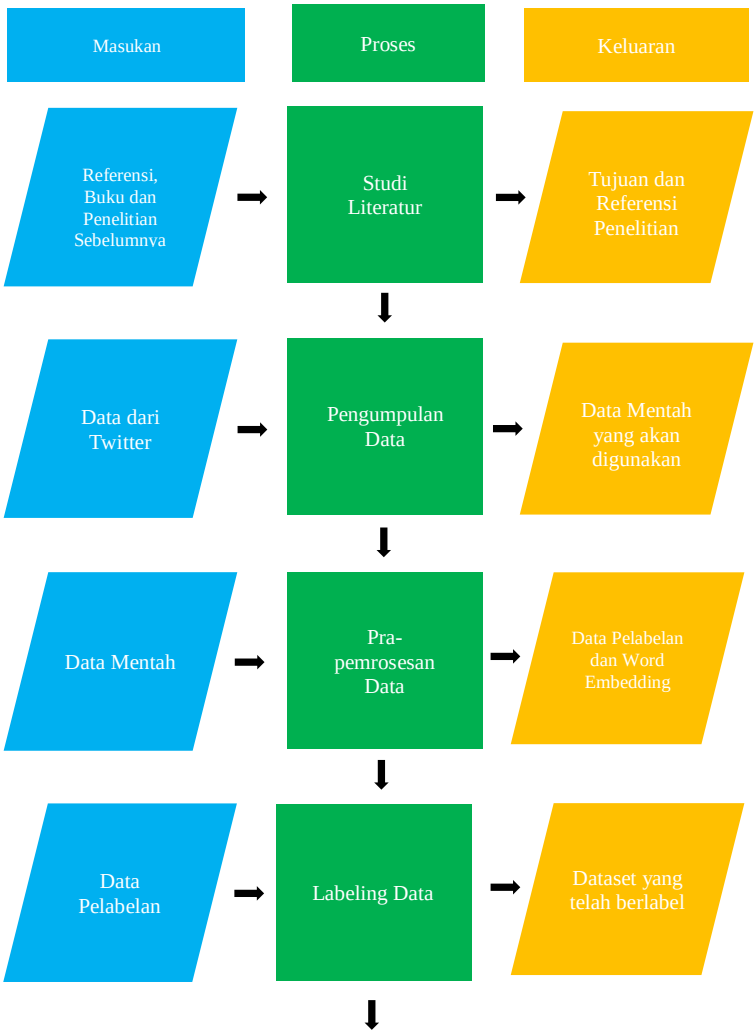
Pada bagian ini akan menjelaskan arsitektur pada penelitian yang akan menjelaskan secara garis besar aktivitas. Input, output dan metode di setiap prosesnya. Arsitektur dapat dilihat pada Gambar 3.1.

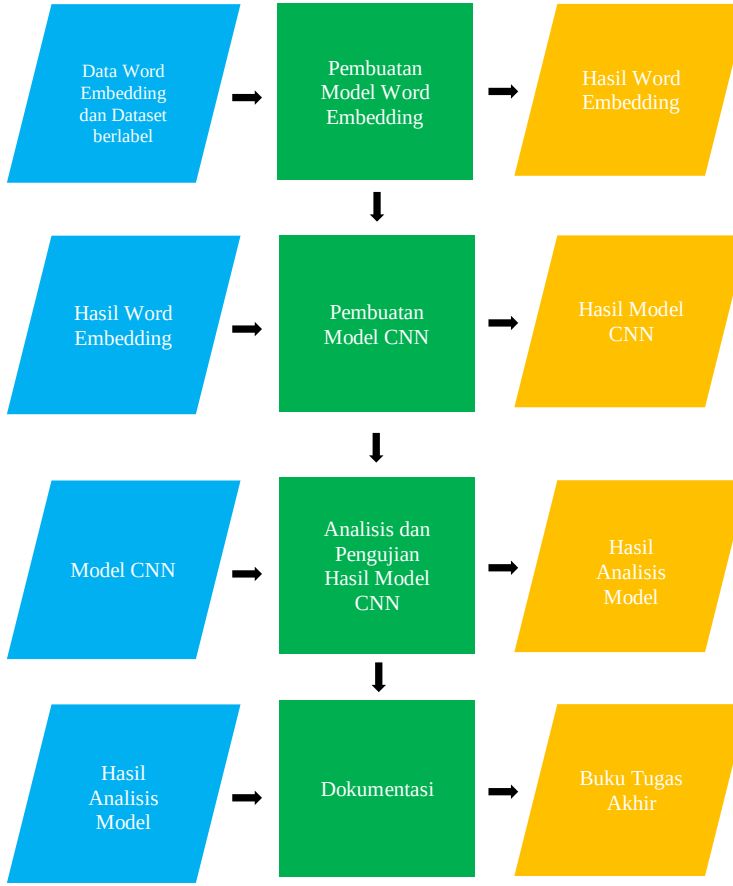


Gambar 3.1 Arsitektur Model

3.2. Tahapan Pelaksanaan Penelitian Tugas Akhir

Pada bagian ini akan dijelaskan mengenai metodologi atau tahapan pelaksanaan penelitian tugas akhir.





Gambar 3.2 Tahapan Pelaksanaan Tugas Akhir

3.2.1. Studi Literatur

Tahapan awal dalam pengerjaan Tugas Akhir ini adalah studi literatur. Pada studi literatur digunakan untuk menentukan topik dan penggalian kebutuhan terkait dengan studi kasus yang akan diambil. Dikarenakan masih sedikitnya penelitian yang membahas mengenai hal ini maka paper-paper yang dijadikan referensi berasal dari luar negeri dan beragam metodenya. Paper-paper yang diambil adalah penelitian seputar

Convolutional Neural Network (CNN). Paper utama yang menjadi rujukan tentang *Convolutional Neural Network (CNN)* adalah "*Convolutional Neural Networks for Sentence Classification*" dan "*A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification*". Kedua paper tersebut membahas tentang cara dalam merepresentasikan sebuah algoritma *deep learning* yang dikembangkan untuk *image classification.* Namun ternyata, baik juga diimplementasikan dalam permasalahan *Natural Languange Processing.* Metode ini yang akan dijadikan sebagai dasar untuk pembuatan model *Convolutional Neural Network (CNN)* pada tugas akhir nantinya.

3.2.2. Pengumpulan Data

Data yang digunakan merupakan data yang telah dikumpulkan dari media sosial Twitter. Data yang dikumpulkan merupakan ulasan dari media sosial 5 marketplace besar di Indonesia yaitu Tokopedia, Lazada, Bukalapak, Blibli, dan Shopee.

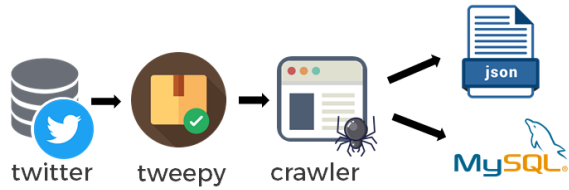
Proses pengumpulan data pada Twitter menggunakan tiga versi *Application Programming Interface (API)* berbeda yang telah disediakan oleh twitter yaitu REST API, Search API, dan Streaming API. Dengan menggunakan REST API dapat dikumpulkan data status dan informasi pengguna. Search API dapat memungkinkan untuk mendapatkan konten yang spesifik. Sedangkan Streaming API dapat digunakan untuk mendapatkan data Twitter secara realtime. Nantinya, data mentah yang telah dihasilkan akan disimpan kedalam dua bentuk yaitu database MySQL dan JSON. Daftar akun Twitter akan diambil datanya dapat dilihat pada Tabel 3.1.

Tabel 3.1 Daftar Keyword Media Sosial Twitter

Twitter
tokopedia
tokopediacare

bukalapak
bukabantuan
bliblidotcom
bliblicare
lazadaid
lazadacare
shopee
shopeecare

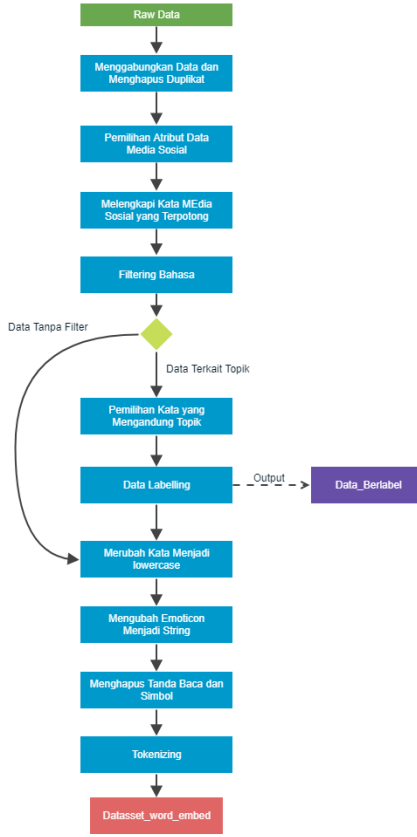
Data mentah akan dikumpulkan dengan menggunakan crawler yang dijalankan sesuai dengan jadwal. Crawler yang digunakan memanfaatkan library tweepy. Nantinya, data mentah yang telah dikumpulkan akan disimpan kedalam dua bentuk yaitu database MySQL dan JSON. Lebih jelasnya dapat dilihat arsitekturnya pada Gambar 3.3.



Gambar 3.3 Proses Crawling Twitter

3.2.3. Pra-Pemrosesan Data

Data yang telah di peroleh dari proses *crawling* dan *scraping* akan masuk ke tahapan pra-pemrosesan data. Dimana setiap data yang akan digunakan pada proses training akan memasuki tahapan pemrosesan data satu per satu.



Gambar 3.4 Proses Pra-Pemrosesan Data

a. Menggabungkan Data dan Menghapus Data Duplikat

Data yang didapatkan dari *crawling* dan *scraping* masih terdapat pada sumber yang terpisah sehingga perlu digabungkan untuk menjadi satu dataset. Selain itu, tidak menutup kemungkinan saat proses memasukkan data kedalam database, terdapat data yang terduplikasi sehingga harus dihapus. Dengan melalui tahapan ini maka akan diperoleh data teks yang lebih layak diproses pada tahap berikutnya.

b. Pemilihan Atribut

Pada tahapan ini akan memisahkan antara atribut yang penting dan tidak penting dalam penelitian ini. Tujuan dari pemilihan atribut ini untuk mengurangi ukuran dari dataset sehingga mempercepat proses training data.

c. Penyaringan Bahasa

Pada tahapan ini, proses untuk menghilangkan *tweets* yang mengandung bahasa asing. Untuk melakukan aktivitas ini akan memanfaatkan library dari python yaitu *polyglot*. Library tersebut untuk memudahkan penyaringan bahasa.

d. Menghapus Data yang Tidak Mengandung Topik dalam Teks untuk Data CNN

Pada tahapan Pra-Pemrosesan dataset pelabelan ini akan dipilih data yang didalam pesan tersebut yang mengandung kata-kata yang termasuk dalam topik. Sedangkan kata-kata yang tidak mengandung topik tidak dipilih dalam untuk masuk dalam dataset pelabelan. Proses ini dilakukan untuk menjaga konsistensi sentimen di dalam teks terhadap topik. Berikut merupakan contohnya yang dapat dilihat pada Tabel 3.4.

Tabel 3.2 Proses Pemilihan Teks yang Mengandung Topik

Topik	Text	Dipilih	Tidak Dipilih
BukaBantuan	@BukaBantuan @GrabID @bukalapak dari kemarin, belum ada driver dari pihak grab yg datang. Mau selesai kapan transaksi saya?	✔	
TokopediaCare	Hai Faisal, mohon maaf atas		✔

	keterlambatan kami dalam membalas pesan kamu dan mohon maaf atas ketidaknyamanannya dalam melakukan transaksi. Terkait dengan hal yang kamu maksudkan tersebut, apabila kamu melakukan pembelanjaan dlm satu toko yang sama -		

e. Mengubah Kata menjadi Lowercase

Pada tahapan ini akan mengubah setiap kata menjadi *lowercase*/huruf kecil. Sehingga nantinya saat pemrosesan data tidak terjadi ambiguitas. Berikut adalah contoh dari proses lowercase yang dapat dilihat pada Tabel 3.4.

Tabel 3.3 Proses Mengubah menjadi Lowercase

Sebelum Pra-proses	Setelah Pra-proses
@tokopedia @TokopediaCare KETERLALUAN! Saya melaporkan penjual atas dugaan penipuan malah AKUN SAYA YANG DIBLOK, lalu diminta SEGALA MACAM dokumentasi dan MASIH BELUM CUKUP. HARI GINI SIAPA YANG KE	@tokopedia @tokopediacare keterlaluan! Saya melaporkan penjual atas dugaan penipuan malah akun saya yang diblok, lalu diminta segala macam dokumentasi dan masih belum cukup. hari gini siapa yang ke bank nyetak mutasi ommmmmmm??????

BANK NYETAK MUTASI OMMMMMMM??????	
@BukaBantuan PEMBELI TIDAK BACA DESKRIPSI YANG SAYA TULIS. APAKAH SEKARANG PENJUAL YANG DI SALAHKAN? TERIMA KASIIIIIHHHHHHHHHHH	@bukabantuan pembeli tidak baca deskripsi yang saya tulis. apakah sekarang penjual yang di salahkan? terima kasiiiiihhhhhhhhhhh

f. Mengubah Emoticon menjadi Token String yang Bersesuaian

Pada tahapan ini, kumpulan teks yang terdapat *emoticon (emotion icon)* akan dikonversi kedalam string yang bersesuaian. Berikut adalah contoh dari proses konversi *emoticon* yang dapat dilihat pada Tabel 3.5

Tabel 3.4 Proses Mengubah Emoticon menjadi String

Sebelum Pra-proses	Setelah Pra-proses
@bukabantuan @bukalapak mohon konfirmasi pembayaran saya, agar segera di proses penjual atau pelapak thanks :)	@bukabantuan @bukalapak mohon konfirmasi pembayaran saya, agar segera di proses penjual atau pelapak thanks <senang>
@margana274 @lazadaidcare @xiaomiindonesia @imannugraha @lazadaid sudah melewati waktu estimasi lebih dari 7 hari pihak lazadanya tiap ditanya bulet aja :(	@margana274 @lazadaidcare @xiaomiindonesia @imannugraha @lazadaid sudah melewati waktu estimasi lebih dari 7 hari pihak lazadanya tiap ditanya bulet aja <sedih>

g. Mengganti URL dan Mention menjadi Token String

Pada tahapan ini, kumpulan dari teks yang dimiliki masih mengandung URL maupun mention. Semua URL dan mention tidak akan dihapus melainkan dikonversi ke dalam bentuk string. Berikut adalah contoh dari proses konversi yang dapat dilihat pada Tabel 3.5.

Tabel 3.5 Proses Mengubah URL dan Mention menjadi Token String

Sebelum Pra-proses	Setelah Pra-proses
@bukabantuan @bukalapak mohon konfirmasi pembayaran saya, udah berapa lama nih kok gak diproses-proses	<mention> <mention> mohon konfirmasi pembayaran saya, udah berapa lama nih kok gak diproses-proses
@tokopediacare min tokopedia lagi bermasalah yak kok dibuka gak bias-bisa nih coba cek deh https://t.co/vpi77sqWLR	<mention> min tokopedia lagi bermasalah yak kok dibuka gak bias-bisa nih coba cek deh <url>

h. Menghapus Tanda Baca, Simbol dan Angka

Pada tahapan ini, setiap teks akan dihapus tanda bacanya dan simbol-simbol yang tidak bermakna. Tujuan dari proses ini adalah menghilangkan karakter yang tak bermakna serta memperkecil peluang kata yang memiliki makna sama namun berbeda penulisan karena penambahan tanda baca. Berikut adalah contoh dari proses menghapus tanda baca dan simbol yang dapat dilihat pada Tabel 3.6

Tabel 3.6 Proses Menghapus Tanda Baca, Simbol dan Angka

Sebelum Pra-proses	Setelah Pra-proses
@bukalapak bisa bantu batalkan resi otomatis j&t untuk no.transaksi ini 180642595049 sebab kantor j&t nya sdh pindah, sdh terlanjur klik resi otomatis	bukalapak bisa bantu batalkan resi otomatis untuk no transaksi ini sebab kantor nya sdh pindah sdh terlanjur klik resi otomatis
@shopeecare kenapa statusnya masih kaya begini ya kak ? bsk udah terakhir	shopeecare kenapa statusnya masih kaya begini ya kak bsk udah terakhir

i. Tokenizing

Tokenizing adalah proses membuat token/pemisahan per kata dari sebuah korpus. Nantinya token-token yang telah terbentuk digunakan untuk merepresentasikan sebuah kata kedalam vektor menggunakan word2vec. Berikut merupakan contohnya yang dapat dilihat pada Tabel 3.7.

Tabel 3.7. Proses Tokenizing

Sebelum Pra-proses	Setelah Pra-proses
bukabantuan trus kenapa dikirim dan pengiriman tidak sesuai dengan alamat	{bukabantuan, trus, kenapa, dikirim, dan, pengiriman, tidak, sesuai, dengan, alamat}
lazadaidcare min ini kok ga bisa di bayar ya di status pemesanan ada tapi di bayar ga bisa gimana ini	{lazadaidcare, min, ini, kok, ga, bisa, di, bayar, ya, di, status, pemesanan, ada, tapi,

	di, bayar, ga, bisa, gimana, ini}

3.2.4. Data Labeling

Proses pelabelan data ini menggunakan metode *Human Intelligence Task (HIT)* dimana proses ini harus memiliki tingkat persetujuan lebih besar dari 95%. Setiap HIT akan dilakukan oleh tiga orang. Dimana setiap orang diharuskan menunjukkan polaritas keseluruhan dari pesan tweet (pada skala 5 poin) serta keseluruhan polaritas pesan terhadap topik target yang diberikan (pada skala 5 poin) [25]. Berikut merupakan contoh proses pelabelan yang dapat dilihat pada Tabel 3.9.

Tabel 3.8 Proses Pelabelan Data

Tweet	Label Sentimen	Label Topik
Big Thx @LazadaIDCare perubahan yg positif, tdk spt FS sblmnya, lama diterima. Kali ini cepat, 2 hr sdh ditangan.	Sangat Positif	LazadaIDCare : Sangat Positif
@BukaBantuan @JNECare Saya tunggu 1 x 60 menit buat anda menyelesaikan bukan 3 x 24 jam !!! Sdh cukup sy menunggu..!!!	Sangat Negatif	BukaBantuan : Sangat Negatif
@BukaBantuan min , , perpanjang dong kode cashback "bayartagihan" utk bulan depan atau	Netral	BukaBantuan : Positif

ada cashback lain bulan depan min? Mhn infonya		

Human Intelligence Task (HIT) dapat ditolak jika terjadi masalah sebagai berikut :

- Satu atau lebih tanggapan tidak memiliki keseluruhan sentimen yang ditandai
- Satu atau lebih tanggapan tidak memiliki sentimen terhadap topik yang ditandai
- Satu atau beberapa tanggapan tampaknya dipilih secara acak

Dalam pemilihan anotasi label dibagi dalam skala lima poin yaitu sentimen sangat positif, positif, netral, negatif, dan sangat negatif yang jika direpresentasikan dalam bentuk angka menjadi +2, +1, 0, -1, dan -2, maka selanjutnya dalam pemilihan label akan dilakukan prosedur dalam dua langkah :

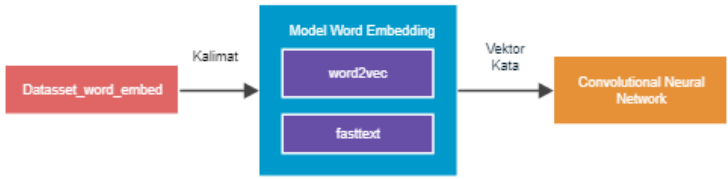
1. Jika 2 dari 3 annotator menyetujui sebuah label yang sama, maka label tersebut akan langsung diterima
2. Jika tidak, maka langkah awal adalah memetakan label kategoris dengan nilai integer -2, -1, 0, 1, 2. Kemudian dihitung nilai rata-rata dan akhirnya akan dipetakan atau dibulatkan ke nilai integer terdekat. Untuk menyeimbangkan kecenderungan rata-rata untuk menjauh dari -2 dan 2, dan juga untuk memilih 0. Maka tidak dilakukan pembulatan pada ± 0.5 dan ± 1.5, tapi pada ± 0.4 dan ± 1.4 sebagai gantinya.

Dataset nantinya akan dibagi menjadi beberapa subtask. Namun, untuk menjaga konsistensi maka untuk setiap subtask akan dilakukan pelabelan dengan cara yang sama. Pembagian data akan dilakukan setelah dilakukan pelabelan. Subtask A akan dibagi menjadi skala dua poin dengan menggabungkan sentimen sangat positif dan positif menjadi positif, dan

menggabungkan sentimen sangat negatif dengan negatif menjadi negatif, sedangkan untuk sentimen netral dihilangkan. Subtask B data akan akan dibagi menjadi skala tiga poin dengan menggabungkan sentimen sangat positif dan positif menjadi positif, dan menggabungkan sentimen sangat negatif dengan negatif menjadi negatif. Untuk, subtask C data yang digunakan sama dengan proses pelabelan.

3.2.5. Pembuatan Model Word Embedding

Pada tahapan *word embedding*, data yang telah melewati pra-pemrosesan akan dibuat model yang bertujuan untuk merubah setiap kata pada data pelatihan menjadi nilai vektor agar dapat diproses dalam algoritma CNN.



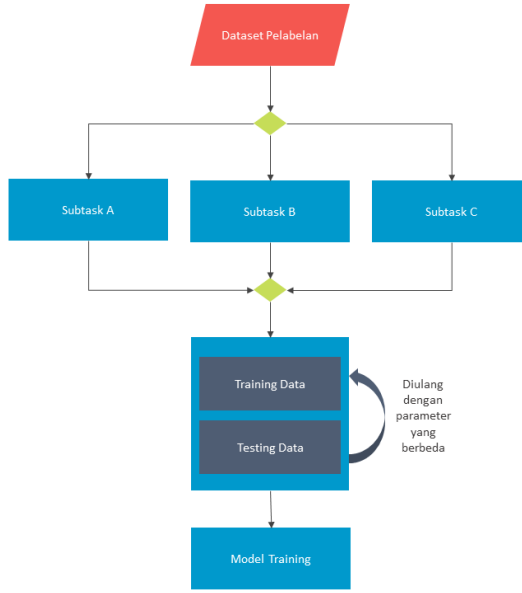
Gambar 3.5. Proses Model Word Embedding

Pada tahap ini akan dilakukan pembuatan model Word2Vec yang dibuat dengan memanfaatkan *library* Pytorch yang diimplementasikan menggunakan Python. Model yang dibuat akan menggunakan skenario yang dihasilkan pada penelitian Tomas Mikolov dengan menggunakan parameter yang tersedia sehingga mendapatkan model yang menghasilkan hasil klasifikasi dengan akurasi tertinggi. Data yang masuk pada model Word2Vec merupakan dataset yang didapatkan tanpa filter dan menggunakan Bahasa Indonesia.

3.2.6. Pembuatan Model CNN

Pada tahapan ini akan dibuat model *Convolutional Neural Network (CNN)*. Dimana model akan dibangun

menggunakan *library pytorch*. *CNN* akan menggunakan ukuran *filter size* dalam setiap aktifitas urutan pembelajarannya.



Gambar 3.6. Alur Proses Model CNN

Model *CNN* yang paling baik akan digunakan untuk diujikan pada dataset yang tersedia, kemudian selanjutnya akan dilakukan proses analisis dan pengujian untuk melihat nilai akurasi dan error yang dimiliki.

3.2.7. Analisis dan Pengujian Hasil Model CNN

Dalam setiap proses *training* model CNN yang dilakukan akan terdapat akurasi pelatihan, yang mana menunjukkan kualitas dari model yang dihasilkan. Semakin besar tingkat akurasi maka semakin bagus model yang dihasilkan.

Setelah model yang paling terbaik ditentukan, maka proses analisis dan pengujian hasil akan dilakukan dengan mengujikan data tes yang terdiri dari kata-kata yang telah

dilabeling berdasarkan tingkat sentimennya. Nantinya, tingkat keakuratan dihitung dari seberapa banyak sentiment yang dipilih sesuai dengan isi dari pesan di dalamnya.

3.2.8. Dokumentasi

Pada tahapan terakhir ini akan dilakukan pembuatan laporan dalam bentuk buku tugas akhir yang disusun sesuai format yang telah ditentukan. Buku ini berisi dokumentasi langkah-langkah pengerjaan tugas akhir secara rinci. Buku ini diharapkan dapat bermanfaat sebagai referensi untuk pengerjaan penelitian lain, serta sebagai acuan untuk pengembangan lebih lanjut terhadap topik penelitian yang serupa.

BAB IV
PERANCANGAN

Pada bab ini akan dijelaskan mengenai rancangan dari luaran penelitian tugas akhir ini. Perancangan yang dibuat berupa rancangan *crawler*, rancangan pra-pemrosesan data, dan rancangan model.

4.1. Akuisisi Data Media Sosial

Tahap awal pada perancangan dimulai dengan mengumpulkan seluruh data yang dibutuhkan dari media sosial untuk tahap pengolahan data selanjutnya. Pengambilan data dilakukan melalui *crawler* yang dirancang untuk mengumpulkan data dari media sosial twitter yang berbahasa Indonesia untuk proses *word embedding* dan data dari akun milik 5 *e-commerce* di Indonesia yang telah dijabarkan dalam sub bab 3.2.2 yaitu Tokopedia, Bukalapak, Lazada, Blibli, dan Shopee.

4.2. Perancangan *Crawler*

Proses untuk melakukan pengambilan data dari media sosial twitter dirancang dengan menggunakan sebuah *crawler* yang akan mengambil data dan menyimpan data yang akan digunakan baik untuk *word embedding* maupun dari akun milik *e-commerce* yang telah ditentukan. *Crawler* yang dibuat mengacu pada *library* tweepy untuk mengambil data Twitter.

4.3. Perancangan Read Data JSON

Hasil dari dataset yang diambil dari pross *crawling* Twitter menghasilkan data json yang berisi banyak atribut yang berisi informasi tentang tweet tersebut. Untuk itu akan dibuat sebuah kode program yang bertugas untuk mengambil informasi yang penting dari keseluruhan data Json.

Kode program ini akan dibuat dengan menggunakan Bahasa Pemrograman Python dengan bantuan *library* json.

4.4. Desain *Database*

Untuk melakukan perancangan *crawler*, maka akan perlu dilakukan perancangan *database* untuk menyimpan data yang diambil dari media sosial Twitter.

Data yang diambil dari Twitter adalah data tweet berbahasa Indonesia dan data akun 5 e-commerce di Indonesia. Tabel 4.1 menunjukkan desain *database* untuk menyimpan data posting dari Twitter.

Tabel 4.1 Desain Database Crawler Post Twitter

Atribut	Tipe Data	Penjelasan	Batasan
Id_Tweet	*bigint*	Berisi *id* unik dari setiap *tweet*	Not Null
Topik	*varchar*	Berisi topik atau kata kunci dari *tweet* yang digunakan dalam *crawling*	Not Null
Message	*date*	Berisi tanggal dari dibuatnya sebuah *tweet*	Not Null
Username	*varchar*	Berisi pesan teks dari *tweet* yang di publikasikan	Not Null
Created_time	datetime	Berisikan tanggal sebuah posting twitter diterbitkan	Not Null

Dalam pemilihan atribut pada proses *word embedding* atribut yang akan digunakan adalah *atribut* message saja. Sedangkan untuk proses pelatihan model *Convolutional Neural Network* yang dibutuhkan adalah id_tweet, topik, username dan message.

4.5. Desain Crawler

Untuk melakukan pengambilan data dari media sosial, maka perlu dibuat sebuah crawler. Inti *source code* yang digunakan dari *library* Tweepy dengan basis pemrograman yang digunakan adalah Python. Kemudian dilakukan sebuah kostumisasi agar dapat mengambil data pada banyak akun sekaligus dan dapat mengambil keseluruhan dari text tweet. Dikarenakan adanya Batasan dari API twitter.

4.6. Perancangan Pengumpulan Kata

Data yang akan dikumpulkan merupakan data yang akan digunakan dalam proses word embedding maupun pemodelan *Convolutional Neural Network*. Maka untuk data word embedding akan digunakan *keyword* yang berasal dari kata-kata yang sering digunakan dalam kalimat Bahasa Indonesia. Sedangkan untuk dataset yang digunakan untuk pemodelan *Convolutional Neural Network* akan digunakan *keyword* berdasarkan pada sub bab 3.2.2.

4.7. Perancangan Pra-Pemrosesan Data

Sebelum diproses, data harus terlebih dahulu mengalami proses persiapan data atau pra-pemrosesan. Data pra-pemrosesan menunjukkan tipe-tipe proses yang menggunakan data mentah untuk ditransformasi ke suatu format yang lebih mudah dan efektif untuk kebutuhan rekomendasi agar dapat diolah dengan baik saat pembuatan model pada langkah selanjutnya. Berikut ini merupakan tahapan yang dilakukan pada pemrosesan data berikut.

Dataset yang digunakan pada penelitian kali ini terbagi menjadi 2 bagian. *Dataset* yang pertama akan digunakan pada *word embedding* sedangkan *dataset* yang kedua akan digunakan

sebagai pemodelan pada *Convolutional Neural Network*. Tahap pra-pemrosesan data pada kedua *dataset* sama yang membedakan hanya pada *dataset* pemodelan *Convolutional Neural Network* di mana *dataset* tersebut akan melewati pra-pemrosesan data yang tidak mengandung topik

4.7.1. Perancangan Penggabungan Dataset

Dataset yang dikumpulkan akan digabungkan menjadi satu *database* dikarenakan pengumpulan *dataset* dilakukan di waktu yang berbeda. Penggabungan dataset nantinya akan dibedakan menjadi dua *dataset*, *dataset* yang pertama merupakan *dataset* yang tidak berhubungan dengan topik dan sedangkan *dataset* yang kedua adalah data yang berhubungan dengan topik.

4.7.2. Perancangan Penghapusan Data yang Terduplikasi

Data yang didapatkan dari proses sebelumnya masih terdapat data yang sama, oleh karena itu harus dihapus salah satunya hingga setiap data yang ada merupakan data yang unik.

```
-- Step 1
CREATE TABLE twitter_temp LIKE twitter;
-- Step 2
INSERT INTO twitter_temp SELECT * FROM twitter GROUP BY
message;
-- Step 3
DROP TABLE twitter;

ALTER TABLE twitter_temp
RENAME TO twitter;
```

Gambar 4.1 Query untuk Menghapus Data yang Terduplikasi

Proses penghapusan data yang dilakukan menggunakan *query* diatas yang ditampilkan pada Gambar 4.1.

4.7.3. Perancangan Filtering Bahasa Indonesia

Dalam proses pengumpulan data akan terdapat data yang mengandung bahasa selain Bahasa Indonesia sehingga diperlukan filter terhadap *dataset* agar keseluruhan data

menggunakan Bahasa Indonesia. Dalam proses filtering Bahasa Indonesia *library* yang digunakan adalah Polyglot dimana nilai *confidency* harus bernilai lebih dari 90%.

4.7.4. Perancangan Perubahan Emoticon menjadi Token

Dalam *dataset* yang telah dikumpulkan akan ditemui banyak penggunaan *emoticon* atau yang lebih dikenal dengan kombinasi dari simbol-simbol berbentuk teks tertulis yang disusun sebagai lambang untuk mengungkapkan luapan perasaan.

Penggunaan *emoticon* ini dapat membantu untuk mengetahui lebih jelas ekspresi dalam sebuah kalimat sehingga untuk membedakan *emoticon* dengan teks biasa. Maka akan dilakukan perubahan menjadi sebuah *token* yang merepresentasikan *emoticon* tersebut. *Emoticon* yang akan diubah antara lain :

Tabel 4.2 Daftar Mapping Emoticon

Kategori	Emoticon
Senyum	:)), :), :-)), :-), ((:, (:, ((-:, (-:, =)), =), ^_^
Sedih	:((, :(, :-((, :-(,)):,):,))-:,)-:
Berkedip	;)), ;)
Berduka	:’)), :’), :’((, :’(, ((‘:, (‘:
Terganggu	:/, :\\
Wajah Datar	:|, :-|

4.7.5. Perancangan Perubahan URL dan Mention menjadi Token

Pada akun media sosial Twitter sering ditemui URL dan *Mention*. Tentu saja dalam teks URL dan *Mention* tidak mempunyai arti, Namun dalam struktur kalimat mungkin saja mempunyai maksud tertentu. Dalam penelitian ini juga difokuskan untuk sesedikit mungkin mengurangi isi dari kalimat dalam *dataset*. Sehingga URL dan *Mention* akan diubah bentuknya menjadi *token*.

- Setiap URL yang diidentifikasi sebagai URL dengan awalan "://" akan diubah menjadi <url>
- Setiap *mention* yang diidentifikasi dengan awalan "@" akan diubah menjadi <mention>

4.7.6. Perancangan Penghapusan Huruf Berulang

Pada penulisan sebuah *tweet* sering ditemui dilakukan penulisan kata yang berulang seperti "maaassaaaa" yang sering digunakan sebagai gaya penulisan tweet dalam Bahasa Indonesia sehingga kata tersebut akan di ubah dengan batasan perulangan yang dilakukan adalah membatasi perulangan huruf menjadi maksimal 2 huruf saja. Sehingga kata "maaassaaaa" akan diubah menjadi "maassaa".

4.7.7. Perancangan Penghapusan Tanda Baca, Simbol dan Angka

Dalam data teks yang telah didapatkan dari media sosial, kurang lebih pasti mengandung banyak simbol serta tanda baca yang tidak memiliki arti dan kurang berpengaruh dalam proses *training*. Oleh karena itu, hal ini harus dihilangkan dalam *dataset* sehingga *dataset* menjadi bersih.

Untuk menghilangkan tanda baca dan simbol, maka dilakukan proses *replace* menggunakan *regular expression* dalam bahasa python. Simbol-simbol yang akan dihilangkan adalah :

- Menghapus semua karakter ASCII, seperti "$", "~", "@"

- Menghapus semua karakter non-ASCII, seperti “é”,”'ö”, “ĉ”
- Menghapus semua karakter angka, seperti 1, 2, 3, 4, 5, 6, 7, 8, 9, 0

4.7.8. Perancangan Penghapusan Data yang Tidak Mengandung Topik dalam Teks untuk Data CNN

Pada *dataset* yang akan digunakan pada pemodelan *Convolutional Neural Network* harus dipastikan di dalam teks yang digunakan mengandung kata topik. Jika dalam teks tersebut tidak terdapat data topik, maka teks tersebut tidak dapat digunakan dalam pemodelan.

4.8. Perancangan Pelabelan Data

Dataset yang berasal dari akun 5 e-commerce akan dilakukan proses pelabelan. Pelabelan data akan menggunakan metode *Human Intelligence Task* (HIT) yang diadaptasi dari *Amazon’s Mechanical Turk*. Pada setiap tweet akan dilakukan anotasi oleh tiga orang yang berbeda. Dengan beberapa syarat :

- Satu atau lebih tanggapan tidak memiliki keseluruhan sentimen yang ditandai
- Satu atau lebih tanggapan tidak memiliki sentimen terhadap topik yang ditandai
- Satu atau beberapa tanggapan tampaknya dipilih secara acak

Hasil dari proses *tagging* kemudian akan dilakukan kalkulasi untuk menentukan label final yang akan digunakan dalam teks. Pada proses kalkulasi *tagging* data akan dilakukan dengan proses sebagai berikut :

1. Jika 2 dari 3 *annotator* menyetujui sebuah label yang sama, maka label tersebut akan langsung diterima.

2. Jika tidak, maka langkah awal adalah memetakan label kategoris dengan nilai integer -2, -1, 0, 1, 2. Kemudian dihitung nilai rata-rata dan akhirnya akan dipetakan atau dibulatkan ke nilai integer terdekat. Untuk menyeimbangkan kecenderungan rata-rata untuk menjauh dari -2 dan 2, dan juga untuk memilih 0. Maka tidak dilakukan pembulatan pada ± 0.5 dan ± 1.5, tapi pada ± 0.4 dan ± 1.4 sebagai gantinya.

4.9. Perancangan Pembuatan Model *Word Embedding*

Dataset yang telah dikumpulkan akan direpresentasikan ke dalam vektor menggunakan metode *Word Embedding* yang nantinya akan menghasilkan model melalui proses *Training* kemudian *output* dari model dapat digunakan sebagai kamus *word vector* dalam melakukan proses *embedding* kata. Ada dua jenis model *Word Embedding* yang akan digunakan yaitu Word2Vec dan FastText.

4.9.1. Perancangan *Training* Word2Vec

Tahapan *training* word2vec bertujuan untuk menghasilkan model Word2Vec, kemudian *output* dari model Word2Vec. Untuk melakukan hal ini maka akan dibuat yang dibangun menggunakan *library* Pytorch.

Proses *training* akan dilakukan dengan komposisi parameter berdasarkan pada penelitian yang telah dilakukan oleh Tomas Mikolov [7]. Berikut adalah parameter yang akan digunakan dalam proses *training* Word2Vec :

1. *Learning Algorithm*: Parameter ini merupakan salah satu parameter penting karena akan menentukan kandidat-kandidat yang dihasilkan dari proses prediksi menggunakan model yang telah terbentuk. Terdapat dua *learning algorithm*, yaitu *Continuous Skip-Gram* dan *Continuous Bag of Words*.
2. *Layer Size*: Parameter ini digunakan untuk menentukan dimensionalitas dari sebuah vektor, nilainya mulai dari 0 sampai 1000

3. *Window Size*: Parameter yang menentukan berapa banyak kata setelah dan sesudah dari sebuah kata yang termasuk konteks dari kata tersebut.
4. *Minimun Word Frequency* : Parameter yang menentukan bahwa sebuah kata akan di *training* apabila kata tersebut muncul minimal dalam jumlah yang ditentukan.
5. *Iteration:* Berapa banyak neural net dapat mengubah koefisiennya dalam sekali proses *mini training.*
6. *Epochs*: Angka untuk menentukan berapa banyak proses *training* dilakukan

4.9.2. Perancangan *Training* FastText

Pada model yang akan digunakan untuk FastText yang akan digunakan adalah model yang telah dibuat oleh Bojanowski di mana *dataset* yang digunakan merupakan data dari Wikipedia yang menggunakan Bahasa Indonesia.

4.10. Perancangan *Convolutional Neural Network*

Dalam pembuatan model *Convolutional Neural Network* dilakukan klasifikasi dalam 3 bentuk *subtask* :

- *Subtask* A : Sentiment positif dan negatif
- *Subtask* B : Sentimen positif, negatif dan netral
- *Subtask* C : Sentimen sangat positif, positif, netral, negatif, sangat negatif

Pendekatan yang dilakukan dalam pembuatan model *Convolutional Neural Network* didasarkan pada penelitian yang dilakukan sebelumnya. Penelitian yang dilakukan oleh Ye Zhang berfokus dalam mencari model terbaik dilakukan dengan perubahan parameter pada *Filter Region Size* dan *Feature Maps*, serta variasi model yang akan digunakan pada penelitian ini akan mengikuti model yang diusulkan oleh Yon Kim.

Skenario yang dilakukan berdasarkan pada penelitian yang dilakukan oleh Ye Zhang. Skenario diawali dengan

menggunakan *single filter region size* sebagai *baseline* pada awal skenario setiap *subtask* akan dilakukan proses pelatihan dengan menggunakan *single filter region size* dengan ukuran 1 sampai 10 dengan nilai *feature maps* 100. Dari hasil setiap *single filter region size* yang dilakukan akan didapatkan *single filter region size* dengan nilai yang paling baik sesuai dengan pengukuran yang digunakan dalam setiap *subtask*.

Single filter region size yang memiliki nilai paling baik akan dijadikan dasar dalam melakukan kombinasi angka untuk skenario pada *multiple region size* dengan nilai *feature maps* 100 sampai 300.

Proses *training* yang dilakukan berulang kali dengan komposisi parameter yang berbeda. Berikut adalah parameter yang akan digunakan dalam proses *training Convolutional Neural Network* :

Tabel 4.3 Parameter yang digunakan dalam Model *Convolutinal neural Network*

Parameter	Nilai	Keterangan
Model *Word Embeddings*	1. Model *word2vec Skip-gram* 2. Model Bojanowski	Model yang digunakan untuk mengubah *input* teks menjadi *input* vektor. Model *word2vec* adalah model yang dihasilkan dari tahap pembuatan model *word2vec* sebelumnya
Filter Region Size	1. *Single region size* 1 - 10 2. *Multiple region size*	Menentukan ukuran jumlah dan ukuran dari filter kata yang digunakan dari region *size* pada proses konvolusi input vektor. Untuk *multiple region size* menggunakan

		kombinasi lebih dari 1 nilai
Feature Number	1. 100 2. 200 3. 300	Menentukan jumlah dan ukuran *feature maps* yang dihasilkan dari proses konvolusi dari fitur kata yang masuk, pada nilai *feature maps* yang digunakan hanya sampai angka 300 karena keterbatasan *hardware*.
Subtask	1. *Subtask* A 2. *Subtask* B 3. *Subtask* C	Jenis *subtask* yang dilakukan *training*

Pada *Convolutional Neural Network* juga akan dibuat dua variasi model berdasarkan pada penelitian yang dilakukan oleh Yon Kim. Variasi model yang diambil adalah *non-static* dan *static.* Berikut adalah penjelasan dari kedua variasi model tersebut :

- *Static* : sebuah model akan dilatih dengan proses *word embedding* kemudian diambil nilai vektornya. Semua kata termasuk kata yang tidak dikenal akan diberikan dilakukan inisialisasi nilai vektor secara acak yang kemudian akan disimpan statis. Dimana yang melakukan pembelajaran hanya parameter yang lain.
- *Non-static* : Model akan dilatih dengan menggunakan *word embedding* kemudian diambil nilai vektornya. Setiap nilai vektor akan berubah dan terus disesuaikan selama proses pelatihan berjalan.

Learning rate method yang digunakan pada model *Convolutional Neural Network* adalah Adadelta (*An Adaptive Learning Rate Method*). Adadelta merupakan metode tingkat

pembelajaran yang baik untuk diterapkan dalam berbagai situasi [26].

4.11. Perancangan Evaluasi Pengukuran

Setiap hasil dari percobaan pada model *Convolutional Neural Network* akan dilakukan perhitungan "*Macroaveraged*" untuk setiap *subtask*. *Macroaveraged* dipilih dikarenakan pembobotan perhitungan yang dilakukan pada proses evaluasi diperhitungkan berdasarkan *class* dan akan sama untuk setiap *class*. Sehingga lebih baik dari perhitungan menggunakan *accuracy* di mana bobot yang diberikan akan sama untuk setiap keputusan klasifikasi dalam setiap data tanpa memperhitungkan ketidakseimbangan perhitungan dalam jumlah data setiap *class*.

Selain dari perhitungan *Macroaveraged* sebagai *baseline* untuk setiap *subtask* pengukuran evaluasi berdasarkan nilai akurasi juga akan tetap digunakan untuk melakukan perbandingan antara pengukuran evaluasi menggunakan perhitungan *macro-average* dengan menggunakan *accuracy*. Berikut adalah evaluasi pengukuran yang digunakan dalam penelitian ini.

- *Average Recall (AvgRec)*

Average Recall adalah perhitungan yang melihat suatu nilai diprediksi dengan benar. Dengan mengukur tingkat keberhasilan suatu sistem dalam menentukan nilai kebenaran dari suatu label.

Rumus *Average Recall*

$$AvgRec = \frac{1}{\mathrm{C}} \left(R^P + R^N + R^U\right)$$

Dimana :

C : Jumlah label di dalam *class*

R^P : Recall terhadap data positif

R^N : Recall terhadap data negatif

R^U : Recall terhadap data netral

Kelebihan menggunakan AvgRec dibandingkan dengan "standar" akurasi adalah lebih baik dalam proses yang melibatkan class imbalance.

- *Macro-average Mean Absolute Error (MAE^M)*

MAE^M adalah pengembangan dari perhitungan nilai *macro* berdasarkan *recall*. Pengukuran evaluasi ini melakukan pengukuran berdasarkan label, kemudian mengambil nilai *error* rata-rata yang dihasilkan. Pengukuran ini sangat baik untuk mengukur data *imbalance*.

Rumus *Macro Mean Absolute Error*

$$MAE^M(h, Te) = \frac{1}{|C|} \sum_{j=1}^{|C|} \frac{1}{|Te_j|} \sum_{x_i \in Te_j} |h(x_i) - y_i|$$

Dimana :

C : Jumlah label di dalam *class*

y_i : Label Actual pada item x_i

$h(x_i)$: Predicted Label

Te_j : Class label data actual

- *F1-Score*

F1-Score merupakan pengukuran akurasi. Evaluasi pengukuran ini mempertimbangkan nilai baik dari presisi maupun dari recall untuk menghitung skor. F1-Score adalah rata-rata penggabungan dari presisi dan recall.

Rumus *F1-Score*

$$F_1{}^{PN} = \frac{1}{\mathrm{C}} \left(F_1{}^{P} + F_1{}^{N}\right)$$

Dimana :

C : Jumlah label di dalam *class* selain *class* netral

F_1^P : Nilai F1-Score pada *class positive*

F_1^N : Nilai F1-Score pada *class negative*

- *Standar Deviation*

Standar Deviation adalah ukuran yang digunakan untuk mengukur jumlah variasi atau penyebaran dari sejumlah data. Berikut merupakan rumus perhitungan yang dilakukan untuk mencari nilai standar deviation.

Rumus Standar Deviation :

$$s = \sqrt{\frac{n\ \sum_{i=1}^{n} {x_i}^2 - (\sum_{i=1}^{n} x_1)^2}{n(n-1)}}$$

Dimana :

S : standar deviasi

n : ukuran sampel

x_i : nilai x ke-i

Standar Deviation yang rendah menunjukkan bahwa titik data cenderung mendekati nilai rata-rata dari kumpulan data, sementara standar deviasi yang tinggi menunjukkan bahwa titik data tersebar di rentang nilai yang lebih luas.

BAB V
IMPLEMENTASI

Pada bab ini, akan dijelaskan mengenai implementasi dari perancangan yang telah dilakukan sesuai dengan metode pengembangan yang dibuat. Bagian implementasi akan menjelaskan mengenai lingkungan implementasi, pembuatan fitur-fitur aplikasi dalam bentuk kode, serta pengujian aplikasi.

5.1. Persiapan Implementasi

Penelitian ini menggunakan perangkat keras dan lunak untuk membantu proses pengerjaan. Untuk spesifikasi perangkat keras yang digunakan dapat dilihat pada Tabel 5.1.

Tabel 5.1 Spesifikasi Perangkat Keras Laptop

Nama Perangkat	Laptop
Processor	Intel® Core™ i5-7200 CPU @ 2.7GHz
Memory	8192 MB RAM
Sistem Operasi	Linux Ubuntu 16.10
Arsitektur Sistem	64-bit *Operating System, x64-based processor*

Aplikasi dikembangkan dengan menggunakan beberapa teknologi seperti code editor, database, bahasa pemrograman, dan library yang disajikan dalam Tabel 5.2.

Tabel 5.2. Daftar Library

Bahasa Pemrograman	Python
Database	MySQL
Code Editor (IDE)	Sublime Text, Visual Studio Code
Virtual Environment	Anaconda
Library	• Tweepy • Jsonpickle • Gensim • Torchtecxt

	• Emot • Matplotlib • Ijson • Numpy • Pandas • Json • Pytorch

5.2. Pembuatan Filtering Bahasa

```
try:
  message_value = value
  obj_value = id_value, message_value
  language_code = []
  for language in Detector(message_value).languages:
    code = language.code
    confiden_value = language.confidence
    type_body = code, confiden_value
    language_code.append(type_body)
  for y in range(len(language_code)):
    if      language_code[y][0]      ==      "id"      and
language_code[y][1] >= 90.0:
        check_languange_id = True
  if check_languange_id == True:
    id_languange.write('INSERT   INTO   twitter   VALUES
{};'.format(str(obj_value)))
    id_languange.write('\n')
    print('Terdeteksi Bahasa Indonesa')
  else:
    other_languange.write('INSERT  INTO  twitter  VALUES
{};'.format(str(obj_value)))
    other_languange.write('\n')
    print("Terdeteksi     Tidak     Mengandung     Bahasa
Indonesia")

except:
  error_languange.write('INSERT   INTO   twitter   VALUES
{};'.format(str(obj_value)))
  error_languange.write('\n')
  print("ERROR !!!")
```

Kode 5.1 Proses Filtering Bahasa dengan Menggunakan Polyglot

Kode 5.1 menjelaskan proses filtering bahasa menggunakan *library* polyglot. Sebelum melakukan filtering bahasa *dataset pada database* dikeluarkan terlebih dahulu dalam bentuk file json. Pada *library* polyglot terdapat fungsi *Detector* di mana akan dideteksi bahasa yang dikandung dalam tweet, serta menunjukkan nilai *confidency* yang menunjukkan

seberapa besar kandungan bahasa dalam tweet tersebut. Nilai *confidency* diatur pada angka 90 untuk memastikan data yang masuk adalah data yang memiliki kandungan Bahasa Indonesia yang sangat tinggi.

```
with open(filename, 'r', encoding='latin1') as fd:
  parser = ijson.parse(fd)
  for prefix, event, value in parser:
    check_languange_id = False

    if (prefix, event) == ('RECORDS.item.id_tweet',
'string'):
      id_value = value
    if (prefix, event) == ('RECORDS.item.message',
'string'):
      message_value = value
```

Kode 5.2 Proses Membaca File Json dengan Library Ijson

Kode 5.2 menjelaskan proses *reading* data json. Pada python proses *read* data json dengan menggunakan *library* json maupun pandas memiliki batasan terhadap jumlah data yang besar maka digunakan *library* ijson.

5.3. Pembuatan Crawler

Proses yang dilakukan dalam pembuatan Twitter *crawler* digunakan *library* Tweepy yang dikhususkan untuk melakukan proses *crawling* pada *social media* Twitter. *Crawler* yang dibuat akan melakukan pengambilan data dari twitter dengan menggunakan *keyword*.

Dilihat dari sisi waktu pengambilan data aturan yang digunakan dari twitter API hanya memperbolehkan mengakses *posting* twitter hingga satu minggu sebelum hari pengambilan data. Jadi, semisal mengambil data yang lebih dari satu minggu sebelumnya maka Twitter API akan mengembalikan nilai *null*. Namun, pada library tweepy data akan diambil dari data paling

baru hingga satu minggu sebelumnya sesuai dengan ketersedian data twitter.

```
# Memasukkan API_KEY dan API_SECRET dengan key dan secret
di aplikasi twitter.
auth = tweepy.AppAuthHandler('API_KEY','API_SECRET')
api    =    tweepy.API(auth,    wait_on_rate_limit=True,
wait_on_rate_limit_notify=True)
if (not api):
    print ("Can't Authenticate")
    sys.exit(-1)
```

Kode 5.3 Autentikasi Twitter

Kode 5.3 menjelaskan proses kode untuk membuat sebuah instance yang menghubungkan client dengan twitter API menggunakan API_Key dan API_Secret yang telah didapatkan dari website Twitter API. Kode diatas juga untuk memastikan autentikasinya apabila koneksi sudah dapat terhubung dengan API Twitter.

Kode 5.4 Pengambilan Keyword

Kode 5.4 menjelaskan proses yang digunakan untuk melakukan pengambilan keyword dari file yang berisi keyword yang dapat menghasilkan data Bahasa Indonesia, juga akun 5 e-commerce yang akan digunakan dalam model CNNs. Pada kode

```
with open(listname) as listsearch:
    for line in listsearch:
        print('Keyword : {}'.format(line.strip()))
        searchQuery = line.strip()  # keyword yang ingin
dicari
        maxTweets = 10000000 # Maximun nilai tweet
        tweetsPerQry = 100  # jumlah max tweet yang dapat
diambil API
        fName = ''+st+'_'+ line.strip()[1:] +'.json' #
membuka file json
        with open('list_dir.txt', 'a') as listdir:
            listdir.write(fName + '\n')
```

di atas juga membatasi jumlah maximal tweet yang diambil untuk membatasi ukuran file yang diambil.

```
sinceId = None

tweetCount = 0
print("Downloading max {0} tweets".format(maxTweets))
with open(fName, 'w') as f:
  while tweetCount < maxTweets:
    try:
      if (not sinceId):
        new_tweets      =      api.search(q=searchQuery,
count=tweetsPerQry, tweet_mode='extended')
      else:
        new_tweets      =      api.search(q=searchQuery,
        count=tweetsPerQry,            since_id=sinceId,
        tweet_mode='extended')

      if not new_tweets:
        print("No more tweets found")
        break
      for tweet in new_tweets:
        f.write(jsonpickle.encode(tweet._json,
unpicklable=False) + '\n')
      tweetCount += len(new_tweets)
      print("Downloaded {0} tweets".format(tweetCount))
      max_id = new_tweets[-1].id
    except tweepy.TweepError as e:
      # Exit jika ada error
      print("some error : " + str(e))
      break
```

Kode 5.4 Pengambilan Data Twitter sesuai dengan Keyword

Kode 5.4 menunjukkan script yang digunakan untuk melakukan pengambilan data twitter sesuai dengan keyword pencarian yang ada. Pada implementasi program ini akan dilakukan pencarian semua tweet yang mengandung keyword. Kemudian pada proses ini juga menghasilkan sebuah file json Twitter yang disimpan untuk keperluan berikutnya. Proses ini akan berulang hingga hasil yang didapatkan dari pengambilan

data twitter tidak menemukan tweet lagi. Proses ini menggunakan *search API twitter*,

5.4. Pra-Pemrosesan Dataset

5.4.1. Penggabungan Dataset

Data yang didapat dari hasil *crawling* yang terpisah akan digabungkan dalam satu file sql.

```
listname = 'list_dir.txt'

with open(listname) as listsearch:
  for line in listsearch:
    namedir = line.strip()[:-5]
    topic = line.strip()[9:-5]
    with open('READ_ALL.sql','a') as saveFile:
      with open(line.strip()) as json_data:
        for line in json_data:
          try:
            tweet = json.loads(line.replace('\r\n', ''))
            id = tweet['id']
            username = tweet['user']['screen_name']
            message = tweet['full_text']
            ts   =   time.strftime('%Y-%m-%d   %H:%M:%S',
time.strptime(tweet['created_at'],'%a   %b   %d   %H:%M:%S
+0000 %Y'))

            file_obj = id,topic,message,username,ts
            print('INSERT     INTO     twitter     VALUES
{};'.format(file_obj))
            saveFile.write('INSERT  INTO  twitter  VALUES
{};'.format(str(file_obj)))
            saveFile.write('\n')
            continue
        print('\n'     +     'COMPLETE     READ     :     {}
'.format(namedir) + '\n')
```

Kode 5.5 Menggabungkan File Twitter menjadi Satu File SQL

Kode 5.5 menjelaskan proses penggabungan *dataset* yang diambil dari hasil file json proses *crawler*. Dari file json yang didapat akan diambil beberapa atribut yang mendukung antara lain id_tweet, text, username dan timestamp. Hasil pembacaan dari file json akan dituliskan dalam bentuk file sql. Jika berhasil dieksekusi maka program akan mengembalikan pesan berhasil. Sehingga nantinya akan didapatkan file sql yang akan dimasukkan ke dalam *database* yang memuat seluruh data gabungan dari *dataset* file json yang ada.

5.4.2. Penghapusan Kata yang Terduplikasi

Proses yang di lakukan untuk menghapus kata yang terduplikasi, dilakukan dengan menggunakan bantuan *query database*. Semua data yang telah dikumpulkan dimasukkan ke dalam satu tabel. Kemudian dibuat *query* untuk mengambil data dari tabel tersebut dan memastikan bahwa data yang duplikat hanya diambil salah satu saja.

```
-- Step 1
CREATE TABLE twitter_temp LIKE twitter;
-- Step 2
INSERT INTO twitter_temp SELECT * FROM twitter GROUP BY
message;
-- Step 3
DROP TABLE twitter;

ALTER TABLE twitter_temp
RENAME TO twitter;
```

Kode 5.6 Query Database untuk Menghapus Kata Duplikat

Pada Kode 5.6 menjelaskan proses menggunakan *query database* yang memastikan bahwa setiap nilai tidak ada yang terduplikasi. Proses ini untuk memastikan bahwa tidak ada tweet yang terduplikasi.

5.4.3. Penghapusan Tanda Baca, Simbol dan Angka

```
def clean_str(self, string):
  string = re.sub(r"\.", " . ", string)
  string = re.sub(r",", " , ", string)
  string = re.sub(r":", " : ", string)
  string = re.sub(r";", " ; ", string)
  string = re.sub(r"!", " ! ", string)
  string = re.sub(r"\?", " ? ", string)
  string = re.sub(r"\(", " ( ", string)
  string = re.sub(r"\)", " ) ", string)
  string = re.sub(r"#", " <hash_tag> ", string)
  string = re.sub(r"\[", " [ ", string)
  string = re.sub(r"\]", " ] ", string)
  string = re.sub(r"\'m ", " \'m ", string)
  string = re.sub(r"\'s ", " \'s ", string)
  string = re.sub(r"\'re ", " \'re ", string)
  string = re.sub(r"\'ll ", " \'ll ", string)
  string = re.sub(r"\'d ", " \'d ", string)
  string = re.sub(r"\'ve ", " \'ve ", string)
  string = re.sub(r"n\'t ", " n\'t ", string)
  string = re.sub(r"[^A-Za-z().,<_>!?\'`]", " ", string)
  string = re.sub(r"\s{2,}", " ", string)
  return string.strip()
```

Kode 5.7 Proses Menghapus Tanda Baca, Simbol dan Angka

Kode 5.7 menunjukkan proses penghapusan tanda baca dan simbol didasarkan pada pre-processing yang dilakukan oleh Yon Kim dengan melakukan beberapa modifikasi sesuai dengan kebutuhan dataset.

Modifikasi yang dilakukan pada pre-processing adalah dengan menghapus angka yang terdapat pada dataset.

5.4.4. Menghapus Huruf yang Berulang

```
 def replace_mult_occurences(self, string):
return re.sub(r'(.)\1{2,}', r'\1\1', string)
```

Kode 5.8 Proses Menghapus Huruf yang Berulang lebih dari 2 kali

Kode 5.8 menjelaskan proses penghapusan huruf yang berulang. Pada proses ini dibatasi untuk setiap huruf hanya boleh berulang dua kali.

5.4.5. Mengubah *Emoticon* menjadi Token String

Pada *dataset* terdapat penggunaan *emoticon* dalam penulisan teks. *Emoticon* dapat membantu untuk mengetahui perasaan penulis tweet dalam menulis sebuah tweet. Untuk membedakan tweet teks biasa dengan *emoticon* maka *emoticon* akan diubah menjadi token.

```
def replace_emoticons(self, string):
  # Senyum
  string = string.replace(':))', ' <senyum_senyum> ')
  string = string.replace(':)', ' <senyum> ')
  string = string.replace(':-))', ' <senyum_senyum> ')
  string = string.replace(':-)', ' <senyum> ')
  string = string.replace('((:', ' <senyum_senyum> ')
  string = string.replace('(:', ' <senyum> ')
  string = string.replace('((-:', ' <senyum_senyum> ')
  string = string.replace('(-:', ' <senyum> ')
  string = string.replace('=))', ' <senyum_senyum> ')
  string = string.replace('=)', ' <senyum> ')
  string = string.replace('^_^', ' <senyum> ')

  # Sedih
  string = string.replace(':((', ' <sedih_sedih> ')
  string = string.replace(':(', ' <sedih> ')
  string = string.replace(':-((', ' <sedih_sedih> ')
  string = string.replace(':-(', ' <sedih> ')
  string = string.replace(')):', ' <sedih_sedih> ')
  string = string.replace('):', ' <sedih> ')
  string = string.replace('))-:', ' <sedih_sedih> ')
  string = string.replace(')-:', ' <sedih> ')

 # Berkedip
  string = string.replace(';))', ' <senyum_berkedip> ')
  string = string.replace(';)', ' <senyum_berkedip> ')
  # Tears
  string = string.replace(":'))", ' <menangis_bahagia>
')
  string = string.replace(":')", ' <menangis_bahagia> ')
  string = string.replace(":'((", ' <menangis_sedih> ')
  string = string.replace(":'(", ' <menangis_sedih> ')
  string = string.replace("((':", ' <menangis_bahagia>
')
  string = string.replace("(':", ' <menangis_bahagia> ')
```

```
  # Some annoyed
  string = string.replace(':/', ' <terganggu> ')
  string = string.replace(':\\', ' <terganggu> ')

  # Straight face
  string = string.replace(':|', ' <muka_datar> ')
  string = string.replace(':-|', ' <muka_datar> ')

  string = ' '.join([self.replace_token_emoticon(token)
                     for token in string.split()])
  return string
```

Kode 5.9 Perubahan Emoticon Menjadi Token String

Kode 5.9 menunjukkan proses mengubah *emoticon* yang kemungkinan akan terdapat pada teks menjadi token yang sesuai.

5.4.6. Mengubah URL dan Mention menjadi Token String

```
def replace_URL(self, string):
  tokens  = ['<url>' if '://' in token else token
             for token in string.split()]
  return ' '.join(tokens)

def replace_mention(self, string):
  tokens  = ['<mention>' if token.startswith('@') else
token
```

Kode 5.10 Perubahan URL dan Mention menjadi Token String

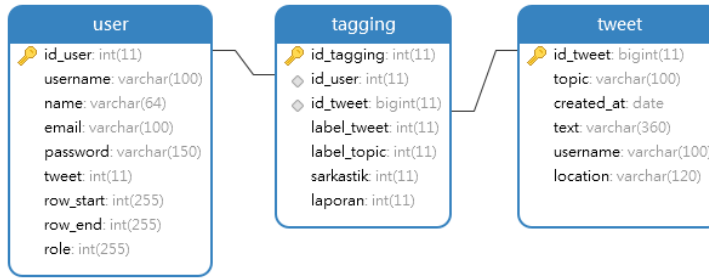
Kode 5.10 menunjukkan proses mengubah URL pada *method* dengan melakukan indikasi untuk setiap kata yang diawali dengan "://" akan diubah menjadi <url>. Sedangkan, pada *method* merupakan proses mengubah setiap mention menjadi token string dengan indikasi untuk setiap kata yang diawali dengan "@" akan diubah menjadi <mention>.

5.5. Pembuatan Pelabelan Data

Data yang akan diberi label diambil dari kumpulan data *twitter* hasil dari pra-pemrosesan sebelumnya. Data diambil secara acak berdasarkan topik yang telah ditentukan. Penentuan topik berdasarkan pada sub bab 3.2.2.

5.1.1 Pembuatan Basis Data Anotasi

Langkah selanjutnya adalah membuat basis data untuk anotasi. Basis data tersebut terdiri dari 3 tabel, yaitu *user*, *tweet*, dan *tagging*. Skema basis data yang dibuat dapat dilihat pada Gambar 5.1.



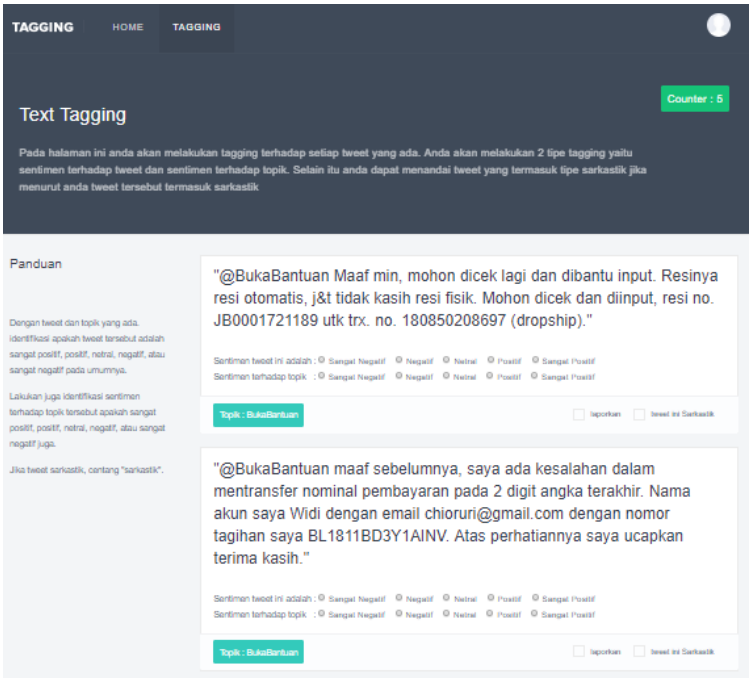
Gambar 5.1 Skema Basis Data Anotasi

Tabel *tweet* adalah tabel yang berisi data *twitter* yang siap untuk diberi label hasil dari pemilihan data sebelumnya. Tabel *user* berisi data untuk anotator yang digunakan untuk otentikasi dan rekap kinerja. Hasil dari pemberian label oleh anotator akan disimpan pada tabel *tagging*.

5.1.2 Pembuatan Aplikasi Anotasi

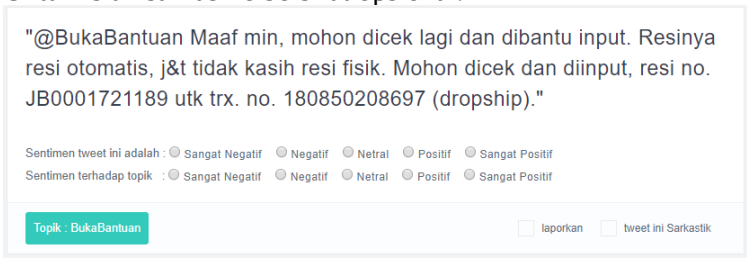
Aplikasi berbasis web untuk anotasi data dibangun menggunakan *framework* CodeIgniter. CodeIgniter dipilih pada penelitian ini karena kemudahan dalam pengembangan dan pemeliharaannya. Aplikasi anotasi hanya memiliki 1 tampilan antar muka. Untuk memudahkan anotator, terdapat bantuan berupa panduan dalam pemberian label terhadap *tweet* yang

ditampilkan. Desain antar muka dari aplikasi dapat dilihat pada Gambar 5.2.



Gambar 5.2 Desain Antar Muka Aplikasi Anotasi

Bagian *card* pada desain antar muka akan memuat *tweet* dan topik yang akan diberi label. terdapat 2 isian yang wajib di isi oleh anotator yaitu label *tweet* dan label terhadap topik. Untuk isian sarkasme bersifat opsional.



Gambar 5.3 Bagian *Card* pada Aplikasi Anotasi

Aplikasi dan basis data anotasi akan diunggah ke VPS untuk dapat diakses secara *online* oleh anotator.

5.6. Pembuatan Model Word2Vec

Proses pembuatan model *Word2Vec* ini akan dibuat menggunakan *library gensim*. Proses ini menggunakan algoritma *Skip-gram*

```
def __init__(self):
  self.size = 300
  self.num_features = 300
  self.num_workers = multiprocessing.cpu_count()
  # sg = 0 is CBOW model, sg = 1 is skip-gram
  self.sg = 1
  self.iter = 1
  self.window = 5
  self.seed = 1
  self.min_word_count = 5
  self.context_size = 7
  self.alpha = 0.025
  self.downsampling = 1e-3

def model(self):

  model_w2v = Word2Vec(size=self.size,
alpha=self.alpha, sg = self.sg, min_count =
self.min_word_count,
                       window = self.window, iter =
self.iter)
  return model_w2v
```

Kode 5.11 Inisialisasi Parameter Model Word2Vec

Kode 5.11 menjelaskan *method* yang memasukkan nilai parameter pada model. Kemudian pada kode tersebut mendefinisikan pemanggilan method Word2Vec yang berisikan proses pembelajaran yang dilakukan. Selanjutnya untuk menentukan *learning algorithm* ditentukan dengan parameter sg

dimana nilai 0 untuk learning algorithm *CBOW* dan 1 untuk learning algorithm *Skip-Gram*.

```
print("\nLoading Word Embedding RAW data ...")
file_read =
ReadDataWordEmbed('data/twitter_indo_new.json')
sentences = []
print("\nInsert into sentences")
# count = 0
for id, message in file_read:
  if len(message) > 0:
    sentences.append(sentence_to_wordlist(message))

print("\nStart Build Vocab")
model_w2v = Word2vec()
model_w2v = model_w2v.model()
model_w2v.build_vocab(sentences)
print("\nWord2Vec vocabulary length :
{}".format(len(model_w2v.wv.vocab)))
token_count = sum([len(sentence) for sentence in
sentences])
print("\nCorpus contains {0:,}
tokens".format(token_count))
print("\nBuilding word2vec model...")
start = time.time()
model_w2v.train(sentences, total_examples =
token_count, epochs=model_w2v.iter)
end = time.time()

print("word2vec training done in {} seconds".format(end
- start))

model_w2v.wv.save_word2vec_format("socmed_w2v_model_3.b
in", binary=False)

print('Training Word2Vec Done')
```

Kode 5.12 Training pada Word2Vec

Kode 5.12 menjelaskan proses training yang dilakukan pada pemodelan *Word Embedding* dengan menggunakan algoritma Word2Vec. Setiap data akan dibaca dan dipisahkan per kata. Kemudian data akan masuk dalam suatu list dan dilakukan perhitungan vocab yang didapatkan dari dataset training model. Kemudian setiap kata akan diberikan nilai vektor dengan besar dimensi yang telah ditentukan.

```
def sentence_to_wordlist(raw):
  words = raw.split()
  return words
```

Kode 5.13 *Method Tokenize Word*

Kode 5.13 menjelaskan proses *tokenize* data di mana sebelum dilakukan *embedding* pada pemodelan Word2Vec, data akan dipisahkan per kata.

5.7. Pembuatan Model Convolutional Neural Network

Kode program yang dibuat untuk melakukan *training* model CNN menggunakan *library* Pytorch. *library* tersebut akan memanfaatkan kemampuan komputasi arsitektur *cuda* pada GPU untuk melakukan komputasi secara pararel.

Kumpulan data yang digunakan untuk *training* model CNN adalah data *tweet* berlabel luaran tahap anotasi data. Untuk menyiapkan data pembuatan model, kumpulan data *tweet* berlabel pada basis data aplikasi anotasi, diekspor ke dalam bentuk file txt.

Model *convolutional neural network* ini rata-rata performa dikalkulasi menggunakan 10-fold cross validation. Proses pembuatan Model CNN ini menggunaka 4 file dengan penjelasan sebagai berikut :

- Model.py

```
assert (len(self.feature_num) == len(self.kernel_width))
  self.kernel_num = len(self.kernel_width)

  self.embeddings     =     nn.Embedding(self.embed_num,
self.embed_dim,                max_norm=self.norm_limit,
norm_type=2,padding_idx=1)
  if self.embed_mode == 'non-static' or self.embed_mode
== 'static' or self.embed_mode == 'multichannel':

self.embeddings.weight.data.copy_(torch.from_numpy(init
ial_embeddings))
    if self.embed_mode == 'static':
      self.embeddings.weight.requires_grad = False
    elif self.embed_mode == 'multichannel':
      self.embeddings2  =  nn.Embedding(self.embed_num,
self.embed_dim,                max_norm=self.norm_limit,
norm_type=2,padding_idx=1)

self.embeddings2.weight.data.copy_(torch.from_numpy(ini
tial_embeddings))
      self.embeddings2.weight.requires_grad = False
      self.channel_in = 2
  self.convs                                          =
nn.ModuleList([nn.Conv1d(self.channel_in,
self.feature_num[i],

self.embed_dim*self.kernel_width[i],
stride=self.embed_dim)
                                  for       i       in
range(self.kernel_num)])

  self.linear     =     nn.Linear(sum(self.feature_num),
self.label_num)
```

Kode 5.14 Pembuatan Variasi Model CNN

Kode 5.14 diawali dengan memastikan jumlah *filter region size* sama banyaknya dengan jumlah *feature maps*. Jika, jumlahnya tidak sama maka proses *training* akan keluar karena di anggap *error*. Dalam penelitian ini hanya difokuskan pada variasi model *non-static* dan *static*. Perbedaan paling mendasar dari kedua variasi model ini adalah pada model *static* tidak diberikan kesempatan oleh model untuk melakukan pembelajaran terhadap kata dan melakukan pembaharuan nilai vector kata tersebut, sehingga pada kode ditambahkana self.embeddings.weight.requires_grad = False. Baris kode ini membuat model tidak akan melakukan update pada pembobotan kata yang dilakukan menggunakan word embedding seiring dengan berjalannya proses training.

```
def forward(self, input):
  batch_width = input.size()[1]
  x      =      self.embeddings(input).view(-1,      1,
self.embed_dim*batch_width)
  if self.embed_mode == 'multichannel':
    x2     =     self.embeddings2(input).view(-1,     1,
self.embed_dim*batch_width)
    x = torch.cat((x, x2), 1)

  conv_results                                          =
[F.max_pool1d(F.relu(self.convs[i](x)),  batch_width  -
self.kernel_width[i]           +           1).view(-1,
self.feature_num[i])
                for i in range(len(self.feature_num))]

  x = torch.cat(conv_results, 1)
  x      =      F.dropout(x,      p=self.dropout_rate,
training=self.training)
  x = self.linear(x)
  return x
```

Kode 5.15 Method Forward Pembelajaran Model CNN

Kode 5.15 menunjukkan method forward yang berfungsi sebagai proses pembelajaran pada model yang akan

mendefinisikan setiap proses konvolusi pada model. Pada kode tersebut akan dimulai dengan melakukan proses *embeeding* kata. Proses ini akan melakukan perubahan kata menjadi vektor. Jumlah kata yang dirubah menjadi vector disesuaikan dengan jumlah kata pada kalimat yang masuk dikalikan dengan jumlah dimensi satu vektor kata. Selanjutnya hasil dari proses perkalian tersebut akan dikalikan lagi dengan jumlah kata yang masuk dikalikan dengan jumlah *batch*.

Proses selanjutnya akan menghasilkan nilai *output* dari proses penggunaan *feature maps*. Nilai dari *feature maps* yang telah ditentukan akan dikalikan dengan nilai *batch* kemudian dilakukan perkalian lagi terhadap jumlah kata dikurangi dengan nilai *filter region size* yang ditentukan. Proses ini akan berulang sesuai dengan jumlah *filter region size* yang diguanakan. Proses aktivasi dalam pemodelan ini akan dilakukan dengan fungsi *ReLu* dimana prosesnya adalah hanya merubah angka negatif menjadi 0.

- Read_embed.py

```
def read_social_media_model(self):
  embeddings_godin = 'model/socmed_w2v_model_2.vec'
  embeddings_path = embeddings_godin

  print('Loading Indonesian Social Media Word Embedding
Model...')

  start = time.time()
  word2vec_model                                          =
KeyedVectors.load_word2vec_format(embeddings_path,
binary=False, unicode_errors='ignore')
  end = time.time()
  print("Indonesian Social Media Word Embedding Model
loading done in {} seconds".format(end - start))
  self.word2vec = word2vec_model
  self.embed_dim = 300
```

Kode 5.16 Method untuk membaca model Word Embedding

Kode 5.16 menjelaskan proses untuk membaca dan memuat *vector* kata yang terdapat pada *output* data pelatihan *word embedding* baik untuk model *Word2Vec* maupun model *FastText* dengan penentuan jumlah nilai dimensi vektor yang disesuaikan dengan output dari model *word embedding* yang dihasilkan. Parameter yang dibutuhkan dalam proses tersebut yaitu lokasi output model *word embedding* disimpan, metode *format binary* dan pengabaian *error* yang terjadi didalam proses *encoding* karakter.

- Read_data.py

```
def read_dataset(self):
if args == 'subtaskA':
    file_name = 'data/E-commerce_SubtaskA.txt'
  else:
    file_name = 'data/E-commerce_SubtaskB_SubtaskC.txt'
  indo_ecommerce_data = ReadDataTopik(file_name, args)
  twt_id_field   =   torchtext.data.Field(use_vocab=False,
sequential=False)
  label_field = torchtext.data.Field(sequential=False)
  text_field = torchtext.data.Field()
  fields    =    [('twt_id',    twt_id_field),    ('label',
label_field), ('text', text_field)]
  self.fields = fields
  if args == 'subtaskA':
    examples  =  [torchtext.data.Example.fromlist([twt_id,
self.polarity_to_label_subtaskA(polarity), text], fields)
 for twt_id, polarity, text in indo_ecommerce_data]
  elif args == 'subtaskB':
    examples  =  [torchtext.data.Example.fromlist([twt_id,
self.polarity_to_label_subtaskB(polarity), text], fields)
              for     twt_id,     polarity,     text     in
indo_ecommerce_data]
  else:
    examples  =  [torchtext.data.Example.fromlist([twt_id,
self.polarity_to_label_subtaskC(polarity), text], fields)
              for     twt_id,     polarity,     text     in
indo_ecommerce_data]
```

Kode 5.17 Method untuk membaca dataset dan melakukan perubahan polarity label

Kode 5.17 menjelaskan proses dari pembacaan dataset yang akan digunakan pada proses pelatihan dimana dataset tersebut dibagi menjadi tiga bagian. Bagian yang memuat

pelabelan data akan diubah sesuai dengan *polarity* dari label tersebut seusai dengan jumlah label yang diberikan pada tiap *subtask*. Sebelum dataset dilakukan pemanggilan terhadap kelas ReadDataTopik yang berada pada file prepare_data.py yang berisi *method* yang digunakan dalam proses *cleansing* data. Pada kode program tersebut membuat variable baru bernama *field* yang memanfaatkan library *torchtext*. Variabel ini akan berisikan id dari setiap tweet, label pada tweet dan text yang digunakan dalam tweet tersebut. Proses ini akan diulangi sebanyak jumlah data yang digunakan dalam proses *training*. Semua proses tersebut disimpan di dalam variabel *examples*.

- Prepare_data.py

```
def __iter__(self):
  with open(self.file_name, 'r', encoding='latin1') as
f:
    for line in f:
      id_tweet,    topik,    lbl_sntmnt,    lbl_topik,
lbl_sarkas, message = self.divide_line(line)
      message = self.replace_URL(message)
      message = html.unescape(message)
      message = message.replace('"', ' <kutip> ')
      message = self.replace_mention(message)
      message = self.replace_mult_occurences(message)
      message = message.replace('..', ' <elipsis> ')
      message = self.replace_emoticons(message)
      message = self.clean_str(message)
      message = self.replace_mention(message)
      message = message.lower()
      yield id_tweet, lbl_sntmnt, message
```

Kode 5.18 Proses perulangan pembacaan data

Kode 5.18 Menunjukkan pre-processing yang dilakukan pada dataset dengan menggunakan proses yang sama dengan dilakukan pada word embedding yang kode programnya dapat dilihat pada sub bab 5.4.

```
def create_fold_embeddings(self, embeddings, args):
  emb_init_values = []
  for i in range(self.idx_to_vocab.__len__()):
    word = self.idx_to_vocab.get(i)
    count_word += 1
    if word == '<unk>':
      emb_init_values.append(np.random.uniform(-0.25,
0.25, embeddings.embed_dim).astype('float32'))
      unk_value += 1
    elif word == '<pad>':

emb_init_values.append(np.zeros(embeddings.embed_dim).a
stype('float32'))
      pad_count += 1
    elif word in embeddings.word2vec.vocab:

emb_init_values.append(embeddings.word2vec.word_vec(wor
d))
      word_masuk += 1
    else:
      emb_init_values.append(np.random.uniform(-0.25,
0.25, embeddings.embed_dim).astype('float32'))
      no_embed += 1

  self.emb_init_values = emb_init_values
  return emb_init_values
```

Kode 5.19 Proses Inisialisasi Vektor Kata

Kode 5.19 menunjukkan proses inisialisasi *vector* kata dari dataset untuk proses pelatihan model CNN dengan melihat kesesuaian kata pada kamus kata yang tersedia. Jika kata terdapat pada kamus kata pada *word embedding* maka nilai akan disamakan dengan vektor kata yang terdapat dalam kamus. Namun, jika kata tidak terdapat pada kamus kata maka akan diambil nilai secara *random* dengan besar dimensi kata disesuaikan dengan model *word embedding* yang digunakan.

- Train.py

```
def cross_validate(fold, data, embeddings, args):
  split_width = int(ceil(len(data.examples)/fold))

  for i in range(fold):
    print('###============FOLD
[{}]============###'.format(i + 1))
    train_examples  = data.examples[:]
    del
train_examples[i*split_width:min(len(data.examples),
(i+1)*split_width)]
    test_examples                                          =
data.examples[i*split_width:min(len(data.examples),
(i+1)*split_width)]
    total_len = len(data.examples)
    train_len = len(train_examples)
    test_len  = len(test_examples)
    train_counts = defaultdict(int)
    test_counts  = defaultdict(int)
    for example in train_examples:
      train_counts[example.label] += 1
    for example in test_examples:
      test_counts[example.label] += 1
    high_pos  = train_counts['HighlyPositive']
    pos       = train_counts['Positive']
    neu       = train_counts['Neutral']
    neg       = train_counts['Negative']
    high_neg  = train_counts['HighlyNegative']

    high_pos  = test_counts['HighlyPositive']
    pos       = test_counts['Positive']
    neu       = test_counts['Neutral']
    neg       = test_counts['Negative']
    high_neg  = test_counts['HighlyNegative']
```

Kode 5.20 Proses Pembagain Data Testing dan Data Training

Kode 5.20 menjelaskan pembagian data pada proses *training* dan *testing*. Data akan dibagi menjadi 10 bagian untuk melakukan proses *10-fold cross valdation*, dimana pada kode program tersebut data yang akan menjadi data testing tidak akan menjadi data training. Setelah itu akan dilakukan perhitungan penyebaran datanya baik untuk data testing maupun data training. Dalam penelitian ini menggunakan 10 *fold* data, sehingga semisal terdapat 10,000 data/tweet maka disetiap bagian data akan dilakukan pembagian dengan proporsi training data 9,000 dan testing data 1,000.

```
text_field  = None
    label_field = None
    for field_name, field_object in fields:
      if field_name == 'text':
        text_field = field_object
      elif field_name == 'label':
        label_field = field_object
    text_field.build_vocab(train_set)
    label_field.build_vocab(train_set)
    data.vocab_to_idx = dict(text_field.vocab.stoi)
    data.idx_to_vocab   =   {v:   k   for   k,   v   in
data.vocab_to_idx.items()}
    data.label_to_idx = dict(label_field.vocab.stoi)
    data.idx_to_label   =   {v:   k   for   k,   v   in
data.label_to_idx.items()}
    embed_num = len(text_field.vocab)
    label_num = len(label_field.vocab)
    emb_init_values                                    =
np.array(data.create_fold_embeddings(embeddings, args))
    train_iter,               test_iter               =
torchtext.data.Iterator.splits((train_set, test_set),
batch_sizes=(args.batch_size,           len(test_set)),
device=-1,repeat=False)
```

Kode 5.21 Proses Pembatan Vocabulary

Kode 5.21 menunjukkan proses pembagian antara text dan label. Data label akan dimasukkan pada proses inisialisasi *embedding* nilai vektor. Setiap kata yang memiliki duplikat akan

dihapus menggunakan *method build_vocab()*. Selanjutnya, setiap vektor kata yang unik akan dimasukan ke dalam *dict()* dan diberikan *id* untuk setiap kata. Hasil dari method *dict()* disimpan didalam variabel *data.vocab_to_idx* untuk kalimat tweet sedangkan *data.label_to_idx* untuk label tweet. Variabel *data.idx_to_vocab* dan *variabel idx_to_label* adalah kebalikan dari variabel sebelumnya. Setelah itu akan dimasukkan dalam method *create_fold_embedding* yang terdapat pada prepare_data.py, untuk ditentukan kata yang terdapat dalam *vocabulary word embedding* dan kata yang tidak terdapat dalam *vocabulary word embedding* untuk dilakukan pemetaan vektor kata untuk setiap kata.

```
train_bulk_dataset = train_set,
    train_bulk__size   = len(train_set),
    train_bulk_iter                                    =
torchtext.data.Iterator.splits(datasets=train_bulk_data
set,
batch_sizes=train_bulk__size,
device=-1, repeat=False)[0]
    kim2014 = model.CNN_Kim2014(embed_num, label_num -
1,args.embeddings_dim,
args.kim2014_embeddings_mode,emb_init_values, args)
    if args.cuda:
      kim2014 = kim2014.cuda()
    trained_model    =    train(kim2014,    train_iter,
test_iter,    data.label_to_idx,    data.idx_to_label,
train_bulk_iter, args, i)
```

Kode 5.22 Proses Pemanggilan Method Training Model CNN

Kode 5.22 menjelaskan proses training yang akan dilakukan dengan pemanggilan model dengan parameter yang telah dilakukan pada file model.py yang kemudian akan dimasukkan dalam proses training dan proses testing. Pada kode tersebut juga menunjukkan proses pemanggilana cuda() yang berfungsi untuk membuat model melakukan processing menggunakan GPU.

```
def train(model, train_iter, test_iter, label_to_idx,
idx_to_label, train_bulk_iter, args, fold):
  parameters = filter(lambda p: p.requires_grad,
model.parameters())
 optimizer = torch.optim.Adadelta(parameters)
  if args.cuda:
    model = model.cuda()
  model.train()
  for epoch in range(1, args.epoch_num+1):
print("###_______FOLD[{}]/EPOCH[{}]_______###".format(f
old+1,epoch))
    for batch in train_iter:
      text_numerical, target = batch.text, batch.label
      if args.cuda:
        text_numerical, target = text_numerical.cuda(),
target.cuda()
      text_numerical.data.t_()
      target.data.sub_(1)
      optimizer.zero_grad()
      forward = model(text_numerical)
      loss = F.cross_entropy(forward, target)
      loss.backward()
      optimizer.step()
      corrects = (torch.max(forward,
1)[1].view(target.size()).data == target.data).sum()
      accuracy = 100.0 * corrects / batch.batch_size
```

Kode 5.23 Proses Training Model CNN

Kode 5.23 menjelaskan proses training pada model CNN. Proses training ini menggunakan sistem batching dan menggunakan optimizer ADADELTA dalam proses learning.

```
def evaluate(model, data_iter, type, fold, epoch):
  model.eval()
  corrects, avg_loss = 0, 0
  data_iter.sort_key = lambda x: len(x.text)
  for batch in data_iter:
    text_numerical, target = batch.text, batch.label
    if args.cuda:
      text_numerical, target = text_numerical.cuda(),
target.cuda()
    text_numerical.data.t_()
    target.data.sub_(1)
    forward = model(text_numerical)
    loss = F.cross_entropy(forward, target,
size_average=False)
    avg_loss += loss.data[0]
    corrects += (torch.max(forward,
1)[1].view(target.size()).data == target.data).sum()
  size = len(data_iter.dataset)
  avg_loss = avg_loss/size
  accuracy = 100.0 * corrects/size

output['fold_{}'.format(fold+1)]['epoch_{}'.format(epoc
h)][type]['avg_loss'] = avg_loss
  return target.data, torch.max(forward,
1)[1].view(target.size()).data
```

Kode 5.24 Proses Testing Model CNN

Kode 5.24 Menunjukkan proses testing terhadap hasil dari proses training dimana dari method ini akan dihasilkan label actual dan prediksi label berdasarkan dari hasil performa training data.

```
def      calculate_fold_counts(actual,      predicted,
label_to_idx, idx_to_label):
  assert len(actual)  ==  len(predicted)
  fold_actual_counts    = defaultdict(int)
  fold_predicted_counts = defaultdict(int)
  fold_match_counts     = defaultdict(int)
  mae_calculate_label   = defaultdict(int)
  std_calculate_label   = defaultdict(float)
  mean  = 0
  for i in range(len(actual)):
    idx   = actual[i] + 1
    mean += idx
  mean = mean/len(actual)
  for i in range(len(actual)):
    idx   = actual[i] + 1
    diff_label = abs((predicted[i]+1)-idx)
    diff_mean  = (idx-mean)**2
    label = idx_to_label[idx]
    fold_actual_counts[label] += 1
    mae_calculate_label[label] += diff_label
    std_calculate_label[label] += diff_mean
    if actual[i] == predicted[i]:
      fold_match_counts[label] += 1
  for i in range(len(predicted)):
    idx = predicted[i] + 1
    label = idx_to_label[idx]
    fold_predicted_counts[label] += 1
  return   fold_actual_counts,   fold_predicted_counts,
fold_match_counts,                mae_calculate_label,
std_calculate_label
```

Kode 5.25 Proses Perhitungan Evaluasi Model

Kode 5.25 menjelaskan proses perhitungan evaluasi pengukuran. Pengukuran evaluasi ini akan menggunakan nilai hasil prediksi dengan nilai actual. Pada method ini dimasukkan seluruh pengukuran evaluasi per foldnya jumlah text yang nilai labelnya sesuai.

```
def
calculate_and_display_SemEval_metrics(actual_counts,
predicted_counts,  match_counts,  mae_calculate_label,
std_calculate_label, args, type, fold, epoch):
  precisions  = defaultdict(float)
  recalls     = defaultdict(float)
  f_measures  = defaultdict(float)
  macro_mae   = defaultdict(float)
  std_dev     = defaultdict(float)
  klds        = defaultdict(float)
  standard_mae  = 0
  test_size = sum(actual_counts.values())
  label_count = 0
  for label in actual_counts.keys():
    macro_mae_avg   =   mae_calculate_label[label]   /
actual_counts[label] if actual_counts[label] > 0 else 0
    standard_mae  += mae_calculate_label[label]
    std_avg     =     std_calculate_label[label]     /
(actual_counts[label]-1) if (actual_counts[label]-1) >
0 else 0
    kld_actual    = actual_counts[label]/test_size
    kld_predicted = predicted_counts[label]/test_size
    diff = kld_actual/kld_predicted if kld_predicted >
0 else 0
    kld_calculate = kld_actual * math.log(diff) if diff
> 0 else 0
    precision       =       match_counts[label]       /
predicted_counts[label] if predicted_counts[label] > 0
else 0
    recall                =    match_counts[label]    /
actual_counts[label] if actual_counts[label] > 0 else 0
    f_measure = 2 * precision * recall / (precision +
recall) if (precision + recall) > 0 else 0
    label_count += 1
output['fold_{}'.format(fold+1)]['epoch_{}'.format(epoc
h)][type][label]  =  [precision,  recall,  f_measure,
macro_mae_avg, std_avg]
```

Kode 5.26 Evaluasi Pengukuran Setiap Epoch

Kode 5.26 menunjukkan evaluasi pengukuran yang dilakukan per epochnya pada setiap label. Pengukuran evaluasi ini mengambil nilai precision, recall, MAE, dan accuracy pada setiap label. Hasil dari pengukuran setiap label akan di hitung untuk mendapatkan nilai evaluasi pengukuran dari setiap epoch.

BAB VI
HASIL DAN PEMBAHASAN

Pada bab ini akan dijelaskan mengenai hasil dan analisis terhadap hasil yang diperoleh dari proses implementasi penelitian.

6.1. Data *Crawling*

Proses pengambilan data yang didapatkan dari tanggal 20 Desember 2017 hingga 1 Mei 2018 adalah sebanyak 64,245,029 untuk *dataset word embedding*. Setelah menghapus data yang terduplikasi maka dihasilkan 32,399,220 data posting Twitter yang unik yang akan masuk dalam proses filtering bahasa sebelum masuk proses pelatihan model *word embedding*. Data post inilah yang akan digunakan untuk melakukan pembuatan model *Word2Vec*. Sedangkan untuk data yang berhubungan dengan topik *e-commerce* dihasilkan 211,897 data *posting* dan setelah melalui proses penghapusan data terduplikasi dihasilkan 172,682 data posting Twitter yang unik. Data yang berhubngan dengan topik ini nantinya akan dipilih sejumlah data yang akan melalui proses *labeling* untuk proses *training* model *Convolutional Neural Network.*

6.2. *Filtering* Bahasa

Total data yang didapatkan dari proses *crawling* adalah 32,399,220 data posting Twitter yang unik. Namun di dalam dataset tersebut masih mengandung data yang tidak mengandung Bahasa Indonesia. Maka dilakukan proses filterisasi bahasa terhadap dataset untuk memastikan bahwa keseluruhan bahasa yang digunakan dari dataset adalah data yang mengandung Bahasa Indonesia. Untuk melakukan filterisasi bahasa digunakan *library* polyglot dengan nilai *confidency* 90%. Nilai *confidency* sebesar 90% digunakan untuk memastikan bahwa seluruh data yang didapat setelah proses *filter* adalah data yang menggunakan Bahasa Indonesia serta untuk mencegah data yang menggunakan bahasa serumpun seperti Bahasa Melayu ikut masuk ke dalam kumpulan dataset.

Setelah dilakukan proses *filter* dihasilkan data sebesar 26,004,595 data posting Twitter yang unik. Selain itu dilakukan juga pada dataset sesuai dengan topik dari total data sebanyak 172,682 dataset yang berhasil dikumpulkan 153,339 data yang menggunakan Bahasa Indonesia.

Presisi dari *filter* Bahasa Indonesia yang digunakan adalah 100 % dengan pengecekan secara manual pada 1,000 data sampel hasil *filtering* yang di ambil secara acak, seliruh data menggunakan Bahasa Indonesia.

6.3. Hasil Pelabelan Data

Data yang telah dikumpulkan melalui proses crawling yang digunakan dalam model *Convolutional Neural Netwok* memiliki jumlah total 10,686 data yang berasal dari *keyword* berdasarkan akun 5 e-commerce yang berdasarkan pada sub bab 3.2.2 dengan pembagian data seperti pada Tabel 6.1.

Tabel 6.1 Pembagian Data Berdasarkan Topik

Topik	Jumlah
Bliblicare	355
Bliblidotcom	165
BukaBantuan	2,559
Bukalapak	943
LazadaID	680
LazadaIDCare	1,248
ShopeeCare	1,291
ShopeeID	970

Tokopedia	659
TokopediaCare	1,816
Total	10,686

Pelabelan data yang dilakukan pada tahap ini dilakukan oleh 3 orang yang berbeda kemudian dilakukan kalkulasi untuk mendapatkan label final yang akan digunakan pada dataset tersebut. Berikut merupakan hasil akhir yang didapatkan dari proses pelabelan dari 3 orang berbeda.

Tabel 6.2 Penyeberan Data berdasarkan masing-masing pelabel

Label	**Jumlah Data**		
	Pelabel 1	Pelabel 2	Pelabel 3
Sangat Positif	76	18	2
Positif	1,547	1,391	964
Netral	6,129	7,496	9,123
Negatif	2,311	1,395	542
Sangat Negatif	623	386	55
Total Data	10,686	10,686	10,686

Pada proses pelabelan, hasil yang diberikan seluruhnya memenuhi persyaratan pertama dalam proses kalkulasi di mana 2 dari 3 pelabel memilih label yang sama akan diambil sebagai label final dan tidak terdapat proses kalkulasi yang terdapat 3 pelabel memilih label yang berbeda-beda.

Pada proses pelabelan yang telah dilakukan kembali pemeriksaan untuk memastikan pelabelan untuk memastikan apabila terdapat data yang dapat ditolah pelabelannya. Dari hasil data yang didapatkan terdapat 1,378 data yang ditolak diakibatkan oleh penyebaran data yang dianggap random pada pelabelan data. Sehingga dilakukan pelabelan kembali. Maka setelah dilakukan kalkulasi kembali dihasilkan dataset final dengan penyebaran sebagai berikut :

Tabel 6.3 Penyeberan Data Hasil Final Pelabelan

Label	Jumlah Data
Sangat Positif	33
Positif	1493
Netral	7151
Negatif	1553
Sangat Negatif	456
Total Data	10,686

Tabel 6.4 Penyebaran Data Berdasarkan Topik pada Hasil Pelabelan Final

Topik	Label Data	Jumlah
Bliblicare	Sangat Positif	2
	Positif	36
	Netral	205

	Negatif	65
	Sangat Negatif	47
Bliblidotcom	Sangat Positif	1
	Positif	17
	Netral	109
	Negatif	26
	Sangat Negatif	12
BukaBantuan	Sangat Positif	6
	Positif	684
	Netral	1591
	Negatif	233
	Sangat Negatif	45
Bukalapak	Sangat Positif	0
	Positif	30
	Netral	706
	Negatif	159
	Sangat Negatif	48
LazadaID	Sangat Positif	6

	Positif	71
	Netral	396
	Negatif	151
	Sangat Negatif	56
LazadaIDCare	Sangat Positif	4
	Positif	77
	Netral	797
	Negatif	258
	Sangat Negatif	112
ShopeeCare	Sangat Positif	1
	Positif	75
	Netral	1054
	Negatif	142
	Sangat Negatif	19
ShopeeID	Sangat Positif	5
	Positif	55
	Netral	721
	Negatif	159

	Sangat Negatif	30
Tokopedia	Sangat Positif	4
	Positif	50
	Netral	469
	Negatif	94
	Sangat Negatif	42
TokopediaCare	Sangat Positif	4
	Positif	398
	Netral	1103
	Negatif	266
	Sangat Negatif	45
Total		10,686

Dari hasil ditunjukkan pada daset yang akan digunakan terdapat imbalance data baik secara topik maupun secara pelabelan. Hal ini diakibatkan dari keterbatasan yang dimiliki dari proses pengumpulan data.

Selanjutnya, untuk mengukur tingkat keberagaman pelabelan data berdasarkan proporsi antara pelabel akan dilakukan perhitungan menggunakan metode *analisis Cohen's Kappa*.

Rumus *Cohen's Kappa*

$$k = \frac{\sum_{i=1}^{I} \pi_{ii} - \sum_{i=1}^{I} \pi_{i+}\pi_{+i}}{1 - \sum_{i=1}^{I} \pi_{i+}\pi_{+i}}$$

Dimana :

$\sum_{i=1}^{I} \pi_{ii}$ = Total proporsi diagonal utama dari frekuensi obeservasi

$\sum_{i=1}^{I} \pi_{i+}\pi_{+i}$ = Total proporsi total marginal dari frekuensi observasi

Berikut adalah hasil untuk mengukur tingkat persetujuan secara menyeluruh pada hasil dari pelabelan yang dilakukan.

Tabel 6.5 Range Nilai Kappa berdasarkan Tingkat Kesepakatan

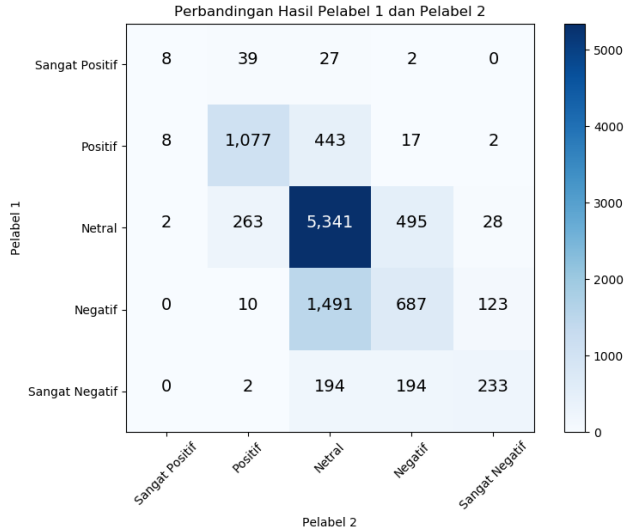
Nilai K	Tingkat Kesepakatan
< 0.20	Rendah (*Poo*r)
0.21 – 0.40	Sedang (*Fair*)
0.41 – 0.60	Cukup (*Moderate*)
0.61 – 0.80	Kuat (*Good*)
0.81 – 1.00	Sangat Kuat (*Very Good*)

Pada Tabel 6.5 menunjukkan tingkatan yang digunakan sebagai tolak ukur untuk mengukur tingkat persetujuan pada pelabelan yang dilakukan.

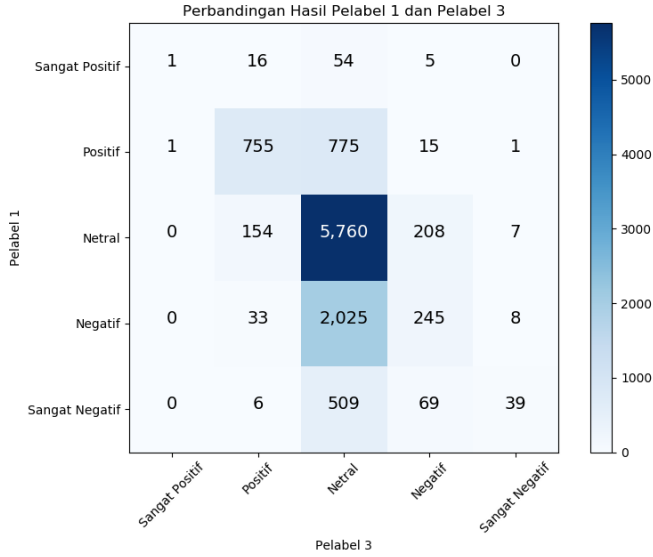
Tabel 6.6 Nilai Kappa Antar tiap Pelabelan

Pelabel	Kappa Value (Tingkat Kesepakatan)
1 – 2	0.430 (Cukup)
1 – 3	0.252 (Sedang)
2 – 3	0.363 (Sedang)
Rata-rata	0.348 (Sedang)

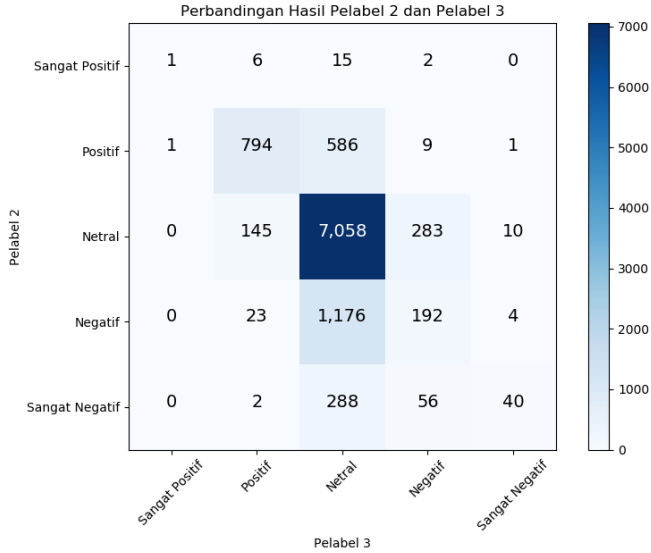
Setelah dilakukan pengujian nilai kappa untuk mengukur persetujuan antara tiap hasil pelabelan maka didapatkan hasil nilai dari hasil pelabelan 1 dengan pelabelan 3 dan pelabelan 2 dengan pelabelan 3 memiliki nilai persetujuan yang sedang. Sedangkan, antara pelabelan 1 dan pelabelan 2 memiliki nilai persetujuan cukup. Berdasarkan hasil rata-rata yang didapat menghasilkan nilai rata-rata 0.348, dengan hasil ini maka nilai persetujuan yang didapatkan memiliki tingkat persetujuan yang sedang. Hasil dari proses perhitungan dapat dilihat pada Tabel 6.6.



Gambar 6.1 Confusion Matrix antara pelabel 1 dan pelabel 2



Gambar 6.2 Confusion Matrix antara pelabel 1 dan pelabel 3



Gambar 6.3 Confusion Matrix antara pelabel 2 dan pelabel 3

Setelah melakukan proses perhitungan maka dilakukan klasifikasi berdasarkan confusion matrix antara tiap pelabel. Berdasarkan dari Gambar 6.1, Gambar 6.2 dan Gambar 6.3 menunjukkan bahwa perbedaan persepsi antara tiap pelabel paling besar adalah membedakan antara label netral dan label negatif.

Berdasarkan hasil dari tingkat persetujuan tersebut maka pada Tabel 6.7 akan ditampilkan beberapa sampel data yang menandakan bahwa adanya perbedaan persepi antar pelabel terhadap suatu teks.

Tabel 6.7 Contoh label dari masing-masing pelabel

Message	Pelabel 1	Pelabel 2	Pelabel 3
udah ngehubungin CS pun, masa saya disuruh	Sangat Negatif	Negatif	Netral

menunggu sampai 3 hari cuma untuk verifikasi transaksi doang. basi coy. 3 hari itu kalo transaksi lancar, hari ini barang udah sampe rumah. masa slogan "jual beli mobile 30 detik" berubah jadi "jual beli mobile 3 hari" @ShopeeID			
@ShopeeCare coba di cek status pengirimannya bagaimana. Jangan males, itu foto bisa diperbesar dan kelihatan no pesanan dan resi.	Negatif	Negatif	Netral
@BlibliCare PERTANYAAN NYA, KENAPA SOLUSI SAYA DIRUBAH TANPA PERSETUJUAN SAYA????	Sangat Negatif	Sangat Negatif	Negatif
@LazadaIDCare @LazadaID @xiaomiindonesia Baiklah min terimakasih semoga kedepannya lebih baik, dan semoga berikutnya bisa beruntung	Positif	Netral	Netral
@TokopediaCare oke min makasih responnya. Tolong ya min di pickup ulang. Spy gak pending lagi. Krn saya tanya seller cma dia diam saja	Positif	Positif	Netral

6.4. Model *Word Embedding*

Salah satu model word embedding yang digunakan adalah *Word2Vec* dimana pada model yang dibangun menggunakan dataset yang telah dikumpulkan dari proses *crawling* pada media sosial Twitter. Dari hasil pengumpulan data maka digunakan scenario yang diambil dari hasil penelitian yang telah dilakukan oleh Tomas Mikolov, dkk. Dengan parameter sebagai berikut.

Tabel 6.8 Parameter Model Yon Kim

Description	Values
Embedding Dimension	300
Batch Size	50
Window Size	5
Iteration	1
Learning Rate	0.025

Dari hasil pembuatan model menggunakan *library pytorch* didapatkan *output* dari model yang telah dibuat menghasilkan 600,978 kosa kata yang unik untuk algoritma *Word2Vec*. Kemudian pada algoritma *FastText* menggunakan model yang telah dihasilkan oleh Bojanowski menghasilkan 300,686 kosa kata.

Komponen terbaik model klasifikasi kalimat diawali dengan menentukan seberapa representasi kata yang digunakan dalam pembuatan model. Pada percobaan ini akan diambil nilai

average recall dikarenakan dataset yang digunakan *imbalance* untuk melihat *effect* yang diberikan *word embedding* terhadap dataset.

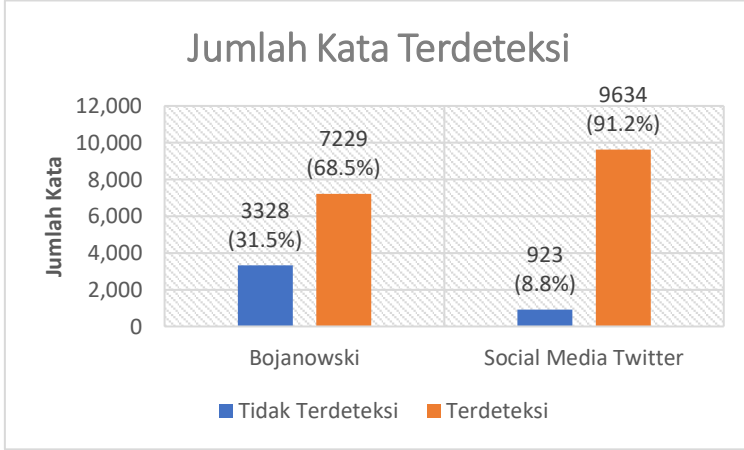
Tabel 6.9 Percobaan Algoritma *Word Embedding* pada Model *Non-Static*

Subtask	Non-Static Word2Vec-SkipGram-CNN (Social Media Twitter)	Non-Static Word2Vec-CBOW-CNN (Social Media Twitter)	Non-Static FastText-CNN (Wikipedia)
A	89.036	88.944	88.903
B	65.763	65.245	65.320
C	39.553	39.161	38.609

Tabel 6.10 Percobaan Algoritma *Word Embedding* pada Model *Static*

Subtask	Static Word2Vec-SkipGram-CNN (Social Media Twitter)	Static Word2Vec-CBOW-CNN (Social Media Twitter)	Static FastText-CNN (Wikipedia)
A	88.961	88.918	88.704
B	66.260	63.581	65.205
C	39.625	38.958	39.410

Dari Tabel 6.9 dan 6.10 menunjukkan hasil dari percobaan dengan melaukan perndingan antara 3 bentuk model algoritma *word embedding* pada 3 bentuk subtask yang digunakan pada penelitian. Berdasarkan hasil percobaan yang dilakukan menunjukkan bahwa pada tiga subtask berbeda Model *Word2Vec* dengan menggunakan *learning algoritma Skip-Gram* menghasilkan nilai paling baik dibandingkan dengan Model *Word2Vec* dengan menggunakan *learning algoritma CBOW* dan Model yang menggunakan *FastText.*



Gambar 6.4 Jumlah Kata Terdeteksi dari Model *Word Embedding*

Dari Tabel 6.4 menunjukkan bahwa model *word embedding* yang dihasilkan dengan menggunakan data media sosial twitter meghasilkan lebih banyak kata yang terdeteksi dari *dataset* pelatihan *Convolutional Neural Network*. Sehingga hasil evaluasi pengukuran yang didapat lebih baik daripada model *Bojanowski* yang mengambil data dari *Wikipedia*.

6.5. Hasil Pengujian Data

a. Perbedaan Penggunaan GPU dengan CPU

Berikut adalah hasil penggunaan menggunakan CPU dan GPU pada subtask yang dilakukan :

Tabel 6.11 Perbandingan Kecepatan Penggunaan GPU dengan CPU

Subtask	GPU	CPU
A	76.02863	1119.88
B	231.9695	3017,92
C	232.7708	3082.39

Berdasarkan Tabel 6.11 dari hasil percobaan yang dilakukan performa kecepatan dengan menggunakan GPU meningkat 13x lipat dari penggunaan CPU. Sehingga proses training dapat dilakukan lebih cepat.

b. Hasil Percobaan Tiga Subtask

1) *Subtask* A

Subtask A merupakan subtask dengan 2 *point scale* dengan label yang digunakan adalah label positif dan label negatif. Dengan penyebaran data sebagai berikut :

Tabel 6.12 Penyebaran Data pada *Subtask* A

Label	Jumlah Data
Positif	1,526
Negatif	2,009
Total	3,535

a) Single Filter Region Size Subtask A

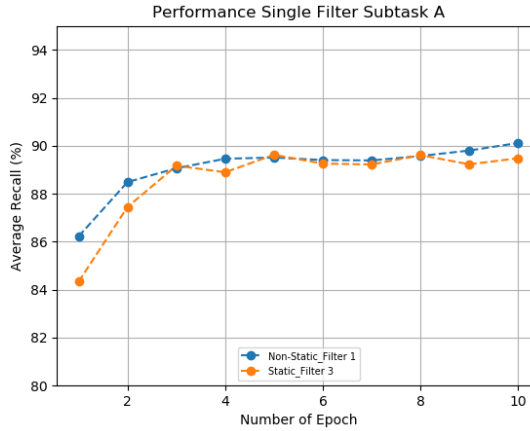
Tabel 6.13 Percobaan *Single Filter Region Size Subtask A* Pada Model CNN-non-Static

Region Size	*AvgRec (%)*	*F1-Score (%)*	*Accuracy (%)*
1	90.12_{1}	89.93_{2}	92.84_{10}
2	89.68_{3}	89.69_{5}	93.52_{3}
3	89.97_{2}	90.43_{1}	93.38_{6}
4	89.43_{5}	89.81_{3}	93.43_{5}
5	88.94_{7}	89.20_{6}	93.69_{1}
6	89.53_{4}	89.76_{4}	93.65_{2}
7	88.48_{9}	88.40_{10}	93.15_{7}
8	88.16_{10}	88.47_{9}	93.43_{4}
9	88.97_{6}	89.07_{7}	93.12_{8}
10	88.73_{8}	88.97_{8}	92.90_{9}

Tabel 6.14 Percobaan *Single Filter Region Size Subtask A* Pada Model CNN-static

Region Size	*AvgRec (%)*	*F1-Score (%)*	*Accuracy (%)*
1	88.63_{9}	88.63_{9}	92.28_{10}
2	88.72_{6}	89.05_{7}	93.49_{3}
3	89.48_{1}	89.87_{1}	93.63_{1}
4	89.30_{3}	89.66_{3}	92.95_{9}
5	89.36_{2}	89.78_{2}	93.46_{4}
6	88.65_{8}	89.04_{8}	93.24_{5}
7	88.01_{10}	88.48_{10}	93.01_{7}
8	88.77_{5}	89.06_{6}	93.12_{6}
9	88.97_{4}	89.59_{4}	93.55_{2}
10	88.71_{7}	89.20_{5}	92.98_{8}

Pada percobaan yang dilakukan disimpulkan bahwa model terbaik dengan *baseline* perhitungan evaluasi menggunakan evaluasi pengukuran *Average Recall* adalah *filter region size* dengan nilai 1 untuk model CNN-non-static. Sedangkan untuk model CNN-static *filter region size* dengan hasil pengukuran terbaik adalah 3.



Gambar 6.5 Performa Model Terbaik Single Filter Subtask A

Pada Gambar 6.5 Jika dilihat dari performa yang dilakukan berdasarkan perubahan setiap *epoch* didapatkan bahwa grafik *non-static* terlihat lebih stabil dibandingkan dengan grafik *static*. Untuk melihat kestabilan pada masing-masing grafik maka dilakukan perhitungan *standar deviation* pada model terbaik yang dihasilkan. Sehingga didapatkan hasil *standar deviation* untuk model terbaik pada *non-static* adalah 1.0425. Sedangkan untuk model terbaik pada *static* menghasilkan nilai standar deviation sebesar 1.5474. Dari hasil yang didapat nilai paling stabil dihasilkan oleh model *non-static*. Jika dilihat dari proses pembelajaran model *non-static* terlihat lebih baik karena mengalami kenaikan seiring dengan kenaikan *epoch*.

Setelah mendapatkan nilai *single filter region size* terbaik maka kemudian dilakukan kombinasi untuk mendapatkan model terbaik dengan menggunakan *multiple region size*.

b) Multiple Region Size Subtask A

Pada percobaan di *multiple region size* akan dilakukan kombinasi dari hasil terbaik yang didapatkan dari skenario menggunakan *single filter region size*. Hasil terbaik dari *single filter region size* dapat dilihat pada Tabel 6.12 dan 6.13 untuk model *non-static* dan *static* dengan melakukan kombinasi dari hasil paling optimal pada skenario *single filter region size*, yang diharapkan dapat meningkatkan performa pada model yang dihasilkan.

Pada *subtask* A digunakan *average recall* sebagai *baseline* perhitungan evaluasi pengukuran untuk mendapatkan model terbaik pada *multiple region size*. Pada skenario awal nilai *feature maps* akan dipertahankan pada nilai 100 untuk melihat performa yang didapat. Pada skenario ini juga digunakan dua model CNN *non-static* maupun *static*.

Tabel 6.15 Hasil Evaluasi Pengukuran *Multiple Filter Subtask A Non-Static*

Filter Size	*AvgRec (%)*	*F1-Score (%)*	*Accuracy (%)*
1,1,1	89.71_2	89.51_2	92.50_2
1,2,3	90.34_1	90.58_1	93.66_1

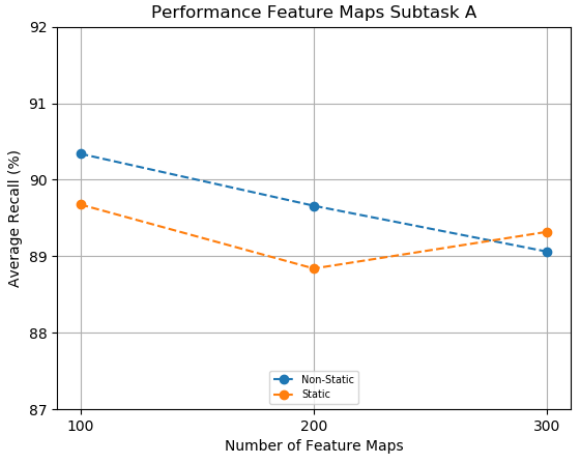
Tabel 6.16 Hasil Evaluasi Pengukuran *Multiple Filter Subtask A Static*

Filter Size	*AvgRec (%)*	*F1-Score (%)*	*Accuracy (%)*
3,3,3	89.05_3	89.20_2	93.12_3
1,2,3	89.68_1	89.79_1	93.26_1
2,3,4	89.10_2	89.18_3	93.18_2
3,4,5	88.83_4	88.75_4	92.67_4

Dari hasil percobaan yang dilakukan pada Tabel 6.15 dan Tabel 6.16 dapat disimpulkan bahwa nilai terbaik yang didapatkan menggunakan evaluasi pengukuran *average recall* maupun evaluasi pengukuran yang lain pada model *CNN-non-static* adalah *multiple region size* dengan nilai 1, 2, 3. Sedangkan untuk model *CNN-static* adalah *multiple region size* dengan nilai 1, 2, 3.

c) Pengaruh dari *Feature Maps Subtask A*

Setelah didapatkan model terbaik untuk *multiple region size* maka selanjutnya dilakukan konfigurasi nilai *feature maps*. Nilai *feature maps* akan diubah dengan jarak 100 sampai dengan 300 untuk melihat pengaruh yang dimiliki oleh perubahan nilai *feature maps* terhadap dataset. Pada perubahan nilai *feature maps* akan berfokus pada nilai perubahan *average recall* pada tiap *feature maps*.



Gambar 6.6 Perubahan *Feature Maps Multiple Filter Subtask A*

Perubahan Gambar 6.6 menunjukkan model *CNN-non-static* selalu mengalami penurunan seiring bertambahnya nilai *feature maps*. Sedangkan pada *CNN-static* nilai evaluasi pengukuran yang dihasilkan mengalami tren turun pada *feature maps* dengan nilai 200 kemudian naik pada *feature maps* 300. Pada nilai *feature maps* 300 nilai evaluasi pengukuran pada *subtask* A untuk model *CNN-static* lebih baik dibandingkan dengan model *CNN-non-static*.

Sehingga dari hasil pengukuran evaluasi untuk model *CNN-non-static* maupun *static* nilai *feature maps* paling baik adalah 100.

d) Pembahasan Hasil dari *Subtask* A

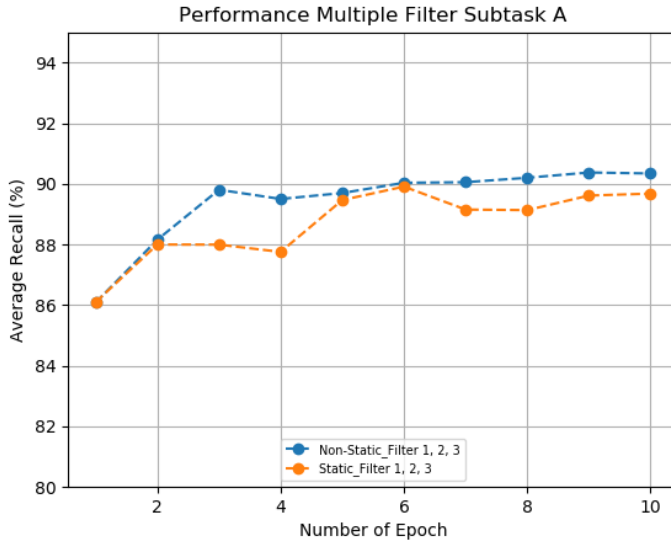
Dari hasil *subtask* A didapatkan performa terbaik yang dapat dilihat sebagai berikut

Tabel 6.17 Pengukuran Evaluasi Model Terbaik dari *Subtask* A

Model CNN	***Filter Size***	***Feature Maps***	***AvgRec (%)***
Non-Static	1,2,3	100	90.34_1

Static	1,2,3	100	89.68_2

Dari hasil Tabel 6.17 menunjukkan bahwa hasil terbaik dihasilkan model *CNN-non-static* dengan nilai *filter region size* 1, 2, 3 dengan *feature maps* 100. Selanjutnya untuk kinerja dari model terbaik dalam melakukan pengklasifikasian data akan dilihat berdasarkan *confusion matrix* yang dihasilkan.



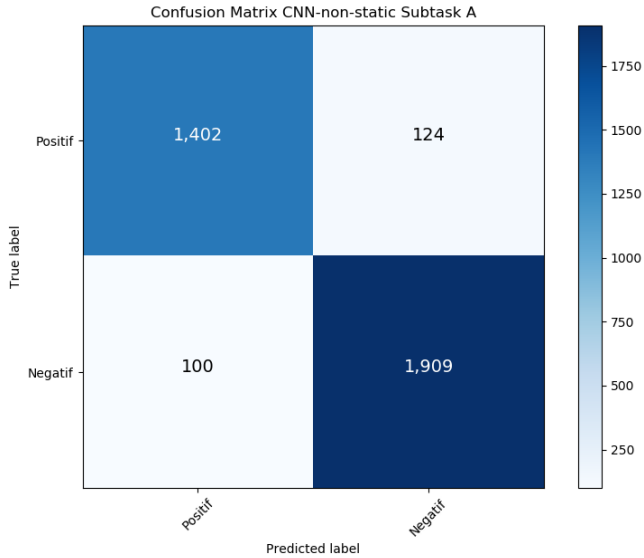
Gambar 6.7 Grafik Performa Model Terbaik *Multiple Region Size Subtask* A

Pada Gambar 6.7 menunjukkan bahwa grafik *static* lebih stabil daripada grafik *non-static*. Untuk membuktikan hal tersebut maka dilakukan perhitungan *standar deviation* pada masing-masing grafik. Sehingga nilai *standar deviation* yang dihasilkan model *non-static* adalah 1.269, sedangkan untuk model *static* menghasilkan nilai 1.135. Namun, jika dilihat dari seberapa baik model dalam melakukan proses pembelajaran, model *non-static* terlihat lebih baik karena selalu mengalami peningkatan. Namun, juga terjadi proses penurunan hasil evaluasi pengukuran pada *epoch* 7. Perbedaan nilai hasil

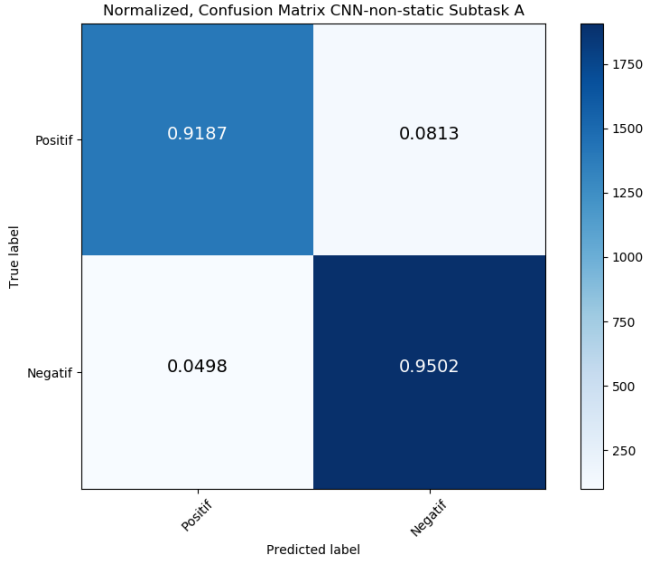
standar deviation juga tergolong sangat kecil yang memiliki selisih 0.134.

- **Model *CNN-non-static***

Selanjutnya ditamplikan *confusion matrix* pada epoch ke 10 untuk melihat kualitas klasifikasi dari model. Pada *confusion matrix* yang dihasilkan, kesalahan dalam melakukan klasifikasi untuk label positif dan negatif sangat kecil dilihat dari perbandingan kebenaran model dalam melakukan klasifikasi terhadap labelnya yang dapat dilihat pada Gambar 6.8.



Gambar 6.8 *Confusion Matrix* Model *Non-static* subtask A



Gambar 6.9 *Normalized Confusion Matrix* Model *Non-static* subtask A

Berdasarkan Gambar 6.9 menunjukkan hasil normalisasi *confusion matrix* untuk melihat perbandingan nilai antar tiap label dalam *confusion matrix*. Normalisasi dilakukan untuk menyamakan tingkat pengukuran dalam skala tertentu, pada proses ini skala yang diambil adalah 0 – 1. Hasil yang didapatkan perbandingan dalam skala yang sama menunjukkan proses klasifikasi pada label negatif dan positif masih menghasilkan proses klasifikasi yang baik.

Pengecekan selanjutnya dilihat berdasarkan penyebaran nilai *recall* pada tiap labelnya Perhitungan evaluasi pengukuran untuk setiap labelnya akan dilihat penyebarannya. Setelah dilakukan perhitungan maka dihasilkan data sebagai berikut :

Tabel 6.18 Perhitungan Recal per Label Model *Non-static Subtask* A

Label	***Recall (%)***
Positif	86.70

Negatif	93.98

Dari hasil perhitungan yang dilakukan pada *subtask* A, nilai recall dalam proses klasifikasi adalah negatif. Hal ini membuktikan bahwa klasifikasi label negatif dan positif sudah sangat baik. Hal ini yang menyebabkan nilai *average recall* menyentuh nilai 90.34.

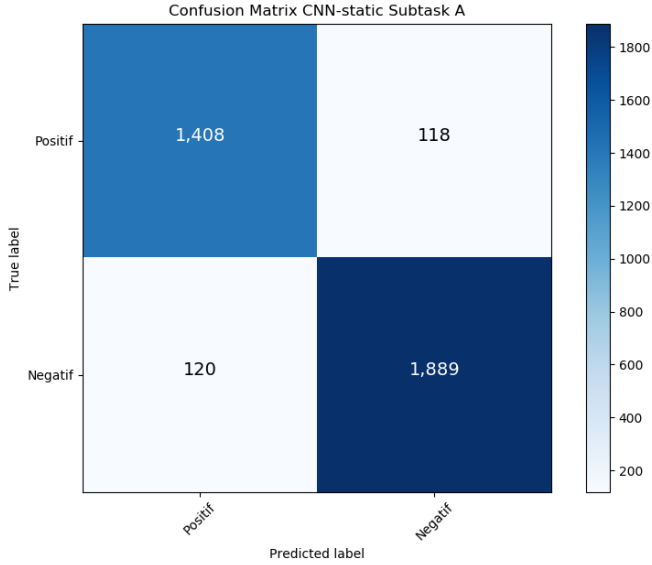
Tabel 6.19 Perbandingan Nilai Pengukuran Evaluasi *Single* dan *Multiple Filter Subtask A Non-static*

Filter Type	***Filter Size***	***AvgRec (%)***
Single Filter	1	90.116_2
Multiple Filter	1, 2, 3	90.340_1

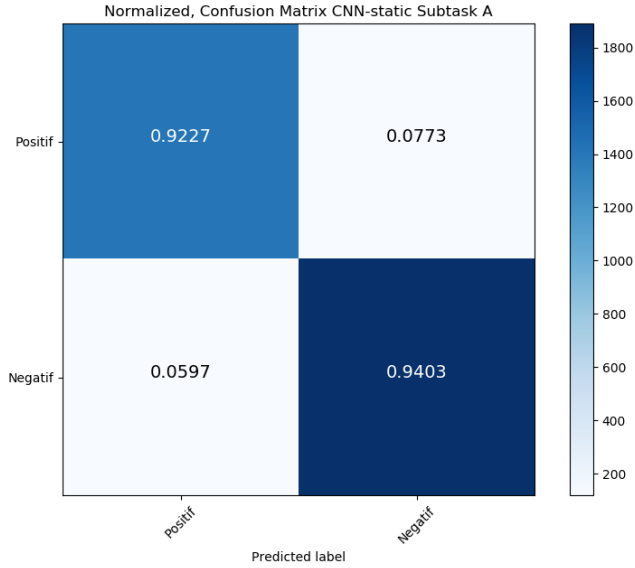
Selanjutnya dilihat performa yang dihasilkan pada proses perubahan *filter size* paling baik dihasilkan oleh model yang menggunakan *multiple filter*. Sehingga, dapat disimpulkan performa model akan semakin baik dengan menggunakan *multiple filter region size* pada model *CNN-non-static*.

- **Model *CNN-static***

Selanjutnya ditamplikan *confusion matrix* pada epoch ke 10 untuk melihat kualitas klasifikasi dari model *CNN-static*. Pada *confusion matrix* yang dihasilkan, kesalahan dalam melakukan klasifikasi untuk label positif dan negatif sangat kecil dilihat dari perbandingan kebenaran model dalam melakukan klasifikasi terhadap labelnya. Hasil yang didapat ini sama dengan yang didapatkan pada model *CNN-non-static.*



Gambar 6.10 *Confusion Matrix* Model *Static subtask* A



Gambar 6.11 *Normalized Confusion Matrix* Model *Static subtask* A

Berdasarkan Gambar 6.11 menunjukkan hasil normalisasi *confusion matrix* untuk melihat perbandingan nilai antar tiap label dalam *confusion matrix.* Normalisasi dilakukan untuk menyamakan tingkat pengukuran dalam skala tertentu, pada proses ini skala yang diambil adalah 0 – 1. Hasil yang didapatkan perbandingan dalam skala yang sama menunjukkan proses klasifikasi pada label negatif dan positif masih menghasilkan proses klasifikasi yang baik.

Tabel 6.20 Perhitungan Recal per Label Model *Static* Subtask A

Label	*Recall (%)*
Positif	86.61
Negatif	92.73

Dari hasil perhitungan yang dilakukan pada subtask A, nilai recall dalam proses klasifikasi adalah negatif. Hal ini membuktikan bahwa klasifikasi label negatif dan positif sudah sangat baik. Hal ini yang menyebabkan nilai *average recall* pada tiap label tinggi yang mencapai angka 0.8661 dan 0.9273.

Tabel 6.21 Perbandingan NiliaPengukuran Evaluasi Single dan Multiple Filter Subtask A Static

Filter Type	*Filter Size*	*AvgRec (%)*
Single Filter	3	89.479_2
Multiple Filter	1, 2, 3	89.677_1

Selanjutnya dilihat performa yang dihasilkan pada proses perubahan *filter size* paling baik dihasilkan oleh model yang menggunakan *multiple filter*. Sehingga, dapat disimpulkan performa model akan semakin baik dengan menggunakan *multiple filter region size* pada model *static.*

2) Subtask B

Subtask B merupakan subtask dengan 3 *point scale* dengan label yang digunakan adalah label positif, label netral dan label negatif. Dengan penyebaran data sebagai berikut :

Tabel 6.22 Penyebaran Data pada Subtask B

Label	Jumlah Data
Positif	1,526
Netral	7,151
Negatif	2,009
Total	10,686

a) Single Filter Region Size Subtask B

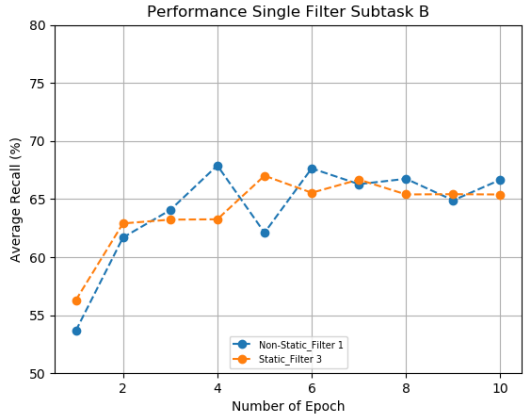
Tabel 6.23 Percobaan *Single Filter Region Size Subtask B* Pada Model *CNN-non-Static*

Region Size	*AvgRec (%)*	*F1-Score (%)*	*Accuracy (%)*
1	66.66_{1}	58.38_{2}	77.30_{10}
2	66.54_{2}	58.64_{1}	78.92_{9}
3	65.55_{3}	57.94_{3}	79.57_{1}
4	64.52_{4}	57.11_{4}	79.24_{6}
5	63.83_{6}	55.87_{6}	79.21_{7}
6	64.08_{5}	56.50_{5}	79.39_{3}
7	62.97_{9}	55.17_{9}	79.35_{5}
8	63.16_{8}	55.33_{8}	79.21_{8}
9	62.91_{10}	55.13_{10}	79.39_{4}
10	63.18_{7}	55.63_{7}	79.44_{2}

Tabel 6.24 Percobaan *Single Filter Region Size Subtask* B Pada Model *CNN-static*

Region Size	*AvgRec (%)*	*F1-Score (%)*	*Accuracy (%)*
1	64.97_{2}	57.35_{1}	78.54_{10}
2	64.39_{3}	57.40_{2}	78.87_{8}
3	65.39_{1}	58.50_{1}	78.93_{7}
4	64.36_{4}	56.98_{4}	79.24_{4}
5	63.44_{6}	55.71_{6}	78.85_{9}
6	63.60_{5}	56.25_{5}	79.16_{5}
7	63.17_{7}	55.50_{7}	79.37_{1}
8	63.05_{8}	55.39_{8}	79.30_{3}
9	62.74_{9}	55.02_{9}	78.99_{6}
10	62.51_{10}	54.54_{10}	79.35_{2}

Pada percobaan yang dilakukan disimpulkan bahwa model terbaik dengan menggunakan perhitungan evaluasi *average recall* adalah *filter region size* dengan nilai 1 untuk model *CNN-non-static*. Sedangkan untuk model *CNN-static* yang dihasilkan adalah *filter region size* dengan nilai 3.



Gambar 6.12 Performa Model Terbaik *Single Filter Subtask* B

Pada Gambar 6.12 jika dilihat dari performa yang dilakukan berdasarkan perubahan setiap *epoch* didapatkan

bahwa grafik *static* terlihat lebih stabil dibandingkan dengan grafik *non-static*. Untuk melihat kestabilan pada masing-masing grafik maka dilakukan perhitungan *standar deviation* pada model terbaik yang dihasilkan. Sehingga didapatkan hasil *standar deviation* untuk model terbaik pada *non-static* adalah 4.06. Sedangkan untuk model terbaik pada *static* adalah 2.93. Berdasarkan hasil yang didapat nilai paling stabil dihasilkan oleh model *static*. Jika dilihat dari proses pembelajaran model *static* juga yang paling baik karena mengalami kenaikan seiring dengan kenaikan *epoch*.

Setelah mendapatkan nilai *single filter region size* terbaik maka kemudian dilakukan kombinasi untuk mendapatkan model terbaik dengan menggunakan *multiple region size*.

b) Multiple Region Size Subtask B

Pada percobaan di *multiple region size* akan dilakukan kombinasi dari hasil terbaik yang didapatkan dari skenario menggunakan *single filter region size*. Dengan melakukan kombinasi dari hasil paling optimal pada *single filter region size* diharapkan dapat meningkatkan performa.

Pada *subtask* B akan digunakan nilai *Average Recall* sebagai baseline perhitungan evaluasi untuk mendapatkan model terbaik pada *multiple region size*. Pada skenario di *subtask* B akan digunakan *feature maps* untuk keseluruhan skenario dengan nilai 100 untuk melihat performa yang didapat. Berikut merupakan hasil yang didapatkan dari hasil percobaan skenario yang dilakukan.

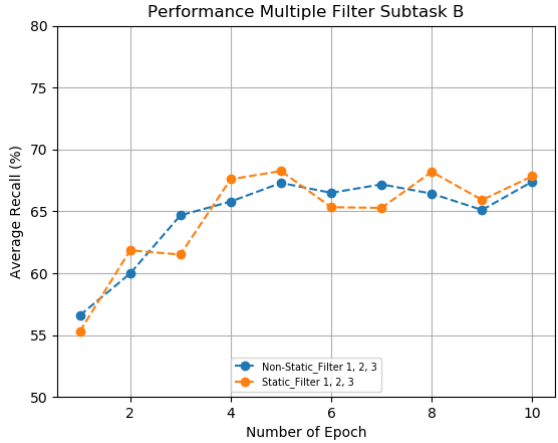
Tabel 6.25 Hasil Evaluasi Pengukuran *Multiple Filter Subtask B Non-Static*

Filter Size	***AvgRec (%)***	***F1-Score (%)***	***Accuracy (%)***
1, 1, 1	66.782_{2}	59.383_{2}	77.558_{2}
1, 2, 3	67.407_{1}	60.173_{1}	78.540_{1}

Tabel 6.26 Hasil Evaluasi Pengukuran *Multiple Filter Subtask B Static*

Filter Size	*AvgRec (%)*	*F1-Score (%)*	*Accuracy (%)*
1, 2, 3	67.839_1	59.559_1	78.521_3
2, 3, 4	67.117_3	59.040_3	79.139_1
3, 3, 3	67.670_2	59.206_2	78.503_4
3, 4, 5	66.725_4	58.788_4	78.989_2

Dari hasil yang didapatkan pada Tabel 6.25 dan Tabel 6.26 dapat disimpulkan bahwa nilai terbaik yang didapatkan menggunakan pengukuran evaluasi *Average Recall* untuk model *CNN-non-static* adalah *multiple region size* dengan nilai 1, 2, 3. Sedangkan untuk model *CNN-static* adalah *multiple region size* dengan nilai 1, 2, 3.



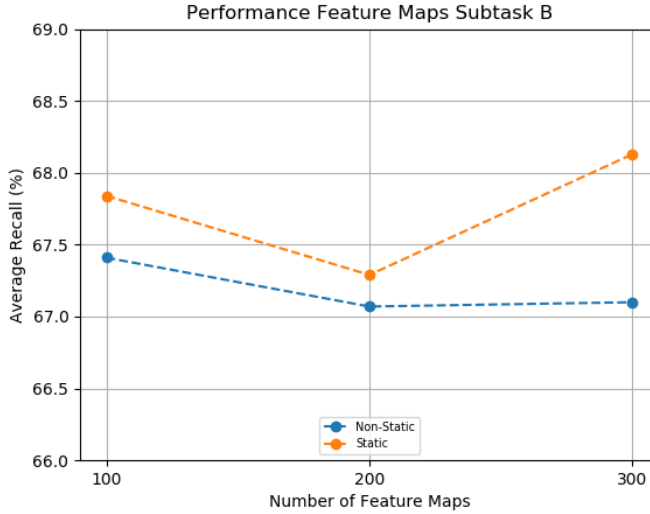
Gambar 6.13 Grafik Performance Multiple Filter Subtask B

Pada Gambar 6.13 menunjukkan bahwa grafik non-static lebih stabil daripada grafik static. Untuk membuktikan hal tersebut maka dilakukan perhitungan standar deviation pada masing-masing grafik. Sehingga nilai standar deviation yang dihasilkan model *non-static* adalah 3.40, sedangkan untuk

model static menghasilkan nilai 3.89. Sehingga hasil paling stabil adalah *non-static*. Jika dilihat dari seberapa baik model dalam melakukan proses pembelajaran, model *non-static* juga terlihat lebih baik karena selalu mengalami peningkatan. Namun, terjadi tren penurunan pada *epoch* ke 7 setelah itu meningkat kembali pada *epoch* ke 10.

c) Pengaruh dari Feature Maps Subtask B

Selanjutnya setelah didapatkan model terbaik untuk model multiple region size dengan nilai feature maps 100. Maka selanjutnya akan dilakukan percobaan skenario dengan melakukan perubahan pada nilai feature maps. Untuk melihat pengaruh yang diberikan pada data Subtask B.



Gambar 6.14 Perubahan Feature Maps Multiple Filter Subtask B

Pada Gambar 6.14 dapat disimpulkan bahwa perubahan terjadi pada *feature maps* dengan nilai 200. Dimana hasil performa yang didapat mengalami penurunan. Namun, nilai tersebut meningkat pada *feature maps* 300 pada model *CNN-*

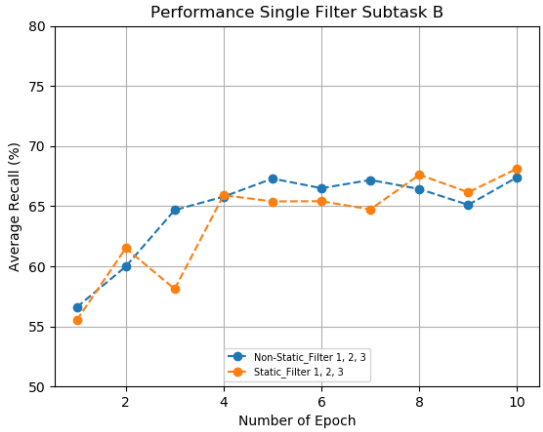
static dimana hasil yang didapatkan lebih optimal daripada *feature maps* dengan nilai 100.

d) Pembahasan Hasil dari Subtask B

Tabel 6.27 Pengukuran Evaluasi Model Terbaik dari Subtask B

Model CNN	***Filter Size***	***Feature Maps***	***AvgRec (%)***
Non-Static	1,2,3	100	67.41_2
Static	1,2,3	300	68.13_1

Dari hasil Tabel 6.26 Menunjukkan bahwa hasil terbaik dihasilkan model *CNN-static* dengan nilai *filter region size* 1, 2, 3 dengan *feature maps* 300. Selanjutnya untuk kinerja dari model terbaik dalam melakukan pengklasifikasian data akan dilihat berdasarkan *confusion matrix* dan juga dilihat berdasarkan perubahan pada setiap *epoch* dari model *non-static* maupun static yang dihasilkan.



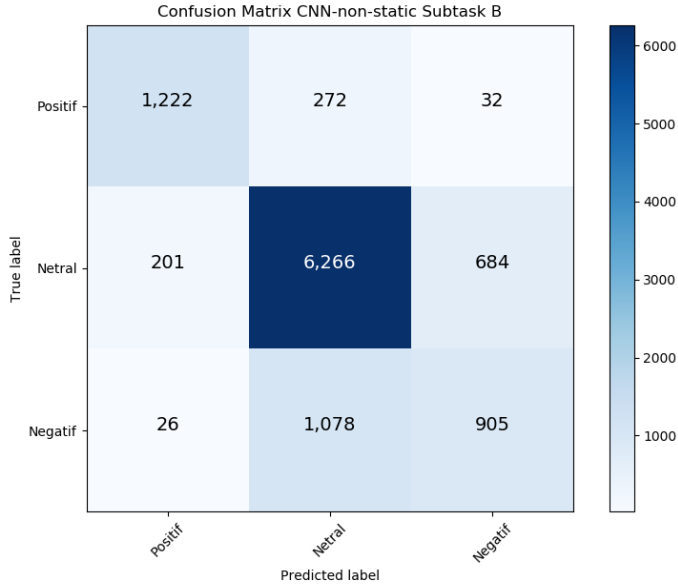
Gambar 6.15. Grafik Performance Model Terbaik *Subtask* B

Pada Gambar 6.15 menunjukkan bahwa grafik *non-static* lebih stabil daripada grafik *static*. Untuk membuktikan hal tersebut maka dilakukan perhitungan *standar deviation* pada

masing-masing grafik. Sehingga nilai *standar deviation* yang dihasilkan model *non-static* adalah 3.404, sedangkan untuk model static menghasilkan nilai 3.943. Dari hasil yang didapat dari nilai *standar deviataion* grafik *non-static* lebih stabil. Jika dilihat dari seberapa baik model dalam melakukan proses pembelajaran, model non-static juga lebih baik karena mengalami peningkatan pada proses pembelajaran. Namun, terjadi tren penurunan pembelajaran pada *epoch* ke 7 setelah itu meningkat kembali pada *epoch* ke 10.

- **Model *CNN-non-Static***

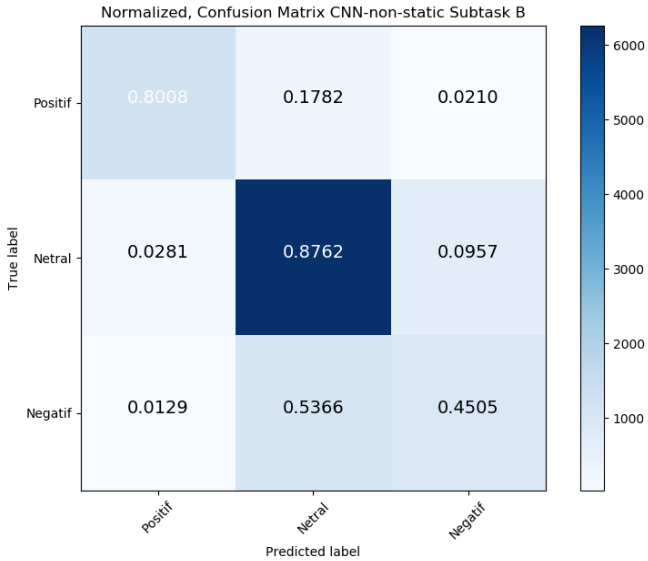
Selanjutnya ditamplikan *confusion matrix* pada *epoch* ke 10 untuk melihat kualitas klasifikasi dari model.



Gambar 6.16 *Confusion Matrix Model Non-static subtask B*

Pada *confusion matrix* yang dihasilkan, kesalahan dalam melakukan klasifikasi untuk label negatif cukup tinggi dilihat dari perbandingan kebenaran model dalam melakukan

klasifikasi, dimana data negatif lebih banyak diklasifikasikan menjadi label netral.



Gambar 6.17 ***Normalized, Confusion Matrix Model Non-static subtask B***

Berdasarkan Gambar 6.14 menunjukkan hasil normalisasi *confusion matrix* untuk melihat perbandingan nilai antar tiap label dalam *confusion matrix.* Normalisasi dilakukan untuk menyamakan tingkat pengukuran dalam skala tertentu, pada proses ini skala yang diambil adalah 0 – 1. Hasil yang didapatkan perbandingan dalam skala yang sama menunjukkan proses klasifikasi pada negatif masih menghasilkan proses klasifikasi yang kurang baik dimana sebesar 0.53 data yang memiliki label negatif diprediksi menjadi label netral, sedangkan yang berhasil melakukan proses klasifikasi hanya sebesar 0.45. Jumlah yang gagal diklasifikasikan secara benar lebih dari 50% jumlah data negatif.

Pengecekan selanjutnya dilihat berdasarkan penyebaran nilai *recall* pada tiap labelnya Perhitungan evaluasi pengukuran

untuk setiap labelnya akan dilihat penyebarannya. Setelah dilakukan perhitungan maka dihasilkan data sebagai berikut :

Tabel 6.28 Perhitungan Recal per Label Model *Non-static Subtask* B

Label	***Recall (%)***
Positif	69.112
Netral	87.592
Negatif	45.520

Berdasarkan hasil perhitungan yang dilakukan pada *subtask* B, nilai *recall* dalam proses klasifikasi paling rendah adalah negatif. Hal ini membuktikan bahwa klasifikasi label negatif sering mengalami kesalahan. Hal ini yang menyebabkan nilai *average recall* model secara keseluruhan menjadi rendah.

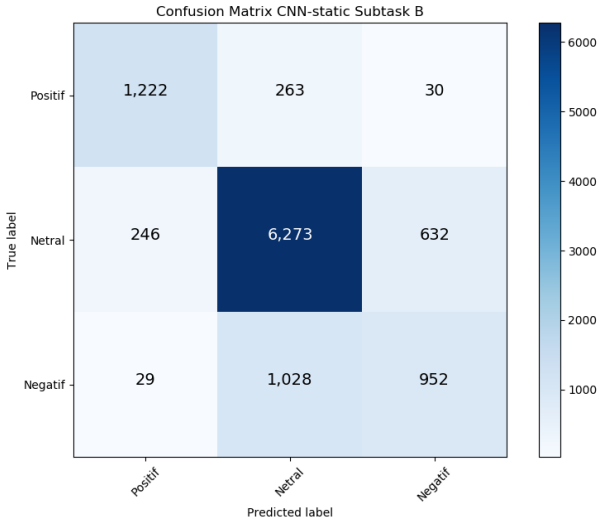
Tabel 6.29 Perbandingan Nilai Pengukuran Evaluasi *Single* dan *Multiple Filter Subtask B Non-static*

Filter Type	***Filter Size***	***AvgRec (%)***
Single Filter	1	66.66_2
Multiple Filter	1, 2, 3	67.41_1

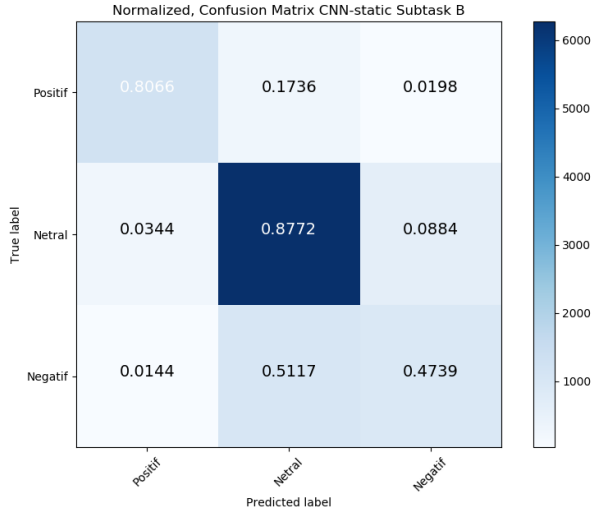
Selanjutnya dilihat performa yang dihasilkan pada proses perubahan *filter size* dan *multiple filter* nilai paling baik dihasilkan oleh model yang menggunakan *multiple filter*. Sehingga, dapat disimpulkan performa model akan semakin baik dengan menggunakan *multiple filter region size*.

- **Model *CNN-static***

Selanjutnya ditamplikan *confusion matrix* pada *epoch* ke 10 untuk melihat kualitas klasifikasi dari model.



Gambar 6.18 ***Confession Matrix Model Static Subtask B***



Gambar 6.19 ***Normalized, Confession Matrix Model Static Subtask B***

Berdasarkan Gambar 6.18 pada *confusion matrix* yang dihasilkan, kesalahan dalam melakukan klasifikasi untuk label negatif cukup tinggi dilihat dari perbandingan kebenaran model

dalam melakukan klasifikasi, dimana data negatif lebih banyak diklasifikasikan menjadi label netral. Hasil yang didapatkan sama dengan hasil yang didapatkan dari model *non-static*

Berdasarkan Gambar 6.19 menunjukkan hasil normalisasi *confusion matrix* untuk melihat perbandingan nilai antar tiap label dalam *confusion matrix.* Normalisasi dilakukan untuk menyamakan tingkat pengukuran dalam skala tertentu, pada proses ini skala yang diambil adalah 0 – 1. Hasil yang didapatkan perbandingan dalam skala yang sama menunjukkan proses klasifikasi pada negatif masih menghasilkan proses klasifikasi yang kurang baik dimana sebesar 0.51 data yang memiliki label negatif diprediksi menjadi label netral, sedangkan yang berhasil melakukan proses klasifikasi hanya sebesar 0.47. Jumlah yang gagal diklasifikasikan secara benar lebih dari 50% jumlah data negatif. Hasil yang didapat dalam melakukan proses klasifikasi masih lebih tinggi dari proses yang dihasilkan pada model *non-static.*

Pengecekan selanjutnya dilihat berdasarkan penyebaran nilai *recall* pada tiap labelnya Perhitungan evaluasi pengukuran untuk setiap labelnya akan dilihat penyebarannya. Setelah dilakukan perhitungan maka dihasilkan data sebagai berikut :

Tabel 6.30 Perhitungan Recal per Label Model Static Subtask B

Label	***Recall (%)***
Positif	71.337
Netral	87.746
Negatif	45.307

Dari hasil perhitungan yang dilakukan pada *subtask* B, nilai *recall* dalam proses klasifikasi paling rendah adalah negatif. Hal ini membuktikan bahwa klasifikasi label negatif sering mengalami kesalahan. Hal ini yang menyebabkan nilai *average recall* model secara keseluruhan menjadi rendah.

Tabel 6.31 Perbandingan Nilai Pengukuran Evaluasi *Single* dan *Multiple Filter Subtask B Static*

Filter Type	*Filter Size*	*AvgRec (%)*
Single Filter	3	65.39_2
Multiple Filter	1, 2, 3	68.13_1

Selanjutnya dilihat performa yang dihasilkan pada proses perubahan *single filter size* dan *multiple filter size* nilai paling baik dihasilkan oleh model yang menggunakan *multiple filter*. Sehingga, dapat disimpulkan performa model akan semakin baik dengan menggunakan *multiple filter region size*.

3) Subtask C

Subtask C merupakan subtask dengan 5 *point scale* dengan label yang digunakan adalah label sangat positif, positif, netral, negatif, dan sangat negatif. Dengan penyebaran data sebagai berikut :

Tabel 6.32 Penyebaran Data pada *Subtask* C

Label	Jumlah Data
Sangat Positif	33
Positif	1493
Netral	7151
Negatif	1553
Sangat Negatif	456
Total Data	10,686

a) Single Filter Region Size Subtask C

Tabel 6.33 Percobaan *Single Filter Region Size Subtask* C Pada Model *CNN-non-static*

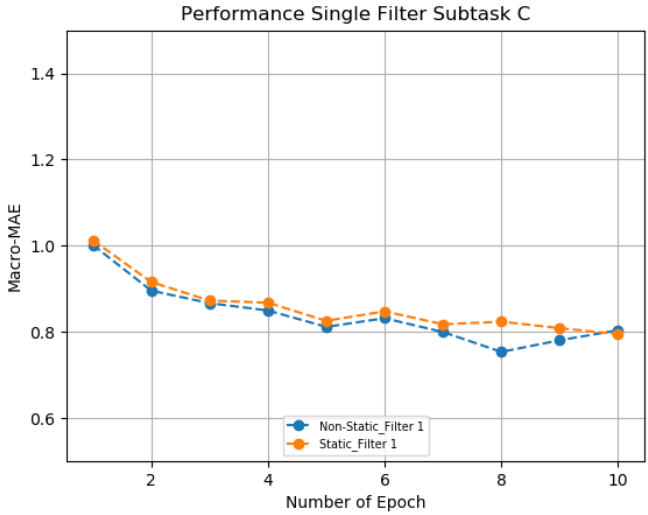
Region Size	*Macro-MAE*	*AvgRec*	*F1-Score*	*Accuracy*
1	0.804_1	41.228_1	31.565_1	75.815_9
2	0.814_2	40.122_2	30.549_2	76.160_5
3	0.855_9	39.952_3	30.477_3	75.966_7
4	0.844_6	39.189_4	29.631_4	76.180_3
5	0.842_5	38.750_7	28.882_8	75.927_8
6	0.834_4	38.707_8	28.956_7	76.245_1
7	0.847_8	38.966_5	29.559_5	76.226_2
8	0.823_3	38.612_9	28.678_{10}	76.161_4
9	0.845_7	38.474_{10}	28.697_9	76.011_6
10	0.865_{10}	38.751_6	29.161_6	64.884_{10}

Tabel 6.34 Percobaan *Single Filter Region Size Subtask* C Pada Type Model *CNN-static*

Region Size	Macro-MAE	AvgRec	F1-Score	Accuracy
1	0.795_1	40.073_3	30.290_3	75.226_{10}
2	0.812_2	40.999_1	31.387_1	75.983_9
3	0.816_3	40.570_2	30.342_2	76.292_6
4	0.810_4	39.753_4	29.420_4	76.620_4
5	0.833_5	39.191_6	28.946_6	76.188_7
6	0.818_6	38.616_8	28.268_9	76.188_8
7	0.817_7	38.345_{10}	28.032_{10}	76.647_3
8	0.798_8	38.568_9	28.270_8	76.938_1
9	0.828_9	38.805_7	28.594_7	76.90_2
10	0.811_{10}	39.247_5	29.260_5	76.442_5

Dari Tabel 6.33 dan Tabel 6.34 dapat disimpulkan bahwa model terbaik dengan baseline perhitungan evaluasi

menggunakan Macro-MAE adalah Filter Region Size 1 untuk type model CNN Non-Static. Sedangkan Untuk Type model CNN static yang memiliki nilai terbaik adalah filter region size 2.



Gambar 6.20 Performa Model Terbaik *Single Filter Subtask* C (semakin kecil semakin baik)

Pada Gambar 6.20 Jika dilihat dari performa yang dilakukan berdasarkan perubahan setiap *epoch* didapatkan bahwa grafik *static* terlihat lebih stabil dibandingkan dengan grafik *non-static*. Untuk melihat kestabilan pada masing-masing grafik maka dilakukan perhitungan *standar deviation* pada model terbaik yang dihasilkan. Sehingga didapatkan hasil *standar deviation* untuk model terbaik pada *non-static* adalah 0.066. Sedangkan untuk model terbaik pada static adalah 0.061. Dari hasil yang didapat nilai paling stabil dihasilkan oleh model *static*.

Setelah mendapatkan nilai *single filter* region terbaik maka kemudian dilakukan kombinasi untuk mendapatkan model terbaik dengan menggunakan *multiple region size*.

b) Multiple Region Size Subtask C

Pada percobaan di *multiple region size* akan dilakukan kombinasi dari hasil terbaik yang didapatkan dari skenario menggunakan *single filter region size*. Dengan melakukan kombinasi dari hasil paling optimal pada *single filter region size* diharapkan dapat meningkatkan performa.

Pada *subtask* C akan digunakan nilai Macro-MAE sebagai baseline perhitungan evaluasi untuk mendapatkan model terbaik pada *multiple region size*. Pada skenario di *subtask* C akan digunakan *feature maps* untuk keseluruhan skenario dengan nilai 100 untuk melihat performa yang didapat. Berikut merupakan hasil yang didapatkan dari hasil percobaan skenario yang dilakukan.

Tabel 6.35 Hasil Evaluasi Pengukuran *Multiple Filter Subtask* C *Non-Static*

Filter Size	*Macro-MAE*	*AvgRec*	*F1-Score*	*Accuracy*
1, 1, 1	0.825	39.173	29.36	74.766
1, 2, 3	0.818	40.448	30.91	76.573

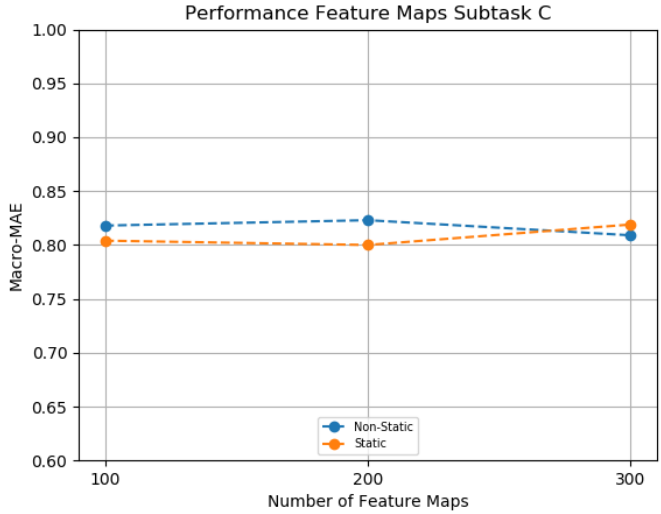
Tabel 6.36 Hasil Evaluasi Pengukuran *Multiple Filter Subtask C Static*

Filter Size	*Macro-MAE*	*AvgRec*	*F1-Score*	*Accuracy*
1, 1, 1	0.819	42.054	30.895	72.794
1, 2, 3	0.804	40.739	30.76	76.414

Dari hasil yang didapatkan pada Tabel 6.35 dan Tabel 6.36 dapat disimpulkan bahwa nilai terbaik yang didapatkan menggunakan pengukuran evaluasi *Macro-MAE* untuk type model CNN *non-static* adalah *multiple region size* dengan nilai 1, 2, 3. Sedangkan untuk *type* model CNN *static* adalah multiple region size dengan nilai 1, 2, 3.

c) Pengaruh dari Feature Maps Subtask C

Pada subtask C telah didapatkan model terbaik pada *multiple region size* dengan nilai 1, 2, 3. Langkah selanjutnya akan dilihat perubahan yang diakibatkan oleh nilai feature maps. Pada percobaan ini akan dilakukan perubahan nilai feature maps dengan nilai 100 sampai 300 dengan menggunakan model terbaik dari type model CNN static maupun non-static.



Gambar 6.21 Perubahan Feature Maps pada Subtask C (semakin kecil semakin baik)

Perubahan pada Gambar 6.21 menunjukkan *feature maps* pada *subtask* C menunjukkan bahwa nilai pada model *CNN-static* mengalami perubahan yang semakin meningkat pada nilai *feature maps* 200, kemudian mengalami penurunan kembali pada *feature maps* 300. Sedangkan pada model *CNN-non-static* mengalami penurunan nilai *feature maps* 200, kemudian mengalami kenaikan pada *feature maps* 300.

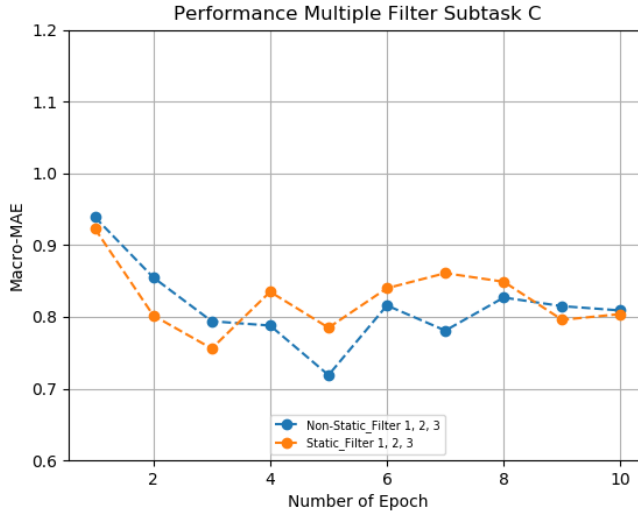
Pada model *CNN-static* nilai pada *feature maps* 300 lebih baik daripada yang dihasilkan pada *feature maps* dengan nilai 100. Sedangkan *CNN-non-static* evaluasi pengukuran terbaik ada pada nilai *feature maps* 100.

d) Pembahasan Hasil dari Subtask C

Tabel 6.37 Pengukuran Evaluasi Model Terbaik dari Subtask C

Model CNN	*Filter Size*	*Feature Maps*	*Macro-MAE*
Non-Static	1,2,3	300	0.809
Static	1,2,3	100	0.804

Dari Tabel 6.37 menunjukkan bahwa nilai yang dihasilkan oleh model *CNN-static* dengan filter region size 1,2,3 dab feature maps 100 memiliki nilai paling baik.

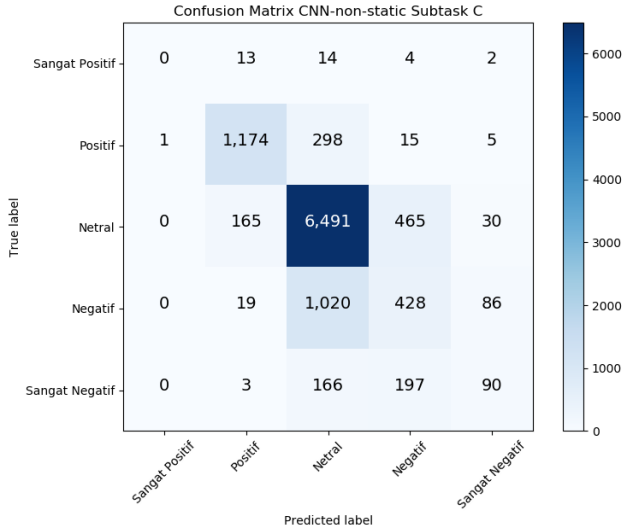


Gambar 6.22 Grafik Performa Model Terbaik Subtask C

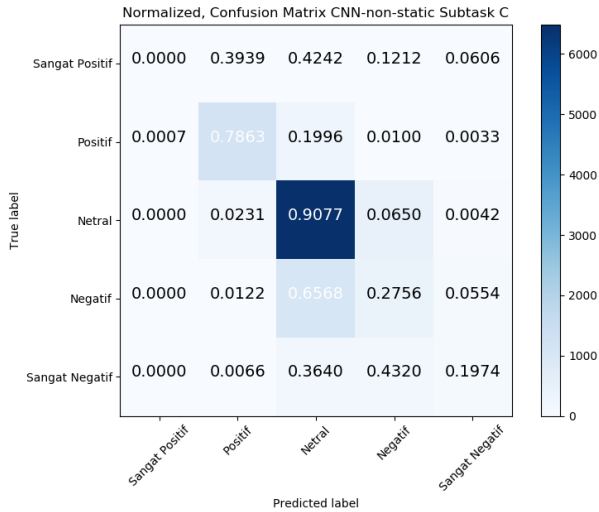
Pada Gambar 6.22 menunjukkan bahwa grafik static lebih stabil daripada grafik static. Untuk membuktikan hal tersebut maka dilakukan perhitungan standar deviation pada masing-masing grafik. Sehingga nilai standar deviation yang dihasilkan model *non-static* adalah 0.053, sedangkan untuk model static menghasilkan nilai 0.044. Namun, jika dilihat dari seberapa baik model dalam melakukan proses pembelajaran,

model non-static terlihat lebih baik karena nilai error yang dihasilkan lebih kecil. Namun, nilai error mulai meningkat pada epoch ke 5.

- Model Non-Static



Gambar 6.23 ***Confusion Matrix Model Non-static subtask C***



Gambar 6.24 ***Normalized, Confusion Matrix Model Non-static subtask C***

Berdasarkan Gambar 6.23 menunjukkan hasil *confusion matrix* pada *epoch* ke 10 untuk melihat kualitas klasifikasi dari model *non-static*. Pada *confusion matrix* yang dihasilkan, kesalahan dalam melakukan klasifikasi untuk label sangat positif memiliki hasil sangat jelek karena tidak ada yang berhasil melakukan proses klasifikasi secara benar. Hal ini juga dikarenakan jumlah data pelatihan yang digunakan sangat sedikit. Pada label sangat negatif juga menghasilkan kesalahan sangat besar dalam proses klasifikasi, kebanyakan dari hasilnya diklasifikasikan menjadi label netral.

Berdasarkan Gambar 6.24 menunjukkan hasil normalisasi *confusion matrix* untuk melihat perbandingan nilai antar tiap label dalam *confusion matrix.* Normalisasi dilakukan untuk menyamakan tingkat pengukuran dalam skala tertentu, pada proses ini skala yang diambil adalah 0 – 1. Hasil yang didapatkan perbandingan dalam skala yang sama menunjukkan proses klasifikasi pada negatif masih menghasilkan proses klasifikasi yang kurang baik dimana sebesar 0.65 dari keseluruhan data yang memiliki label negatif diprediksi menjadi

label netral, sedangkan yang berhasil melakukan proses klasifikasi hanya sebesar 0.27. Hasil yang didapat juga memiliki nilai yang lebih kecil daripada label sangat negatif, dimana hasil proses klasifikasi sebesar 0.43. Sedangkan yang diklasifikasi sebagai netral sebesar 0.36. Sedangkan untuk label sangat positif tidak mengalami proses klasifikasi secara benar sama sekali, hasil dari proses klasifikasi juga cenderung diprediksi sebagai label netral, hasilnya lebih besar dibandingkan dengan label positif yang seharusnya menjadi label terdekat label sangat positif memprediksi hasil label.

Pengecekan selanjutnya dilihat berdasarkan penyebaran nilai *recall* pada tiap labelnya Perhitungan evaluasi pengukuran untuk setiap labelnya akan dilihat penyebarannya. Setelah dilakukan perhitungan maka dihasilkan data sebagai berikut :

Tabel 6.38 Perhitungan Recal per Label Model *Non-static Subtask C*

Label	*Recall (%)*
Sangat Positif	0
Positif	68
Netral	90.77
Negatif	26.10
Sangat Negatif	17.37

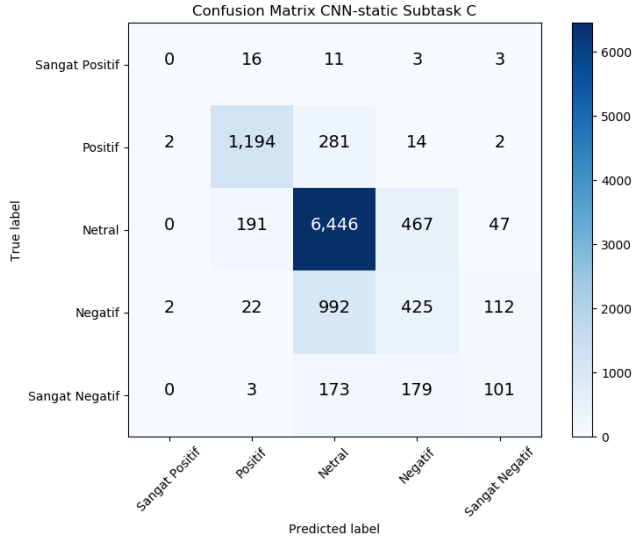
Dari hasil perhitungan yang dilakukan pada subtask C non-static, nilai recall dalam proses klasifikasi yang paling baik adalah netral. Sedangkan nilai label sangat positif, negatif dan sangat negatif menghasilkan nilai yang kurang baik. Hal ini yang menyebabkan proses klasifikasi menghasilkan nilai perhitungan *error* pada evaluasi pengukuran Macro-MAE menjadi besar.

Tabel 6.39 Perbandingan Nilai Pengukuran Evaluasi *Single* dan *Multiple Filter Subtask C Non-static*

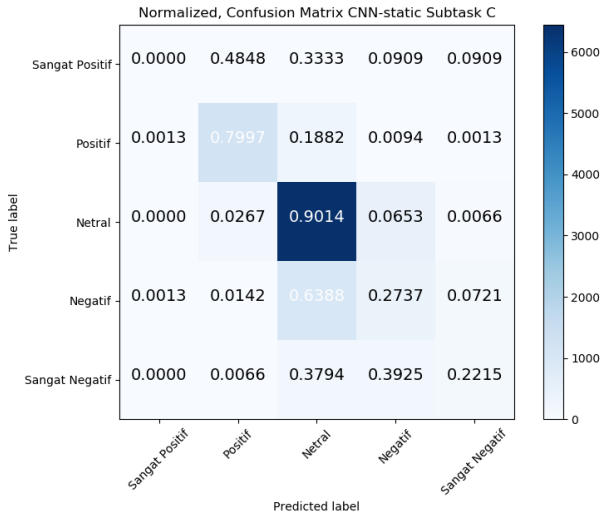
Filter Type	***Filter Size***	***Macro-MAE***
Single Filter	1	0.804
Multiple Filter	1, 2, 3	0.809

Selanjutnya dilihat performa yang dihasilkan pada proses perubahan *single filter size* dan *multiple filter* nilai paling baik dihasilkan oleh model yang *menggunakan multiple filter* dengan selisih antara *single filter* dan *multiple filter* adalah 0.005. Maka dapat disimpulkan peningkatan *filter region size* dapat meningkatkan nilai evaluasi pengukuran.

- Model Static



Gambar 6.25 *Confusion Matrix Model Static subtask C*



Gambar 6.26 ***Normalized, Confusion Matrix Model Static subtask C***

Berdasarkan Gambar 6.25 menunjukkan *confusion matrix* pada *epoch* ke 10 untuk melihat kualitas klasifikasi dari model *static*. Pada *confusion matrix* yang dihasilkan sama dengan yang dihasilkan pada model *non-static* di mana klasifikasi untuk label sangat positif menghasilkan hasil yang buruk karena tidak ada yang berhasil di klasifikasi secara benar pada tabel tersebut. Pada label sangat negatif juga sangat besar kebanyakan dari hasilnya diklasifikasikan menjadi label netral.

Berdasarkan Gambar 6.26 menunjukkan hasil normalisasi *confusion matrix* untuk melihat perbandingan nilai antar tiap label dalam *confusion matrix.* Normalisasi dilakukan untuk menyamakan tingkat pengukuran dalam skala tertentu, pada proses ini skala yang diambil adalah 0 – 1. Hasil yang didapatkan perbandingan dalam skala yang sama menunjukkan proses klasifikasi pada negatif masih menghasilkan proses klasifikasi yang kurang baik dimana sebesar 0.63 dari keseluruhan data yang memiliki label negatif diprediksi menjadi label netral, sedangkan yang berhasil melakukan proses klasifikasi hanya sebesar 0.27. Hasil yang didapat juga memiliki

nilai yang lebih kecil daripada label sangat negatif, dimana hasil proses klasifikasi sebesar 0.39. Namun, nilai ini lebih kecil daripada yang dihasilkan pada model *non-static*. Sedangkan yang diklasifikasi sebagai netral sebesar 0.37. Sedangkan untuk label sangat positif tidak mengalami proses klasifikasi secara benar sama sekali, hasil dari proses klasifikasi cenderung diprediksi sebagai label positif, hasil ini beebeda dengan model *non-static* yang cenderung diprediksi sebagai label netral.

Pengecekan selanjutnya dilihat berdasarkan penyebaran nilai *recall* pada tiap labelnya. Perhitungan evaluasi pengukuran untuk setiap labelnya akan dilihat penyebarannya. Setelah dilakukan perhitungan maka dihasilkan data sebagai berikut :

Tabel 6.40 Perhitungan Recal per Label *Model Static Subtask C*

Label	***Recall***
Sangat Positif	0
Positif	68.927
Netral	90.166
Negatif	25.454
Sangat Negatif	19.151

Dari hasil perhitungan yang dilakukan, nilai recall dalam proses klasifikasi yang paling baik adalah netral. Sedangkan nilai label sangat positif, negatif dan sangat negatif sangat jelek. Hal ini yang menyebabkan proses klasifikasi menghasilkan nilai perhitungan error pada evaluasi pengukuran Macro-MAE menjadi besar.

Tabel 6.41 Perbandingan Nilai Pengukuran Evaluasi *Single* dan *Multiple Filter Subtask C Static*

Filter Type	***Filter Size***	***Macro-MAE***
Single Filter	1	0.795
Multiple Filter	1, 2, 3	0.804

Selanjutnya dilihat performa yang dihasilkan pada proses perubahan *single filter size* dan *multiple filter* nilai paling baik dihasilkan oleh model yang menggunakan *single filter*. Perbedaan dari hasil perubahan *filter size* sebesar 0.009.

c. Pembahasan Hasil Percobaan Tiga Subtask

Berdasarkan percobaan yang telah dilakukan pada tiga *subtask* berbeda didapatkan model terbaik pada tiap subtaksnya. Hasil evaluasi pengukuan model terbaik yang dihasilkan dapat dilihat sebagai berikut dimana evaluasi pengukuran yang digunakan untuk melakukan perbadingan antar tiap subtask adalah *average recall*.

1) Pengaruh dari *Filter Region Size*

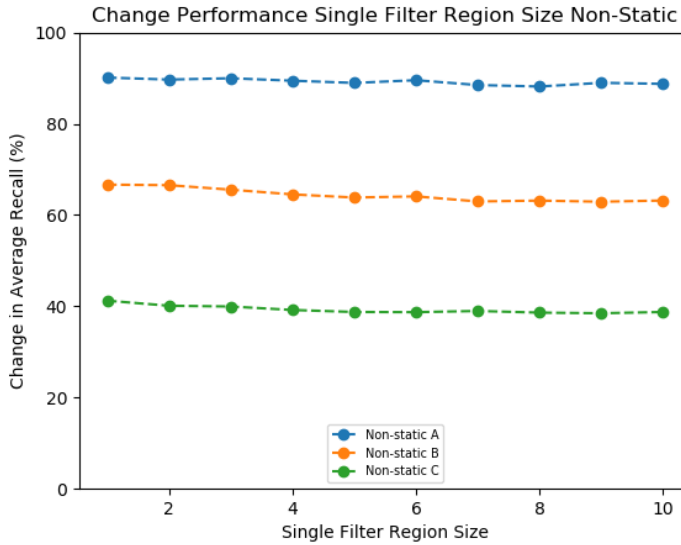
Berdasarkan hasil dari penelitian dengan melihat pengaruh dari perubahan pada nilai *filter region size* dengan menggunakan nilai region size dari 1 hingga 10 pada tiap subtask didapatkan hasil sebagai berikut :

Tabel 6.42 Model Terbaik Setiap Subtask Berdasarkan *Single Filter Size*

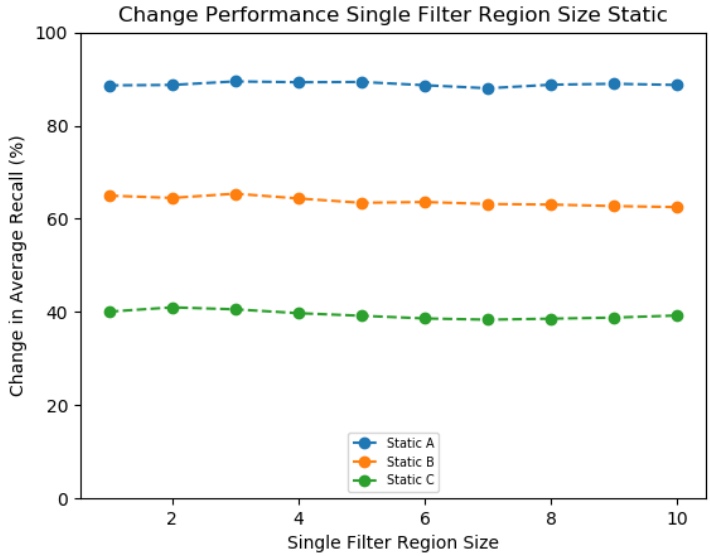
Subtask	Model CNN	*Single Filter Size*	*AvgRec*
A	*CNN-non-static*	1	**90.11**
	CNN-static	3	89.47
B	*CNN-non-static*	1	**66.66**

	CNN-static	3	65.39
C	*CNN-non-static*	1	**40.50**
	CNN-static	1	40.07

Berdasarkan hasil dari Tabel 6.42 didapatkan hasil pada CNN-*non-static* selalu menghasilkan nilai *filter size* 1. Sedangkan, untuk model CNN-*static* nilainya bervariasi antara filter size 1 dan filter size 3.



Gambar 6.27 Pengaruh *single filter region size* CNN-*non-static* pada perubahan evaluasi pengukuran



Gambar 6.28 Pengaruh *single filter region size* CNN-*static* pada perubahan evaluasi pengukuran

Berdasarkan Gambar 6.27 dan Gambar 6.28 dapat dilihat perubahan pengukuran evaluasi yang terjadi pada *single filter region size* dari 1 hingga 10. Dapat dilihat setelah region size 3 pada model CNN-*non-static* maupun CNN-*static.* Dapat disimpulkan pada studi kasus dataset menggunakan *e-commerce* semakin tinggi nilai single filter size akan semakin memperkecil hasil evaluasi pengukuran model.

Tabel 6.43 Model Terbaik Setiap Subtask Berdasarkan *Multiple Filter Size*

Subtask	Jumlah Label	*MultipleFilter Size*	*CNN-non-static*	*CNN-static*
A	2 label	1,2,3	**90.34**	89.68
B	3 label	1,2,3	67.41	**68.13**

C	5 label	1,2,3	40.45	**40.74**

Kemudian, setelah dilakukan pengukuran pada single filter selanjutnya dilakukan pengukuran pada *multiple region size* pada tiap subtask. Berdasarkan Tabel 6.43 nilai evaluasi pengukuran meningkat dengan melakukan kombinasi dari *filter region size* yang berbeda berdasarkan dari nilai evaluasi pengukuran terbaik pada *single filter region size*. Kemudian, pada hasil evaluasi pengukuran menunjukkan pada proses *multiple region size* nilai *average recall* yang dihasilkan semakin kecil seiring dengan pertambahan jumlah label, hal ini disebabkan oleh keragaman label yang ada pada tiap subtask. Proses pembelajaran makin sulit untuk menentukan label yang akan dihasilkan. Maka dapat disimpulan berdasarkan percobaan yang dilakukan menambah jumlah filter dari proses *single filter size* menjadi *multiple filter size* meningkatkan hasil proses evaluasi pengukuran yang dilakukan menjadi lebih baik pada semua jenis subtask. Namun, jika dilihat perbedaan pada subtask B adalah 0.72 dan pada subtask C adalah 0.29. Hasil ini merupakan nilai yang kecil. Pada Tabel 6.43 didapatkan bahwa nilai *filter region size* terbaik pada semua subtask bernilai 1,2,3 dengan hasil rata-rata penggunaan *feature maps* paling baik bernilai 100.

2) Proses Uji Signifikansi

Berdasarkan pada penelitian yang dilakukan dengan menggunakan *multiple filter* pada subtask B dan subtask C didapatkan bahwa model terbaik merupakan model *CNN-static*. Sehingga dilakukan proses uji signifikansi untuk menentukan apakah hipotesis yang dibuat akan diterima atau ditolah dari hasil percobaan yang dilakukan pada subtask B dan subtask C.

Terdapat dua hipotesis yang dibuat yaitu hipotesis nul (H_0) dan hipotesis alternative (H_A). Berikut merupakan proses uji signifikansi yang dilakukan pada subtask C dan subtask C. Model dengan parameter terbaik yang dihasilkan pada subtask

B dan subtask C akan dilakukan proses training sebanyak 15 kali untuk tiap model *CNN-static* dan *CNN-non-static*.

a) Subtask B

Percobaan yang dilakukan dengan menggunakan subtask B akan menggunakan hasil dari pengukuran evaluasi yang menjadi baseline pada subtask B. Pengukuran evaluasi yang digunakan adalah *average recall*.

Tabel 6.44 Datat Percobaan Uji Signifikansi Subtask B

Percobaan	*CNN-static*	*CNN-non-static*
1	66.642	66.366
2	66.672	65.566
3	66.609	65.687
4	66.885	65.856
5	66.231	65.267
6	66.131	66.258
7	66.502	65.752
8	66.454	65.724
9	67.16	64.978
10	66.602	65.454
11	66.911	65.595
12	67.046	65.611

13	67.404	64.398
14	66.17	64.933
15	66.803	65.826

Level signifikan : 0.05

H_0 : Tidak ada perbedaan antara model CNN-static dengan model *CNN-non-static*

H_A : Ada perbedaan antara model CNN-static dengan model *CNN-non-static*

Proses perhitungan signifkansi dapat dilihat pada perhitungan berikut :

$\mu_s = 66.68147$ $s_s = 0.364252$ $Sd = 0.160573$

$\mu_{ns} = 65.5514$ $s_{ns} = 0.504058$

$t = 7.03773$ $df = 28$

Berdasarkan hasil perhitungan yang dilakukan didapatkan nilai p-value < 0.00001. Hasil yang didapatkan adalah signifikan karena nilai $p <$ 0.05. Maka, dapat disimpulkan ada perbedaan signifikan antara model *CNN-static* dan *CNN-non-static* dengan hasil nilai rata-rata model *CNN-static* lebih baik dari *CNN-non-static.*

b) Subtask C

Percobaan yang dilakukan dengan menggunakan subtask C akan menggunakan hasil dari pengukuran evaluasi yang menjadi baseline pada subtask C. Pengukuran evaluasi yang digunakan adalah *Macro-Mean Absolute Error*.

Tabel 6.45 Datat Percobaan Uji Signifikansi Subtask C

Percobaan	*CNN-static*	*CNN-non-static*
1	0.786	0.796
2	0.788	0.819
3	0.808	0.802
4	0.811	0.806
5	0.8	0.791
6	0.789	0.816
7	0.784	0.791
8	0.812	0.806
9	0.781	0.806
10	0.783	0.8
11	0.796	0.818
12	0.79	0.801
13	0.797	0.818
14	0.802	0.801
15	0.776	0.818

Level signifikan : 0.05

H_0 : Tidak ada perbedaan antara model CNN-static dengan model *CNN-non-static*

H_A : Ada perbedaan antara model CNN-static dengan model *CNN-non-static*

Proses perhitungan signifkansi dapat dilihat pada perhitungan berikut :

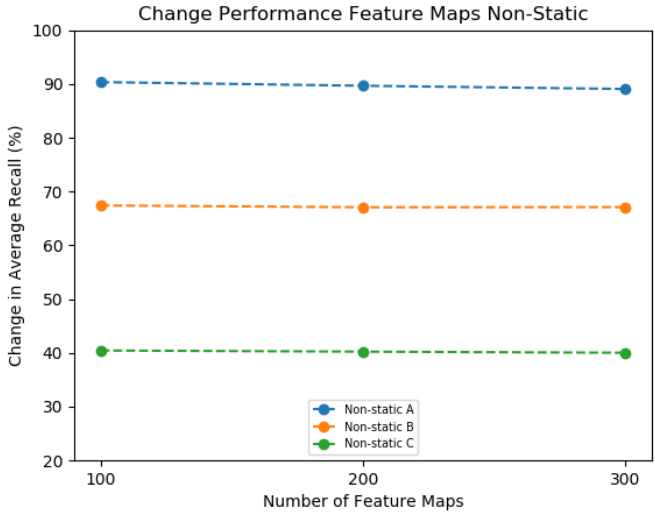
$\mu_s = 0.793533$ $s_s = 0.011256$ $Sd = 0.003859$

$\mu_{ns} = 0.805933$ $s_{ns} = 0.00983$

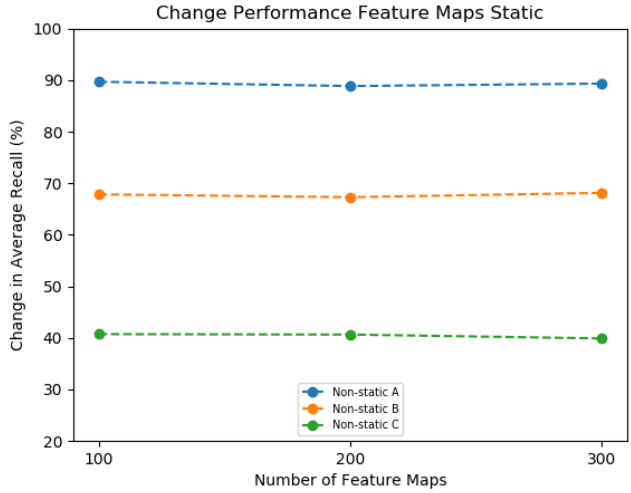
$t = -3.21359$ $df = 28$

Berdasarkan hasil perhitungan yang dilakukan didapatkan nilai p-value < 0. .00329. Hasil yang didapatkan adalah signifikan karena nilai $p <$ 0.05. Maka, dapat disimpulkan ada perbedaan signifikan antara model *CNN-static* dan *CNN-non-static* dengan hasil nilai rata-rata model *CNN-static* lebih baik dari *CNN-non-static.*

3) Pengaruh dari Jumlah *Feature Maps*



Gambar 6.29 Pengaruh *feature maps* CNN-*non-static* pada perubahan evaluasi pengukuran



Gambar 6.30 Pengaruh *feature maps CNN-static* pada perubahan evaluasi pengukuran

Berdasarkan hasil dari perubahan *feature maps* yang digunakan pada setiap model terbaik pada tiap subtaksnya. Rata-rata *feature maps* 100 menjadi model yang terbaik. Hal ini juga dapat dilihat dari hasil pada Gambar 6.29 dan Gambar 6.30 yang menunjukkan nilai evaluasi pengukuran pada model yang dihasilkan cenderung turun pada *feature maps* 200 dan pada beberapa model meningkat kembali pada *feature maps* 300. Namun, peningkatan yang naik rata-rata pada tiap *subtask* tidak lebih tinggi dari nilai evaluasi pengukuran yang dihasilkan pada nilai *feature maps* 100.

4) Perbandingan dengan Algoritma lain

Berdasarkan percobaan algoritma *Convolutional Neural Network* telah didapatkan hasil terbaik untuk tiap *subtask*. Maka untuk mengukur tingkat performa yang dihasilkan dalam proses pembelajaran dilakukan perbandingan dengan algoritma lain. Algoritma yang digunakan untuk dilakukan perbandingan adalah *Naïve Bayes* dan *Support Vector Machine (SVM)*. Pada percobaan ini semua algoritma menggunakan metode *k-fold cross validation* sebagai proses validasi keakuratan model dengan jumlah fold sebesar 10. Berikut adalah hasil dan perbandingan yang dilakukan antara algoritma *Naïve Bayes* dan *Support Vector Machine (SVM)* dengan model terbaik pada algoritma *Convolutional Neural Network* dengan menggunakan evaluasi pengukuran *Average Recall*.

Tabel 6.46 Perbandingan *Convolutional Neural Network* dengan Algoritma lain

Subtask	***Naïve Bayes***	***SVM***	***Convolutional Neural Network***
A	85.61	86.95	90.34
B	54.28	57.02	68.13

C	32.20	34.50	40.74

Berdasarkan pada Tabel 6.44 menunjukkan bahwa dibandingkan dengan algoritma *Naïve Bayes* dan *Support Vector Machine (SVM)* hasil evaluasi pengukuran yang dilakukan didapatkan nilai dari *Convolutional Neural Network* mendapatkan hasil paling baik dibandingkan dengan algortima lain. Dapat disimpulkan bahwa proses pembelajaran model yang dilakukan oleh *Convolutional Neural Network* lebih baik dibandingkan dengan algoritma lain yang dalam hal ini adalah *Naïve Bayes* dan *Support Vector Machine (SVM).*

BAB VII
KESIMPULAN DAN SARAN

Pada bab ini dibahas mengenai kesimpulan dari semua proses yang telah dilakukan dan saran yang dapat diberikan untuk pengembangan yang lebih baik

7.1. Kesimpulan

Kesimpulan yang didapatkan dari proses pengerjaan tugas akhir yang telah dilakukan antara lain :

1. Proses pengumpulan data di media sosial Twitter dapat dikumpulan dengan metode *crawling*. Proses *crawling* data dengan menggunakan *search API* lebih cepat dari pada menggunakan *streaming API*. Proses dengan menggunakan *search API* bisa didapatkan data dalam jumlah banyak dengan lebih cepat. Selain itu, proses menggunakan *search API* dapat mengatasi limit kata yang dibatasi dalam proses *crawling* dengan menggunakan tweet_mode='extended' untuk mendapatkan *full text* dari media sosial Twitter

2. Metode pra-pemrosesan yang dilakukan pada data disesuaikan dengan topik yang digunakan. Contohnya *cleansing* data disesuaikan dengan bentuk dan keragaman data. Proses *cleansing* yang tepat akan mempengaruhi hasil dari evaluasi pengukuran terhadap data. Pada dataset *e-commerce* proses pra-pemrosesan ditambahkan dengan menghapus angka pada dataset.

3. Hasil dari tingkat persetujuan pada proses pelabelan dari 3 pelabel menghasilkan nilai rata-rata koefisien kappa sebesar 0.374. Hal ini menandakan kesepakatan antara ketiga pelabel masih sedang dalam melakukan pelabelan terhadap data.

4. Proses pembuatan model *Word2Vec* menggunakan algoritma *Word2Vec* dengan *learning algorithm* yang digunakan adalah *CBOW* dan *Skip-Gram*. Hasil yang

didapatkan dari proses percobaan *learning algorithm Skip-Gram* menghasilkan nilai lebih baik daripada *CBOW*.

5. Jumlah kosa kata terdapat pada *output* dari model *word embedding* serta sumber data yang digunakan akan mempengaruhi hasil dari proses *embedding* terhadap dataset yang digunakan dalam proses pelatihan.
6. Jumlah *single filter region size* pada pada setiap subtask paling baik berada pada range 1-3. Hal ini dikarenakan semakin tinggi nilai *single filter region size*, semakin menurun hasil evaluasi pengukuran terhadap model.
7. Perubahan terhadap *filter region size* dari *single* menjadi *multiple* rata-rata dapat meningkatkan hasil evaluasi pengukuran baik pada model *non-static* maupun model *static*.
8. Semakin banyak jumlah label dalam proses pelatihan membuat nilai evaluasi pengukuran semakin menurun. Hal ini disebabkan oleh kompleksitas yang semakin tinggi dalam proses klasifikasi untuk menentukan label suatu teks.
9. Berdasarkan hasil klasifikasi *confusion matrix* pada tiap subtask disimpulkan bahwa kesalahan prediksi label paling sering terjadi pada label negatif di mana hasil yang didapatkan menunjukkan label di prediksi menjadi label netral. Hal ini dipengaruhi dari kemampuan pelabel yang melakukan tagging terhadap dataset untuk membedakan sentimen terhadap suatu teks di mana para pelabel sulit untuk menentukkan label dalam suatu teks termasuk label negatif atau netral.
10. Dibandingkan dengan algoritma *Naive Bayes* dan SVM dalam proses klasifikasi. Model yang dihasilkan oleh *Convolutional Neural Network* lebih baik dalam melakukan proses klasifikasi. Hal ini dibuktikan dengan evaluasi pengukuran yang dihasilkan lebiih tinggi daripada dua algoritma yang lain.

7.2. Saran

Dari pengerjaan tugas akhir ini, adapun beberapa saran untuk pengembangan penelitian ke depan.

1. Sumber data pada dataset ditambahkan untuk beberapa media sosial yang berbeda.
2. Proses cleansing yang dilakukan pada tahap pra-pemrosesan data ditambahlan untuk menghapus data yang termasuk dalam karakter ASCII.
3. Proses pelabelan yang dilakukan dibuat standar bahasa dalam mengkategorikan setiap label positif, netral dan negatif. Hal ini untuk dapat dijadikan sebagai panduan bagi annotator dalam melakukan proses pelabelan daya
4. Data untuk proses tagging diperbanyak lagi untuk setiap label. Khusus pada topik tentang e-commerce diperbanyak pada data negatif untuk meningkatkan proses klasifikasi.
5. Dalam melakukan percobaan untuk mengukur kualitas algoritma *word embedding* diuji coba dengan menggunakan data dari sumber yang sama namun dengan parameter dan algoritma yang berbeda.
6. Penyebaran data dalam proses pelatihan K-fold cross validation dapat dilihat pada proses training setiap folding untuk melihat kesalahan prediksi yang sering terjadi pada dataset agar dataset dapat dievaluasi kembali.
7. Dalam melakukan pemilihan pelabel juga akan mempengaruhi kualitas dari hasil dataset. Kualitas data tersebut berpengaruh terhadap pemahaman seseorang tentang suatu topik.
8. Jumlah percobaan pada proses training yang dilakukan pada setiap scenario sebaiknya dilakukan berulang kali. Hal tersebut untuk mengakomodasi perubahan nilai yang dihasilkan pada setiap proses training dengan mengkalkulasi nilai minimal, maksimal dan rata-rata.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] Asosiasi Penyelenggara Jasa Internet Indonesia - APJII, "Penetrasi & Perilaku Pengguna Internet Indonesia - Survey 2017," p. 34, 2017.

[2] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality," pp. 1–9.

[3] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural Language Processing (almost) from Scratch," 2011.

[4] Y. Kim, "Convolutional Neural Networks for Sentence Classification," 2014.

[5] Y. Zhang and B. Wallace, "A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification," 2015.

[6] M. Cliche, "BB_twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNs and LSTMs," no. 2014, pp. 573–580, 2017.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," pp. 1–12, 2013.

[8] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of Tricks for Efficient Text Classification," 2016.

[9] T. M. Mitchell, *Machine learning in ecosystem informatics and sustainability*. 1997.

[10] I. Goodfellow, Y. Bengio, and A. Courville, "Machine Learning Basics," *Intell. Sens. Networks Integr. Sens. Networks, Signal Process. Mach. Learn.*, pp. 3–29, 2012.

[11] "sentiment analysis | Definition of sentiment analysis in English by Oxford Dictionaries." [Online]. Available: https://en.oxforddictionaries.com/definition/sentiment_analysis. [Accessed: 17-Feb-2018].

[12] S. S. Neeraj, "Intuitive Understanding of Word Embeddings: Count Vectors to Word2Vec," pp. 1–31,

2017.

[13] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," 2016.

[14] "fastText – Facebook Research." [Online]. Available: https://research.fb.com/fasttext/. [Accessed: 24-Feb-2018].

[15] W. Yih, X. He, and C. Meek, "Semantic Parsing for Single-Relation Question Answering," *Proc. 52nd Annu. Meet. Assoc. Comput. Linguist. (Volume 2 Short Pap.*, pp. 643–648, 2014.

[16] Andrew Ng *et al.*, "Unsupervised Feature Learning and Deep Learning Tutorial." [Online]. Available: http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/. [Accessed: 01-Feb-2018].

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient Based Learning Applied to Document Recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[18] M. A. Hearst, "Untangling text data mining," *Proc. 37th Annu. Meet. Assoc. Comput. Linguist. Comput. Linguist.* -, pp. 3–10, 1999.

[19] J. Ananiadou, S., McNaught, *Text Mining for Biology and Biomedicine (book)*, vol. 53, no. 9. 2005.

[20] C. D. Manning, P. Raghavan, and H. Schutze, "An Introduction to Information Retrieval," *Online*, no. c, p. 569, 2009.

[21] A. Copestack, "Natural Language Processing," *Nat. Lang. Process.*, pp. 2003–2004, 2004.

[22] B. Liu, *Web Data Mining*. 2011.

[23] I. H. Witten, E. Frank, and M. a Hall, *Data Mining: Practical Machine Learning Tools and Techniques (Google eBook)*. 2005.

[24] G. Pant, P. Srinivasan, and F. Menczer, "Crawling the Web," *Springer*, pp. 153--177, 2004.

[25] P. Nakov, A. Ritter, S. Rosenthal, V. Stoyanov, and F. Sebastiani, "{SemEval}-2016 Task 4: Sentiment

Analysis in {T}witter," *Proc. 10th Int. Work. Semant. Eval.*, pp. 1–18, 2016.

[26] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," 2012.

Halaman ini sengaja dikosongkan

BIODATA PENULIS

Penulis lahir di Masohi pada tanggal 02 Juni 1996. Merupakan anak pertama dari 3 bersaudara. Penulis telah menempuh beberapa pendidikan formal yaitu; SD Negeri 2 Masohi, SMP Negeri 1 Masohi, dan SMA Negeri Siwalima Ambon.

Pada Tahin 2014 pasca kelulusan SMA, penulis melanjutkan pendidikan di Jurusan Sistem Informasi Fakultas Teknologi Informasi dan Komunikasi – Institut Teknologi Sepuluh Nopember (ITS) Surabaya dan terdaftar sebagai mahasiswa dengan NRP 05211441007007. Selama menjadi mahasiswa, penulis mengikuti berbagai kegiatan kemahasiswaan seperti beberapa kepanitian serta pernah menjabat sebagai Staf Departemen Media Informasi BEM FTIK ITS dan pada tahun ketiga menjabat sebagai Staf Ahli Departemen Media Informasi BEM FTIK ITS. Di bidang akademik, penulis aktif menjadi asisten dosen praktikum pada mata kuliah Sistem Cerdas. Selain itu, pada tahun 2017 penulis menjadi Top 10 Finalis pada perlombaan ASEAN Data Science Explores yang diadakan oleh SAP dan ASEAN FOUNDATION di Jakarta.

Pada tahun keempat, karena penulis memiliki ketertarikan di bidang pengolahan data, maka penulis mengambil bidang minat Akuisisi Data dan Diseminasi Informasi (ADDI). Penulis dapat dihubungi melalui *email* di aldendelfian@gmail.com.

Halaman ini sengaja dikosongkan

LAMPIRAN A

Contoh Data Mentah Twitter untuk Word Embedding

Id_Tweet	Message
979309087551774720	Angga suka tebar pesona. Tapi gadis itu miliki pesona bertebaran yang harus Angga berjuang kumpulkan. Pada hari itu, suatu putusan terjadi. Bahwa, Angga miliki cinta pertama. Kakak kelas tiga prodi IPA yang buat ia berkelana.
962171455524896773	Atensinya terpaku pada sudut ruangan dengan sebuah suara decitan-decitan kecil. Kedua maniknya bisa melihat dengan jelas makhluk apa yang ia lihat. Dengan segera, ia pun berjalan pelan dan sedkit mengendap namun dengan langkah yang lebar.
960847334148993024	Baru makan beberapa, mendadak pandangannya menjadi kabur. Ia tidak bisa melihat dengan jelas. Ia mengerjapkan matanya berkali-kali, namun pandangannya tetap kabur. "I-ini kenapa?" #KTZ905
974614732211609600	Berjalan melewati beberapa ruangan dan menaiki beberapa anak tangga menuju kelas Demonology. Sesekali melirik jam tangannya untuk mengetahui apa ia terlambat atau tidak. #KTZ905
981131801056894977	Bincang-bincang ringan di selaras trotoar jalan. Tawa membangun suka cita bahwa harus menerima kini ia sedang yang dipuja. "Teteh curang."

	"Kenapa gitu?" "Teteh tau nama saya, tapi saya nggak tau nama Teteh siapa?"
975373889491513347	Dan aku yakin,aku tak akan bisa lolos karena aku hanyalah anak keluaran SMA." Ia pun menghela napas. " Lagipula,aku bekerja untuk membiyayai adikku bersekolah agar ia tak akan semenderita kakaknya." Tawanya kemudian [@Maid_Urushibara]
960862616229834752	Dan kemudian mengisi absen.
962183977476931584	Derap langkah kakinya hampir tidak terdengar sama sekali, ia membawa beberapa benda untuk ia gunakan untuk ritualnya. Tentu tidak benar menamakan hal ini ritual, tapi, ia memang sedang ingin melakukan sesuatu. Disebuah tempat yang sudah ia siapkan.
960863808913776641	Ia melihat sensei yang masuk dengan cara memanjat. Dan segera membenarkan posisi duduknya. "Selamat malam, sensei." Ucapnya. #KTZ905
963055709456801792	Ia memasuki ruangan yang pencahayaannya remang-remang. Kemudian, melihat seseorang telah berdiri di sana, ia pun hanya bisa diam.
960043856078151680	Ia menunjukan senyuman tipis dari bibirnya, kemudian memeluk kakeknya. Ia tahu betul, kakeknya tidak ingin ditinggal. "Aku akan baik-baik saja, percayalah." "Jika kau ingin, kami bisa mengajarimu." ucapnya kembali.
960398744851132416	ID saya, 905. Mengapa 905? Tanyakan kepada peta, karena saya tidak tahu. Hobby saya, memancing ikan. Karena, memancing keributan itu tidak baik.

	Kenapa saya memancing ikan? Kalau ada kucing lewat, tinggal saya lemparkan ikan.
975373452713500672	Jelasnya. " Alasanku kesini sebenarnya aku hanya ingin bekerja,tidak ada maksud lebih dan kebetulan aku mendapati lowongan ini di internet, meskipun mungkin bekerja di amerika lebih menguntungkan namun persaingan disana sangatlah ketat
980464173476794373	Karena memang radius mobik dan kursi mereka bersinggah tak terlampau jauh. Sudah diperkiraan. Anggukan kecil dari Keira membuat Angga segera bangkit. Menjemput minum untuk si nyonya cilik. (@rentetanilusi)
962183979716640768	Kedua maniknya seolah menyapu pandangan sekitar, kepalanya terus menoleh ke kanan, depan, belakang. Tidak ingin seorang pun mengetahui apa yang dilakukannya. Mungkin terlihat mencurigakan, tapi ia sangat benci menjadi tontonan.
981133552635924481	Lagi-lagi senyuman, tahu memang apa yang jadi kelemahan. Irama kembali tak tertata, berusaha melawan. Apa daya tak sanggup. Sampai lupa tungkai mengarah kemana, "Eh Ga, itu rumah aku. Duluan ya? Hati-hati." Seperti tak berakal dibuat. Dasar, gadis pujaan!
979312999750975490	Melipir, melipir, sepedanya melipir ke sisi jalan untuk menghampiri rambut sebahu. Sepeda tepat berhenti dihadapan sang Hawa. Pun juga yang berjalan. Bergeming di tempat. Wajah keduanya saling bertemu, Kembali.

Halaman ini sengaja dikosongkan

LAMPIRAN B

Contoh Data Mentah Twitter untuk Convolutional Neural Network

991588284169183232	Tokopedia	0	0	0	(1) Berniat memberikan kado spesial untuk yang terkasih? Bisaa... Indogold melalui official store di Tokopedia memberikan berbagai pilihan paket bundling emas dengan perhiasan perak yang bisa Anda jadikan sebagai bingkisan cantik. https://t.co/zC7SXyTUA0
993774046436253696	Tokopedia	0	0	0	{mu} halo maaf bgt oot, bisa ga sih bayar tokopedia di indomart tp nyuruh abang gojek dan sklian beli shampo? Nnt byr sm abanya pake gopay, makasih
993752957018628096	Tokopedia	0	0	0	{mu} maaf oot, toko ini original kan ya? Btw ini di tokopedia, makasih https://t.co/LGf0BxnCvk
991605985763471360	TokopediaCare	1	1	0	@TokopediaCare Ok.. terima kasih atas infonya..krn cashback trx yg seblum2nya

					selalu real-time msk setelah pembayaran dilakukan..salam
995477640634032128	TokopediaCare	1	1	0	@TokopediaCare Ok..terimakasih atas bantuannya
995115399158026240	TokopediaCare	1	1	0	@TokopediaCare Ok.Makasih kak
994043868218773504	Tokopedia	1	1	0	@TokopediaCare Okay deh. Sedianya pesanan tsb utk hadiah kado. Tp ndak papa ya. Terimakasih atas attensinya ya. Sukses selalu TOKOPEDIA
990855694147502080	TokopediaCare	1	1	0	@TokopediaCare okay kak terima kasih
941208137087909888	TokopediaCare	-1	-1	0	@tokopediacare saya tarik dana sudah hampir 2 jam blom masuk ke rekening, biasanya cepat ngga smpe 10 menit sudah masuk
956464265270984704	TokopediaCare	-2	-2	0	@TokopediaCare saya tunggu dana masuk ke saldo dana saya 1x24jam!!!
953157071611969536	BlibliCare	-1	-1	0	@BlibliCare sudah dicoba min tetap tidak bisa, pertama saat di install bisa tetapi setelah dicoba masuk kedua kali tetap tidak bisa min :(
953506766398013440	BlibliCare	0	0	0	@BlibliCare Sudah didm min

988998459423272961	BukaBantuan	0	0	0	@BukaBantuan selamat siang, orderan dengan no transaksi 180797769632 udah saya kirim dengan no resi j&t 888066996904, tapi dicancel oleh sistem. mohon bantuan tim BL, terima kasih banyak
995534907844608002	BukaBantuan	0	0	0	@BukaBantuan selamat siang.., saya salah transfer ke rek. Bukalapak, ada Bukalapak dapat memasukkannya ke dompet Bukalapak Saya ?, mohon bantuannya, Terima kasih
992682377242525696	BukaBantuan	0	0	0	@BukaBantuan selamat sore admin buka lapak,saya hanya mau konfirmasi,di aplikasi, barang sudah sampai,tetapi sebenarnya belum, status pengiriman nya adalah gagal antar,jadi saya belum anggap transaksi selesai ya admin Terimakasih https://t.co/HCiMbnmPJr
994130448371539968	BukaBantuan	0	0	0	@BukaBantuan selamat sore bukalapak mohon untuk mengembalikan dana pelanggan sesuai permohonan pelanggan dengan nomor transaksi 180837529653 dan bukti SS nya . terima kasih https://t.co/xM86GqNce3

994898247620726784	BukaBantuan	0	0	0	@BukaBantuan selamat sore bukalapak...bagaimana cara menghilangkan jejak iklan saya di google,padahal iklan di bukalapak sudah saya hapus...mohon bantuannya,terima kasih
994509033758314496	BukaBantuan	0	0	0	@BukaBantuan selamat sore min, bantu cek min 180841810125 Dari tadi Drivernya gk datang2. Trimakasih
991629397563133952	BukaBantuan	0	0	0	@BukaBantuan selamat sore untuk pesanan no.transaksi :180818281228 barang SUDAH SAYA TERIMA 3pc dengan baik... Tapi status masih... Di proses pelapak... Minta bantuan remit dana ke pelapak ya... Terima kasih
993794381525078016	BukaBantuan	0	0	0	@BukaBantuan selamat sore,, mohon infonya,,, saya order di bukalapak dan sdh tranfser tp smpai skrg blm terverifikasi pembayarannya. Terimakasih
991561173546876928	BukaBantuan	0	0	0	@BukaBantuan selmat siang bukalapak ,mohon bantuannya untuk mengembalikan dana pembeli dengan no transaksi

					180818021073,terima kasih https://t.co/LqEBNaD1yt
990055156136865792	BukaBantuan	0	0	0	@BukaBantuan Semoga nggak lama, terima kasih min
956057693730541569	LazadaIDCare	-1	-1	0	@LazadaIDCare HI LAZ. udh tgl 24 nih. Tiap dm pas jawabnya mohon maaf dan mau menunggu hingga tgl sekian. Trs abis ini saya harus nunggu sampe tgl 27 gitu? No Trims.
946754605194518529	LazadaIDCare	0	0	0	@LazadaIDCare hy, orderan #3966284846 statusnya gimana yah? Kok ngga dateng2?
953115804001755136	LazadaIDCare	0	0	0	@LazadaIDCare info dong status barang saya dengan pesanan #3942132226 seperti apa? Di email dibatalkan tapi uang sudah terpotong dan blm ada pengembalian ke rekening
956017917442908160	LazadaIDCare	-1	-1	0	@LazadaIDCare ini apa apaan. Klik beli duluan mau bayar gagal terus langsung ada pemberitahuan sold out Apa apaan ini . Masih saat ini barang masih di keranjang dan mau bayar susah
957936068019367937	LazadaIDCare	-1	-1	0	@LazadaIDCare ini buktinya, kalo LASADA tuh isinya tukang tepu semua!

958557650362425344	LazadaIDCare	-1	-1	0	@LazadaIDCare kalo pesan barang, udah berhasil dapet nomor pesanan, tapi email ternyata salah input, status pesanannya gimana ya? Apakah valid atau invalid?
948474904155799552	LazadaIDCare	-1	-1	0	@LazadaIDCare kapan ?? tiap mau konfirmasi pesanan gagal terus !! ;(
949626096982487040	LazadaIDCare	0	0	0	@LazadaIDCare Kapan lg flashsale redmi 5a nya min?
948408714758799361	LazadaIDCare	-2	-2	0	@LazadaIDCare KECEWA AMA LAZADA! BARANG SUDAH DI TROLI PAS DI KLIK KONFIRMASI PESANAN HILANG GITU AJA , FAK LAH!
948456225456340992	LazadaIDCare	0	0	0	@LazadaIDCare kenapa ada pemberitahuan ke no sya bhwa pesanan 3566169846 ditunda pesanannya sesuai prmintaan. padahl sya tdk prnah mengonfrmasi untuk menunda pngiriman pesanan. Mhn pnjelasannya.mksih.

LAMPIRAN C

Hasil Pengujian Skenario Model
Single Filter Size

Subtask : A
Embed : Social Media Twitter
Type CNN: Non-Static
Feature
Num : 100
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.9078	0.9279	0.9310	0.9296	0.9310	0.9307	0.9290	0.9284	0.9290	0.9313
2	0.9052	0.9279	0.9301	0.9321	0.9321	0.9321	0.9326	0.9304	0.9318	0.9332
3	0.9197	0.9318	0.9372	0.9375	0.9369	0.9369	0.9372	0.9355	0.9369	0.9378
4	0.9174	0.9327	0.9360	0.9383	0.9360	0.9360	0.9366	0.9360	0.9377	0.9369
5	0.9069	0.9278	0.9321	0.9315	0.9295	0.9324	0.9324	0.9318	0.9335	0.9332

6	0.9157	0.9307	0.9338	0.9346	0.9343	0.9352	0.9355	0.9341	0.9343	0.9346
7	0.9123	0.9309	0.9312	0.9318	0.9312	0.9309	0.9318	0.9303	0.9315	0.9318
8	0.9123	0.9262	0.9270	0.9293	0.9287	0.9287	0.9284	0.9298	0.9304	0.9293
9	0.9168	0.9264	0.9290	0.9310	0.9287	0.9298	0.9287	0.9309	0.9298	0.9290
10	0.9109	0.9273	0.9287	0.9295	0.9293	0.9318	0.9307	0.9278	0.9307	0.9301

Measure : AvgRec

	Epoch									
Filter	1	2	3	4	5	6	7	8	9	10
1	0.8622	0.8849	0.8906	0.8946	0.8951	0.8940	0.8939	0.8957	0.8980	0.9012
2	0.8616	0.8841	0.8875	0.8951	0.8964	0.8936	0.8952	0.8964	0.8955	0.8968
3	0.8678	0.8910	0.8871	0.8963	0.9000	0.8973	0.8992	0.9026	0.8977	0.8997
4	0.8708	0.8921	0.8879	0.8940	0.8946	0.8911	0.8942	0.8979	0.8951	0.8943
5	0.8595	0.8802	0.8803	0.8821	0.8850	0.8839	0.8852	0.8925	0.8888	0.8895
6	0.8701	0.8860	0.8860	0.8926	0.8950	0.8929	0.8955	0.9001	0.8950	0.8954
7	0.8616	0.8847	0.8798	0.8823	0.8850	0.8843	0.8841	0.8880	0.8841	0.8848

8	0.8659	0.8827	0.8740	0.8788	0.8836	0.8797	0.8804	0.8893	0.8823	0.8816
9	0.8742	0.8873	0.8837	0.8876	0.8902	0.8894	0.8897	0.8967	0.8905	0.8897
10	0.8675	0.8855	0.8820	0.8833	0.8895	0.8870	0.8878	0.8919	0.8877	0.8874

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.8637	0.8886	0.8946	0.8946	0.8965	0.8974	0.8946	0.8942	0.8972	0.8994
2	0.8598	0.8888	0.8912	0.8963	0.8967	0.8968	0.8964	0.8946	0.8954	0.8969
3	0.8755	0.8955	0.8933	0.9027	0.9029	0.9029	0.9034	0.9031	0.9020	0.9043
4	0.8741	0.8959	0.8949	0.8994	0.8973	0.8964	0.8978	0.8999	0.8991	0.8981
5	0.8598	0.8823	0.8864	0.8857	0.8877	0.8870	0.8872	0.8937	0.8906	0.8921
6	0.8720	0.8883	0.8922	0.8961	0.8976	0.8978	0.8988	0.8997	0.8973	0.8976
7	0.8631	0.8850	0.8828	0.8854	0.8858	0.8853	0.8842	0.8869	0.8837	0.8840
8	0.8667	0.8848	0.8790	0.8840	0.8862	0.8839	0.8838	0.8903	0.8858	0.8847
9	0.8761	0.8885	0.8870	0.8908	0.8902	0.8914	0.8902	0.8939	0.8917	0.8908
10	0.8673	0.8864	0.8845	0.8881	0.8920	0.8913	0.8903	0.8909	0.8904	0.8897

Subtask : B
Embed : Social Media Twitter
Type CNN: Non-Static
Feature
Num : 100
Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.5367	0.6169	0.6405	0.6789	0.6210	0.6764	0.6629	0.6675	0.6489	0.6666
2	0.5456	0.6224	0.6452	0.6794	0.6271	0.6784	0.6553	0.6699	0.6634	0.6654
3	0.5490	0.6188	0.6357	0.6685	0.6238	0.6786	0.6455	0.6642	0.6550	0.6555
4	0.5534	0.6154	0.6295	0.6676	0.6311	0.6665	0.6340	0.6474	0.6494	0.6452
5	0.5520	0.6186	0.6420	0.6565	0.6354	0.6591	0.6314	0.6468	0.6480	0.6383
6	0.5462	0.6130	0.6415	0.6611	0.6389	0.6477	0.6234	0.6461	0.6469	0.6408
7	0.5496	0.6116	0.6353	0.6550	0.6378	0.6478	0.6231	0.6417	0.6375	0.6297
8	0.5506	0.6180	0.6334	0.6548	0.6417	0.6493	0.6219	0.6444	0.6405	0.6316
9	0.5531	0.6121	0.6353	0.6506	0.6377	0.6437	0.6194	0.6405	0.6361	0.6291

10	0.5552	0.6146	0.6379	0.6612	0.6374	0.6441	0.6171	0.6439	0.6400	0.6318

Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.7616	0.7778	0.7983	0.7936	0.7941	0.7703	0.7718	0.7616	0.7783	0.7730
2	0.7658	0.7762	0.8022	0.7904	0.7951	0.7580	0.7855	0.7738	0.7938	0.7892
3	0.7687	0.7827	0.8023	0.7919	0.7944	0.7644	0.7917	0.7935	0.7907	0.7957
4	0.7703	0.7803	0.7982	0.7861	0.7932	0.7688	0.7917	0.7901	0.7885	0.7924
5	0.7686	0.7780	0.7965	0.7874	0.7925	0.7870	0.7931	0.7953	0.7904	0.7921
6	0.7685	0.7830	0.7985	0.7860	0.7933	0.7870	0.7937	0.7954	0.7907	0.7939
7	0.7697	0.7806	0.7934	0.7861	0.7931	0.7898	0.7944	0.7955	0.7894	0.7935
8	0.7702	0.7801	0.7889	0.7895	0.7928	0.7913	0.7920	0.7946	0.7888	0.7921
9	0.7699	0.7809	0.7946	0.7839	0.7912	0.7912	0.7935	0.7936	0.7909	0.7939
10	0.7710	0.7823	0.7912	0.7869	0.7926	0.7932	0.7933	0.7946	0.7917	0.7944

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.3885	0.5289	0.5702	0.6038	0.5496	0.6054	0.5932	0.5929	0.5777	0.5838
2	0.3987	0.5338	0.5711	0.5993	0.5575	0.6012	0.5883	0.5963	0.5944	0.5864
3	0.4019	0.5294	0.5578	0.5884	0.546	0.6019	0.5765	0.5923	0.5825	0.5795
4	0.4069	0.5214	0.5473	0.5936	0.5545	0.5926	0.5633	0.5765	0.5777	0.5711
5	0.4048	0.5209	0.5626	0.5763	0.5577	0.5865	0.5554	0.5749	0.5732	0.5587
6	0.399	0.5198	0.5665	0.5876	0.564	0.5721	0.5494	0.5768	0.5739	0.565
7	0.4047	0.5196	0.5548	0.5817	0.565	0.5749	0.5477	0.5708	0.563	0.5517
8	0.4048	0.5231	0.5469	0.5774	0.5714	0.5755	0.5425	0.5739	0.5669	0.5534
9	0.4069	0.5182	0.5541	0.5748	0.5638	0.5701	0.5412	0.5703	0.5617	0.5513
10	0.4108	0.5201	0.5572	0.5865	0.5677	0.5706	0.5376	0.5754	0.5687	0.5563

Subtask : C
Embed : Social Media Twitter
Type CNN: Non-Static
Feature
Num : 100
Measure : Macro_MAE

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	1.6029	1.4682	1.3758	1.2910	1.3932	1.2804	1.2273	1.1746	1.1563	1.1720
2	1.5696	1.4514	1.3500	1.2628	1.3543	1.2681	1.1955	1.1912	1.2239	1.2283
3	1.5793	1.4298	1.2701	1.2522	1.3124	1.2874	1.1872	1.2518	1.2538	1.2396
4	1.5765	1.4337	1.3204	1.2635	1.3265	1.3203	1.1773	1.2401	1.2526	1.2455
5	1.5638	1.4259	1.2986	1.3273	1.3014	1.3318	1.2406	1.2764	1.2752	1.2802
6	1.5715	1.4015	1.3053	1.2895	1.2882	1.3258	1.2333	1.2623	1.2767	1.2741
7	1.5642	1.4080	1.3247	1.2863	1.3068	1.3534	1.2697	1.2822	1.2672	1.2646
8	1.5724	1.4039	1.2773	1.2798	1.3167	1.3296	1.2439	1.2886	1.2538	1.2698
9	1.5699	1.4213	1.2890	1.3024	1.3103	1.3337	1.2765	1.3103	1.2737	1.2676

10	1.5557	1.4121	1.3554	1.3175	1.3092	1.3555	1.2886	1.3142	1.2955	1.2808

Measure : Accuracy

	Epoch									
Filter	1	2	3	4	5	6	7	8	9	10
1	0.7507	0.7609	0.7651	0.7693	0.7681	0.7595	0.7476	0.7439	0.7391	0.7454
2	0.7559	0.766	0.7714	0.7726	0.7704	0.7589	0.7323	0.7488	0.7486	0.7616
3	0.7601	0.7702	0.7699	0.7698	0.7637	0.7585	0.74	0.7552	0.756	0.7597
4	0.7565	0.7681	0.7684	0.7727	0.7617	0.7607	0.7463	0.7582	0.7604	0.7618
5	0.7587	0.7675	0.7652	0.7664	0.7624	0.7635	0.7497	0.7608	0.7608	0.7593
6	0.7593	0.7695	0.7666	0.7688	0.7653	0.7673	0.7593	0.7665	0.766	0.7625
7	0.7566	0.7669	0.763	0.7627	0.7648	0.7631	0.7589	0.7626	0.7649	0.7623
8	0.76	0.7685	0.7664	0.7654	0.7652	0.7677	0.7592	0.7645	0.7652	0.7616
9	0.7599	0.7666	0.763	0.7674	0.7641	0.7637	0.7615	0.7666	0.7627	0.7601
10	0.7595	0.7699	0.7659	0.7663	0.7656	0.7643	0.7616	0.7686	0.7666	0.7649

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.3153	0.3391	0.3676	0.384	0.365	0.3817	0.3948	0.396	0.4079	0.4051
2	0.3206	0.3408	0.3788	0.3892	0.3676	0.3792	0.4041	0.4006	0.3981	0.4012
3	0.3238	0.3464	0.3818	0.3901	0.3751	0.3806	0.4104	0.3983	0.3978	0.3995
4	0.3222	0.3493	0.3795	0.3879	0.3723	0.3741	0.4013	0.3909	0.3882	0.3919
5	0.3233	0.3476	0.3848	0.3767	0.3764	0.366	0.3931	0.3849	0.3871	0.3875
6	0.3243	0.3487	0.373	0.3825	0.3849	0.3681	0.3944	0.3897	0.3899	0.3871
7	0.3238	0.349	0.3788	0.3833	0.3809	0.3638	0.3902	0.3871	0.388	0.3897
8	0.3253	0.354	0.3898	0.3844	0.3821	0.3696	0.3906	0.383	0.3923	0.3861
9	0.3237	0.3531	0.3865	0.3807	0.3797	0.3709	0.3841	0.3814	0.3863	0.3847
10	0.3257	0.3546	0.375	0.3786	0.3842	0.3743	0.3865	0.3829	0.3847	0.3875

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.165	0.2065	0.2583	0.2746	0.257	0.2838	0.2915	0.2923	0.3008	0.3018
2	0.1707	0.2106	0.2733	0.2848	0.2573	0.2793	0.2975	0.2967	0.2953	0.3055
3	0.1768	0.2194	0.2761	0.2881	0.2739	0.2834	0.3148	0.3025	0.3036	0.3048
4	0.1738	0.2276	0.2727	0.2868	0.2696	0.2737	0.2998	0.2921	0.2902	0.2963
5	0.1759	0.2245	0.2764	0.2692	0.2715	0.2613	0.2969	0.2817	0.2896	0.2888
6	0.1764	0.2236	0.2644	0.281	0.2877	0.2653	0.2952	0.2921	0.2943	0.2896
7	0.1768	0.2299	0.2704	0.2782	0.2789	0.2578	0.2902	0.2885	0.2914	0.2956
8	0.1805	0.2352	0.2862	0.2822	0.2822	0.2676	0.2886	0.2813	0.296	0.2868
9	0.1763	0.2365	0.2813	0.2823	0.2765	0.2696	0.2838	0.2822	0.2897	0.287
10	0.1796	0.2376	0.272	0.2775	0.2851	0.2735	0.285	0.2853	0.2864	0.2916

Information Table
Subtask : A
Social Media
Embed : Twitter
Type CNN: Static
Feature
Num : 100
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.8920	0.9171	0.9179	0.9199	0.9228	0.9248	0.9247	0.9236	0.9208	0.9222
2	0.8993	0.9222	0.9205	0.9230	0.9281	0.9281	0.9256	0.9278	0.9295	0.9284
3	0.9061	0.9256	0.9256	0.9304	0.9335	0.9341	0.9327	0.9363	0.9369	0.9355
4	0.9123	0.9253	0.9307	0.9284	0.9386	0.9349	0.9341	0.9349	0.9360	0.9341
5	0.9083	0.9264	0.9290	0.9244	0.9355	0.9352	0.9341	0.9352	0.9349	0.9343
6	0.9134	0.9247	0.9253	0.9228	0.9276	0.9296	0.9293	0.9295	0.9295	0.9293
7	0.9089	0.9245	0.9225	0.9253	0.9298	0.9312	0.9290	0.9312	0.9310	0.9298

8	0.9137	0.9293	0.9295	0.9278	0.9315	0.9326	0.9318	0.9324	0.9332	0.9312
9	0.9084	0.9290	0.9290	0.9287	0.9338	0.9355	0.9324	0.9349	0.9360	0.9335
10	0.9120	0.9267	0.9262	0.9290	0.9310	0.9310	0.9307	0.9335	0.9315	0.9321

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.8323	0.8683	0.8831	0.8713	0.8904	0.8855	0.8863	0.8882	0.8817	0.8863
2	0.8433	0.8691	0.8865	0.8760	0.8923	0.8831	0.8815	0.8862	0.8870	0.8872
3	0.8434	0.8745	0.8917	0.8889	0.8962	0.8925	0.8922	0.8961	0.8922	0.8948
4	0.8562	0.8770	0.9002	0.8881	0.9012	0.8935	0.8938	0.8932	0.8927	0.8930
5	0.8500	0.8783	0.9001	0.8836	0.8952	0.8935	0.8941	0.8923	0.8895	0.8936
6	0.8552	0.8770	0.8928	0.8837	0.8849	0.8864	0.8872	0.8813	0.8803	0.8866
7	0.8523	0.8713	0.8800	0.8799	0.8826	0.8808	0.8801	0.8809	0.8775	0.8801
8	0.8556	0.8819	0.8967	0.8908	0.8904	0.8880	0.8876	0.8858	0.8857	0.8877
9	0.8506	0.8851	0.8893	0.8863	0.8914	0.8916	0.8893	0.8888	0.8880	0.8897
10	0.8570	0.8805	0.8893	0.8918	0.8894	0.8875	0.8865	0.8872	0.8843	0.8871

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.8362	0.8736	0.8788	0.8786	0.887	0.8872	0.8876	0.8877	0.8833	0.8863
2	0.8434	0.8747	0.882	0.8817	0.8909	0.8859	0.8831	0.8884	0.8904	0.8905
3	0.8457	0.8809	0.8881	0.8937	0.8975	0.8974	0.8952	0.8996	0.8971	0.8987
4	0.8594	0.8827	0.8968	0.8918	0.9039	0.8978	0.8968	0.8977	0.8978	0.8966
5	0.8537	0.883	0.8955	0.887	0.8978	0.8974	0.8976	0.8973	0.8939	0.8978
6	0.8564	0.8824	0.8896	0.8867	0.8887	0.8912	0.8909	0.8861	0.886	0.8904
7	0.8548	0.8761	0.8788	0.8861	0.8847	0.8862	0.8842	0.8863	0.8835	0.8848
8	0.8608	0.8866	0.8935	0.8911	0.8929	0.8913	0.8901	0.8905	0.8913	0.8906
9	0.8572	0.8907	0.8913	0.89	0.8959	0.8983	0.8946	0.8962	0.8958	0.8959
10	0.8583	0.8841	0.8891	0.8939	0.8915	0.8911	0.8894	0.8924	0.8898	0.892

Subtask : B
Embed : Social Media Twitter
Type CNN: Static
Feature
Num : 100
Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.546	0.618	0.614	0.641	0.661	0.664	0.662	0.653	0.672	0.65
2	0.557	0.631	0.626	0.635	0.675	0.661	0.664	0.652	0.675	0.644
3	0.563	0.629	0.632	0.633	0.67	0.655	0.667	0.654	0.654	0.654
4	0.561	0.629	0.629	0.63	0.66	0.652	0.656	0.652	0.65	0.644
5	0.557	0.62	0.625	0.625	0.651	0.647	0.663	0.653	0.647	0.634
6	0.56	0.627	0.621	0.632	0.65	0.648	0.649	0.65	0.65	0.636
7	0.559	0.627	0.637	0.636	0.645	0.645	0.654	0.651	0.645	0.632
8	0.566	0.616	0.628	0.629	0.641	0.649	0.65	0.651	0.649	0.631

9	0.562	0.625	0.629	0.634	0.645	0.644	0.646	0.648	0.646	0.627
10	0.567	0.624	0.63	0.643	0.64	0.644	0.643	0.644	0.647	0.625

Measure : Accuracy

	Epoch									
Filter	1	2	3	4	5	6	7	8	9	10
1	0.7627	0.7861	0.7906	0.7961	0.7937	0.7948	0.7830	0.7806	0.7754	0.7854
2	0.7678	0.7941	0.7965	0.7993	0.7949	0.7926	0.7904	0.7930	0.7881	0.7887
3	0.7680	0.7920	0.7976	0.7945	0.7899	0.7887	0.7874	0.7918	0.7877	0.7893
4	0.7688	0.7941	0.7945	0.7941	0.7928	0.7888	0.7904	0.7914	0.7905	0.7924
5	0.7694	0.7889	0.7905	0.7905	0.7891	0.7917	0.7920	0.7907	0.7899	0.7885
6	0.7713	0.7924	0.7885	0.7915	0.7928	0.7920	0.7918	0.7918	0.7943	0.7916
7	0.7709	0.7899	0.7934	0.7929	0.7963	0.7946	0.7977	0.7917	0.7898	0.7937
8	0.7694	0.7875	0.7871	0.7906	0.7937	0.7949	0.7957	0.7925	0.7907	0.7930
9	0.7722	0.7874	0.7879	0.7915	0.7921	0.7921	0.7933	0.7911	0.7886	0.7899
10	0.7730	0.7922	0.7919	0.7945	0.7941	0.7970	0.7958	0.7919	0.7925	0.7935

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.3891	0.5299	0.5251	0.5638	0.5809	0.5932	0.5636	0.5746	0.5921	0.5735
2	0.3973	0.5482	0.5432	0.5517	0.5955	0.5903	0.5739	0.5776	0.5932	0.574
3	0.411	0.5461	0.5542	0.5473	0.5934	0.5819	0.5881	0.5815	0.5807	0.585
4	0.4123	0.5414	0.5467	0.5466	0.5815	0.5801	0.5789	0.5763	0.5725	0.5698
5	0.4038	0.5272	0.5361	0.5308	0.5694	0.5752	0.5823	0.5778	0.573	0.5571
6	0.4109	0.5416	0.5329	0.5527	0.5746	0.5788	0.5722	0.58	0.581	0.5625
7	0.4056	0.5423	0.5591	0.5531	0.5681	0.5734	0.5796	0.5809	0.5714	0.555
8	0.4205	0.521	0.543	0.543	0.5633	0.5802	0.5766	0.5811	0.5775	0.5539
9	0.4154	0.536	0.5432	0.553	0.5684	0.57	0.5726	0.5769	0.5711	0.5502
10	0.4236	0.5374	0.5444	0.565	0.5565	0.5713	0.5654	0.5694	0.5724	0.5454

Subtask : C
Embed : Social Media Twitter
Type CNN: Static
Feature Num : 100
Measure : Macro_MAE

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	1.5437	1.4171	1.3205	1.2491	1.2642	1.1234	1.2211	1.1972	1.1535	1.2062
2	1.5091	1.3642	1.2818	1.2632	1.2914	1.1773	1.2702	1.2104	1.2097	1.2031
3	1.4994	1.4148	1.3354	1.2681	1.3203	1.2203	1.2851	1.2538	1.2351	1.2407
4	1.4821	1.3917	1.3055	1.2573	1.2937	1.2280	1.2550	1.2599	1.2365	1.2487
5	1.4603	1.3798	1.3369	1.2475	1.2977	1.2145	1.2764	1.2451	1.2839	1.2519
6	1.4752	1.3864	1.3352	1.2850	1.3260	1.2549	1.2958	1.2781	1.3094	1.2643
7	1.4923	1.3969	1.3243	1.2579	1.3457	1.2901	1.3323	1.2895	1.3266	1.2699
8	1.4712	1.3928	1.3365	1.3043	1.3414	1.2666	1.3117	1.2852	1.3574	1.2874
9	1.4751	1.3990	1.3156	1.3163	1.3298	1.2689	1.3433	1.2893	1.3364	1.3041
10	1.4353	1.3709	1.3152	1.2803	1.3137	1.2762	1.3209	1.2826	1.3219	1.2596

Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.7554	0.7691	0.7738	0.7699	0.7678	0.7519	0.7674	0.7645	0.7366	0.7553
2	0.7628	0.7777	0.7787	0.7731	0.7714	0.7509	0.7735	0.7658	0.7550	0.7641
3	0.7624	0.7733	0.7753	0.7690	0.7660	0.7564	0.7657	0.7598	0.7608	0.7629
4	0.7626	0.7740	0.7739	0.7672	0.7629	0.7531	0.7653	0.7641	0.7656	0.7662
5	0.7645	0.7763	0.7704	0.7591	0.7658	0.7595	0.7657	0.7634	0.7651	0.7619
6	0.7639	0.7771	0.7669	0.7564	0.7626	0.7603	0.7694	0.7650	0.7672	0.7619
7	0.7624	0.7754	0.7685	0.7551	0.7637	0.7643	0.7672	0.7641	0.7674	0.7665
8	0.7659	0.7754	0.7695	0.7591	0.7687	0.7626	0.7697	0.7703	0.7703	0.7694
9	0.7643	0.7746	0.7700	0.7634	0.7707	0.7697	0.7702	0.7694	0.7711	0.7690
10	0.7631	0.7734	0.7661	0.7635	0.7646	0.7645	0.7665	0.7663	0.7696	0.7644

Measure : AvgRec

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.3233	0.3513	0.3856	0.409	0.3975	0.4297	0.4061	0.4118	0.4219	0.4095
2	0.334	0.3636	0.3938	0.3996	0.3947	0.419	0.3954	0.4092	0.421	0.4166
3	0.3352	0.3599	0.387	0.3997	0.3908	0.4102	0.3929	0.4043	0.4072	0.4057
4	0.3391	0.3616	0.3872	0.4012	0.3867	0.4054	0.3891	0.3911	0.4015	0.3975
5	0.3437	0.3656	0.3814	0.4045	0.388	0.4003	0.3889	0.3966	0.3912	0.3919
6	0.3426	0.3659	0.381	0.3951	0.3843	0.3922	0.3857	0.3877	0.386	0.3862
7	0.3398	0.3657	0.3775	0.4004	0.377	0.3907	0.3762	0.3839	0.3772	0.3835
8	0.3458	0.3648	0.3781	0.3881	0.3799	0.387	0.3797	0.387	0.3769	0.3857
9	0.3457	0.3646	0.3819	0.3907	0.3809	0.3925	0.3784	0.3859	0.3789	0.388
10	0.3441	0.3672	0.3853	0.3955	0.376	0.3885	0.3799	0.3835	0.3846	0.3925

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1	0.181	0.2307	0.284	0.3146	0.302	0.3289	0.3083	0.3108	0.3092	0.2986
2	0.1958	0.249	0.2963	0.3037	0.2942	0.3183	0.2992	0.3105	0.3181	0.3123
3	0.2008	0.2448	0.2891	0.3033	0.2924	0.3141	0.2969	0.3065	0.3097	0.3034
4	0.2084	0.2463	0.288	0.304	0.2845	0.3043	0.2923	0.2918	0.3027	0.2942
5	0.2162	0.2543	0.2814	0.3053	0.2867	0.3002	0.2905	0.2967	0.2893	0.2895
6	0.2157	0.2556	0.2816	0.2997	0.2875	0.2945	0.2904	0.2901	0.2864	0.2827
7	0.2108	0.2561	0.2762	0.3046	0.2748	0.2925	0.2772	0.2877	0.274	0.2803
8	0.219	0.2508	0.2772	0.2866	0.2804	0.2863	0.2822	0.2904	0.2728	0.2827
9	0.2214	0.2516	0.2801	0.2928	0.2837	0.293	0.2797	0.2866	0.2764	0.2859
10	0.2178	0.2535	0.2862	0.3021	0.2747	0.2877	0.2837	0.2845	0.2861	0.2926

Multiple Filter

Subtask : A
Embed : Social Media Twitter
Type CNN: Non-Static
Feature Num : 100
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,2,3	0.89817	0.92050	0.92700	0.92956	0.92955	0.93012	0.93068	0.93012	0.93181	0.93011
2,3,4	0.90437	0.92473	0.92843	0.93210	0.93211	0.93238	0.93324	0.93266	0.93295	0.93182
3,3,3	0.90551	0.91908	0.92671	0.92982	0.93180	0.92870	0.92897	0.92981	0.93038	0.93066
3,4,5	0.91430	0.92614	0.93207	0.93178	0.93491	0.93520	0.93548	0.93433	0.93604	0.93660

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.86397	0.87046	0.88614	0.88793	0.89033	0.891	0.89969	0.89497	0.89799	0.89714
1,2,3	0.86084	0.88156	0.89804	0.895	0.89696	0.90034	0.90053	0.90199	0.90373	0.90341

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.85852	0.87217	0.88178	0.88469	0.88906	0.88975	0.89619	0.89361	0.89639	0.89509
1,2,3	0.85201	0.88362	0.89724	0.90019	0.90001	0.90301	0.90346	0.90565	0.90665	0.9058

Feature Num : 200
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,2,3	0.89448	0.91173	0.92786	0.92899	0.93012	0.93041	0.93154	0.93097	0.93238	0.93012
2,3,4	0.90014	0.91371	0.92557	0.93041	0.93069	0.93153	0.93238	0.93153	0.93210	0.93097
3,3,3	0.89929	0.90776	0.91962	0.92587	0.92701	0.92757	0.92983	0.92841	0.92898	0.93039
3,4,5	0.90862	0.91994	0.92982	0.92955	0.93012	0.93126	0.93295	0.93182	0.93239	0.93210

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.85478	0.87066	0.89041	0.88285	0.88583	0.8879	0.89281	0.89089	0.89432	0.89157
1,2,3	0.85903	0.86864	0.89073	0.88641	0.89142	0.89293	0.89302	0.89351	0.8947	0.89665

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.84655	0.86733	0.88216	0.87963	0.88577	0.89171	0.89133	0.89028	0.89254	0.89093
1,2,3	0.84698	0.86933	0.88595	0.89076	0.89421	0.89501	0.89489	0.89567	0.89633	0.89826

Feature Num : 300
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,2,3	0.88968	0.90270	0.92332	0.92758	0.92786	0.93182	0.93210	0.93267	0.93267	0.93238
2,3,4	0.89589	0.90663	0.92130	0.92956	0.93125	0.93209	0.93097	0.93067	0.93125	0.93238
3,3,3	0.90043	0.89958	0.92558	0.92926	0.92955	0.92843	0.92899	0.92927	0.92899	0.92927
3,4,5	0.90097	0.90353	0.91965	0.92644	0.92673	0.92617	0.92786	0.92588	0.92673	0.92757

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.8572 6	0.8655 4	0.8915 1	0.8846 1	0.8934 9	0.8856 4	0.8972 9	0.8950 7	0.8955 5	0.894 2
1,2,3	0.8620 2	0.8789 5	0.8905 8	0.8838	0.8900 2	0.8906 1	0.8904 4	0.8911 7	0.8915 1	0.890 6

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.8455	0.8644 3	0.8834 5	0.8823 1	0.8925 2	0.8910 5	0.8930 4	0.8937 8	0.8942 6	0.8926 8
1,2,3	0.8486 9	0.8780 1	0.8850 5	0.8865 2	0.8911 2	0.8919 4	0.8907	0.8914 4	0.8914 7	0.8906 9

Subtask : B
Embed : Social Media Twitter
Type CNN: Non-Static
Feature Num : 100
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,2,3	0.77958	0.79793	0.77492	0.78558	0.78905	0.79176	0.7928	0.79344	0.79597	0.79541
2,3,4	0.77817	0.79241	0.76744	0.77847	0.79054	0.79429	0.79681	0.79541	0.79532	0.79513
3,3,3	0.77929	0.79091	0.77427	0.7808	0.78783	0.79345	0.794	0.79691	0.79597	0.79803
3,4,5	0.77527	0.79044	0.75779	0.78221	0.79054	0.79382	0.79588	0.79541	0.79447	0.79522

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.5588	0.59248	0.65346	0.65868	0.67371	0.65312	0.67599	0.67868	0.65515	0.66782
1,2,3	0.56594	0.59974	0.64675	0.65791	0.67297	0.66497	0.67181	0.66431	0.65111	0.67408

Feature Num : 200
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,2,3	0.7792	0.79812	0.77212	0.77941	0.78653	0.79486	0.79663	0.79635	0.80019	0.79925
2,3,4	0.78033	0.79643	0.76491	0.78464	0.79204	0.79504	0.80149	0.7985	0.79831	0.80196

3,3,3	0.7790 2	0.7939 1	0.7598 5	0.7851 1	0.7915 7	0.7935 4	0.7959 7	0.7957 9	0.7987 8	0.7978 4
3,4,5	0.7762 1	0.7951 3	0.7510 5	0.7872 6	0.7902 7	0.7976 6	0.7983 1	0.8009 3	0.7988 7	0.7970 9

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.5633 5	0.5976 6	0.6522 7	0.6555 7	0.6713 5	0.6478 6	0.671 7	0.6678 7	0.6471 5	0.6525 1
1,2,3	0.5696 4	0.6029 2	0.6476 7	0.6452 3	0.6630 3	0.6608 1	0.675 8	0.6514	0.6492 5	0.6707 2

Feature Num : 300
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,2,3	0.77154	0.80084	0.75677	0.76724	0.78324	0.79401	0.79682	0.79467	0.79747	0.79804
2,3,4	0.76854	0.79859	0.76799	0.77585	0.79251	0.79831	0.79944	0.79635	0.80084	0.79831
3,3,3	0.77257	0.79672	0.76406	0.77706	0.78316	0.7956	0.80084	0.79832	0.79906	0.80009
3,4,5	0.76808	0.79129	0.73355	0.78577	0.79073	0.79513	0.79756	0.79737	0.79653	0.7984

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.55847	0.59958	0.65399	0.64629	0.67172	0.65251	0.67403	0.67143	0.65156	0.64577
1,2,3	0.5688	0.61042	0.65627	0.65293	0.66458	0.6651	0.67203	0.6571	0.64881	0.67095

Subtask : C
Embed : Social Media Twitter
Type CNN: Non-Static
Feature Num : 100
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
8,9,10	0.76544	0.76769	0.76349	0.73196	0.77004	0.77135	0.77387	0.76957	0.77368	0.76461
10,10,10	0.76788	0.76581	0.7559	0.72681	0.76526	0.76834	0.76759	0.76638	0.77087	0.76638

Measure : Macro_MAE

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	1.58002	1.3403	1.23194	1.28803	1.18554	1.21795	1.18578	1.2671	1.19912	1.22209
1,2,3	1.57636	1.29104	1.18165	1.24844	1.12783	1.19078	1.13869	1.19956	1.17317	1.19197

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.32072	0.36745	0.40104	0.38383	0.40717	0.41417	0.40698	0.40179	0.40363	0.39174
1,2,3	0.325	0.37724	0.41693	0.39686	0.4217	0.41825	0.42245	0.40666	0.41921	0.40448

Feature Num : 200
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
8,9,10	0.76694	0.76966	0.75722	0.7243	0.76769	0.77134	0.77134	0.76957	0.77162	0.76686
10,10,10	0.76816	0.76413	0.75824	0.7155	0.77022	0.77031	0.77144	0.76947	0.77087	0.76199

Measure : Macro_MAE

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	1.52291	1.3081	1.20155	1.25045	1.14769	1.20555	1.18346	1.23208	1.15747	1.22902
1,2,3	1.53666	1.26217	1.17959	1.21287	1.10873	1.21027	1.17588	1.19366	1.21499	1.20297

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.33205	0.37795	0.40304	0.38995	0.41289	0.41965	0.40876	0.40473	0.40806	0.39385
1,2,3	0.33527	0.38339	0.41874	0.39605	0.42025	0.416	0.41343	0.41164	0.40581	0.40238

Feature Num : 300
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,2,3	0.77154	0.80084	0.75677	0.76724	0.78324	0.79401	0.79682	0.79467	0.79747	0.79804
2,3,4	0.76854	0.79859	0.76799	0.77585	0.79251	0.79831	0.79944	0.79635	0.80084	0.79831
3,3,3	0.77257	0.79672	0.76406	0.77706	0.78316	0.7956	0.80084	0.79832	0.79906	0.80009
3,4,5	0.76808	0.79129	0.73355	0.78577	0.79073	0.79513	0.79756	0.79737	0.79653	0.7984

Measure : Macro_MAE

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	1.50188	1.29646	1.20551	1.23632	1.15866	1.18727	1.21684	1.2897	1.18437	1.26269
1,2,3	1.46222	1.2863	1.16678	1.20464	1.09496	1.2179	1.17263	1.21749	1.19392	1.22524

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,1,1	0.33907	0.3825	0.41188	0.39033	0.41393	0.41456	0.40493	0.38951	0.40228	0.38962

1,2,3	0.34598	0.38318	0.42136	0.39323	0.42027	0.41299	0.41502	0.40479	0.40576	0.40033

Subtask : A
Embed : Social Media Twitter
Type CNN: Static
Feature Num : 100
Measure : Accuracy

	Epoch									
Filter	1	2	3	4	5	6	7	8	9	10
1,2,3	0.90944	0.927	0.92671	0.92699	0.9318	0.93094	0.93292	0.93349	0.9332	0.93377
2,3,4	0.91341	0.92727	0.9318	0.93181	0.93321	0.93292	0.93575	0.93604	0.93519	0.93575
3,3,3	0.91226	0.92161	0.93066	0.93378	0.92982	0.93291	0.93292	0.93237	0.93292	0.9332

3,4,5	0.9117	0.92162	0.92868	0.93123	0.93067	0.92925	0.9301	0.9301	0.92982	0.93124

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
3,3,3	0.86119	0.88392	0.88204	0.88133	0.88946	0.88759	0.89121	0.88932	0.89097	0.89046
1,2,3	0.86095	0.87997	0.87994	0.87754	0.89464	0.89908	0.89151	0.89132	0.89612	0.89677
2,3,4	0.85666	0.88809	0.87733	0.87248	0.8914	0.89102	0.8886	0.88727	0.89248	0.89104
3,4,5	0.85872	0.88054	0.87488	0.87628	0.88935	0.89355	0.89242	0.89024	0.89346	0.8883

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
3,3,3	0.8619	0.8859	0.8859	0.8834	0.8919	0.8919	0.8934	0.8927	0.8930	0.8920
	9	1	9	7	4	5	4	1	4	3
1,2,3	0.8598	0.8833	0.8863	0.8802	0.8983	0.9026	0.8950	0.8952	0.8996	
	2	6	1	9	3	9	1	5	2	0.8979
2,3,4	0.8596	0.8896	0.8821	0.8763	0.8937	0.8933	0.8904	0.8901		0.8917
	7	4	6	4	8	9	9	6	0.8943	8
3,4,5	0.8600	0.8781		0.8736	0.8896	0.8948	0.8910	0.8906		0.8875
	4	8	0.8773	2	6	6	2	7	0.8928	1

Feature Num : 200

Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10

1,2,3	0.91341	0.92927	0.93239	0.93436	0.93182	0.93634	0.93803	0.93633	0.93661	0.93689
2,3,4	0.90663	0.92414	0.92755	0.92643	0.92952	0.92839	0.92782	0.92981	0.92811	0.92924
3,3,3	0.90632	0.92274	0.93095	0.92672	0.92869	0.93095	0.9318	0.9318	0.93152	0.93237
3,4,5	0.91055	0.9233	0.92642	0.93096	0.93093	0.93179	0.93406	0.93435	0.93321	0.93319

Measure : AvgRecall

	Epoch									
Filter	1	2	3	4	5	6	7	8	9	10
3,3,3	0.86003	0.88367	0.87212	0.87686	0.8856	0.88529	0.88716	0.88288	0.88536	0.88782
1,2,3	0.85963	0.8799	0.87085	0.87044	0.88862	0.88823	0.89025	0.88952	0.88887	0.88837
2,3,4	0.85543	0.87804	0.86729	0.8709	0.885	0.88765	0.88766	0.88629	0.88798	0.88638

3,4,5	0.8592 5	0.8854 4	0.8797 9	0.8778 1	0.8895 5	0.8917 7	0.8926 5	0.8922 2	0.8949 1	0.8904 1

Measure : F1-Score

	Epoch									
Filter	1	2	3	4	5	6	7	8	9	10
3,3,3	0.8624 4	0.8842 1	0.8776 9	0.8763 4	0.8865 6	0.8892 7	0.8888 6	0.8862 7	0.8873 7	0.8861 6
1,2,3	0.8617 6	0.8824 6	0.8795 7	0.8731 8	0.8920 5	0.8935 8	0.8934 7	0.8938 9	0.8912 4	0.8899 8
2,3,4	0.8530 6	0.8759 7	0.8692 4	0.8708 1	0.8859 7	0.8902	0.8875 2	0.8889 5	0.8885 5	0.8858 1
3,4,5	0.8611 5	0.8855 1	0.8878 2	0.879	0.8928 1	0.8952	0.8942 3	0.8951 8	0.8972 1	0.8898 6

Feature Num : 300
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,2,3	0.89895	0.91962	0.92588	0.92757	0.92841	0.92982	0.92982	0.9304	0.92897	0.92954
2,3,4	0.90859	0.92387	0.92786	0.93294	0.93066	0.93349	0.93604	0.93493	0.93519	0.9349
3,3,3	0.90857	0.92641	0.92559	0.9304	0.93349	0.93321	0.93549	0.93521	0.93492	0.93491
3,4,5	0.90659	0.92445	0.92927	0.92927	0.92925	0.92869	0.93067	0.93096	0.93181	0.93067

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
3,3,3	0.8640 1	0.8821	0.8762 7	0.8792 7	0.8882 3	0.8900 2	0.8882 4	0.8866 6	0.8905 9	0.8834 2
1,2,3	0.8547 9	0.8811 3	0.8704 2	0.8675 3	0.8945 9	0.8922 4	0.8950 8	0.8947 7	0.8947 3	0.8931 8
2,3,4	0.8569 3	0.8846 1	0.8693 5	0.8707 4	0.8913 6	0.8862 1	0.8904 1	0.8890 5	0.8923 1	0.8853 1
3,4,5	0.8552 2	0.8858 6	0.879	0.8776 1	0.8936 9	0.8921 9	0.8930 4	0.8892 3	0.8896 7	0.8883 4

Measure : F1-Score

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
3,3,3	0.8629 2	0.8840 2	0.8822 9	0.8807 1	0.8909 3	0.8924 3	0.8902 2	0.8897 5	0.8918 9	0.8812

1,2,3	0.85099	0.88141	0.87305	0.86673	0.89481	0.89356	0.89583	0.89628	0.89502	0.88982
2,3,4	0.85466	0.8843	0.87318	0.8732	0.89185	0.88776	0.89102	0.89149	0.89262	0.88467
3,4,5	0.85481	0.88861	0.88385	0.87883	0.89602	0.89556	0.89389	0.89247	0.89237	0.88695

Subtask : B
Embed : Social Media Twitter
Type CNN: Static
Feature Num : 100
Measure : Accuracy

	Epoch									
Filter	1	2	3	4	5	6	7	8	9	10
5,6,7	0.77051	0.75592	0.77276	0.79522	0.79325	0.79503	0.7954	0.78353	0.79588	0.77165
6,7,8	0.7721	0.75854	0.76911	0.79419	0.79194	0.79354	0.79821	0.77604	0.79456	0.7767
7,7,7	0.77079	0.75246	0.77061	0.79026	0.7925	0.79288	0.79633	0.77454	0.79447	0.77342
7,8,9	0.77378	0.74573	0.76977	0.79213	0.79325	0.79185	0.79194	0.75845	0.7925	0.77623

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,2,3	0.55301	0.61851	0.61507	0.67593	0.68259	0.65334	0.65279	0.68192	0.65926	0.6784
2,3,4	0.55597	0.62755	0.60869	0.66443	0.65672	0.64946	0.65187	0.66472	0.64879	0.67117
3,3,3	0.54491	0.62256	0.62106	0.67261	0.65478	0.65662	0.64957	0.66961	0.65556	0.6767
3,4,5	0.55748	0.62238	0.60391	0.65511	0.65236	0.65133	0.64892	0.64895	0.63783	0.66725

Feature Num : 200
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
5,6,7	0.76986	0.73581	0.76884	0.79719	0.79503	0.79728	0.79634	0.76641	0.79962	0.76837
6,7,8	0.77285	0.73048	0.77024	0.79353	0.79372	0.79503	0.79774	0.77576	0.79409	0.76622
7,7,7	0.76994	0.72543	0.7635	0.7955	0.79287	0.79353	0.79512	0.76818	0.79438	0.76472
7,8,9	0.7691	0.73394	0.76752	0.79334	0.79512	0.7984	0.79848	0.77333	0.79699	0.77239

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
1,2,3	0.54913	0.61266	0.5936	0.66464	0.65825	0.64594	0.64628	0.67718	0.65202	0.67295
2,3,4	0.55414	0.61395	0.58787	0.65227	0.62831	0.64791	0.64178	0.65918	0.64421	0.67472
3,3,3	0.55816	0.61314	0.58263	0.65887	0.63225	0.65301	0.64421	0.66705	0.65038	0.67184
3,4,5	0.55938	0.60985	0.58676	0.64493	0.64654	0.64906	0.64215	0.64936	0.63024	0.66489

Feature Num : 300

Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10

5,6,7	0.76797	0.71841	0.75957	0.79475	0.79063	0.794	0.79371	0.76594	0.79372	0.75527
6,7,8	0.7692	0.71019	0.76744	0.79653	0.7941	0.79597	0.79372	0.76566	0.79822	0.76051
7,7,7	0.76554	0.71823	0.75658	0.79615	0.79765	0.79438	0.79709	0.75518	0.79999	0.75762
7,8,9	0.76995	0.70616	0.74683	0.79718	0.79644	0.79606	0.79915	0.75911	0.79747	0.76099

Measure : AvgRecall

	Epoch									
Filter	1	2	3	4	5	6	7	8	9	10
1,2,3	0.55519	0.61506	0.58113	0.65912	0.65397	0.65413	0.64734	0.67616	0.66158	0.6813
2,3,4	0.54782	0.60461	0.57602	0.64383	0.62295	0.65058	0.64309	0.65482	0.63984	0.67248
3,3,3	0.54799	0.61559	0.57459	0.65998	0.62521	0.64857	0.64395	0.65557	0.6411	0.66995

3,4,5	0.54835	0.60874	0.5718	0.62976	0.64481	0.65362	0.63938	0.651	0.63066	0.66504

Subtask : C
Embed : Social Media Twitter
Type CNN: Static
Feature Num : 100
Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
6,7,8	0.76077	0.76768	0.76377	0.7617	0.77013	0.77003	0.77228	0.76059	0.76676	0.77349
7,8,9	0.76039	0.7674	0.75918	0.76826	0.76704	0.76835	0.77078	0.75507	0.76451	0.77425
8,8,8	0.76516	0.76881	0.76096	0.76012	0.76835	0.76863	0.77116	0.75694	0.76751	0.77406

8,9,10	0.76413	0.76601	0.75881	0.77022	0.77097	0.76779	0.76854	0.75834	0.76339	0.77228

Measure : Macro_MAE

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
2,2,2	1.59445	1.41641	1.27972	1.23361	1.03737	1.19044	1.11214	1.18486	1.18398	1.27949
1,2,3	1.4699	1.27536	1.16254	1.3222	1.23288	1.2956	1.27304	1.25541	1.19203	1.18615
2,3,4	1.46683	1.26435	1.14625	1.32905	1.25343	1.31007	1.30703	1.28423	1.24287	1.23098

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
2,2,2	0.31819	0.35499	0.38415	0.40114	0.43487	0.40125	0.41189	0.40477	0.40382	0.37843
1,2,3	0.33979	0.38731	0.41096	0.3858	0.40138	0.38986	0.38384	0.38965	0.40678	0.4074
2,3,4	0.34644	0.39204	0.41684	0.38446	0.40254	0.38898	0.38933	0.40522	0.40777	0.41058

Feature Num : 200

Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
6,7,8	0.75768	0.76637	0.75011	0.76022	0.77097	0.76928	0.77425	0.76349	0.76442	0.77743

7,8,9	0.7583 3	0.7613	0.7507 7	0.7684 5	0.7697 6	0.7686 3	0.7719 1	0.7609 7	0.7643 3	0.7758 4
8,8,8	0.7593 6	0.7638 4	0.7553 5	0.7699 5	0.7737 8	0.7757 4	0.7748 1	0.7637 7	0.7693 8	0.7752 7
8,9,10	0.7562 8	0.7597 2	0.7564 7	0.7654 5	0.7706	0.7697 5	0.7724 7	0.7639 5	0.7662	0.7755 5

Measure : Macro_MAE

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
2,2,2	1.5692 2	1.4235 9	1.2718 1	1.2503 5	1.0712 7	1.1769 7	1.1160 1	1.2087 8	1.2153 3	1.2879 1
1,2,3	1.4450 8	1.2612 4	1.1467 8	1.3528 6	1.2904 9	1.3106 6	1.3247 9	1.2672 6	1.2396 7	1.2589 3
2,3,4	1.4212 6	1.1985 8	1.1250 2	1.3199 8	1.2712 3	1.2942 7	1.3032 8	1.2776 6	1.2388 2	1.2357 1

Measure : AvgRecall

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
2,2,2	0.32184	0.35941	0.3878	0.40663	0.43344	0.41499	0.42564	0.40918	0.40715	0.38488
1,2,3	0.34551	0.39028	0.41275	0.38069	0.39224	0.39755	0.37515	0.40092	0.40286	0.40654
2,3,4	0.35081	0.39199	0.41273	0.379	0.39168	0.38804	0.38345	0.38983	0.40149	0.39901

Feature Num : 300

Measure : Accuracy

Filter	Epoch									
	1	2	3	4	5	6	7	8	9	10
6,7,8	0.74963	0.75757	0.75825	0.76873	0.77154	0.77144	0.77219	0.76424	0.76929	0.77612

7,8,9	0.75646	0.76168	0.75441	0.76667	0.77116	0.77172	0.775	0.76517	0.76957	0.77593
8,8,8	0.74814	0.75943	0.76143	0.76386	0.77331	0.77256	0.77172	0.76405	0.7677	0.77574
8,9,10	0.75581	0.76224	0.76162	0.77069	0.77369	0.77087	0.77696	0.76826	0.77266	0.77443