

## Transkrip

Nama : PETER YUDHISTIRA  
 NRP : C14190067  
 Tempat/Tgl.Lahir : JAKARTA, 04/08/2001  
 Tahun Masuk : 1-2019/2020

Fakultas : TEKNOLOGI INDUSTRI  
 Program Studi : INFORMATIKA  
 Program : INFORMATIKA  
 Program Pendidikan : SI

Kode	Mata Kuliah	SMT	SKS	Nilai
DU4161	ETIKA / ETHICS	1-19/20	2	A
DU4101	PANCASILA / STATE IDEOLOGY	1-19/20	2	A
TF4503	APLIKASI KOMPUTER / COMPUTER APPLICATION	1-19/20	3	A
TF4205	DASAR SISTEM KOMPUTER / INTRODUCTION TO COMPUTER SYSTEM	1-19/20	2	A
TF4244	MATEMATIKA DASAR / BASIC MATHEMATICS	1-19/20	3	A
TF4204	ALGORITMA DAN PEMROGRAMAN / ALGORITHMS AND PROGRAMMING	1-19/20	5	A
TF4220	PENGANTAR MANAJEMEN DAN BISNIS / INTRODUCTION TO MANAGEMENT AND BUSINESS	1-19/20	2	A
DU4167	FILSAFAT AGAMA / PHILOSOPHY OF RELIGION	2-19/20	2	A
DU4164	PENDIDIKAN KEWARGANEGERAAN / CIVICS EDUCATION	2-19/20	2	A
TF4229	BASIS DATA / DATABASE	2-19/20	3	A
TF4504	BAHASA INGGRIS / ENGLISH	2-19/20	2	A
TF4235	PEMROGRAMAN BERORIENTASI OBYEK / OBJECT-ORIENTED PROGRAMMING	2-19/20	3	B+
TF4505	INTERNET OF EVERYTHING / INTERNET OF EVERYTHING	2-19/20	2	A
TF4245	MATEMATIKA DISKRIT / DISCRETE MATHEMATICS	2-19/20	3	B+
TF4249	PENGANTAR AKUNTANSI / INTRODUCTION TO ACCOUNTING	2-19/20	2	A
DU4122	BAHASA INDONESIA / INDONESIAN	1-20/21	2	B+
TF4255	REKAYASA PERANGKAT LUNAK / SOFTWARE ENGINEERING	1-20/21	3	A
TF4372	ARSITEKTUR DAN ORGANISASI KOMPUTER / COMPUTER ARCHITECTURES AND ORGANIZATION	1-20/21	3	B+
TF4253	JARINGAN KOMPUTER / COMPUTER NETWORKS	1-20/21	3	A
TF4270	DESAIN DAN ANALISIS ALGORITMA / DESIGN AND ANALYSIS OF ALGORITHMS	1-20/21	3	A
TF4343	TEKNOLOGI WEB / WEB TECHNOLOGIES	1-20/21	3	A
TF4219	STRUKTUR DATA / BASIC STRUCTURES	1-20/21	3	A
FD4507	ALJABAR LINIER / LINEAR ALGEBRA	1-20/21	3	A
DU4171	ENGLISH FOR ACADEMIC PURPOSES / ENGLISH FOR ACADEMIC PURPOSES	2-20/21	2	A
DU4191	SERVANT LEADER / SERVANT LEADER	2-20/21	2	A

Kode	Mata Kuliah	SMT	SKS	Nilai
TF4227	STATISTIKA DASAR / BASIC STATISTICS	2-20/21	2	B+
TF4267	KOMUNIKASI INTERPERSONAL / INTERPERSONAL COMMUNICATION	2-20/21	3	A
TF4243	SISTEM OPERASI / OPERATING SYSTEM	2-20/21	3	A
TF4327	ANALISIS DAN DESAIN SISTEM INFORMASI / ANALYSIS AND DESIGN OF INFORMATION SYSTEM	2-20/21	3	B+
TF4237	INTERAKSI MANUSIA DAN KOMPUTER / HUMAN-COMPUTER INTERACTION	2-20/21	3	A
TF4409	KECERDASAN BUAATAN / ARTIFICIAL INTELLIGENCE	2-20/21	3	A
TF4426	KRIPTOGRAFI DAN STEGANOGRAFI / CRYPTOGRAPHY AND STEGANOGRAPHY	1-21/22	3	A
DU4163	ETIKA PROFESI / PROFESSIONAL ETHICS	1-21/22	2	A
TF4374	TEKNOLOGI OPEN SOURCE / OPEN SOURCE TECHNOLOGY	1-21/22	3	A
TF4548	BUSINESS INTELLIGENCE / BUSINESS INTELLIGENCE	1-21/22	3	B+
TF4514	MACHINE LEARNING / MACHINE LEARNING	1-21/22	3	A
TF4257	TEORI BAHASA DAN AUTOMATA / LANGUAGE AND AUTOMATA THEORY	1-21/22	2	A
TF4333	DATA MINING / DATA MINING	1-21/22	3	A
FD4501	TECHNOPRENEURSHIP I / TECHNOPRENEURSHIP I	1-21/22	2	A
FD4502	TECHNOPRENEURSHIP II / TECHNOPRENEURSHIP II	2-21/22	3	B+
TF4415	GRAFIKA KOMPUTER / COMPUTER GRAPHICS	2-21/22	3	A
TF4259	METODOLOGI PENELITIAN / RESEARCH METHODOLOGY	2-21/22	3	B+
TF4542	DEEP LEARNING / DEEP LEARNING	2-21/22	3	A
TF4317	MANAJEMEN PROYEK TEKNOLOGI INFORMASI / INFORMATION TECHNOLOGY PROJECT MANAGEMENT	2-21/22	3	B+
TF4523	ANALISIS BIG DATA / BIG DATA ANALYSIS	2-21/22	3	B+
TF4549	PEMODELAN DATA UNTUK DATA SCIENCE / DATA MODELING FOR DATA SCIENCE	2-21/22	3	A
TF4540	INDUSTRIAL TRAINING / INDUSTRIAL TRAINING	1-22/23	4	B+
TF4554	ON-THE-JOB PROBLEM SOLVING / ON-THE-JOB PROBLEM SOLVING	1-22/23	5	A
TF4539	PROFESSIONAL DEVELOPMENT / PROFESSIONAL DEVELOPMENT	1-22/23	6	A
TF4991	PRA SKRIPSI / PRE THESIS	1-22/23	2	E

Keterangan :

Indeks Prestasi Kumulatif : 3.82  
 Jumlah SKS : 141 SKS

Nilai Huruf	Keterangan	Nilai Bobot
A	Istimewa	4.00
B+	Baik Sekali	3.50
B	Baik	3.00
C+	Cukup Baik	2.50
C	Cukup	2.00
D	Kurang	1.00
E	Sangat Kurang	0.00

## Daftar Bimbingan Tugas Akhir Mahasiswa

**Nama** : PETER YUDHISTIRA      **Fakultas** : TEKNOLOGI INDUSTRI  
**NRP** : C14190067      **Jurusan** : INFORMATIKA  
**Dosen Wali** : Ir. KARTIKA GUNADI, M.T.  
**Judul TA** : Topic Modelling dan Clustering untuk Anomaly Detection pada Transkrip Diskusi Komunitas

Tanggal	Dosen Pembimbing	Bimbingan	Topik Bimbingan	Setuju?
25-01-2023	Dr. GREGORIUS SATIA BUDHI, S.T., M.T.	1	Perubahan Fokus Topik Skripsi	<input checked="" type="checkbox"/>
25-01-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	2	Perubahan Fokus Topik Skripsi	<input checked="" type="checkbox"/>
03-02-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	3	Pembahasan Alur Eksperimen	<input checked="" type="checkbox"/>
09-02-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	4	Preprocessing dan Feature Extraction Awal	<input checked="" type="checkbox"/>
14-02-2023	Dr. GREGORIUS SATIA BUDHI, S.T., M.T.	5	Laporan progress pipeline eksperimen	<input checked="" type="checkbox"/>
16-02-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	6	Laporan progress pipeline eksperimen	<input checked="" type="checkbox"/>
24-02-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	7	Laporan progress pipeline eksperimen	<input checked="" type="checkbox"/>
03-03-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	8	Weighted Averaging menghasilkan cluster yang baik	<input checked="" type="checkbox"/>
10-03-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	9	Percobaan parameter DBSCAN	<input checked="" type="checkbox"/>
23-03-2023	Dr. GREGORIUS SATIA BUDHI, S.T., M.T.	10	Laporan penemuan dan perubahan pipeline eksperimen	<input checked="" type="checkbox"/>
29-03-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	11	Laporan penemuan dan perubahan pipeline eksperimen	<input checked="" type="checkbox"/>
17-04-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	12	Penyelesaian Pipeline Eksperimen secara penuh.	<input checked="" type="checkbox"/>
03-05-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	13	Laporan Pembuatan Program	<input checked="" type="checkbox"/>
10-05-2023	Dr. GREGORIUS SATIA BUDHI, S.T., M.T.	14	Laporan Pembuatan Program	
11-05-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	15	Laporan Pembuatan Program dan Saran Buku	<input checked="" type="checkbox"/>
11-05-2023	Dr. GREGORIUS SATIA BUDHI, S.T., M.T.	16	Saran Penggunaan Isolation Forest	
23-05-2023	Dr. GREGORIUS SATIA BUDHI, S.T., M.T.	17	ACC PROGRAM	
25-05-2023	ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.	18	ACC PROGRAM	<input checked="" type="checkbox"/>

Mengetahui  
Pembimbing 1

Mengetahui  
Pembimbing 2

---

Dr. GREGORIUS SATIA BUDHI, S.T., M.T.

---

ALVIN NATHANIEL TJONDROWIGUNO, S.Kom., M.T.

NO: --/--/--

**DENSITY-BASED CLUSTERING UNTUK ANOMALY DETECTION PADA  
TRANSKRIP DISKUSI TERARAH KOMUNITAS**

**SKRIPSI**

Diajukan untuk memenuhi persyaratan penyelesaian program S-1  
Jurusan Informatika Fakultas Teknologi Industri  
Universitas Kristen Petra

Oleh :  
Peter Yudhistira  
NRP : C14190067

**PROGRAM STUDI INFORMATIKA**



**FAKULTAS TEKNOLOGI INDUSTRI  
UNIVERSITAS KRISTEN PETRA  
SURABAYA  
2023**

**LEMBAR PENGESAHAN**

**SKRIPSI**

***TOPIC MODELLING DAN DENSITY-BASED CLUSTERING UNTUK  
ANOMALY DETECTION PADA TRANSKRIP DISKUSI TERARAH KOMUNITAS***

Oleh :

Peter Yudhistira                    NRP : C14190067

Diterima Oleh :

Program Studi Informatika  
Fakultas Teknologi Industri  
Universitas Kristen Petra

Surabaya, 31 Mei 2023

Pembimbing 1

Pembimbing 2

Dr. Gregorius Satiabudhi  
NIP: 02-030

Alvin Tjondrowiguno, S.Kom., M.T.  
NIP: 22-012

Ketua Tim Penguji :

Henry Novianus Palit, Ph.D.  
NIP: 14-001

Ketua Program Studi :

Henry Novianus Palit, Ph.D.  
NIP: 14-001

**LEMBARAN PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH**  
**UNTUK KEPENTINGAN AKADEMIS**

Sebagai mahasiswa Universitas Kristen Petra, yang bertanda tangan dibawah ini, saya :

Nama : Peter Yudhistira

NRP : C14190067

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Kristen Petra Hak Bebas Royalti Non-Eksklusif (*Non-Exclusive Royalty-Free Right*) atas karya ilmiah saya yang berjudul "*TOPIC MODELLING DAN DENSITY-BASED CLUSTERING UNTUK ANOMALY DETECTION PADA TRANSKRIP DISKUSI TERARAH KOMUNITAS*". Dengan Hak Bebas Royalti Non-Eksklusif ini Universitas Kristen Petra berhak menyimpan,mengalih-mediakan/formatkan, mengelolanya dalam bentuk pangkalan data (*database*), mendistribusikannya dan menampilkan/mempublikasinya di internet atau media lain untuk kepentingan akademik tanpa perlu meminta ijin dari saya selama tetap mencantumkan saya sebagai penulisnya.

Saya bersedia untuk menanggung secara pribadi, tanpa melibatkan pihak Universitas Kristen Petra, segala bentuk tuntutan hukum yang timbul atas pelanggaran Hak Cipta dalam karya ilmiah saya ini.

Demikian pernyataan ini yang saya buat dengan sebenarnya.

Surabaya, 31 Mei 2023

Yang Menyatakan,

Peter Yudhistira

## KATA PENGANTAR

Segala puji syukur ke hadirat Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya, sehingga peneliti dapat menyelesaikan laporan Tugas Akhir ini dengan baik. Tugas Akhir ini diajukan untuk memenuhi salah satu persyaratan guna meraih gelar Sarana Strata-1 di Jurusan Informatika, Fakultas Teknologi Industri , Universitas Kristen Petra, Surabaya.

Rasa terima kasih yang tak terhingga penulis sampaikan kepada berbagai pihak yang telah membantu peneliti selama ini, khususnya:

1. Dr. Gregorius Satia Budhi selaku dosen pembimbing 1 yang telah meluangkan waktu dan menyumbangkan saran-saran dan sudut pandang baru dalam penulisan tugas akhir ini.
2. Alvin Tjondrowiguno, S.Kom., M.T. selaku dosen pembimbing 2 yang telah meluangkan waktu dan memberikan arahan-arahan praktis dalam penulisan tugas akhir ini.
3. Henry Novianus Palit, Ph.D. selaku Kepala Bidang Informatika Universitas Kristen Petra Surabaya.
4. Segenap dosen dan staf pengajar di Program Studi Informatika dan Sistem Informasi Bisnis Universitas Kristen Petra.
5. Keluarga tercinta yang telah memberikan banyak dukungan doa, mental, motivasi, dan material, dan pengorbanan-pengorbanan yang memungkinkan penulis untuk menyelesaikan tugas akhir ini.
6. Teman-teman penyelenggara diskusi komunitas Cangkrukan yang telah mau direpotkan untuk menjadi validator, dan mengijinkan penulis untuk menggunakan data notulen dan *platform* untuk mengumpulkan data.

Peneliti menyadari bahwa penulisan tugas akhir ini masih sangat jauh dari sempurna. Oleh karena itu, peneliti mengharapkan segala petunjuk, kritik, dan saran yang membangun dari pembaca agar dapat menunjang pengembangan dan perbaikan penelitian selanjutnya.

Akhir kata peneliti mohon maaf atas banyaknya kekurangan dalam penelitian ini dan dengan hati terbuka menerima kritik, saran, dan masukan dari pembaca.

Surabaya, 31 Mei 2023

Peter Yudhistira

## **ABSTRAK**

Peter Yudhistira:

Skripsi

*TOPIC MODELLING DAN DENSITY-BASED CLUSTERING UNTUK ANOMALY DETECTION  
PADA TRANSKRIP DISKUSI TERARAH KOMUNITAS*

Diskusi terarah dalam Komunitas X adalah sebuah usaha yang dilakukan secara berkelompok oleh tim penyelenggara untuk membuat sebuah *platform* dimana anggota bisa berpendapat dan berdiskusi tentang topik-topik yang disediakan. Karena panjangnya pendapat beberapa peserta, notulis seringkali kehilangan fokus dan tidak mendokumentasikan pendapat peserta dengan baik dan harus melakukan parafrase yang berdasarkan sumber yang tidak lengkap, sehingga data yang dicatat tidak mencerminkan secara akurat pernyataan yang dikeluarkan oleh pembicara, dan mungkin mengandung bias, sehingga menghambat proses evaluasi. Penelitian ini bertujuan untuk membuat sebuah *pipeline* yang memastikan tertangkapnya pendapat setiap peserta diskusi dan menggunakan metode unsupervised learning untuk mencari dan mengidentifikasi anomali dalam data dengan cara mencari kombinasi-kombinasi parameter yang sesuai untuk mendapatkan anomali yang dimaksud.

Kata Kunci : Density-based clustering, DBSCAN, Topic Modeling, Latent Dirichlet Allocation, Word-Embedding, Sentence-Embedding

## **ABSTRACT**

Edward Manhattan

Skripsi

TOPIC MODELLING AND DENSITY-BASED CLUSTERING FOR ANOMALY DETECTION ON  
COMMUNITY DISCUSSION TRANSCRIPT

The directed discussion held regularly by Community X is a group effort by the enabler team to create a platform where members of the community may speak their minds and discuss topics within the group's interest. Due to the volume of speech data and limitations in terms of manpower, the assigned annotator for a given discussion oftentimes loses focus and becomes unable to document members' contributions properly and resort to paraphrasing, which translates to loss of information and context, and presence of personal bias that may impede evaluation. This research is made with the goal of making a pipeline that ensures the retrieval of each member's contributions to a given discussion topic and utilize unsupervised learning to find and identify anomalies in the data by searching for effective combinations of parameters to obtain said anomalies.

**Keywords :** Density-based clustering, DBSCAN, Topic Modeling, Latent Dirichlet Allocation, Word-Embedding, Sentence-Embedding

## DAFTAR ISI

SKRIPSI.....	iii
LEMBAR PENGESAHAN.....	iv
LEMBARAN PERNYATAAN PERSETUJUAN PUBLIKASI KARYA ILMIAH .....	v
KATA PENGANTAR.....	vi
ABSTRAK.....	vii
DAFTAR ISI .....	ix
DAFTAR TABEL.....	xi
DAFTAR GAMBAR.....	xii
DAFTAR RUMUS .....	xiv
DAFTAR SEGMENT KODE.....	xv
1. PENDAHULUAN .....	1
1.1. Latar Belakang Masalah.....	1
1.2. Perumusan Masalah .....	4
1.3. Tujuan Penelitian .....	5
1.4. Ruang Lingkup.....	5
1.5. Metodologi Penelitian .....	6
1.6. Sistematika Penulisan .....	8
2. LANDASAN TEORI .....	10
2.1. Tinjauan Pustaka.....	10
2.2. Tinjauan Studi .....	20
3. ANALISA DAN DESAIN SISTEM .....	23
3.1. Analisa Permasalahan.....	23
3.2. Desain Eksperimen .....	24
3.3. Desain Eksperimen dan Program.....	31

4. IMPLEMENTASI SISTEM.....	42
4.1. Implementasi Umum .....	42
4.2. Implementasi Eksperimen .....	43
4.3. Implementasi Program .....	80
5. ANALISIS HASIL PENGUJIAN .....	169
5.1. Text Preprocessing dan Tokenization .....	169
5.2. Word Embedding.....	171
5.3. Sentence Embedding .....	178
5.4. Tiny Word-Embedding .....	182
5.5. Weighting dengan TF-IDF .....	182
5.6. Topic Modelling dengan LDA .....	185
5.7. Hasil Kuesioner .....	185
5.8. Kombinasi-Kombinasi Parameter .....	190
5.9. Analisa Kesuksesan Algoritma .....	199
6. KESIMPULAN .....	202
DAFTAR PUSTAKA.....	204

## DAFTAR TABEL

Tabel 3.1 Struktur Penyimpanan Data .....	25
Tabel 4.1 Fungsi-fungsi <i>library</i> yang di- <i>import</i> .....	45
Tabel 4.2 Mapping fungsi class AddWindow .....	103
Tabel 4.3 Mapping fungsi class ListenWindow .....	119
Tabel 4.4 Mapping fungsi class ProcessWindow.....	144
Tabel 4.5 Mapping fungsi class PresentWindow.....	154
Tabel 4.6 <i>Mapping</i> fungsi class TranslateText .....	165
Tabel 5.1 10 Kombinasi parameter dengan indeks Jaccard terbaik untuk dataset dengan ID 18 .....	200
Tabel 5.2 10 Kombinasi parameter dengan indeks Jaccard terbaik untuk dataset dengan ID 19 .....	200
Tabel 5.3 10 Kombinasi parameter dengan indeks Jaccard terbaik untuk dataset dengan ID 20 .....	201
Tabel 5.4 10 Kombinasi parameter dengan indeks Jaccard terbaik untuk dataset dengan ID 27 .....	201
Tabel 5.5 10 Kombinasi parameter dengan indeks Jaccard terbaik untuk dataset dengan ID 28 .....	202

## DAFTAR GAMBAR

Gambar 1.1 Dokumentasi Pertemuan Diskusi .....	2
Gambar 1.2 Contoh Notulen Diskusi.....	3
Gambar 2.1 Ilustrasi <i>word tokenization</i> (Hvitfeld & Silge, 2022) .....	11
Gambar 2.2 Perbandingan pendekatan <i>stemming</i> dan <i>lemmatization</i> (Aghammadzada, 2020). .....	12
Gambar 2.3 <i>Stop word removal</i> pada sejumlah kalimat Bahasa Inggris (GeekForGeeks, 2022) .....	12
Gambar 2.4 Beberapa fungsi jarak (Xu & Tian, 2015).....	13
Gambar 2.5 Pseudocode Penerapan DBSCAN (Ester, Kriegel, Sander, & Xu, 1996).....	14
Gambar 2.6 Pseudocode Fungsi <i>ExpandCluster()</i> yang Dinyatakan dalam Pseudocode DBSCAN (Ester, Kriegel, Sander, & Xu, 1996).....	15
Gambar 2.7 Clustering dengan DBSCAN; poin-poin data hitam menunjukkan <i>outlier</i> (Scikit Learn).....	16
Gambar 2.8 Visualisasi <i>Word2Vec</i> (Gautam, 2020). .....	16
Gambar 2.9 Arsitektur <i>skip-gram Word2Vec</i> (Firdaus, 2019).....	17
Gambar 2.10 Arsitektur CBOW <i>Word2Vec</i> (Firdaus, 2019). .....	17
Gambar 2.11 Diagram Model LDA (Lee & et.al., 2018).....	18
Gambar 3.1 Notulen diskusi yang telah terkumpul, disimpan dalam 14 <i>file</i> .....	27
Gambar 3.2 Diagram Alur Eksperimen – Pengumpulan Data .....	32
Gambar 3.3 Diagram Alur Eksperimen – Text Preprocessing .....	33
Gambar 3.4 Diagram Alur Eksperimen – <i>Feature Extraction</i> .....	35
Gambar 3.5 Diagram Alur Eksperimen – <i>Clustering &amp; Anomaly Detection</i> .....	36
Gambar 3.6 State Diagram Program yang Akan Dibuat.....	37
Gambar 3.7 <i>State Diagram</i> Alur Pengumpulan Klip Suara.....	38
Gambar 3.8 Pseudocode Alur Pengumpulan Klip Suara .....	39
Gambar 3.9 State Diagram Alur Proses dan Penyajian Data.....	40
Gambar 3.10 Pseudocode Alur Proses dan Penyajian Data.....	40
Gambar 4.1 Desain window MainMenu dan ListenWindow dalam program Qt Designer .....	81
Gambar 4.2 Desain MainMenu dalam program Qt Designer .....	82

Gambar 4.3 Window MainMenu dalam program yang sudah dirancang dan dijalankan	90
.....	.....
Gambar 4.4 Desain AddWindow dalam QtDesigner .....	90
Gambar 4.5 AddWindow untuk menambahkan pembicara baru.....	105
Gambar 4.6 AddWindow untuk menambahkan sesi diskusi baru.....	106
Gambar 4.7 Desain ListenWindow pada Qt Designer .....	106
Gambar 4.8 ListenWindow dengan data sesi, pembicara, dan pertanyaan yang sudah diisi .....	120
Gambar 4.9 ListenWindow dalam proses rekaman menggunakan pilihan Bahasa Indonesia .....	121
.....	.....
Gambar 4.10 Desain ProcessWindow dalam Qt Designer.....	121
Gambar 4.11 PresentWindow yang sedang menampilkan data sebuah sesi .....	147
Gambar 4.12 Desain PresentWindow pada Qt Designer .....	148
Gambar 4.13 Tampilan PresentWindow dengan satu <i>outlier</i> dalam data (merah)....	155
Gambar 4.14 Skema <i>database</i> untuk program.....	156
Gambar 5.1 Hasil <i>word-embedding</i> dengan model yang dibuat sendiri.....	170
Gambar 5.2 Model <i>word2vec</i> yang di- <i>train</i> sendiri .....	172
Gambar 5.3 Model <i>text8</i> .....	173
Gambar 5.4 Model <i>word2vec-google-news-300</i> .....	175
Gambar 5.5 Model <i>glove-wiki-gigaword-50</i> .....	176
Gambar 5.6 Model <i>glove-wiki-gigaword-100</i> .....	177
Gambar 5.7 Model <i>glove-wiki-gigaword-300</i> .....	178
Gambar 5.8 <i>Sentence-embedding</i> dengan <i>averaging</i> , tanpa <i>weighting</i> .....	180
Gambar 5.9 <i>Sentence-embedding</i> dengan <i>averaging</i> dan <i>weighting</i> dengan TF-IDF score .....	180
.....	.....
Gambar 5.10 <i>Sentence-embedding</i> dengan <i>sum</i> , tanpa <i>weighting</i> .....	181
Gambar 5.11 <i>Sentence-embedding</i> dengan <i>sum</i> dan <i>weighting</i> dengan TF-IDF score	181
Gambar 5.12 Para penyelenggara diskusi yang mengambil kuesioner .....	186

## **DAFTAR RUMUS**

Rumus 2.1 <i>Term Frequency</i> .....	19
Rumus 2.2 <i>Inverse Document Frequency</i> .....	20
Rumus 2.3 TF-IDF .....	20

## DAFTAR SEGMENT KODE

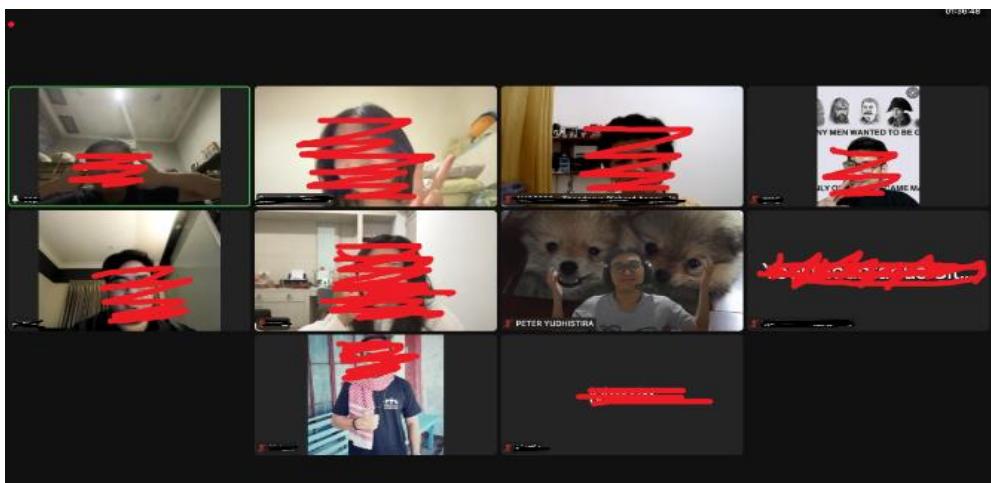
Segmen Kode 4.1 Imports .....	44
Segmen Kode 4.2 Fungsi untuk melakukan <i>text preprocessing</i> .....	45
Segmen Kode 4.3 Fungsi getWordnetPos .....	47
Segmen Kode 4.4 Fungsi untuk mengganti model <i>word embedding</i> yang digunakan dan <i>list</i> modelnya .....	49
Segmen Kode 4.5 Fungsi WordEmbed .....	49
Segmen Kode 4.6 Fungsi untuk melakukan <i>sentence-embedding</i> tanpa <i>weight</i> .....	50
Segmen Kode 4.7 Fungsi untuk melakukan <i>sentence-embedding</i> dengan TF-IDF sebagai <i>weight</i> .....	51
Segmen Kode 4.8 Fungsi tiny word-embedding untuk sebuah dokumen tanpa <i>weighting</i> .....	52
Segmen Kode 4.9 Fungsi <i>tiny word-embed</i> dengan <i>weighting</i> TF-IDF.....	53
Segmen Kode 4.10 Fungsi untuk mendapatkan Matriks TF-IDF .....	54
Segmen Kode 4.11 Fungsi untuk mendapatkan distribusi topik dengan LDA .....	55
Segmen Kode 4.12 Fungsi DBSCAN .....	56
Segmen Kode 4.13 Fungsi yang menjalankan algoritma LOF .....	57
Segmen Kode 4.14 Fungsi yang menjalankan algoritma Isolation Forest.....	59
Segmen Kode 4.15 anomalydetector.py .....	60
Segmen Kode 4.16 Konversi MainMenu.ui ke mainmenu.py .....	82
Segmen Kode 4.17 mainmenu.py .....	82
Segmen Kode 4.18 Class MainWindow .....	88
Segmen Kode 4.19 Konversi AddWindow.ui ke addwindow.py .....	91
Segmen Kode 4.20 addwindow.py.....	91
Segmen Kode 4.21 Class AddWindow.....	100
Segmen Kode 4.22 Konversi ListenWindow.ui ke listenwindow.py .....	107
Segmen Kode 4.23 listenwindow.py .....	107
Segmen Kode 4.24 Class ListenWindow.....	115
Segmen Kode 4.25 Konversi ProcessWindow.ui ke processwindow.py .....	122
Segmen Kode 4.26 Class ProcessWindow.....	134
Segmen Kode 4.27 Konversi PresentWindow.ui ke presentwindow.py .....	148
Segmen Kode 4.28 presentwindow.py .....	148

Segmen Kode 4.29 Class PresentWindow.....	152
Segmen Kode 4.30 Class DatabaseHandler dalam file db.py.....	156
Segmen Kode 4.31 <i>Mapping</i> fungsi class DatabaseHandler .....	162
Segmen Kode 4.32 Class TurnToTextinator dalam file turntotextinator.py .....	164
Segmen Kode 4.33 <i>Class TurnToTextinatorThread sebagai implementasi class TurnToTextinator yang memanfaatkan multithreading dengan QThread .....</i>	166
Segmen Kode 4.34 Class RecorderThread.....	168
Segmen Kode 5.1 Mendapatkan <i>scatter plot</i> untuk setiap <i>pre-trained</i> model. ....	173
Segmen Kode 5.2 Visualisasi kinerja fungsi agregat dan <i>weighting</i> pada <i>sentence-embedding</i> .....	179
Segmen Kode 5.3 <i>Tiny word-embedding</i> .....	182
Segmen Kode 5.4 Weighting pada <i>sentence-embedding</i> .....	183
Segmen Kode 5.5 <i>Tiny Word-embedding</i> dengan <i>weight</i> .....	184
Segmen Kode 5.6 <i>Ground truth</i> untuk <i>event</i> dengan ID 18 .....	187
Segmen Kode 5.7 <i>Ground truth</i> untuk <i>event</i> dengan ID 19 .....	187
Segmen Kode 5.8 <i>Ground truth</i> untuk <i>event</i> dengan ID 20 .....	188
Segmen Kode 5.9 <i>Ground truth</i> untuk <i>event</i> dengan ID 27 .....	189
Segmen Kode 5.10 <i>Ground truth</i> untuk <i>event</i> dengan ID 28.....	189
Segmen Kode 5.11 Fungsi untuk mencari kombinasi-kombinasi parameter.....	192
Segmen Kode 5.12 Fungsi untuk menjalankan <i>anomaly detection</i> untuk setiap kombinasi parameter.....	196
Segmen Kode 5.13 Fungsi untuk mendapatkan Jaccard Similarity antara <i>ground truth</i> dan hasil <i>anomaly detection</i> .....	199

## **1. PENDAHULUAN**

### **1.1. Latar Belakang Masalah**

Sebuah forum adalah sebuah pertemuan atau medium yang memungkinkan para pesertanya bertukar pendapat dan gagasan akan sebuah permasalahan ("Forum", 2022). Forum yang diselenggarakan oleh sekelompok orang yang memiliki ketertarikan pada bidang yang sama disebut forum komunitas. Dalam sebuah forum komunitas, terjadi diskusi dan pertukaran pendapat antar anggota mengenai sebuah topik yang ada dalam bidang yang dimengerti oleh anggota-anggota komunitas tersebut. Sebuah diskusi harus melibatkan peran seorang fasilitator dan moderator – seorang fasilitator membantu dan menginspirasi proses-proses komunikasi untuk membangun pengetahuan, dan seorang moderator bertugas untuk menggabungkan kontribusi-kontribusi para peserta diskusi dengan cara mengambil pernyataan-pernyataan yang dilontarkan peserta dan mengaitkannya dengan topik dan teori dalam diskusi (Kienle & Ritterskamp, 2007). Pada Komunitas X yang diikuti penulis, terdapat sebuah forum yang diselenggarakan secara daring melalui *platform* Zoom Meetings. Diskusi-diskusi yang terjadi dalam forum ini dibantu dengan adanya fasilitator yang mengarahkan diskusi ke topik yang diinginkan, dan moderator yang memastikan koherensi pendapat setiap peserta diskusi. Selama berlangsungnya sebuah diskusi, seorang moderator juga berperan sebagai notulis yang mencatat pernyataan-pernyataan yang dikeluarkan oleh peserta diskusi. Catatan notulen ini akan dievaluasi untuk menilai kontribusi setiap peserta pada diskusi, serta sebagai dasar pengambilan keputusan untuk topik diskusi selanjutnya. Khususnya, catatan notulen digunakan untuk mengidentifikasi pendapat-pendapat peserta yang tidak sesuai dengan topik diskusi.



Gambar 1.1 Dokumentasi Pertemuan Diskusi

Untuk menghargai hak privasi setiap anggota peserta diskusi, pihak penyelenggara diskusi tidak menggunakan fitur rekaman yang disediakan oleh platform Zoom Meeting. Oleh karena itu, dokumentasi berupa notulen dibuat oleh moderator atau seorang penyelenggara secara *real-time*. Notulis mencatat secara langsung pernyataan yang dikeluarkan oleh peserta diskusi, baik secara lisan maupun melalui fitur *chat* yang disediakan oleh platform *Zoom Meeting*. Notulis juga melakukan *filtering* pada pernyataan-pernyataan yang dicatatnya, yaitu melakukan sensor terhadap perkataan-perkataan yang kasar atau tidak berkenan, serta melakukan peringkasan terhadap pernyataan-pernyataan yang panjang. Pada akhir pernyataan, moderator menyampaikan ulang pendapat peserta yang baru saja dicatat secara lebih ringkas sebagai bentuk konfirmasi ketepatan catatan.

Trigger	Respond
Opinion : Jesus lives again vs it is all in their minds vs Jesus fainted vs Jesus was stolen	<p>Ce Sam : I agree that [Jesus is stolen] makes sense. It COULD have happened.</p> <p>Ko Ricky : I agree that [Jesus is stolen] makes sense. They could have stolen jesus' body in order to 'fulfill the prophecy'. When i heard that theory, while i don't know exactly, i agree that that theory is plausible.</p> <p>Greg : Historically, I think that jesus' family wasn't poor. Nazareth is no small town. They COULD have bribed the guards because they had the money.</p> <p>Sam : @Greg How do you believe something, but also are open to the possibility of the opposite being true?</p> <p>Ricky : I am open to possibilities. Whichever is true, so be it.</p> <p>Greg : Elements of the theories might be present, but ultimately I believe that Jesus lives again.</p> <p>Audrey : The bribe theory is implausible. Judas who betrayed Jesus received money. Everybody hated Jesus.</p>

Gambar 1.2 Contoh Notulen Diskusi

Namun, terdapat beberapa kendala dalam sistem pencatatan dan evaluasi yang diterapkan oleh penyelenggara diskusi dalam Komunitas X. Pembuatan keputusan dan penilaian yang dilakukan secara terus menerus dapat menurunkan fungsi eksekutif dan sumber daya mental seseorang (Danziger, Levav, & Avnaim-Pesso, 2011). Karena panjangnya pendapat beberapa peserta, notulis bisa kehilangan fokus dan tidak mendokumentasikan pendapat peserta dengan baik. Seringkali notulis harus melakukan parafrase yang berdasarkan sumber yang tidak lengkap. Akibatnya, data yang dicatat tidak mencerminkan secara akurat pernyataan yang dikeluarkan oleh pembicara, dan mungkin mengandung bias, sehingga menghambat proses evaluasi. Selain itu, karena besarnya volume teks yang harus dievaluasi dan terbatasnya waktu kerja, komite penyelenggara diskusi mengalami kesulitan untuk mengidentifikasi pendapat-pendapat peserta diskusi yang melenceng dari topik.

Oleh karena itu, diperlukan sebuah metode untuk mendapatkan transkrip diskusi, merepresentasikan fitur-fitur dari pendapat setiap peserta diskusi dan mendeteksi pendapat-pendapat yang melenceng dari topik diskusi. Untuk tujuan ini, akan digunakan metode *clustering* dan *topic modelling* untuk melakukan *anomaly detection* pada dataset transkrip diskusi Komunitas X. Hasil dari penelitian akan dievaluasi oleh komite penyelenggara diskusi di Komunitas X dan akan diterapkan dalam bentuk program untuk membantu proses diskusi.

Studi yang dilakukan oleh (Curiskis & al., 2019) mengevaluasi *clustering* dan *topic modelling* pada media sosial Twitter dan Reddit. Dalam studi tersebut, dikumpulkan dataset dari media sosial Twitter dan Reddit yang direpresentasikan dalam empat representasi fitur, yaitu TF-IDF, word2vec dengan *weight*, word2vec tanpa *weight*, dan doc2vec. Dilakukan pula empat metode *clustering*, yaitu *K-means Clustering*, *K-medoids clustering*, *hierarchical agglomerative clustering*, dan *Non-negative matrix factorization*. Kesuksesan penelitian diukur dengan metrik evaluasi intrinsik Normalized Mutual Information (NMI), dan Adjusted Mutual Information (AMI), dan metric evaluasi ekstrinsik Adjusted Rand Index (ARI). Ditemukan bahwa representasi fitur word2vec dan metode *K-means clustering* membuat hasil terbaik dibandingkan kombinasi-kombinasi lainnya.

Penelitian yang dilakukan oleh Wiliams & Betak (Williams & Betak, 2019) membandingkan performa *topic modelling* Latent Dirichlet Allocation (LDA) dengan Latent Semantic Analysis (LSA) dan menemukan bahwa kedua algoritma tersebut menemukan beberapa topik yang sama, namun ada beberapa topik yang eksklusif satu sama lain. Penelitian lainnya dilakukan untuk membandingkan performa LDA dan LSA terhadap satu sama lain menggunakan dataset berita BBC dan ditemukan bahwa dalam kasus ini, *Accuracy* algoritma LDA pada nilai 82.57 lebih tinggi dari *Accuracy* algoritma LSA pada nilai 75.30 (Kalepalli, Tasneem, Phani Teja, & Manne, 2020). Algoritma LDA ditemukan memiliki nilai *divergence* yang lebih besar daripada LSA, dan direkomendasikan untuk dataset berukuran lebih kecil. Studi yang dilakukan terhadap data *e-books* menunjukkan bahwa LDA memiliki performa yang lebih baik daripada LSA (Mohammed & Al-augby, 2020). Dalam kasus terbaiknya, LDA dengan asumsi jumlah topik 20 memiliki *coherence value* sebesar 0.592179, dibandingkan LSA dengan asumsi jumlah topik 10 memiliki *coherence value* sebesar 0.577302.

Survei yang dilakukan oleh Chandola, Banerjee & Kumar (Chandola, Banerjee, & Kumar, 2009) menyatakan beberapa metode yang dapat dilakukan untuk melakukan *anomaly detection*, diantaranya adalah penerapan *density-based clustering* untuk mengelompokkan data dalam beberapa *cluster*. Poin-poin data yang menjadi *outlier* diidentifikasi sebagai titik-titik anomali.

## 1.2. Perumusan Masalah

Berdasarkan latar belakang yang tertulis, dapat dirumuskan permasalahan sebagai berikut :

- Metode *feature extraction* manakah yang lebih baik untuk mencerminkan pendapat peserta diskusi sebagai acuan untuk *anomaly detection*?

- Kombinasi parameter dan *hyperparameter* seperti apakah yang dapat melakukan *anomaly detection* yang sesuai dengan kebutuhan penyelenggara diskusi?

### **1.3. Tujuan Penelitian**

Tujuan skripsi ini adalah membandingkan *feature extraction* berbasis *embedding* dan *topic modelling* dengan LDA untuk melakukan *anomaly detection* pada transkrip diskusi Komunitas X dengan pendekatan *density-based clustering*. Pendekatan yang membawa hasil terbaik menurut evaluasi komite penyelenggara diskusi akan diterapkan dalam sebuah program yang dalam *pipeline*-nya mendapatkan transkrip diskusi, melakukan *pre-processing* dan vektorisasi, lalu mengidentifikasi pendapat-pendapat yang tidak sesuai dengan topik bahasan diskusi.

### **1.4. Ruang Lingkup**

Ruang lingkup dibatasi pada :

1. Menggunakan bahasa pemrograman *Python* dan *library-library* yang dibuat dalam bahasa *Python*.
2. Pembuatan dan uji coba algoritma akan dilakukan pada *IDE Visual Studio Code* dan *Jupyter Notebook*.
3. *Library* yang digunakan adalah *NLTK* dan *gensim* untuk *Natural Language Processing* dan *sklearn* untuk model-model *Machine Learning*. Digunakan pula beberapa *library* yang akan ditentukan untuk pembuatan GUI, sampling transkrip audio, dan integrasi komponen-komponen program.
4. Komponen-komponen berupa *speech-to-text* dan *translator* menggunakan API yang sudah ada.
5. Dataset berupa pernyataan yang tercatat dalam notulen diskusi. Satu rekor data berupa sebuah dokumen yang berisi pendapat seorang peserta diskusi yang dicatat pada satu kesempatan berbicara.
6. Untuk poin 5), dataset didapatkan dari proses transkripsi diskusi menggunakan program *speech-to-text* yang sudah ada.

7. Dataset penelitian untuk mengembangkan sistem *anomaly detection* didapatkan dari gabungan transkrip manual yang sudah dikumpulkan oleh tim penyelenggara dari sesi-sesi terdahulu.
8. Untuk poin 5), bahasa yang digunakan dalam diskusi dan pembuatan materi adalah Bahasa Inggris, dan respon-respon yang diberikan dalam Bahasa Indonesia atau bahasa campuran akan terlebih dahulu diterjemahkan ke Bahasa Inggris menggunakan API yang sudah ada.
9. Akan dilakukan *feature extraction* berupa *vectorization* melalui *word-* dan *sentence-embedding*, serta *topic modelling* untuk mengubah dokumen menjadi vektor numerik.
10. Akan juga diambil *term-document matrix* dengan metode TF-IDF sebagai metode *weighting* untuk pendekatan *vectorization* dan *topic modelling*.
11. Proses *anomaly detection* akan dilakukan dengan algoritma *Density-Based Spatial Clustering of Applications with Noise* (DBSCAN). *Outlier-outlier* yang terbentuk dari proses ini akan diidentifikasi sebagai anomali dalam dataset.
12. Hasil penelitian ini berupa suatu program yang telah disetujui penggunaannya oleh penyelenggara diskusi Komunitas X sejak usulan ini dibuat, dan akan digunakan untuk membantu proses evaluasi diskusi dalam Komunitas X.

### **1.5. Metodologi Penelitian**

Langkah-langkah dalam pengerjaan skripsi meliputi hal di bawah ini:

#### **1.5.1. Studi Literatur**

- Mempelajari prinsip-prinsip *Natural Language Processing* (NLP).
- Mempelajari *library-library* yang menerapkan proses-proses NLP, khususnya NLTK dan gensim.
- Mempelajari konsep dan teknik *vectorization* berupa *word-embedding* dan *sentence-embedding*.
- Mempelajari konsep, teknik, dan penerapan Latent Dirichlet Allocation sebagai metode *feature extraction*.
- Mempelajari konsep dan penerapan TF-IDF.
- Mempelajari definisi anomali dalam data dan mengeksplorasi metode-metode yang dapat digunakan untuk mencari anomali dalam data teks.
- Mempelajari konsep, teknik, dan penerapan metode-metode *clustering* yang ada, khususnya *density-based clustering*.

### **1.5.2. Pengumpulan Dataset Penelitian**

- Pengumpulan dataset berupa notulen diskusi yang berupa teks.
- Pengambilan data berupa transkrip audio diskusi yang akan diolah untuk menghasilkan transkrip berupa teks.
- Penerjemahan data transkrip menjadi Bahasa Inggris untuk memudahkan proses-proses NLP.
- *Cataloging* dataset, dikelompokkan berdasarkan tanggal *event* dan peserta diskusi yang berbicara.

### **1.5.3. Perencanaan Penelitian dan Pengembangan Program**

- Mengumpulkan data transkrip diskusi yang sudah ada, baik berupa teks atau transkrip audio.
- Mengubah data transkrip audio menjadi teks dengan API speech-to-text yang sudah ada, serta melakukan penerjemahan ke dalam Bahasa Inggris secara semi-manual dengan menggunakan bantuan API *translation* yang sudah ada.
- Menyusun data transkrip diskusi dengan struktur yang sesuai, dalam rekor-rekor yang memiliki identifikasi berupa nama peserta, identifikasi acara diskusi, dan pendapat peserta yang berbicara.
- Segmentasi data berdasarkan kategori-kategori yang ada sebagai variabel eksperimen.  
Akan dilakukan segmentasi teks sebagai berikut:
  - Kontribusi total peserta dalam sebuah diskusi. Satu poin data mewakili sebuah dokumen yang menggabungkan seluruh kontribusi tiap-tiap peserta dalam satu sesi diskusi.
  - Sesi berbicara. Setiap poin data mewakili transkrip yang didapatkan dari satu kesempatan seorang peserta berbicara dalam sebuah sesi diskusi.
  - Sesi berbicara dan kalimat. Setiap poin data mewakili transkrip yang didapatkan dari satu kesempatan seorang peserta berbicara dalam sebuah sesi diskusi, tapi dipisahkan per kalimat. Segmentasi ini dilakukan untuk mengubah matriks TF-IDF.
- Melakukan *preprocessing* pada data transkrip berupa *case folding*, *tokenizing*, *lemmatizing*, dan *stop-words removal*. Akan diuji kombinasi-kombinasi dari teknik *preprocessing* untuk menentukan kombinasi *preprocessing* yang menghasilkan *corpus* yang paling baik performanya.
- *Word-tokenizing* untuk mendapatkan *corpus* yang akan menjadi basis eksperimen.

- *Feature extraction* dengan pendekatan *vectorization* melalui *word embedding* dan *sentence embedding*. Akan dilakukan pengujian terhadap beberapa variabel untuk menentukan kombinasi dan parameter seperti apa yang menghasilkan fitur-fitur data yang baik performanya.
- *Feature extraction* dengan pendekatan *topic modelling*, yaitu dengan metode *Latent Dirichlet Allocation* (LDA). Akan dilakukan eksperimen dengan parameter jumlah topik yang diinferensikan untuk mencapai komposisi paling baik.
- Menerapkan *density-based clustering* untuk mendapatkan *cluster-cluster* data, dan mengidentifikasi *outlier* pada data.
- Menggabungkan elemen-elemen yang sudah disebutkan diatas dalam sebuah data *pipeline* yang menerima data transkrip berupa teks dan mengidentifikasi anomali-anomali yang ada dalam transkrip tersebut.
- Mengintegrasikan API-API *speech-to-text* dan *translation* yang ada untuk mengubah transkrip audio menjadi teks.
- Mendesain *use case*, *database*, dan *UI* program.

#### **1.5.4. Pengujian Program**

- Menguji penggunaan program pada beberapa transkrip audio diskusi.
- Mengumpulkan 6 penyelenggara diskusi komunitas untuk mengevaluasi anomali yang diidentifikasi oleh program pada transkrip-transkrip diskusi yang dipakai.

#### **1.5.5. Pengambilan Kesimpulan**

- Membuat kesimpulan dari keputusan-keputusan yang dibuat oleh para penyelenggara diskusi komunitas.
- Merumuskan saran apa yang bisa dilakukan untuk penelitian berikutnya.
- Mengidentifikasi kelemahan pada komponen-komponen program, bila ada, dan merumuskan alternatif yang mungkin memberikan performa lebih baik.

### **1.6. Sistematika Penulisan**

Penulisan laporan skripsi ini dibagi menjadi beberapa bab, yaitu :

- Bab I : Pendahuluan

Bab ini berisikan judul, latar belakang, perumusan masalah, ruang lingkup, tujuan skripsi, dan metodologi penelitian yang akan digunakan dalam skripsi ini.

- Bab II : Landasan Teori

Bab ini berisikan teori-teori yang digunakan dan diterapkan dalam skripsi ini.

- Bab III : Analisa Dan Desain Sistem

Menjelaskan analisa masalah yang dihadapi dan perencanaan pembuatan keseluruhan sistem dalam aplikasi yang akan dibuat.

- Bab IV : Implementasi Sistem

Bab ini berisikan tentang implementasi sistem berdasarkan desain.

- Bab V : Pengujian Sistem

Bab ini berisi tentang hasil pengujian yang dilakukan terhadap aplikasi yang telah dibuat berdasarkan implementasi pada sistem yang telah dirancang dan dibuat pada Bab IV.

- Bab VI : Kesimpulan Dan Saran

Bab ini berisikan kesimpulan yang dapat diambil terhadap hasil yang dicapai, dan saran-saran yang berguna bagi pengembangan selanjutnya.

## **2. LANDASAN TEORI**

### **2.1. Tinjauan Pustaka**

Pada bab ini akan dijelaskan mengenai teori-teori yang akan digunakan dalam penulisan skripsi dan pembuatan aplikasi.

#### **2.1.1. Natural Language Processing (NLP)**

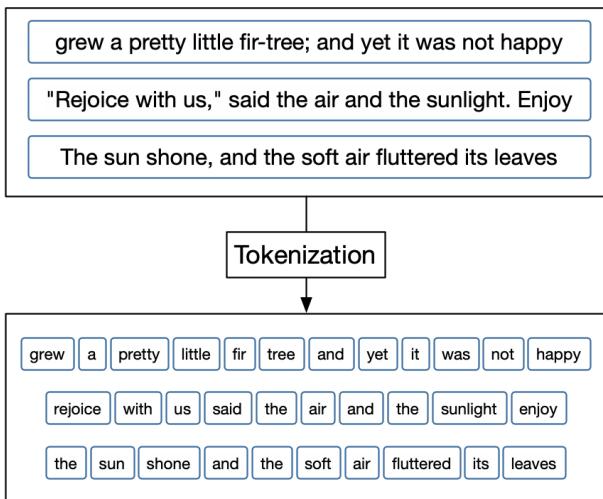
*Natural Language Processing* (NLP) adalah sebuah bidang dalam ilmu komputer dan ilmu bahasa yang mempelajari interaksi antara komputer dan bahasa natural manusia (Kumar, 2013). NLP mencakup studi dalam berbagai tingkatan dalam bahasa manusia, yaitu fonologi, morfologi, leksikon, sintaktik, semantik, pembicaraan, dan pragmatik (Ribeiro, 2021). Penerapan-penerapan NLP dapat berupa Word Processing, pengecekan dan perbaikan ejaan, Information Retrieval (IR), Information Extraction, Information Categorization, penjawaban pertanyaan, Information Summarization, dan Machine Translation (Chowdhary, 2020). Dalam penelitian ini, teknik-teknik NLP dipakai untuk memroses data transkrip yang didapatkan dari sebuah diskusi melalui program *speech-to-text*.

#### **2.1.2. Text Preprocessing**

Dalam NLP, *Text Preprocessing* adalah teknik yang digunakan untuk mempersiapkan teks yang tidak terstruktur menjadi data yang baik dan siap untuk diolah (Katryn, 2020). *Preprocessing* penting untuk dilakukan untuk mengurangi ukuran file teks dokumen yang akan dianalisa (Kannan, Gurusamy, Vijayarani, Ilamathi, & Nithya, 2014). Penggunaan teknik-teknik preprocessing dilakukan sesuai dengan karakteristik data teks dan kebutuhan penelitian. Teknik-teknik yang digunakan dalam penelitian ini antara lain *tokenization*, *stemming*, *lemmatization*, dan *stop word removal*.

*Tokenization* adalah sebuah proses memecah sebuah dokumen menjadi kumpulan kata, frasa, simbol, atau elemen-elemen lainnya yang disebut token (Kannan, Gurusamy, Vijayarani, Ilamathi, & Nithya, 2014). Secara umum, ada tiga tipe token yang bisa didapatkan dari sebuah dokumen, yaitu *word*, *character*, dan *subword* (Pai, 2020). Token bertipe *word* memecah sebuah dokumen menjadi sekumpulan kata, dan *subword* memecah sebuah dokumen menjadi sekumpulan kata yang dipecah ke bentuk dasar dan imbuhanya. Untuk penelitian ini, dilakukan *word-tokenization*, yaitu pemecahan dokumen menjadi kumpulan kata-kata yang akan

direpresentasikan dalam vektor pada proses selanjutnya. Proses *word-tokenization* ini juga dilakukan untuk mempermudah *lemmatizing* dan *stemming* yang akan dilakukan sebagai bagian dari *preprocessing*.

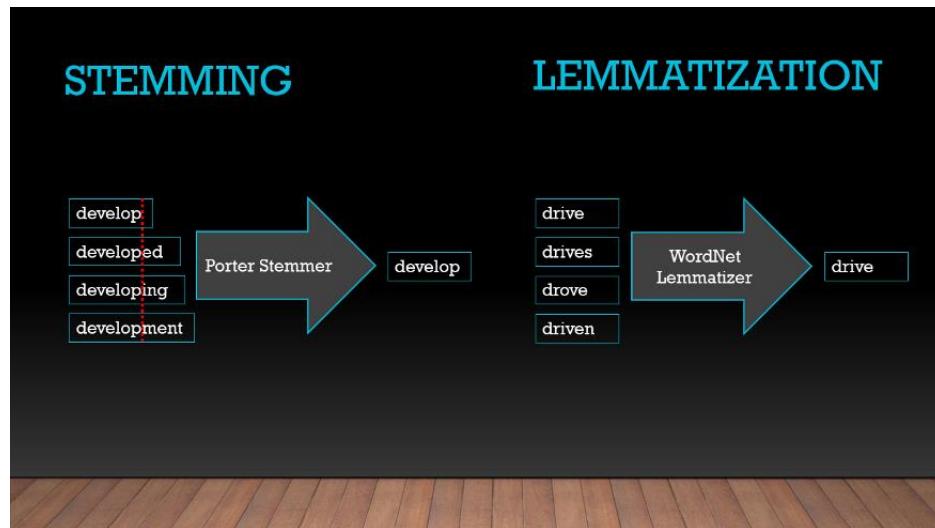


Gambar 2.1 Ilustrasi *word tokenization* (Hvitfeld & Silge, 2022)

*Stemming* adalah sebuah proses mengubah varian sebuah kata menjadi kata dasarnya, yang disebut *stem* atau batang (Kannan, Gurusamy, Vijayarani, Ilamathi, & Nithya, 2014). Teknik ini dilakukan untuk generalisasi kata kerja, supaya kata-kata kerja dalam bentuk bentuk berbeda tidak dianggap token-token yang berbeda. Dalam penelitian yang diajukan, karena keseluruhan data transkrip ada dalam Bahasa Inggris, dapat diterapkan algoritma Porter Stemming yang diterapkan oleh Martin Porter. Algoritma ini memanfaatkan beberapa prinsip dan aturan untuk menghilangkan *suffix* (akhiran) pada setiap kata untuk mendapatkan stem dari kata tersebut. Algoritma ini dimaksudkan untuk meningkatkan kinerja *information retrieval* dan mungkin memproduksi sebuah *stem* yang sama untuk dua kata dengan makna yang berbeda (Porter, 1980).

*Lemmatization* adalah proses pengelompokan bentuk-bentuk terinfleksi dari sebuah kata menjadi satu bentuk dasar (*lemma*) (Dictionary, n.d.). Tidak seperti *stemming* yang menggunakan pendekatan berdasarkan karakter, pendekatan reduksi dengan *lemmatization* bergantung pada pengetahuan tata bahasa yang bersangkutan untuk mengenali *lemma* sebuah kata infleksi. Oleh karena itu, proses ini membutuhkan proses *POS tagging* untuk mengenali bentuk kata (contohnya: “working” sebagai kata kerja atau sebagai kata sifat –tergantung konteks kalimat) untuk melakukan klasifikasi dengan benar. Dengan pendekatan ini, makna kata dapat dipertahankan dan tidak ambigu dengan *lemma* lainnya. Dalam penelitian yang akan

dijalankan, akan diuji pendekatan *stemming* dan *lemmatization* secara terpisah untuk mendapatkan hasil yang lebih baik.



Gambar 2.2 Perbandingan pendekatan *stemming* dan *lemmatization* (Aghammadzada, 2020).

*Stop word removal* adalah sebuah proses penghapusan kata-kata yang frekuensi kemunculannya tinggi, namun tidak memberikan makna yang signifikan pada dokumen. *Stop word* dalam Bahasa Inggris dapat mencapai 20 hingga 30 persen jumlah kata total dalam sebuah teks dokumen (Kannan, Gurusamy, Vijayarani, Ilamathi, & Nithya, 2014). Penghapusan *stop-words* pada sebuah korpus bertujuan untuk mengurangi *noise* dan token-token kata yang redundan.

Sample text with Stop Words	Without Stop Words
GeeksforGeeks – A Computer Science Portal for Geeks	GeeksforGeeks , Computer Science, Portal ,Geeks
Can listening be exhausting?	Listening, Exhausting
I like reading, so I read	Like, Reading, read

Gambar 2.3 *Stop word removal* pada sejumlah kalimat Bahasa Inggris (GeekForGeeks, 2022)

### 2.1.3. Density-Based Clustering

*Clustering* atau *cluster analysis* adalah sebuah metode *unsupervised machine learning* yang bertujuan untuk menemukan hubungan dan pengelompokan yang sudah ada pada data (Brownlee, 2020). Metode ini disebut *unsupervised* karena tidak membutuhkan label pada data; *clustering* berfungsi untuk mengelompokkan data dan menarik kesimpulan berdasarkan kemiripan-kemiripan fitur data tersebut. Ada beberapa pendekatan yang bisa diambil untuk

melakukan *clustering*, diantaranya *centroid-based clustering*, *density-based clustering*, *distribution-based clustering*, dan *hierarchical clustering* (Google, 2022). Masing-masing pendekatan tersebut. Konsep fundamental *clustering* adalah mengidentifikasi kelompok-kelompok yang terbentuk dari kemiripan antar rekord data. Terutama untuk data kuantitatif, digunakan fungsi yang menentukan jarak antar poin data yang dimiliki (Xu & Tian, 2015). Terdapat beberapa algoritma untuk menentukan jarak yang dipakai dalam *machine learning*, diantaranya adalah Euclidean Distance, Cosine Distance, dan Mahalanobis Distance.

Minkowski distance

$$\left( \sum_{l=1}^d |x_{il} - x_{jl}|^n \right)^{1/n}$$

Cosine distance

$$1 - \cos \alpha = \frac{x_i^T x_j}{\|x_i\| \|x_j\|}$$

Gambar 2.4 Beberapa fungsi jarak (Xu & Tian, 2015)

Pendekatan yang diambil untuk suatu kasus tertentu harus sesuai dengan kebutuhan penelitian dan sifat dari dataset yang dimiliki, dan oleh *tools* yang tersedia untuk melakukan *clustering* (Witten, Frank, Hall, & Pal, 2016). Dalam penelitian yang diajukan, akan dilakukan *anomaly detection* dengan mengidentifikasi outlier pada data. Oleh karena itu, *density-based clustering* akan digunakan karena tidak memasukkan *outlier* ke *cluster* manapun. (Google, 2022), dan menentukan cluster-cluster berdasarkan kedekatan relatif antar poin data. Untuk tujuan ini, akan digunakan metode DBSCAN (Density Based Spatial Clustering of Applications with Noise) untuk memproduksi *cluster-cluster* pada dataset, dengan harapan ditemukannya poin-poin data yang tidak tergabung dalam cluster manapun sebagai *outlier*. Adapun akan diuji secara paralel fungsi-fungsi jarak yang ada untuk menentukan pendekatan yang paling efektif dan sesuai dengan kebutuhan penelitian.

#### **2.1.4. Density-Based Spatial Clustering of Applications with Noise (DBSCAN)**

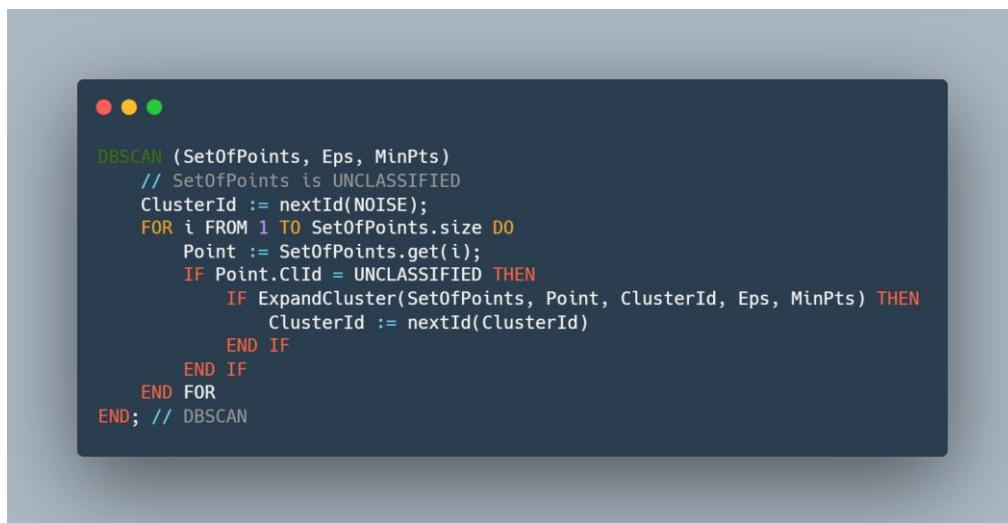
DBSCAN adalah sebuah algoritma *density-based clustering* yang membuat *cluster-cluster* pada poin-poin data yang tidak berlabel berdasarkan kedekatannya. Algoritma ini dibuat sebagai realisasi pendekatan intuitif yang dapat disimpulkan ketika mengevaluasi poin-poin data

yang memiliki kedekatan (density) tertentu yang dapat memisahkan sebuah *cluster* dengan *noise* (Ester, Kriegel, Sander, & Xu, 1996).

Algoritma DBSCAN menerima dua parameter yang ditentukan oleh pengguna, yaitu *MinPts* dan  $\varepsilon$  (eps; epsilon).

Adapun poin-poin data dikategorikan berdasarkan parameter yang ada sebagai berikut:

- *Core point*, jika ada *MinPts* buah poin data yang terletak dalam radius  $\varepsilon$  di sekelilingnya.
- Poin data  $p$  dikatakan *directly density-reachable point* dari poin data  $q$ , jika poin data  $p$  memiliki jarak maksimum  $\varepsilon$  dari  $q$ .
- Poin data  $p$  dikatakan *density-reachable point* dari poin data  $q$ , jika ada sebuah jalur atau path di antara poin data  $p$  dan  $q$  berupa poin-poin data yang berjarak maksimum  $\varepsilon$  antar satu sama lain.
- *Outlier* atau *noise point*, jika poin data tersebut tidak reachable dari poin-poin data lainnya.



```
DBSCAN (SetOfPoints, Eps, MinPts)
    // SetOfPoints is UNCLASSIFIED
    ClusterId := nextId(NOISE);
    FOR i FROM 1 TO SetOfPoints.size DO
        Point := SetOfPoints.get(i);
        IF Point.ClId = UNCLASSIFIED THEN
            IF ExpandCluster(SetOfPoints, Point, ClusterId, Eps, MinPts) THEN
                ClusterId := nextId(ClusterId)
            END IF
        END IF
    END FOR
END; // DBSCAN
```

Gambar 2.5 Pseudocode Penerapan DBSCAN (Ester, Kriegel, Sander, & Xu, 1996)

```

● ● ●

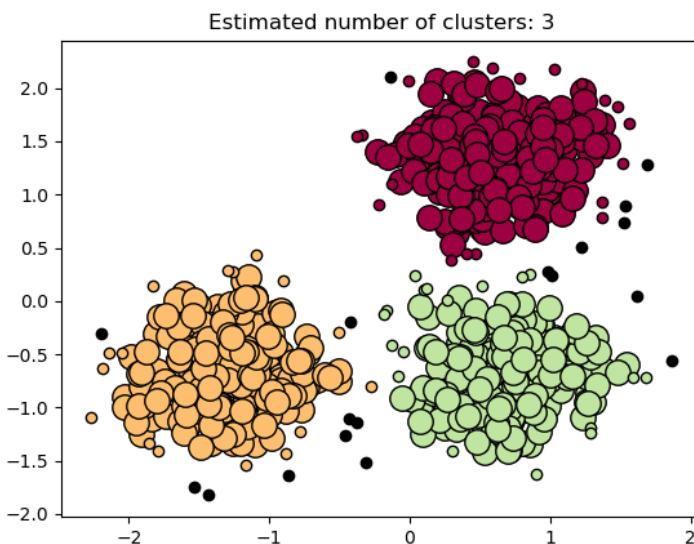
ExpandCluster (SetOfPoints, Point, ClId, Eps,MinPts) : Boolean;
seeds:=SetOfPoints.regionQuery(Point,Eps);
IF seeds.size<MinPts THEN // no core point
    SetOfPoint.changeClId(Point,NOISE);
    RETURN False;
ELSE // all points in seeds are density-
    // reachable from Point
    SetOfPoints.changeClIds(seeds,ClId);
    seeds.delete(Point);
    WHILE seeds <> Empty DO
        currentP := seeds.first();
        result := SetOfPoints.regionQuery(currentP,Eps);
        IF result.size >= MinPts THEN
            FOR i FROM 1 TO result.size DO
                resultP := result.get(i);
                IF resultP.ClId IN {UNCLASSIFIED, NOISE} THEN
                    IF resultP.ClId = UNCLASSIFIED THEN
                        seeds.append(resultP);
                    END IF;
                    SetOfPoints.changeClId(resultP,ClId);
                END IF; // UNCLASSIFIED or NOISE
            END FOR;
        END IF; // result.size >= MinPts
        seeds.delete(currentP);
    END WHILE; // seeds <> Empty
    RETURN True;
END IF;
END; // ExpandCluster

```

Gambar 2.6 Pseudocode Fungsi ExpandCluster() yang Dinyatakan dalam Pseudocode DBSCAN (Ester, Kriegel, Sander, & Xu, 1996).

DBSCAN diterapkan secara iteratif untuk setiap poin yang ada dalam dataset. Setiap iterasi menentukan apakah diperlukan adanya *cluster* baru atau tidak, dengan cara menghitung poin-poin yang berdekatan dengan poin data yang bersangkutan. Dalam Gambar 2.5., perhitungan dilakukan dengan menjalankan fungsi ExpandCluster() yang dijelaskan dalam Gambar 2.6. Dalam fungsi ExpandCluster(), akan dicari poin-poin yang merupakan *core points*. Poin-poin tersebut akan diiterasi secara rekursif untuk mencari dan menandai *density-reachable points* yang ada di sekitarnya. Poin-poin yang tidak memenuhi kriteria akan ditandai sebagai *noise* atau *outlier*.

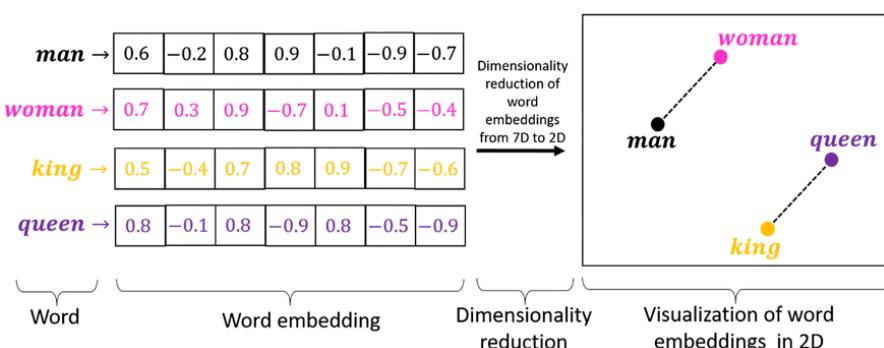
DBSCAN mendefinisikan sebuah *cluster* sebagai kumpulan poin-poin data yang *reachable* antar satu sama lain. *Outlier* adalah poin data yang tidak *reachable* dari poin-poin data lainnya. Oleh karena itu, DBSCAN akan digunakan dalam penelitian ini untuk mengidentifikasi anomali dalam data berupa data point yang dianggap *noise* atau *outlier*.



Gambar 2.7 Clustering dengan DBSCAN; poin-poin data hitam menunjukkan *outlier* (Scikit Learn).

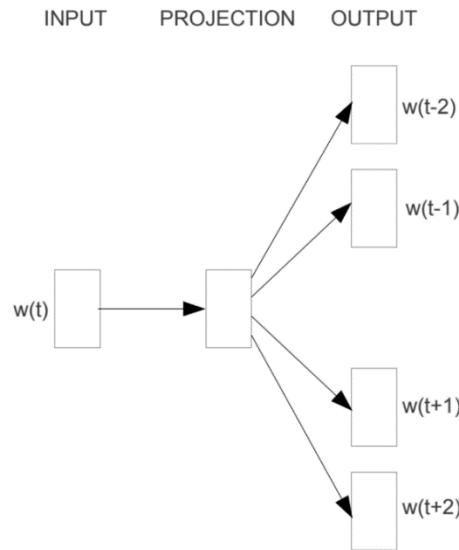
### 2.1.5. Word Embedding

*Word embedding* adalah metode untuk menangkap makna semantik dan sintaktik kata-kata yang ada dalam sebuah korpus teks yang tidak dilabeli (Lai, Liu, He, & Zhao, 2016). Metode ini dilakukan untuk melakukan *encoding* setiap word token yang telah dikumpulkan menjadi sebuah vektor bilangan riil yang bisa direpresentasikan dalam sebuah dimensi tertentu. Vektor-vektor bilangan yang didapatkan dari hasil *embedding* dapat ditransformasikan menjadi vektor dengan dimensi lebih kecil melalui sebuah proses yang disebut *dimensionality reduction*.

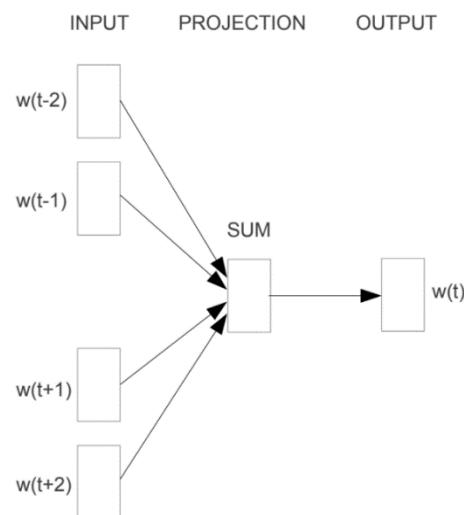


Gambar 2.8 Visualisasi *Word2Vec* (Gautam, 2020).

*Word2Vec* adalah suatu metode untuk membangun *Word Embedding* secara efisien berupa *neural network* yang terdiri dari dua *layer* untuk melakukan proses pengubahan teks menjadi vektor. *Word2Vec* memiliki 2 pilihan arsitektur yaitu *CBOW* (*Continuous Bag of words*) dan *Skip-Gram*.



Gambar 2.9 Arsitektur *skip-gram* *Word2Vec* (Firdaus, 2019)



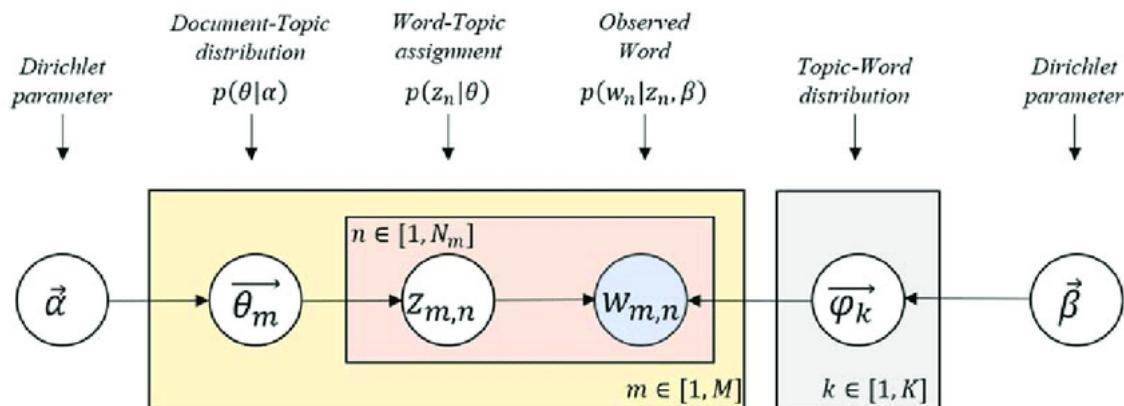
Gambar 2.10 Arsitektur *CBOW* *Word2Vec* (Firdaus, 2019).

Arsitektur *skip-gram* memprediksi kata-kata konteks yang ada di sekitar kata utama. Sebaliknya, arsitektur *CBOW* mencoba memprediksi *encoding* sebuah kata berdasarkan kata-kata konteks di sekitarnya. Dengan arsitektur-arsitektur ini, *Word2Vec* memprediksi konteks sebuah word token dalam korpus berdasarkan sejumlah kata-kata konteks yang berdekatan dengan kata pilihan; banyaknya kata yang diambil sebagai konteks disebut dengan istilah *window*. Dalam proses *training*, pendekatan ini akan berusaha mengatur *weight* dari setiap kata sedekat mungkin dengan makna aslinya. Kata-kata yang dekat maknanya akan memiliki vektor representasi yang berdekatan dalam *embedding* *Word2Vec* (Firdaus, 2019). Dalam studi yang akan dilakukan, sebuah dokumen terdiri dari kumpulan *word-token*. Akan dilakukan *sentence-*

*embedding* untuk mengagregasikan word-embedding untuk setiap token dan mendapatkan sebuah vektor n-dimensi yang merepresentasikan sebuah dokumen.

#### 2.1.6. Topic Modelling

*Topic Modelling* adalah sebuah algoritma yang mengidentifikasi pola-pola tersemat dalam kata-kata dalam sekumpulan dokumen (Jacobi, van Atteveldt, & Welbers, 2015). Proses ini merupakan sebuah penerapan *unsupervised machine learning* yang digunakan untuk memodelkan topik-topik yang terkandung dalam sebuah dokumen secara probabilistik. Salah satu *topic modelling* yang populer adalah *Latent Dirichlet Allocation* (LDA). Dalam penelitian yang diajukan ini, distribusi topik per dokumen yang didapatkan dari proses *topic modelling* dengan LDA akan dianggap sebagai fitur-fitur data.



Gambar 2.11 Diagram Model LDA (Lee & et.al., 2018)

Keterangan:

- $\alpha, \beta$  : distribusi Dirichlet
- $m$  : dokumen ke- $m$  dalam  $M$  buah dokumen
- $M$  : jumlah total dokumen
- $n$  : kata ke- $n$  dalam  $N$  buah kata
- $N$  : jumlah total kata
- $k$  : topik ke- $k$  dalam  $K$  buah topik
- $K$  : jumlah total topik
- $\theta_m$  : distribusi topik dalam dokumen ke- $m$
- $\phi_k$  : distribusi kata dalam topik ke- $K$
- $z_{m,n}$  : topik yang didapatkan dari  $\theta_m$  yang diberikan ke kata ke- $n$
- $w_{m,n}$  : kata ke- $n$  dalam dokumen ke- $m$

LDA adalah sebuah model statistika generatif yang dibuat untuk mengekstraksi variabel-variabel yang tersemat untuk mencari kemiripan dalam data. Metode ini dilakukan dengan mengambil secara stokastik sampel-sampel topik dari kumpulan topik per dokumen serta kata dari kumpulan kata per dokumen, lalu menggabungkannya dalam sebuah dokumen baru yang mengandung kata-kata dan topik yang mencerminkan dokumen awalnya (Blei, Ng, & Jordan, 2001). Dalam satu iterasi untuk generasi topik LDA, akan dicari distribusi kata-kata untuk tiap topik pada kumpulan topik K, dengan memperhatikan distribusi Dirichlet  $\beta$ . Akan juga dicari distribusi topik-topik untuk tiap dokumen pada kumpulan dokumen M dengan memperhatikan distribusi Dirichlet  $\alpha$ . Untuk setiap dokumen m pada kumpulan dokumen M, akan diberikan sebuah topik yang dihasilkan dari distribusi dokumen-topik sebelumnya, dan sebuah kata yang didapatkan dari distribusi topik-kata sebelumnya, untuk menginferensikan distribusi topik per dokumen.

Proses ini akan diiterasi sebanyak yang diinginkan, untuk menghasilkan konvergensi statistika. Dalam penelitian yang akan dilakukan, akan diuji variasi jumlah topik yang akan menghasilkan distribusi topik dengan dimensi yang berbeda-beda. Diharapkan ditemukan jumlah topik yang optimal, sehingga menciptakan vektor fitur yang reflektif atas setiap dokumen.

#### 2.1.7. TF-IDF

Term Frequency—Inverse Document Frequency (TF-IDF) adalah sebuah metode untuk memberikan *weight* sebuah kata dalam sebuah dokumen dalam sebuah korpus. Metode ini menggabungkan nilai *Term Frequency* (TF) yang mengevaluasi seberapa sering sebuah kata muncul dalam sebuah dokumen dibandingkan kata-kata lainnya, dan nilai *Inverse Document Frequency* (IDF) yang mengekspresikan perbandingan terbalik (inverse) jumlah dokumen dimana *term* atau kata tertentu muncul (Salton & Buckley, 1988).

TF dinyatakan dalam rumus sebagai berikut :

$$TF_{(t,d)} = \frac{f_{(t,d)}}{\sum_{(t',d)} f_{(t',d)}}$$

Rumus 2.1 *Term Frequency*

Dimana:

- t adalah *term* atau kata tertentu
- d adalah dokumen tertentu
- $TF_{(t,d)}$  adalah *term frequency* kata t dalam dokumen d.

- $f_{(t,d)}$  adalah jumlah iterasi kata t dalam dokumen d
- $\sum_{(t',d)} f_{(t',d)}$  adalah jumlah seluruh kata yang ada dalam dokumen d

IDF dinyatakan dalam rumus sebagai berikut:

$$IDF_{(t,D)} = \log \frac{N}{|\{d \in D : t \in d\}|}$$

Rumus 2.2 *Inverse Document Frequency*

Dimana :

- $IDF_{(t,D)}$  adalah *inverse document frequency* kata t dalam kumpulan dokumen D
- $N$  adalah jumlah dokumen dalam korpus
- $|\{d \in D : t \in d\}|$  adalah jumlah cacah dokumen d dalam kumpulan dokumen D, dimana kata t ada dalam dokumen d

TF-IDF dinyatakan dalam matriks hasil *dot-product* sebagai berikut:

$$TFIDF_{(t,d)} = TF_{(t,d)} \cdot IDF_{(t,D)}$$

Rumus 2.3 TF-IDF

Dalam penelitian yang akan dilakukan, *term-document matrix* yang dihasilkan dari proses TF-IDF untuk *corpus* yang akan dikumpulkan berfungsi sebagai augmentasi untuk *feature extraction*. *Term-document matrix* akan digunakan sebagai *weight* untuk proses agregasi kumpulan *word-embedding* dalam sebuah dokumen untuk menghasilkan sebuah *sentence-embedding* yang mewakili dokumen tersebut. Dalam LDA, *term-document matrix* akan digunakan sebagai input inisial yang akan mempengaruhi distribusi dan sampling untuk *topic-document matrix* dan *word-topic matrix*.

### 2.1.8. F

## 2.2. Tinjauan Studi

Berikut merupakan penelitian yang telah dilakukan sebelumnya :

### 2.2.1. An evaluation of document clustering and topic modelling in two online social networks: Twitter and Reddit (Curiskis & al., 2019)

Penelitian sebelumnya yang dilakukan oleh (Curiskis & al., 2019) melakukan perbandingan *clustering* dan *topic modelling* pada data *tweet* dari media sosial Twitter dan *thread* dari media sosial Reddit. Dataset direpresentasikan dalam empat representasi fitur, yaitu

*TF-IDF*, *word2vec* dengan *weight*, *word2vec* tanpa *weight*, dan *doc2vec*. Dilakukan teknik *clustering* pada data dengan empat metode, yaitu *K-means Clustering*, *K-medoids clustering*, *hierarchical agglomerative clustering*, dan *Non-negative matrix factorization*. Kesuksesan hasil *clustering* dalam penelitian ini diukur dengan NMI, AMI, dan ARI. Ditemukan bahwa performa paling baik didapatkan oleh representasi fitur dengan *word2vec* dan metode *k-means clustering*.

Penelitian yang diajukan memiliki beberapa perbedaan dengan penelitian yang dilakukan oleh (Curiskis & al., 2019). Pada penelitian yang diajukan, setiap rekor data berupa pernyataan peserta diskusi yang mungkin berisi lebih dari satu kalimat, sedangkan sebuah rekor data *tweet* pada *Twitter* tidak melebihi 255 karakter. Oleh karena itu, diatas *word-embedding* *word2vec*, perlu juga dilakukan *sentence embedding*. Dalam penelitian yang akan dilakukan akan dicoba dua variasi feature extraction, yaitu *word-* dan *sentence-embedding*, serta *topic modelling* dengan LDA. Fitur-fitur data akan menjadi input algoritma *density-based clustering* dengan asumsi pengetahuan dalam *domain* yang mencukupi, dan dengan harapan bahwa hasil *clustering* akan mencerminkan cluster-cluster yang ada dalam data dan memproduksi *outlier*. Selain itu, karena dataset yang tersedia dan yang akan dikumpulkan dalam penelitian ini tidak memiliki label, tidak ada asumsi *ground truth*. Oleh karena itu, pengukuran kesuksesan *clustering* akan dilakukan dengan validasi oleh pengguna.

### **2.2.2. A Comparison of LSA and LDA for the Analysis of Railroad Accident (Williams & Betak, 2019)**

Penelitian selanjutnya (Williams & Betak, 2019) membandingkan performa *topic modelling* LDA dengan LSA, dan menemukan bahwa kedua metode menemukan beberapa topik yang eksklusif satu sama lain. Penelitian ini juga menyatakan bahwa algoritma LDA direkomendasikan untuk dataset yang berukuran lebih kecil. Dalam penelitian yang akan diajukan, LDA akan digunakan untuk mengekstraksi topik dari setiap dokumen yang berupa pendapat peserta diskusi. Distribusi topik dalam tiap dokumen akan dianggap sebagai fitur-fitur data yang menjadi basis density-based clustering untuk mencari poin-poin data yang merupakan *outlier*.

### **2.2.3. Anomaly Detection: A Survey (Chandola, Banerjee, & Kumar, Anomaly Detection: A Survey, 2009)**

Survei yang dilakukan oleh Chandola, Banerjee & Kumar (Chandola, Banerjee, & Kumar, Anomaly Detection: A Survey, 2009) menyatakan beberapa metode yang dapat dilakukan untuk

melakukan *anomaly detection*, diantaranya adalah penerapan *density-based clustering* untuk mengelompokkan data dalam beberapa *cluster*. Poin-poin data yang menjadi *outlier* diidentifikasi sebagai titik-titik anomali. Dalam penelitian yang diajukan, akan dilakukan *density-based clustering* pada vektor-vektor fitur data yang telah didapatkan dari fase *feature extraction* yang telah dilakukan. *Outlier-outlier* yang terbentuk oleh *clustering* yang dilakukan akan diidentifikasi sebagai anomali dalam data dan diuji validitasnya.

### **3. ANALISA DAN DESAIN SISTEM**

Pada bab ini akan lebih dibahas lebih lanjut mengenai analisa dan desain dan rancangan sistem yang digunakan untuk mendekripsi.

#### **3.1. Analisa Permasalahan**

Diskusi terstruktur adalah sebuah proses dimana sekelompok orang berkumpul untuk berbagi informasi, gagasan, dan mengkaji sebuah topik menurut sebuah format yang telah ditentukan. Dalam sebuah diskusi terstruktur, terdapat sebuah rangkaian acara yang didesain untuk menggiring jalannya diskusi terhadap topik yang telah dipilih agar mencapai hasil yang diinginkan. Dalam Komunitas X yang merupakan komunitas multi-agama dengan mayoritas beragama Kristen, dilakukan diskusi terstruktur secara rutin mengenai topik-topik yang berkenaan dengan agama, moralitas, dan etika. Diskusi-diskusi terstruktur yang diadakan oleh Komunitas X mengharapkan para pesertanya mengikuti rangkaian dan peraturan-peraturan yang telah ditentukan oleh penyelenggara, meliputi giliran berbicara, kesesuaian dengan topik, kesopanan dalam menyampaikan kontribusi, dan larangan untuk melakukan interupsi.

Diskusi terstruktur dalam Komunitas X dipimpin oleh seorang fasilitator, dan diawasi oleh seorang moderator. Fasilitator bertugas untuk melakukan eksposisi terhadap topik yang diambil untuk sesi diskusi tersebut, dan menyampaikan pemicu diskusi menurut rincian subtopik diskusi yang telah dipersiapkan sebelumnya. Adapun moderator bertugas untuk menjembatani kontribusi-kontribusi peserta diskusi dengan topik yang sedang dibahas, serta menyaring dan menegur peserta jika kontribusi yang ditawarkan tidak sesuai dengan peraturan dan pedoman diskusi. Moderator juga bertugas untuk membuat notulen diskusi untuk mencatat kontribusi setiap peserta dalam sebuah sesi diskusi. Notulen diskusi ini akan dievaluasi pada pertemuan mingguan untuk menilai apakah kontribusi setiap peserta sesuai dengan harapan, atau malah melenceng dari pertanyaan-pertanyaan pemicu dan tidak menghasilkan input sesuai harapan.

Terdapat beberapa kendala dalam sistem pencatatan dan evaluasi yang diterapkan oleh penyelenggara diskusi dalam Komunitas X, baik pada waktu sesi diskusi maupun rapat evaluasi. Pada sebuah sesi diskusi, moderator menjalankan tugasnya sembari merangkap menjadi notulis. Seringkali, panjangnya kontribusi seorang peserta membuat moderator harus melakukan parafrase agar dapat mencatatnya tepat waktu. Akibatnya, data yang dicatat mungkin mengandung bias moderator dan tidak mencerminkan pendapat peserta yang sebenarnya, dan

akan berpengaruh pada proses evaluasi. Pada rapat evaluasi, terdapat kesulitan untuk mengidentifikasi pendapat-pendapat yang tidak sesuai dengan topik karena data notulen yang terparafrase dengan bias moderator atau kontribusi peserta tertentu yang panjang sehingga sulit untuk dicari esensinya.

Keperluan yang perlu untuk ditangani adalah deteksi kontribusi peserta yang bersifat ganjil (*anomalous*). Anomali atau keganjilan yang diamati berupa ketidaksesuaian konten pendapat peserta diskusi dengan subtopik yang ditawarkan dalam sebuah sesi diskusi. Asumsi didasarkan oleh pengamatan bahwa kontribusi-kontribusi yang ganjil umumnya berasal dari sejumlah kecil individu yang menyatakan pendapat yang melebar dari topik, dan sebagian besar anggota diskusi umumnya menawarkan kontribusi yang sesuai dengan topik dan pertanyaan-pertanyaan pemicu yang diajukan. Oleh karena itu, akan dilakukan *unsupervised learning* untuk mendeteksi kecenderungan *cluster* yang ada dalam data transkrip diskusi.

Studi yang akan dilakukan bertujuan untuk menciptakan sebuah *pipeline* data yang melakukan transkripsi kontribusi setiap peserta, menerjemahkannya ke Bahasa Inggris, melakukan *preprocessing* yang sesuai, lalu mengekstraksi fitur dengan pendekatan *word-* dan *sentence-embedding*, serta *topic modelling*. Setelah fitur-fitur berhasil diekstraksi, akan dilakukan pendekatan *density-based clustering* untuk mengamati kecenderungan *cluster* yang terbentuk dari kontribusi-kontribusi peserta dalam sebuah sesi, dan mengidentifikasi *outlier*. Studi diadakan untuk menguji apakah pendekatan yang dilakukan dengan komponen-komponen yang digabungkan dapat menghasilkan *outlier* yang mencerminkan anomali data yang diharapkan oleh penyelenggara.

### **3.2. Desain Eksperimen**

Secara garis besar, eksperimen yang akan dilakukan bertujuan untuk membuat sebuah *pipeline* inti yang menggabungkan komponen-komponen *preprocessing*, *feature extraction*, dan *density-based clustering* serta *hyperparameter*-nya untuk mengelompokkan data transkrip ke dalam *cluster-cluster* yang mencerminkan kedekatan setiap kontribusi peserta dalam sebuah sesi diskusi. Setelah itu, akan diidentifikasi *outlier-outlier* pada dataset sebagai anomali – kontribusi peserta diskusi yang ganjil dan melenceng dari bahasan topik. Eksperimen yang akan dilakukan akan berfokus pada *feature extraction –vectorization* dengan *word-* dan *sentence-embedding*, dan *topic modelling* dengan LDA, serta pencarian *hyperparameter-hyperparameter* yang tepat pada LDA dan DBSCAN. Dalam eksperimen yang dilakukan, diharapkan dapat dicari

*range* parameter yang dapat mengidentifikasi *outlier* yang sesuai dengan definisi yang diharapkan oleh penyelenggara diskusi dalam komunitas.

### 3.2.1. Corpus Data Transkrip

Data transkrip yang digunakan untuk menjalankan eksperimen tersusun dari data teks notulen dari beberapa sesi diskusi, serta transkrip audio dari beberapa diskusi yang telah diadakan. Transkrip audio akan diubah menjadi transkrip teks dengan menggunakan API Google Speech-to-text. Bagian-bagian yang berbahasa Indonesia akan diterjemahkan dengan menggunakan API penerjemahan DeepL. Adapun data transkrip yang sudah berupa teks akan disimpan dengan format yang sudah ditentukan. Berikut sebuah tabel yang berisi beberapa rekordata:

Tabel 3.1 Struktur Penyimpanan Data

No	Question	Answer	Event	Speaker
1	How did we come to exist?	Everybody should believe that they are creations, made by a creator. It is a prerequisite to believing anything else that comes after.	19-Feb	P
2	How did we come to exist?	We came from evolution. This is undeniable, and the Charles Darwin's evolution model is the most widely accepted theory to this. We evolved from unicellular bacteriae, to fish, to amphibious creatures, to mammals, to apes, then all the way to humans.	19-Feb	G
4	Do you think you need to exist?	I don't think we need to exist. As Peter said, everything would just go on as usual, regardless of your existence or not.	19-Feb	Je
39	What is the difference between a mistake and a sin?	If a man does wicked things while not being sane, he also sins.	26-Feb	G
46	What sin leads to hell?	I think there are no sins that are greater or lesser than others, all sin leads to hell.	26-Feb	P
69	We see from our side that maybe they are miserable. But how do we get the right to decide who lives and who doesn't?	Is it wrong to regret life? That is the right of the individual. The fetus is special because of the human species, even though it may have the same abilities as a chimpanzee. Life goes on, no matter how bad it is.	2-Jul	G

70	We see from our side that maybe they are miserable. But how do we get the right to decide who lives and who doesn't?	Actually, we humans are well aware that life is really difficult. Only he will know what his difficulties and limitations are. It is precisely because we know life is difficult. Science is not based on morals, while we humans have moral standards. Can everything that is presented with standards and data refer to justice? Everything has a consequence. Does that consequence make it better or worse? Not prolife and prochoice but what is from God is what is lived.	2-Jul	Yot
----	--	--	-------	-----

Legenda tabel:

- Kolom “No” menyatakan nomor rekor data, dan akan digunakan untuk mengidentifikasi dokumen pada eksperimen yang akan dilakukan.
- Kolom “Question” menyatakan pertanyaan pemicu yang menghasilkan kontribusi yang tercatat, dan akan digunakan untuk kategorisasi dokumen pada eksperimen yang akan dilakukan.
- Kolom “Answer” berisi kontribusi peserta diskusi terhadap pertanyaan (question) yang menjadi pemicu. Konten dari kolom ini dianggap sebagai dokumen yang akan digunakan sebagai bahan utama eksperimen.
- Kolom “Event” menandakan kode sesi diskusi dimana rekor data dicatat. Kolom ini akan digunakan untuk kategorisasi dokumen pada eksperimen yang akan dilakukan.
- Kolom “Speaker” digunakan untuk mengidentifikasi peserta diskusi yang menghasilkan kontribusi yang tercatat. Identitas peserta diskusi disamarkan untuk menjaga privasi. Kode yang tercatat bisa ditafsirkan oleh penyelenggara diskusi.

### 3.2.1.1.1. Pengumpulan Data

Penyelenggara diskusi di Komunitas X menyimpan notulen-notulen diskusi yang telah dilakukan dalam bentuk teks. Terhitung ada 3 sesi diskusi yang cukup lengkap yang telah tercatat dalam notulen diskusi. Kontribusi-kontribusi peserta yang tercatat dalam notulen-notulen diskusi ini akan digunakan sebagai bahan eksperimen. Selain itu, terhitung sejak 31 Maret 2023,

telah terkumpul dua transkrip audio dari diskusi yang dilakukan oleh Komunitas X. Transkrip-transkrip audio disimpan dalam *file* dengan *extension .ogg*, dan akan diubah menjadi transkrip teks terlebih dahulu sebelum diolah lebih lanjut.



Gambar 3.1 Notulen diskusi yang telah terkumpul, disimpan dalam 14 *file*.

Dataset berupa notulen akan dipilah dan dimasukkan ke dalam tabel dengan struktur yang telah dijelaskan pada Tabel 3.1, sesuai dengan pertanyaan pemicu, sesi diskusi, dan kode pembicara. Beberapa notulen yang berisi parafrase singkat dikembangkan berdasarkan gagasan-gagasan yang telah ditulis sehingga membentuk kalimat-kalimat yang koheren dan sedekat mungkin dengan gagasan parafrase yang ditulis. Adapun untuk data yang berupa transkrip audio, akan dilakukan dua kali proses transkripsi: secara manual dan dengan menggunakan API Speech-to-text Google. Hasil dari kedua transkripsi akan diterjemahkan dengan bantuan API DeepL ke dalam Bahasa Inggris dan akan dibandingkan dalam kerangka eksperimen yang sama.

Dengan menganggap kolom “Answer” dari setiap rekord data sebagai sebuah dokumen, terbentuklah sebuah *corpus* teks yang terdiri dari kumpulan kontribusi peserta diskusi terhadap pertanyaan pemicu (kolom “Question”). *Preprocessing*, *feature extraction*, dan *clustering* akan dilakukan terhadap *corpus* yang terbentuk.

### 3.2.2. Text Preprocessing

Text Preprocessing dilakukan untuk mempersiapkan data teks yang tidak terstruktur untuk diolah (Katrlyn, 2020). Dalam penelitian ini, akan digunakan teknik-teknik *preprocessing* yang mempersiapkan data teks untuk *feature extraction* dengan pendekatan *word-embedding*, *sentence-embedding*, *TF-IDF*, dan *topic modelling*. Oleh karena itu, preprocessing yang dilakukan

berfokus pada upaya mengurangi *noise* dan redundansi pada data untuk menghasilkan token-token yang berkualitas.

#### **3.2.2.1. Case-Folding**

Pada data teks akan dilakukan *case-folding*, yaitu pengubahan semua karakter huruf menjadi huruf kecil. Proses ini dilakukan untuk menghilangkan redundansi *string* yang menyebabkan *noise* pada data.

#### **3.2.2.2. Stop Word Removal**

Pada data teks akan juga dihapus kata-kata yang frekuensi kemunculannya tinggi, namun tidak bermakna signifikan pada dokumen. Proses ini dilakukan untuk mengurangi redundansi yang disebabkan oleh kata-kata yang berfrekuensi tinggi. Rincian kata-kata yang termasuk *stop words* akan didapatkan dari library NLP yang digunakan.

#### **3.2.2.3. Stemming dan Lemmatizing**

Stemming adalah proses pengubahan sebuah kata menjadi sebuah elemen dasar yang disebut *stem* atau batang (Kannan, Gurusamy, Vijayarani, Ilamathi, & Nithya, 2014). Proses ini dilakukan untuk generalisasi kata kerja, supaya variasi-variasi kata kerja dalam Bahasa Inggris (contohnya: “work” dan “working”) dapat dikenali sebagai token yang sama. Dalam eksperimen yang akan dilakukan, akan diterapkan algoritma Porter Stemming yang disediakan oleh *library* NLP yang digunakan.

Lemmatizing adalah proses pengelompokan dan pengubahan sebuah kata berdasarkan *lemma*(kata dasar)-nya. Proses lemmatizing akan mengembalikan sebuah token kata yang valid, namun membutuhkan komponen POS Tagging untuk menentukan unsur kata (contohnya: “working” dapat dikategorikan sebagai *verb* atau *adjective*, tergantung dengan konteks). Proses ini akan dilakukan dengan bantuan library NLP yang digunakan. Akan dilakukan perbandingan penggunaan *lemmatizing* dan *stemming* pada *pipeline* data yang akan dibuat.

#### **3.2.2.4. Tokenization**

Tokenization adalah proses pemecahan sebuah dokumen menjadi kumpulan token (Kannan, Gurusamy, Vijayarani, Ilamathi, & Nithya, 2014). Dalam studi ini, karena studi berada di tingkat *lexical*, setiap dokumen akan dipecah menjadi token-token kata (*word tokens*). Dengan mengubah sebuah dokumen menjadi sekumpulan token, beberapa proses, seperti stemming dan lemmatizing serta *feature extraction* akan lebih mudah dilakukan. Proses *tokenization* akan dilakukan dengan bantuan library NLP yang digunakan.

### **3.2.3. Feature Extraction**

Feature extraction adalah proses untuk mengubah setiap token yang sudah dibersihkan dengan *preprocessing* menjadi sekumpulan angka yang merepresentasikan token tersebut. Hal ini dilakukan agar token bisa menjadi input pada model *density-based clustering* yang akan digunakan. Dalam studi ini, akan dilakukan dua pendekatan *feature extraction*, yaitu *vectorization* dengan *word-* dan *sentence-embedding*, serta *topic modelling* dengan LDA. Kedua pendekatan akan diuji secara paralel untuk menentukan kinerja yang lebih baik untuk studi yang dilakukan.

#### **3.2.3.1. Pendekatan Word- dan Sentence-Embedding**

Pendekatan *word-embedding* digunakan untuk mengekspresikan makna semantik dan sintaktik token-token kata dalam sebuah corpus yang tidak dilabeli (Lai, Liu, He, & Zhao, 2016). Dengan pendekatan ini, setiap word token yang dihasilkan oleh proses *tokenization* akan memiliki representasi dalam bentuk sebuah vektor n-dimensi. Proses *word-embedding* akan dilakukan dengan metode *Word2Vec* yang sudah tersedia dalam *library* yang digunakan. Akan dilakukan perbandingan secara paralel pengujian beberapa variabel, diantaranya jumlah dimensi, serta model yang dibangun sendiri dan *pre-trained model*.

Adapun pendekatan *sentence-embedding* digunakan untuk menangkap makna sebuah kalimat atau dokumen yang terdiri dari sekumpulan *word token* yang sudah *di-embed* menjadi vektor. Vektor-vektor yang dihasilkan lewat *word-embedding* setiap token yang ada dalam satu dokumen akan diagregasikan menjadi satu *sentence vector* yang berdimensi sama. Akan diuji beberapa metode agregasi untuk menentukan kinerja yang paling baik. Metode-metode agregasi yang akan diuji adalah *averaging*, *sum*, *weighted-averaging*, dan *weighted-sum*; *weight* atau berat yang akan digunakan diambil dari nilai TF-IDF setiap token pada dokumen yang bersangkutan. Luaran dari proses ini adalah sebuah vektor berdimensi-n untuk setiap dokumen, dimana n adalah jumlah dimensi yang ditentukan oleh peneliti. Vektor dari setiap dokumen akan menjadi input untuk algoritma *density-based clustering* yang akan digunakan.

#### **3.2.3.2. Pendekatan Topic Modelling**

Metode *feature extraction* menggunakan *topic modelling* akan diuji sebagai alternatif dari pendekatan *embedding*. Dalam pendekatan ini, akan dicari pola-pola tersemat dalam token-token kata yang ada dalam sebuah dokumen (Jacobi, van Atteveldt, & Welbers, 2015), dengan asumsi jumlah topik tertentu. Pendekatan ini adalah pendekatan stokastik, sehingga dapat menghasilkan keputusan yang berbeda-beda dalam setiap operasi dengan parameter dan input yang sama. Akan diuji beberapa variasi parameter yang mempengaruhi algoritma LDA,

diantaranya jumlah topik yang diasumsikan –akan diuji beberapa jumlah topik sesuai dengan studi literatur tentang LDA dan LSA (Williams & Betak, 2019). Akan diuji juga performa algoritma dengan input tambahan berupa TF-IDF sebagai *term-document matrix*. Luaran dari proses ini adalah sebuah vektor berdimensi-n, dimana n merupakan jumlah topik yang diinferensikan. Nilai dari setiap dimensi dari vektor yang terbuat melambangkan proporsi topik yang diinferensikan yang terkandung dalam dokumen yang bersangkutan. Kumpulan vektor yang didapatkan dari setiap dokumen akan menjadi input untuk algoritma *density-based clustering* yang akan digunakan.

### **3.2.3.3. TF-IDF**

TF-IDF (Term Frequency-Inverse Document Frequency) adalah metode statistical untuk mendapatkan *term-document matrix* yang mewakili relevansi setiap kata pada setiap dokumen pada *corpus*. *Matrix* yang didapatkan dari proses TF-IDF untuk *corpus* yang menjadi bahan penelitian akan digunakan untuk augmentasi dan *weighting* metode-metode *feature extraction* yang digunakan, dengan harapan bahwa relevansi setiap token dicerminkan dari matriks yang didapatkan.

### **3.2.4. Density-based Clustering**

Setelah *feature extraction* dijalankan untuk setiap dokumen dalam *corpus*, akan dilakukan proses *density-based clustering* dengan DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*). Penggunaan metode ini bertujuan untuk mendapatkan *cluster-cluster* berdasarkan kedekatan vektor-vektor setiap dokumen yang telah didapatkan dengan metode *feature extraction* yang telah dipilih. *Feature extraction* yang baik akan menghasilkan data yang berkualitas dan merefleksikan dokumen sesungguhnya. Diharapkan akan ditemukan poin-poin data yang tidak tergabung dalam *cluster* manapun untuk diidentifikasi sebagai *outlier*.

### **3.2.5. DBSCAN dan Parameter-Parameternya**

DBSCAN adalah *algoritma density-based clustering* yang membuat *cluster-cluster* berdasarkan kedekatannya (Ester, Kriegel, Sander, & Xu, 1996). Pendekatan ini dipilih karena bisa menghasilkan *outlier* dalam bentuk poin-poin data yang terletak jauh dari *cluster-cluster* yang terbentuk. Dalam algoritma ini, terdapat dua parameter yang dapat diubah-ubah, yaitu *MinPts* –jumlah poin data minimum yang ada di sekeliling sebuah poin data, dan  $\epsilon$  –radius dimana *MinPts* diperhitungkan. DBSCAN akan diterapkan dan divisualisasikan dengan *library* yang telah disediakan.

### **3.2.6. Fungsi Jarak yang Akan Digunakan**

Fungsi jarak yang digunakan dapat mempengaruhi pembentukan cluster karena perbedaan karakteristik masing-masing fungsi jarak yang dikenali. Dalam penelitian ini, akan dibandingkan secara paralel beberapa fungsi jarak, diantaranya Euclidean Distance, Cosine Distance, dan Mahalanobis Distance.

### **3.2.7. Anomaly Detection and Validation**

Dalam eksperimen yang dilakukan, diasumsikan hipotesis utama bahwa *outlier-outlier* yang ditemukan adalah *lexical anomaly* pada data, dan mencerminkan kontribusi peserta diskusi yang ganjil atau tidak sesuai topik. Hasil dari eksperimen-eksperimen ini akan divalidasi oleh tim penyelenggara diskusi Komunitas X, yang berperan sebagai pengguna dan memiliki kapasitas dalam *domain* yang bersangkutan. Akan ditentukan apakah *outlier* yang ditemukan oleh program yang dibuat mencerminkan kontribusi yang ganjil atau tidak.

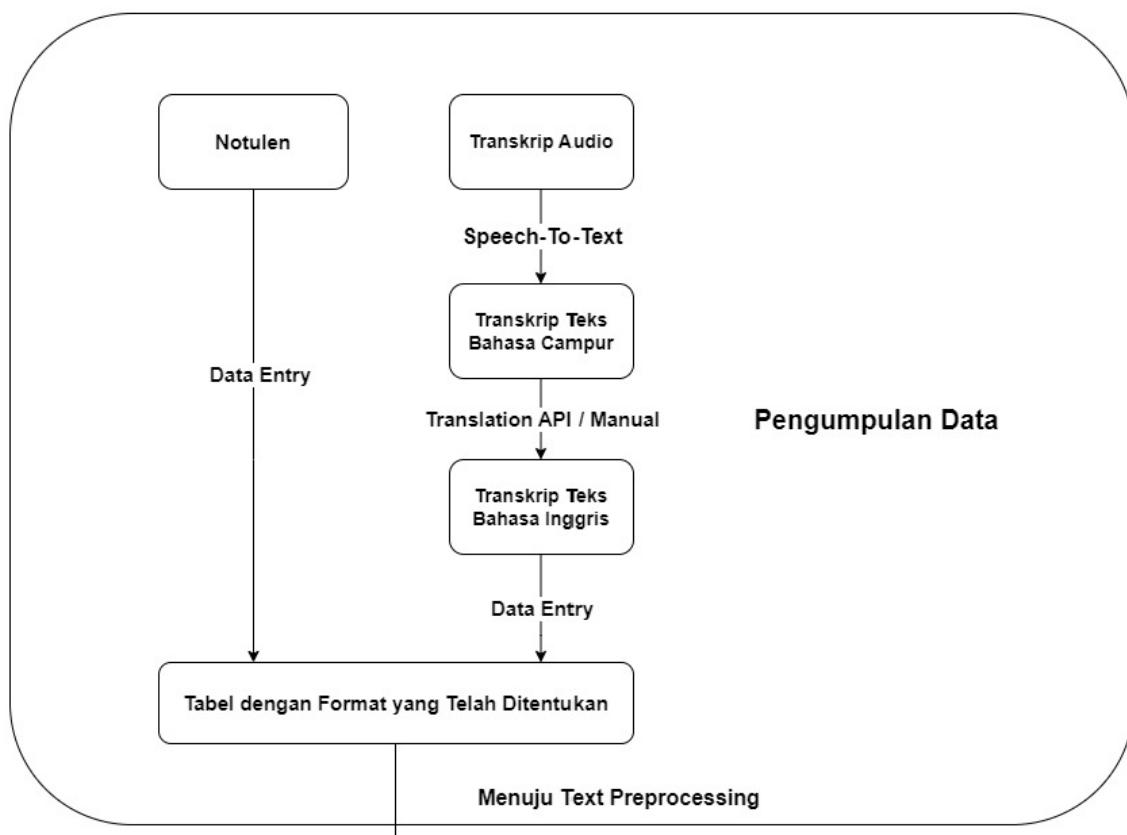
## **3.3. Desain Eksperimen dan Program**

Bagian ini akan menjelaskan tentang desain eksperimen untuk mendeteksi anomali, serta desain program implementasi yang akan dibuat.

### **3.3.1. Alur Eksperimen**

#### **3.3.1.1. Pengumpulan Data**

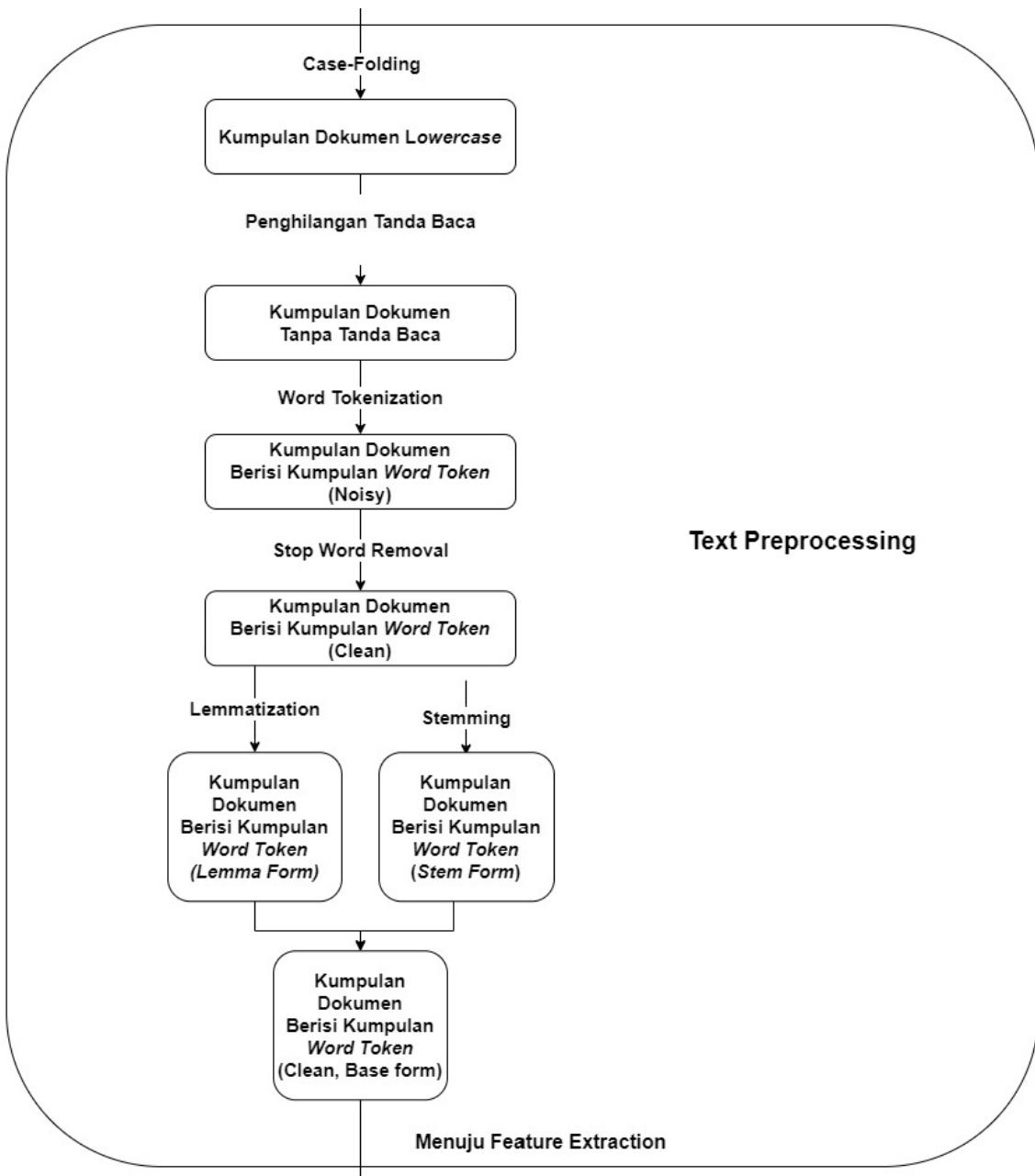
Pada tahap ini *corpus* dibangun dari notulen dan transkrip audio. Data dari Notulen akan langsung dimasukkan ke dalam tabel data dengan format yang telah ditentukan. Adapun transkrip yang berupa audio akan diubah dahulu menjadi transkrip teks melalui API Speech-to-text Google. Setelah itu, untuk input yang tidak berbahasa Inggris, akan diterjemahkan ke dalam Bahasa Inggris dengan API DeepL dan intervensi manual. Kedua proses ini akan dilakukan secara terpisah dan dijalankan secara paralel, untuk menguji efektivitas komponen API speech-to-text dan translation. Pada akhir proses ini, data yang tersimpan dalam tabel akan diakses dengan program dan dikenakan *text preprocessing*.



Gambar 3.2 Diagram Alur Eksperimen – Pengumpulan Data

### 3.3.1.2. Text Preprocessing

*Text preprocessing* dimulai dari *case-folding* untuk meratakan isi *corpus*, menjadikannya *lower case*. Setelah itu, akan dihilangkan tanda baca dan elemen-elemen nonesensial lainnya seperti angka yang tidak relevan. Dokumen-dokumen yang hanya berisi karakter-karakter alfabetik akan dijadikan kumpulan *word tokens* dengan fungsi *word tokenize*, menghasilkan dokumen-dokumen yang berisi kumpulan token. Untuk setiap dokumen, akan dijalankan proses *stop words removal*, untuk menghapus token-token yang frekuensinya tinggi namun tidak menawarkan banyak makna, untuk mengurangi *noise* dalam data. Selanjutnya, akan dijalankan secara paralel *stemming* dan *lemmatization*, untuk dibandingkan hasilnya. Kedua proses itu akan mengubah setiap dokumen menjadi kumpulan token yang berisi *stem* atau *lemma* dasar dari setiap kata, dan siap untuk memasuki tahap *feature extraction*.



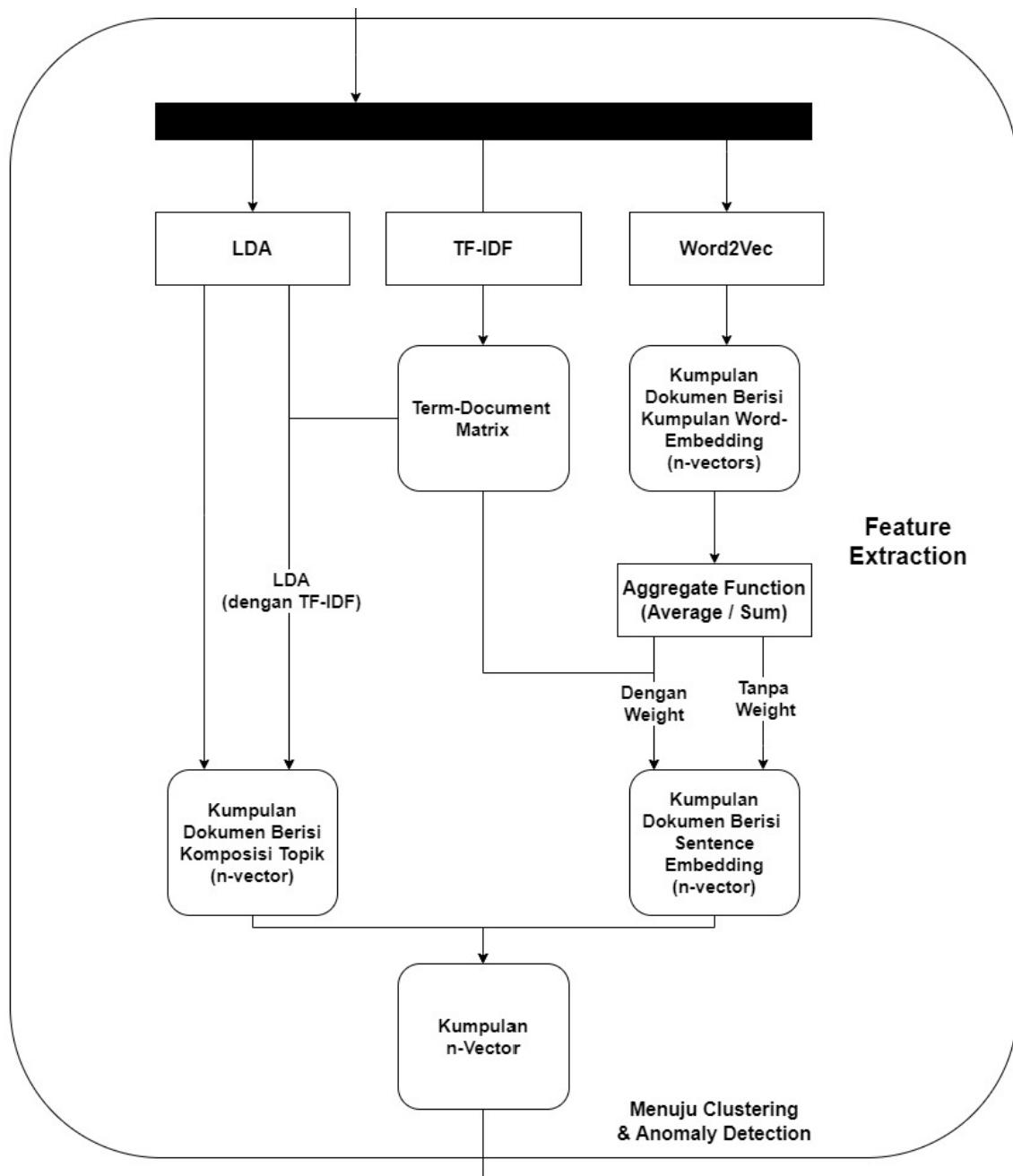
Gambar 3.3 Diagram Alur Eksperimen – Text Preprocessing

### 3.3.1.3. Feature Extraction

Untuk proses *feature extraction*, ada dua pendekatan utama yang akan diuji, yaitu pendekatan *topic modelling* lewat LDA, dan pendekatan *embedding* lewat *word-embedding* dan *sentence-embedding*. Akan juga dijalankan TF-IDF untuk mendapatkan *term-document matrix*, yang merupakan sebuah nilai numerik untuk setiap *word token* di setiap dokumen, sebagai variabel pengujian pada dua metode *feature extraction* yang dipakai. *Topic modelling* dan *word/sentence-embedding* akan dijalankan secara paralel untuk dibandingkan performanya.

Dalam pendekatan *embedding*, akan dilakukan *word-embedding* pada setiap token kata yang terkandung dalam setiap dokumen. Akan digunakan variasi model Word2Vec, dengan menggunakan beberapa *pre-trained model* yang sudah tersedia serta model yang dibangun sendiri. Akan dilakukan juga variasi jumlah dimensi vektor yang dihasilkan. Setelah setiap word-token berhasil di-embed, akan dilakukan agregasi pada setiap word-token yang dimiliki sebuah dokumen, untuk mendapatkan sebuah vektor yang merepresentasikan dokumen tersebut (*sentence-embedding*). Akan dicoba secara paralel dan dibandingkan metode *aggregation* dengan *averaging* dan *sum*, serta *weighted averaging* dan *weighted sum* dengan menggunakan *term-document matrix* sebagai *weight*. Proses agregasi ini akan menghasilkan sebuah *n-dimensional numerical vector* untuk setiap dokumen.

Dalam pendekatan *topic modelling* dengan LDA, akan dilakukan variasi eksperimen, yaitu proses LDA sebagaimana adanya, dibandingkan dengan proses LDA dengan input TF-IDF sebagai *term-document matrix*, dengan harapan weight yang diberikan oleh matriks tersebut mempengaruhi probabilitas stokastik. Akan juga dilakukan variasi jumlah topik yang diinferensikan, untuk mencari jumlah topik optimal yang bisa menghasilkan distribusi topik yang baik. Proses LDA akan menghasilkan sebuah *n-dimensional numerical vector* untuk setiap dokumen, dengan n merupakan jumlah topik yang diinferensikan, dan setiap dimensi vektor melambangkan proporsi topik tersebut pada dokumen yang bersangkutan.

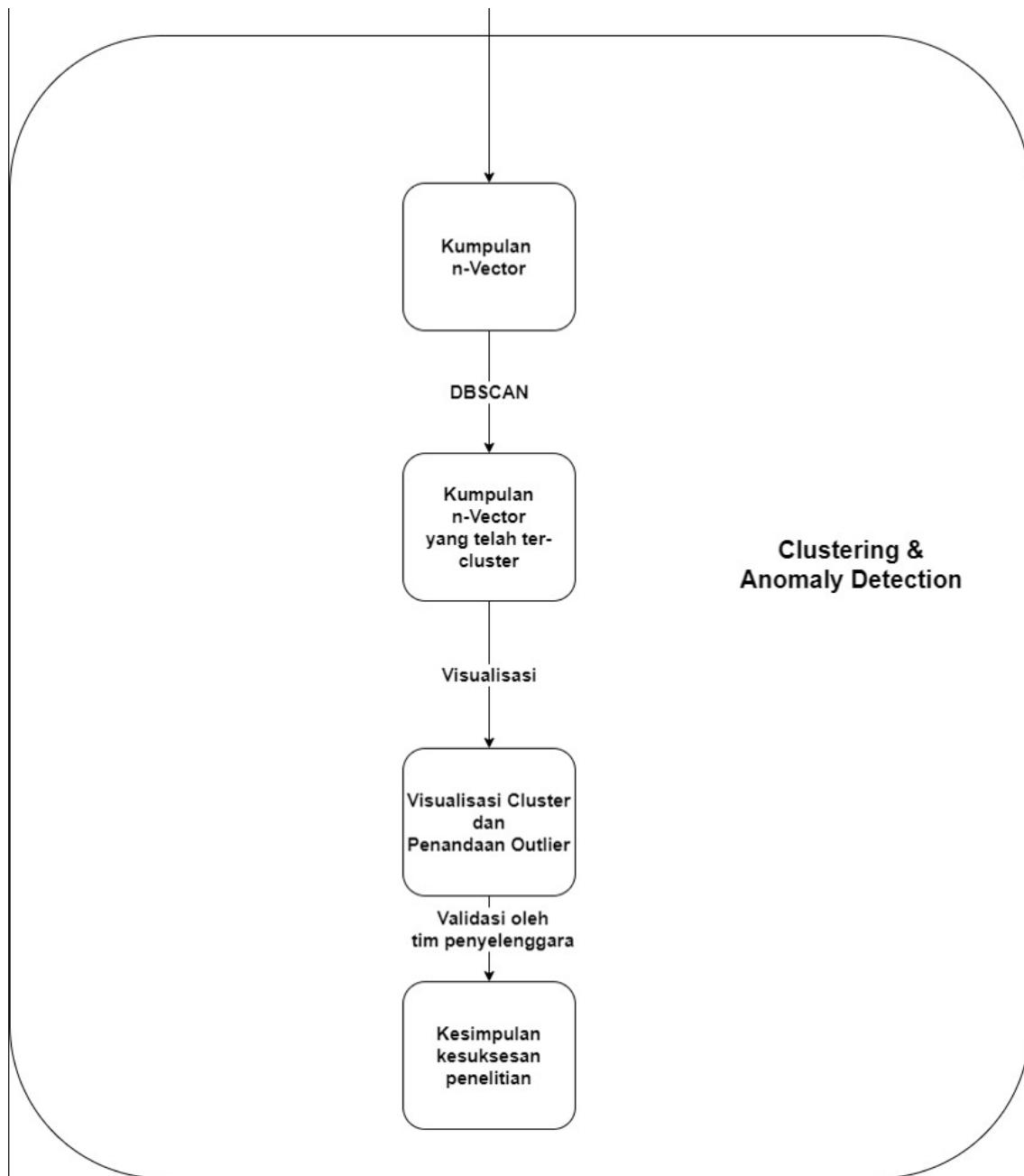


Gambar 3.4 Diagram Alur Eksperimen – *Feature Extraction*

### 3.3.1.4. Clustering & Anomaly Detection

Vektor-vektor n-dimensi yang terbentuk akan di-fit dalam model DBSCAN dan dikelompokkan dalam *cluster-cluster* menurut algoritmanya. Akan dilakukan *tuning* terhadap parameter berupa MinPts dan  $\epsilon$  untuk mendapatkan *cluster-cluster* yang berkualitas, dan mendapatkan *outlier* yang benar. Setiap iterasi dari eksperimen akan direpresentasikan secara visual dalam *point graph*, dan akan divalidasi oleh penyelenggara diskusi Komunitas X sebagai

pengguna, untuk menentukan keakuratan *cluster* yang terbentuk dan validasi *outlier-outlier* yang teridentifikasi sebagai anomali yang dicari dalam data.



Gambar 3.5 Diagram Alur Eksperimen – *Clustering & Anomaly Detection*

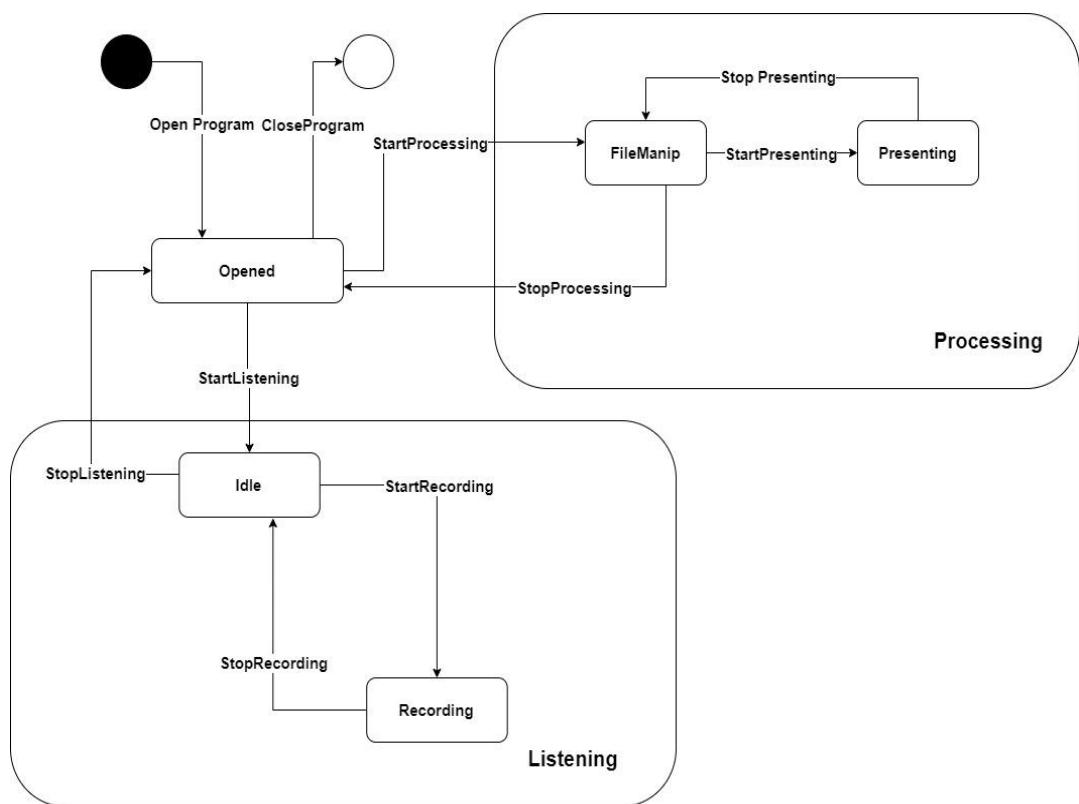
### 3.3.2. Alur Program

#### 3.3.2.1. Alur Keseluruhan Program

Program akan memiliki tiga *state* inti, yaitu Opened, Listening, dan Processing. Pada saat program dijalankan, program akan memasuki *state* Opened. Akan disajikan sebuah menu bagi pengguna untuk memilih aksi selanjutnya, yaitu StartProcessing, StartListening, atau

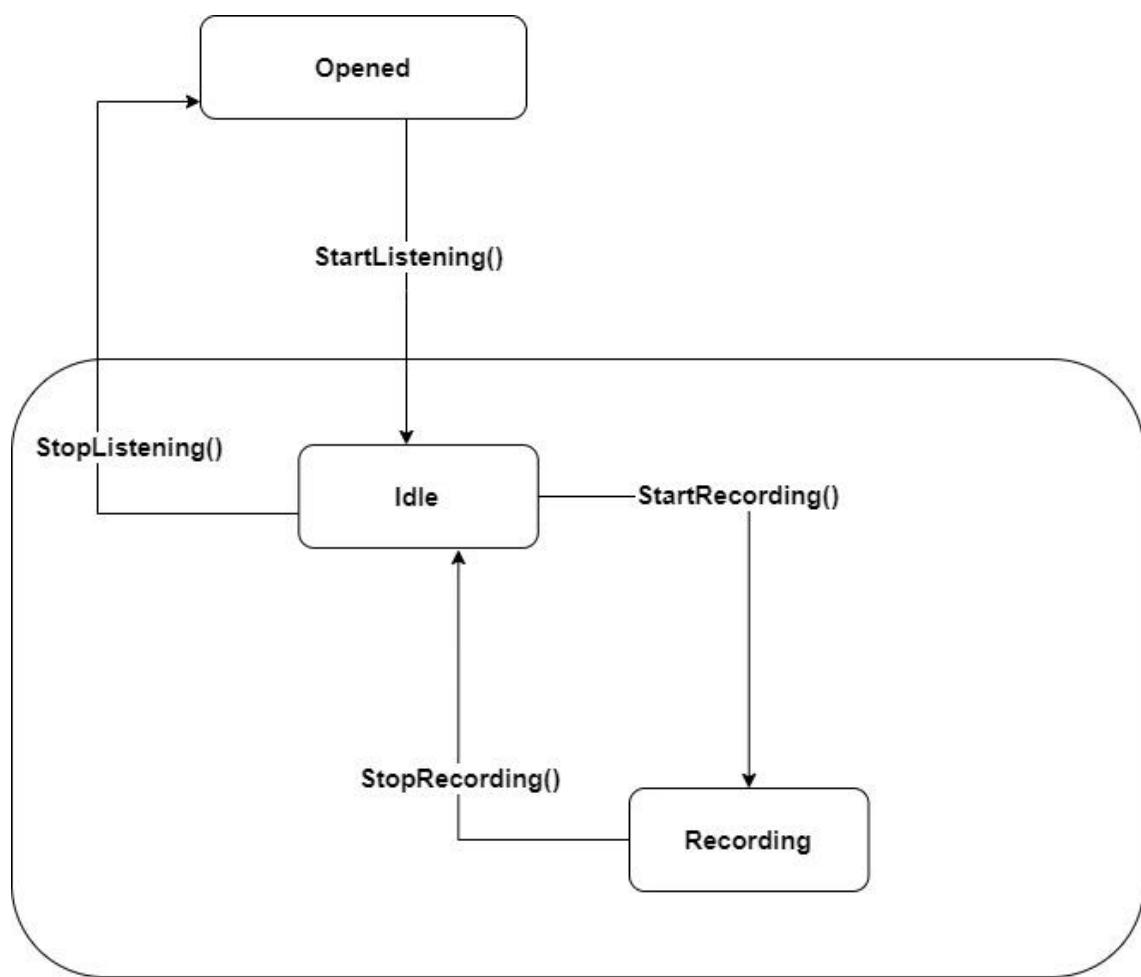
CloseProgram. Ketika pengguna memilih aksi StartListening, program akan memasuki *state* Listening, dimana program akan mendengarkan dan merekam suara yang ada di komputer pengguna, untuk menangkap jalannya diskusi yang sedang berlangsung. Dalam *state* ini, pengguna dapat kembali ke *state* Opened dengan aksi StopListening.

Ketika pengguna memilih aksi StartProcessing, pengguna akan memasuki state Processing. Dalam state ini, pengguna bisa membuka file database untuk mengakses data rekaman terstruktur yang telah dibuat, lalu menjalankan anomaly detection dan mendapatkan visualisasi data. Pengguna dapat kembali ke state Opened dengan aksi StopProcessing. Ketika di state Opened, pengguna dapat menjalankan aksi CloseProgram untuk menghentikan jalan program.

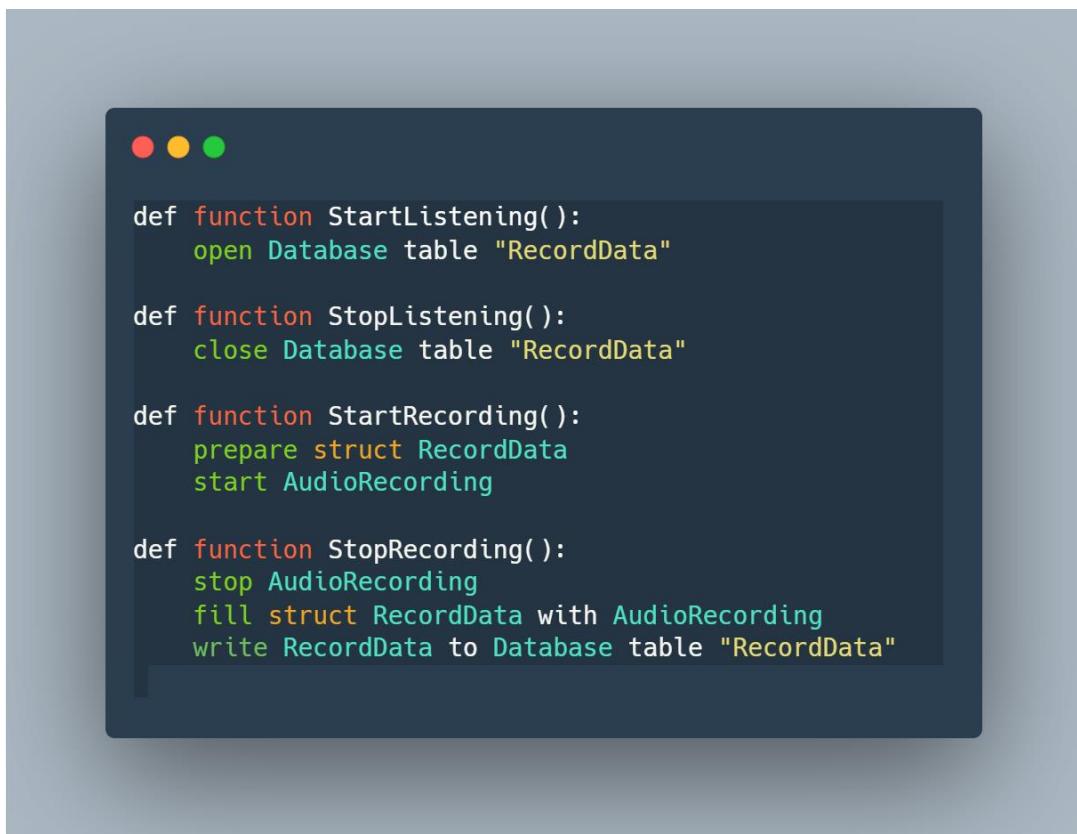


Gambar 3.6 State Diagram Program yang Akan Dibuat

### 3.3.2.2. Alur Pengumpulan Klip Suara



Gambar 3.7 State Diagram Alur Pengumpulan Klip Suara



```
def function StartListening():
    open Database table "RecordData"

def function StopListening():
    close Database table "RecordData"

def function StartRecording():
    prepare struct RecordData
    start AudioRecording

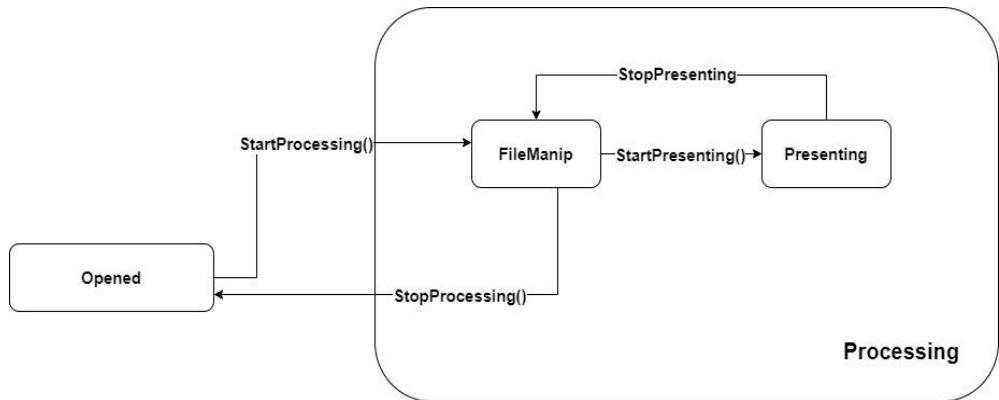
def function StopRecording():
    stop AudioRecording
    fill struct RecordData with AudioRecording
    write RecordData to Database table "RecordData"
```

Gambar 3.8 Pseudocode Alur Pengumpulan Klip Suara

Berikut penjelasan setiap perintah yang ada dalam bagan yang ditunjukkan:

- `StartListening()`: Ketika pengguna memilih perintah `StartListening()`, program akan membuka koneksi ke tabel dalam *database* untuk menyimpan data rekaman suara, lalu memasuki *state* Idle. Dalam *state* Idle, pengguna dapat memulai proses rekaman dengan perintah `StartRecording()`.
- `StartRecording()`: Ketika perintah ini diberikan, akan disiapkan sebuah *struct* untuk menampung data yang akan disimpan dalam *database*. Data mencakup ID rekaman, tanggal rekaman, dan *file* suara yang akan disimpan. Proses rekaman suara akan dimulai sampai perintah `StopRecording()` diberikan.
- `StopRecording()`: Ketika perintah ini diberikan, proses rekaman akan dihentikan. *File* rekaman yang terbentuk akan disimpan dalam *database* berdasarkan *struct* yang telah didefinisikan. *State* program akan kembali ke Idle.
- `StopListening()`: Ketika perintah ini diberikan, program akan menutup koneksi ke *database* dan kembali ke *state* Opened.

### 3.3.2.3. Alur Proses dan Penyajian Data



Gambar 3.9 State Diagram Alur Proses dan Penyajian Data

```
● ● ●

def function StartProcessing():
    open Database table "RecordData"
    open Database table "TranscriptData"

def function StartPresenting(recordID):
    prepare list RecordTable
    prepare list TranscriptTable
    read Database table "RecordData" with recordID to RecordTable
    for recordData in RecordTable:
        prepare struct TranscriptData
        transcribe recordData with SpeechToTextAPI
        translate recordData with TranslateAPI
        fill struct TranscriptData with recordData
        append TranscriptData to TranscriptTable
    process TranscriptTable with DataPipeline

def function StopPresenting():
    open Database table "SaveData"
    prepare struct SaveData
    fill struct SaveData with DataPipeline_Result
    write SaveData to Database table "SaveData"
    close Database table "SaveData"

def function StopProcessing():
    close Database table "RecordData"
    close Database table "TranscriptData"
```

Gambar 3.10 Pseudocode Alur Proses dan Penyajian Data

Berikut penjelasan setiap perintah yang ada dalam bagan yang ditunjukkan:

- StartProcessing(): Pada saat perintah ini diberikan, program akan memasuki *state* FileManip. Dalam *state* ini, dibuka koneksi ke tabel-tabel dalam *database* yang menyimpan transkrip audio dan transkrip teks. Pengguna dapat memilih untuk mengakses rekord transkrip audio yang diinginkan, untuk diproses menjadi transkrip teks dan dijadikan input proses *anomaly detection*.
- StartPresenting(recordID): Perintah ini dijalankan dengan parameter ID dari salah satu rekord data rekaman audio. Data akan diambil dari tabel yang bersangkutan dan diproses dengan komponen-komponen API *speech-to-text* dan *translator* untuk menghasilkan transkrip berupa teks berbahasa Inggris. Kumpulan transkrip teks akan dijadikan input untuk *pipeline* data yang melibatkan *preprocessing*, *tokenization*, *feature extraction*, *clustering*, dan *anomaly detection*. Akhirnya, hasil dari *anomaly detection* akan dipresentasikan. Ketika perintah ini dijalankan, program akan memasuki *state* Presenting, dimana pengguna dapat melihat visualisasi analisa data.
- StopPresenting(): Ketika perintah ini dijalankan, hasil *anomaly detection* akan disimpan dalam sebuah format tertentu dalam sebuah tabel pada *database*. Program akan kembali ke *state* FileManip dimana pengguna bisa memilih rekord-rekord data audio yang akan diproses.
- StopProcessing(): Ketika perintah ini dijalankan, koneksi *database* ke tabel yang bersangkutan akan ditutup dan *state* akan kembali ke Opened.

## 4. IMPLEMENTASI SISTEM

Bab ini akan menjelaskan implementasi desain sistem yang telah dirumuskan dalam Bab sebelumnya, serta mendokumentasikan code dan tools yang dipakai untuk merancang program dan eksperimen.

### 4.1. Implementasi Umum

#### 4.1.1. Tools dan IDE

Implementasi eksperimen dan program ditulis dengan Bahasa Python, berikut library-library yang ada dalam *repository PyPI (Python Package Index)*.

#### 4.1.2. Mapping Konsep dan Implementasi

Konsep di Bab 3	Implementasi di Bab 4	Keterangan
3.3.1.1	4.3.2.3; 4.3.2.4; 4.3.3	Tahap pengumpulan data dalam konsep direalisasikan dengan penyimpanan data rekaman audio yang dijabarkan dalam 4.3.2.3, transkripsi data rekaman audio menjadi teks dalam 4.3.2.4, dan penyimpanan data dalam skema <i>database</i> yang diatur oleh <i>object</i> yang dijabarkan dalam 4.3.3.
3.3.1.2	4.2.3.	Tahap <i>text preprocessing</i> diimplementasikan dalam fungsi <i>PreprocessDoc()</i> sebagaimana dinyatakan dalam Segmen Kode 4.2.
3.3.1.3	4.2.4; 4.2.5; 4.2.7; 4.2.8	Tahap <i>feature extraction</i> dibahas dalam subbab-subbab yang telah ditentukan. <i>Word-embedding</i> dengan model-model dibahas dalam 4.2.4, <i>Sentence-embedding</i> dan fungsi agregasi dibahas dalam 4.2.5, penggunaan TF-IDF sebagai <i>weighting</i> untuk pendekatan <i>embedding</i> dan <i>topic modeling</i> dibahas dalam 4.2.7,

		dan <i>topic modeling</i> dengan LDA dibahas dalam 4.2.8.
-	4.2.6	Pendekatan <i>feature extraction</i> baru yang dinamakan <i>tiny word-embedding</i> dibahas dalam 4.2.6. Pendekatan ini disarankan pada saat implementasi eksperimen.
3.3.1.4	4.2.9	Penerapan algoritma DBSCAN dibahas di 4.2.9.
-	4.2.10; 4.2.11	Penerapan algoritma <i>anomaly detection</i> yang lain (LOF dan IF) dibahas di 4.2.10 dan 4.2.11. Kedua algoritma ini disarankan pada saat implementasi eksperimen.
3.3.2.1	4.3.2	Alur keseluruhan program diimplementasikan dengan GUI Windows yang detailnya dibahas dalam subbab 4.3.2.
3.3.2.2	4.3.2.3; 4.3.4	Pengumpulan klip suara dan penyimpanannya dalam <i>database</i> diatur dalam ListenWindow pada 4.3.2.3, dan dilakukan dengan komponen RecorderThread dalam 4.3.4.
3.3.2.3	4.3.2.4; 4.3.2.5; 4.3.4	Penyajian dan pemrosesan data dilakukan dalam ProcessWindow yang dibahas dalam 4.3.2.4, dan PresentWindow yang dibahas dalam 4.3.2.5. Proses transkripsi dan penerjemahan diatur oleh <i>class</i> TurnToTextinator yang dibahas dalam 4.3.4.

## 4.2. Implementasi Eksperimen

#### **4.2.1. Tools dan IDE**

Eksperimen dilaksanakan dengan bahasa pemrograman Python. IDE yang digunakan adalah Jupyter Notebook, yang memungkinkan pembagian program dalam segmen-segmen yang bisa dijalankan secara individu. *Pipeline* eksperimen yang lengkap dijadikan sebuah *class* yang diimplementasikan dalam program.

#### **4.2.2. Importing Library**

Eksperimen yang dilaksanakan memanfaatkan beberapa *library* untuk mempermudah pelaksanaan komponen-komponen yang ada didalamnya.

Segmen Kode 4.1Imports

```
# imports go here
import numpy as np
import db
import pandas as pd
import inflect
import string
import nltk
import gensim
import matplotlib.pyplot as plt
import gensim.downloader as api
from gensim import corpora, models
from nltk.test.gensim_fixt import setup_module
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.corpus import wordnet
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import DBSCAN
from sklearn.ensemble import IsolationForest
from sklearn.neighbors import LocalOutlierFactor
```

Adapun fungsi dari library-library yang digunakan adalah sebagai berikut:

Tabel 4.1 Fungsi-fungsi *library* yang di-*import*

Nama Library	Fungsi
numpy	Memudahkan operasi-operasi matematika pada sekelompok data.
pandas	Menyediakan struktur data yang mempermudah pengambilan, pengelompokan, dan penyaringan data.
inflect	Mengubah angka menjadi bentuk terbilangnya dalam bahasa Inggris. Contoh : string yang menyatakan angka '92' akan diubah menjadi ' <i>ninety-two</i> '.
string	Memudahkan operasi yang melibatkan tipe data string.
nltk	Natural Language Toolkit, memudahkan tugas-tugas yang berhubungan dengan <i>preprocessing</i> , seperti <i>word tokenization</i> dan <i>POS tagging</i>
gensim	Memudahkan tugas-tugas yang berhubungan dengan <i>Topic Modelling</i> dan <i>preprocessing</i> . Diantaranya, gensim digunakan untuk menyediakan <i>pre-trained model</i> untuk <i>word embedding</i> yang digunakan, serta menyediakan LDA.

#### 4.2.3. Text Preprocessing

*Pipeline text preprocessing* dibuat dalam fungsi `PreprocessDocument()`. Ada beberapa parameter *boolean* yang disertakan dan diberikan nilai *default* sesuai dengan penemuan dalam eksperimen.

Segmen Kode 4.2 Fungsi untuk melakukan *text preprocessing*

```
'''  
inputs:  
- doc : str --> a string representing a sentence/document.
```

```

- isLemma : bool --> use lemmatizer or not? Defaults to not.
- isStopWords : bool --> use stopwords or not? Defaults to
not.
- isInflect : bool --> use inflections (you're --> you are)
or not? Defaults to not.
- isNumberFiltered : bool --> delete numbers in the string?
Defaults to yes.

"""
"""

output : list<str> --> a list of word tokens (list<string>)
"""

def PreprocessDocument(doc: str, isLemma: bool = False,
isStopWords: bool = False, isInflect: bool = False,
isNumberFiltered: bool = True):
    inflector = inflect.engine()
    stopwordSet = set(stopwords.words("english"))
    lemmatizer = WordNetLemmatizer()
    punctuations = string.punctuation
    # if numbers are filtered, add that to the punctuation
    string
    if isNumberFiltered:
        punctuations += "1234567890"

    # case fold
    doc = doc.lower()

    # remove puncs
    doc = "".join([char for char in doc if char not in
punctuations])

    # tokenize it.
    token_list = nltk.word_tokenize(doc)

    for i in range(len(token_list)):

```

```

# if inflect
if isInflect:
    if token_list[i].isdigit():
        token_list[i] = inflector.number_to_words(token_list[i])

# if lemma
if isLemma:
    tagged_word = nltk.pos_tag([token_list[i]])
    wordnet_pos = getWordnetPos(tagged_word[0][1])
    token_list[i] = lemmatizer.lemmatize(
        tagged_word[0][0], pos=wordnet_pos)

# if stopword
if isStopWords:
    if token_list[i] in stopwordSet or
token_list[i].isdigit():
        token_list[i] = "#" # mark as #

# remove the marked strings
token_list = [token for token in token_list if token != "#"]

if token_list:
    return token_list
return []

```

Segmen Kode 4.3 Fungsi getWordnetPos

```

'''
inputs:
- tag : str --> the tag obtained from POS tagging.
'''
outputs:
- str --> Wordnet POS tag.

```

```

"""
def getWordnetPos(self, tag):
    """Map POS tag to WordNet POS tag"""
    if tag.startswith('J'):
        return wordnet.ADJ
    elif tag.startswith('V'):
        return wordnet.VERB
    elif tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN # solves as noun by default.

```

Fungsi ini mengambil input berupa sebuah string yang berisi teks, dan melakukan *preprocessing* atas string tersebut. Pertama, teks akan dijadikan *lowercase* untuk menghindari kerancuan. Setelah itu, akan dihapus tanda-tanda baca yang ada dalam string dengan fungsi `string.lower()`. String yang telah dibersihkan dari tanda baca akan dibagi-bagi menjadi kumpulan *word-token* dengan fungsi `nltk.word_tokenize()`. Untuk setiap *word-token*, akan dilakukan infleksi dengan komponen `inflector_engine()` untuk *word-token* yang berupa angka, untuk mengubahnya menjadi bentuk terucapnya (contoh : “34” akan menjadi “*thirty-four*”). Setelah itu, akan dilakukan *lemmatizing* dengan `nltk.stem.Lemmatizer()` dengan terlebih dahulu ditentukan POS nya dengan *POS-tagging* yang disediakan oleh `nltk.pos_tag()` dan fungsi `getWordnetPos()`. Akhirnya, setiap *token* yang tergolong *stop-words* akan dihapus dari list. Fungsi ini akan mengembalikan sebuah list yang berisi *word-token* dalam bentuk string.

#### 4.2.4. Word-Embedding

*Word-embedding* dilakukan untuk mengubah setiap *word-token* menjadi sebuah vektor n-dimensi. Vektor-vektor ini dimaksudkan menjadi fitur-fitur dari setiap *word-token*, yang akan menjadi input algoritma-algoritma *density-based clustering* dan *anomaly detection* yang akan dijalankan. *Word-embedding* dilakukan menggunakan pendekatan *word2vec* yang disediakan oleh *library Python gensim*, dan menggunakan model-model *pre-trained* yang telah disediakan.

Segmen Kode 4.4 Fungsi untuk mengganti model *word embedding* yang digunakan dan *list* modelnya

```
def SetModel(self, modelName: str = "glove-wiki-gigaword-300"):
    self.model = api.load(modelName)
WordEmbeddingModelList = ["word2vec-google-news-300",
                         "glove-wiki-gigaword-50",
                         "glove-wiki-gigaword-100",
                         "glove-wiki-gigaword-300"]
```

Digunakan pula, pada awal penelitian,

Segmen Kode 4.5 Fungsi WordEmbed

```
# input : list<str> : tokens of one document/sentence
# output : list<(str, list<int>[50|100|300])> : list of word-
vector pair for each word available on the model
def WordEmbed(self, document: list, model):
    word_embed_pairs = []
    for word in document:
        if word in model:
            word_embed_pairs.append((word, model[word]))
    return word_embed_pairs
```

Fungsi ini mengambil sebuah *list* yang berisi *word-token*, dan berupaya melakukan *embedding* terhadap setiap *word-token* didalamnya. Token-token yang tidak ada dalam model *word-embedding* yang digunakan akan dibuang. Ada 4 *pre-trained model* yang digunakan, yaitu word2vec-google-news-300, glove-wiki-gigaword-50, glove-wiki-gigaword-100, glove-wiki-gigaword-300. Model yang digunakan menentukan besar dimensi vektor yang dihasilkan.

#### 4.2.5. Sentence-Embedding

Setelah didapatkan *word-embedding* untuk setiap token dalam setiap dokumen, akan dilakukan proses *sentence-embedding*, yaitu melakukan agregasi setiap *word-vector* yang mewakili sebuah dokumen menjadi satu vektor kalimat atau dokumen yang berdimensi sama. Vektor ini dianggap sebagai representasi sebuah dokumen yang akan dijadikan input algoritma *anomaly detection* yang ada. Terdapat pilihan untuk menambahkan *weight* pada *sentence-*

*embedding* yang dilakukan, menggunakan *TF-IDF score* dari *word-token* tertentu dalam dokumennya.

Segmen Kode 4.6 Fungsi untuk melakukan *sentence-embedding* tanpa *weight*

```
# input : list<(str, list<float>[300])>, str : word-vector pair
list and preferred agg method.

# output : list<float>[300] : 300-d vector that represents an
aggregated value of the input words

def SentenceEmbedUnweightedFunction(self, word_embed_pair_list:
list, aggregateMethod: str = "avg"):

    wvs = []
    for pair in word_embed_pair_list:
        wvs.append(pair[1])
    if aggregateMethod == "avg":
        return np.mean(wvs, axis=0)
    else:
        return np.sum(wvs, axis=0)

# input : list<list<(str, list<float>[300])>>, str : list
containing word-vector pairs and preferred agg method
# output : list<(str, list<int>[300])> : list containing
sentence-vector pairs.

def SentenceEmbedUnweighted(self, word_embedded_docs: list,
aggregateMethod: str = "avg"):

    sentence_embedded_docs = []
    for i in range(len(word_embedded_docs)):

        sentence_embedded_docs.append(self.SentenceEmbedUnweightedFunc
tion(
            word_embedded_docs[i], aggregateMethod))
    return sentence_embedded_docs
```

Dalam pendekatan dengan fungsi ini, setiap *word-vector* n-dimensi yang ada dalam sebuah dokumen akan diagregasi dengan *mean* atau *sum* sebagaimana terlihat pada fungsi

SentenceEmbedUnweightedFunction(), sehingga menghasilkan satu vektor n-dimensi yang dianggap sebagai fitur suatu dokumen. Fungsi SentenceEmbedUnweighted akan mengembalikan sebuah *list* vektor n-dimensi yang merupakan cerminan fitur untuk setiap dokumen.

Segmen Kode 4.7 Fungsi untuk melakukan *sentence-embedding* dengan TF-IDF sebagai *weight*.

```
'''  
input :  
list<list<(str, list<float>[300])>> : word-vector pair list  
matrix : tf-idf matrix for the corresponding doc  
int : the row we want  
str : preferred agg method  
# output : list<float>[300] : 300-d vector that represents an  
aggregated value of the input words  
  
def SentenceEmbedWeightedFunction(self, word_embed_pair_list:  
list, tfidf_matrix, index: int, aggregateMethod: str = "avg"):  
    weighted_wvs = []  
    # multiplies each word with its TF-IDF value in the  
    # corresponding row. Is 0 if word isn't found somehow.  
    for pair in word_embed_pair_list:  
        tfidf_weight = 0  
        if pair[0] in tfidf_matrix:  
            tfidf_weight = tfidf_matrix[pair[0]][index]  
            weighted_wvs.append(pair[1] * tfidf_weight)  
    # turn into array for fast aggregating  
    weighted_wvs = np.array(weighted_wvs)  
    if aggregateMethod == "avg":  
        sentence_vector = np.mean(weighted_wvs, axis=0)  
    else:  
        sentence_vector = np.sum(weighted_wvs, axis=0)  
    return sentence_vector
```

```

# input : list<list<(str, list<float>[300])>>, str : list
containing word-vector pairs, TF-IDF matrix of the corpus, and
preferred agg method
# output : list<(str, list<float>[300])> : list containing
sentence-vector pairs.

def SentenceEmbedWeighted(self, word_embedded_docs: list,
tfidf_matrix, aggregateMethod="avg"):
    sentence_embedded_docs = []
    for i in range(len(word_embedded_docs)):

        sentence_embedded_docs.append(self.SentenceEmbedWeightedFunction(
            word_embedded_docs[i], tfidf_matrix, i,
            aggregateMethod))

    return sentence_embedded_docs

```

Dalam pendekatan ini, matriks TF-IDF score akan digunakan sebagai *weighting* untuk setiap token kata yang ada dalam sebuah dokumen. Fungsi agregat yang dipakai untuk mendapatkan satu vektor dokumen adalah *mean* dan *sum*, sama dengan pendekatan *unweighted*. Fungsi SentenceEmbedWeightedFunction() dipanggil dengan parameter index untuk mendapatkan baris pada matriks TF-IDF yang berkorespondensi dengan dokumen yang bersangkutan. Setiap word-vector akan dikalikan dengan TF-IDF score untuk kata tersebut dalam proses agregasi. Ini dilakukan dengan asumsi bahwa TF-IDF score mencerminkan relevansi sebuah *word-token* dalam sebuah dokumen, dalam dataset yang ada.

#### 4.2.6. Tiny Word-Embedding

Pendekatan ini merupakan pendekatan alternatif dari *sentence-embedding*. Dalam *feature extraction tiny word-embedding*, fitur sebuah dokumen berupa sebuah *list* yang berisi nilai-nilai singuler hasil agregat dimensi-dimensi yang didapatkan dari *word-embedding*.

Segmen Kode 4.8 Fungsi tiny word-embedding untuk sebuah dokumen tanpa *weighting*

```

"""
inputs :
- model          : --> word2vec model
- document       : list<str> --> a list of word tokens to embed.

```

```

- maxLength      : int          --> the maximum length, to pad the
vector with if necessary.
- weighted      : bool         --> multiply each word with respective
TF-IDF score or not.
'''

def TinyWordEmbed(self, document: list, model, maxLength: int):
    features = []
    # this creates a feature length of len(document)
    for word in document:
        if word in model:
            features.append(np.mean(model[word], axis=0))
    # if less than maxLength, pad with zeros.
    if len(features) < maxLength:
        padLength = maxLength - len(features)
        padding = np.zeros(padLength)
        features = np.concatenate((features, padding))

    # i remain pessimistic that this would work.
    return features

```

Karakteristik agregat yang dilakukan menyebabkan kemungkinan panjang fitur setiap dokumen berbeda-beda, yaitu sesuai jumlah *word-token* yang dimiliki sebuah dokumen. Oleh karena itu, digunakan sebuah parameter `maxLength` yang memastikan kesamaan jumlah fitur untuk setiap dokumen dalam dataset.

Segmen Kode 4.9 Fungsi *tiny word-embed* dengan *weighting TF-IDF*

```

# ditto TinyWordEmbed but with weight
def TinyWordEmbedWeighted(self, document: list, model,
maxLength: int, index: int):
    features = []
    for word in document:
        if word in model:
            weight = 0
            if word in self.tfidf_df:

```

```

        weight = self.tfidf_df[word][index]
        features.append(
            np.mean(model[word], axis=0) * weight)

    # if less than maxLength, pad with zeros.
    if len(features) < maxLength:
        padLength = maxLength - len(features)
        padding = np.zeros(padLength)
        features = np.concatenate((features, padding))

    # i still remain pessimistic that this would work.
    return features

```

Fungsi diatas berfungsi sama dengan fungsi TinyWordEmbed, namun dengan tambahan parameter `index` untuk mendapatkan TF-IDF *score* untuk *weighting* nilai agregat setiap *word-token*.

#### 4.2.7. TF-IDF sebagai *Weighting*

Matriks TF-IDF diperlukan sebagai weight dalam *feature extraction*, yaitu ketika mengagregasi *word-vector* menjadi *sentence-vector*, membuat *document-vector* dengan *tiny word-embedding*, atau sebagai *word-document matrix* pada *topic distribution* LDA. Matriks TF-IDF dihitung ketika ditentukan oleh parameter yang bersangkutan, dengan menggunakan library `Python sklearn.feature_extraction.text.TfidfVectorizer`.

Segmen Kode 4.10 Fungsi untuk mendapatkan Matriks TF-IDF

```

'''
inputs:
- doclist : list<str> --> list of doc/sentences.
- isProcessed : bool --> has it already been preprocessed?
Defaults to True.

'''

outputs:
- df_tfidf : Dataframe --> the TFIDF matrix in df form.
- matrix : matrix --> the TFIDF matrix purely. mainly for LDA
purposes.

```

```

"""
def GetTFIDF(self, doclist: list, isPreprocessed=True):
    if not isPreprocessed:
        doclist = [self.PreprocessDocument(
            doc, isLemma=True, isStopWords=True) for doc in
        doclist]
    # else:
    #     # just tokenize the thing
    #     doclist = [nltk.word_tokenize(doc) for doc in doclist]
    # i think the thing has already been tokenized. That's the
problem.

    flat_doclist = [' '.join(doc)
                    for doc in doclist] # turn into one big
corpus

    vectorizer = TfidfVectorizer()
    matrix = vectorizer.fit_transform(flat_doclist)
    tfidf_keys = vectorizer.get_feature_names_out()
    df_tfidf = db.pd.DataFrame(matrix.toarray(),
columns=tfidf_keys)

    return df_tfidf, matrix

```

Fungsi ini mengembalikan data matriks TF-IDF dalam dua format, yaitu dalam format DataFrame dan spmatrix, untuk menangani kebutuhan tipe data yang berbeda.

#### 4.2.8. Latent Dirichlet Allocation (LDA)

LDA adalah metode *feature extraction* yang diajukan, dan diterapkan melalui *library* Python gensim.models.LdaModel() pada sebuah dataset. Parameter yang diuji adalah jumlah topik yang diinferensikan, yang ada dalam parameter num\_topics.

Segmen Kode 4.11 Fungsi untuk mendapatkan distribusi topik dengan LDA

```

def GetLDADistribution(self, doclist: list, topics: int = 5,
use_tfidf: bool = True):
    new_corpus = []

```

```

if use_tfidf:
    for i in range(len(doclist)):
        doc = [(j, self.tfidf_matrix[i, j])
                for j in self.tfidf_matrix[i].indices]
        new_corpus.append(doc)
    gensim_dict =
corpora.Dictionary.from_corpus(new_corpus)
else:
    gensim_dict = corpora.Dictionary(doclist)
    new_corpus = [gensim_dict.doc2bow(doc) for doc in
doclist]

lda_model = gensim.models.LdaModel(
    new_corpus, num_topics=topics, id2word=gensim_dict)
doc_topic_distributions = lda_model[new_corpus]

docFeatureList = []
for doc_topic_dist in doc_topic_distributions:
    featureList = [0.0 for i in range(0, topics)]
    for topic_dist in doc_topic_dist:
        featureList[topic_dist[0]] = topic_dist[1]
    docFeatureList.append(featureList)

return docFeatureList

```

Fungsi ini akan mengembalikan vektor n-dimensi untuk setiap dokumen, dimana n adalah parameter num\_topics pada gensim.models.LdaModel(). Vektor-vektor tersebut digunakan sebagai fitur data yang akan menjadi input algoritma *anomaly detection* yang digunakan.

#### **4.2.9. Density-Based Clustering of Applications with Noise (DBSCAN)**

DBSCAN adalah metode utama yang digunakan untuk mendeteksi anomali. Dalam penelitian ini, DBSCAN diterapkan dengan

Segmen Kode 4.12 Fungsi DBSCAN

```
'''
```

```

inputs:
- vectors : list<list<float>> --> list of features corresponding
to each doc/sentence
- epsilon : float --> the radius within which points are
considered connected.
- min : int --> minimum amount of connected points for a point
to be considered a core point of a cluster.
'''
'''

output:
clusters : list<int> --> a list of integers to assign each data
point to a cluster. -1 means outlier.
'''


def GetDBSCANClusters(self, vectors, epsilon: float, min: int):
    dbscan = DBSCAN(eps=epsilon, min_samples=min)
    clusters = dbscan.fit_predict(vectors)
    return clusters

```

Fungsi ini mengambil *list* fitur-fitur data yang telah diekstraksi dengan metode yang dipilih dan menjalankan algoritma DBSCAN. Data yang dikembalikan adalah pengelompokan *cluster* data. Dokumen yang mendapatkan *cluster* -1 adalah dokumen yang ditandai sebagai *outlier*, dan ditandai sebagai anomali berdasarkan asumsi penelitian.

#### **4.2.10. Local Outlier Factor (LOF)**

Local Outlier Factor digunakan sebagai metode *anomaly detection* pembanding, dan diterapkan dalam penelitian ini menggunakan *library* Python `sklearn.neighbors.LocalOutlierFactor`. Parameter-parameter yang akan diuji adalah `n_neighbors` dan `contamination`.

Segmen Kode 4.13 Fungsi yang menjalankan algoritma LOF

```

def SetLOFClusters(self, vector):
    lof = LocalOutlierFactor(
        n_neighbors=self.AnomalyDetectionParams["neighbors"],
        contamination=self.AnomalyDetectionParams["contamination"])
    lof.fit(vector)

```

```

LOFResults = lof.negative_outlier_factor_

# minus values yield anomalies
if "lof_generalize" in self.AnomalyDetectionParams and
self.AnomalyDetectionParams["lof_generalize"]:
    print("if minus we go bald")
    LOFResults[LOFResults >= 0] = 0
    LOFResults[LOFResults < 0] = -1
else:
    # if not generalize, we assume outliers are within the
    # n-th percentile, with n = contamination rate.
    print(
        "taking                               the                  {}-th
percentile".format(self.AnomalyDetectionParams["contamination"]
] * 100))
    threshold = np.percentile(
        LOFResults,                      100                  *
self.AnomalyDetectionParams["contamination"])
    LOFResults[LOFResults >= threshold] = 0
    LOFResults[LOFResults < threshold] = -1
self.df["Cluster Assignment"] = LOFResults

```

Fungsi ini akan mengembalikan nilai `_negative_outlier_factor` untuk setiap dokumen. Akan dilaksanakan penempatan *cluster* yang berbeda untuk menentukan *outlier*. Jika tidak digeneralisir, maka akan diambil persentil ke-n dengan nilai negatif tertinggi, dimana n adalah nilai parameter `contamination`. Jika digeneralisir, maka semua dokumen yang memiliki nilai `_negative_outlier_factor` negatif akan ditandai sebagai *outlier*. Dalam rangka penelitian ini, dokumen-dokumen yang ditandai sebagai *outlier* akan dianggap sebagai anomali.

#### **4.2.11. Isolation Forest (IF)**

Isolation Forest digunakan sebagai metode *anomaly detection* pembanding. Isolation Forest diterapkan dengan menggunakan *library* Python `sklearn.ensemble.IsolationForest`. Parameter-parameter yang akan diuji adalah `contamination` dan `n_estimators`.

Segmen Kode 4.14 Fungsi yang menjalankan algoritma Isolation Forest

```
'''  
inputs:  
- vectors : list<list<float>> --> list of features corresponding  
to each doc/sentence  
  
'''  
  
def SetIFClusters(self, vector):  
    isolationForest = IsolationForest(  
  
n_estimators=self.AnomalyDetectionParams["estimators"],  
contamination=self.AnomalyDetectionParams["contamination"])  
    isolationForest.fit(vector)  
    IFResults = isolationForest.decision_function(vector)  
  
    # minus values yield anomalies.  
    for i in range(len(IFResults)):  
        if IFResults[i] >= 0:  
            IFResults[i] = 0  
        else:  
            IFResults[i] = -1  
    self.df["Cluster Assignment"] = IFResults
```

Fungsi ini menghasilkan `decision_function` untuk setiap rekord data. Dokumen dengan nilai dibawah nol, atau nilai negatif, merupakan *outlier*, dan dalam rangka penelitian ini ditandai sebagai anomali.

#### 4.2.12. *Class AnomalyDetector sebagai Pipeline Eksperimen dan Komponen Program*

Seluruh komponen-komponen diatas disusun sedemikian rupa dalam sebuah *class* `AnomalyDetector`, yang memiliki fungsi-fungsi pembantu untuk memudahkan pergantian parameter dan mendapatkan hasil *anomaly detection* dalam bentuk yang dapat dimanfaatkan oleh program. *Class* ini dimuat dalam file `anomalydetector.py` yang digunakan sebagai komponen langsung program yang dirancang.

Segmen Kode 4.15 anomalydetector.py

```
# imports go here
import numpy as np
from objects import db
import pandas as pd
import inflect
import string
import nltk
import gensim
import matplotlib.pyplot as plt
import gensim.downloader as api
from gensim import corpora, models
from nltk.test.gensim_fixt import setup_module
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.corpus import wordnet
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import DBSCAN
from sklearn.ensemble import IsolationForest
from sklearn.neighbors import LocalOutlierFactor
from sklearn.decomposition import PCA

# this is where everything we've experimented on will be
implemented.

class AnomalyDetector():
    def __init__(self, dbName: str = "", dh=None, model=None,
modelName="glove-wiki-gigaword-300") -> None:
        if dh is None:
            self.dh = db.DatabaseHandler(dbName=dbName)
        else:
            self.dh = dh
        if model is None:
            if modelName != "":
                if modelName == "glove-wiki-gigaword-300":
```

```

        self.model = api.load(modelName)

    else:
        self.model = model

    self.FeatureExtractionParams = {}
    self.AnomalyDetectionParams = {}

    '''

    inputs :
    - dh : DatabaseHandler --> to retrieve data from database
    - eventID : int --> we're doing this by event, so straight
to the eventID
    - selector : str --> pretty much formality.
    - splitBySentences : bool --> Split each doc into sentences
or not. Defaults to no.

    '''

    '''

    outputs:
None, just setting
    '''

    def SetDFFromDB(self, eventID: int, selector: str =
"event_id", splitBySentences: bool = False):
        self.df = self.dh.get_recordDataJoinedDF(selector=selector, ID=eventID)
        if splitBySentences:
            # df.set_index('id', inplace=True)
            self.df['answer'] = self.df['answer'].str.split('.')
            self.df = self.df.explode("answer", True)
            self.df.drop(self.df[self.df["answer"] == ""].index, inplace=True)
            self.df.reset_index(drop=True, inplace=True)

    # ditto above, but takes a pre-made DF instead.

```

```

        def SetDF(self, df: db.pd.DataFrame, splitBySentences: bool
= False):
            self.df = df
            if splitBySentences:
                # df.set_index('id', inplace=True)
                self.df['answer'] =
self.df['answer'].str.split('.')
                self.df = self.df.explode("answer", True)
                self.df.drop(self.df[self.df["answer"] ==
""].index, inplace=True)
                self.df.reset_index(drop=True, inplace=True)

        def SetModel(self, modelName: str = "glove-wiki-gigaword-
300"):
            self.model = api.load(modelName)

            # these are to add key:value to the dictionaries that dictate
parameters. Indeed, we are refurbishing.

        def SetFeatureExtractionParam(self, key: str, value):
            self.FeatureExtractionParams[key] = value

        def SetAnomalyDetectionParam(self, key: str, value):
            self.AnomalyDetectionParams[key] = value

        '''

        inputs :
        - dh : DatabaseHandler --> to retrieve data from database
        - eventID : int --> we're doing this by event, so straight
to the eventID
        - selector : str --> pretty much formality.
        - splitBySentences : bool --> Split each doc into sentences
or not. Defaults to no.

        '''

        '''

        outputs:

```

```

- df : DataFrame --> dataframe containing the thing we're
gonna be using.

'''


def GetDF(self, dh: db.DatabaseHandler, eventID: int,
selector: str = "event_id", splitBySentences: bool = False):
    df      = dh.get_recordDataJoinedDF(selector=selector,
ID=eventID)

    if splitBySentences:
        # df.set_index('id', inplace=True)
        df['answer'] = df['answer'].str.split('.')
        df = df.explode("answer", True)
        df.drop(df[df["answer"] == ""].index, inplace=True)
        df.reset_index(drop=True, inplace=True)

    return df


'''


inputs:
- doc : str --> a string representing a sentence/document.
- isLemma : bool --> use lemmatizer or not? Defaults to not.
- isStopWords : bool --> use stopwords or not? Defaults to
not.
- isInflect : bool --> use inflections (you're --> you are)
or not? Defaults to not.
- isNumberFiltered : bool --> delete numbers in the string?
Defaults to yes.

'''


'''


output : list<str> --> a list of word tokens (list<string>)
'''


def PreprocessDocument(self, doc: str, isLemma: bool =
False, isStopWords: bool = False, isInflect: bool = False,
isNumberFiltered: bool = True):
    inflector = inflect.engine()

```

```

stopwordSet = set(stopwords.words("english"))

lemmatizer = WordNetLemmatizer()
punctuations = string.punctuation
# if numbers are filtered, add that to the punctuation
string

if isNumberFiltered:
    punctuations += "1234567890"

# case fold
doc = doc.lower()

# remove puncs
doc = "".join([char for char in doc if char not in
punctuations])

# tokenize it.
token_list = nltk.word_tokenize(doc)

for i in range(len(token_list)):
    # if inflect
    if isInflect:
        if token_list[i].isdigit():
            token_list[i] =
inflector.number_to_words(token_list[i])

    # if lemma
    if isLemma:
        tagged_word = nltk.pos_tag([token_list[i]])
        wordnet_pos =
self.getWordnetPos(tagged_word[0][1])
        token_list[i] = lemmatizer.lemmatize(
            tagged_word[0][0], pos=wordnet_pos)

    # if stopword
    if isStopWords:

```

```

        if      token_list[i]      in      stopwordSet      or
token_list[i].isdigit():

            token_list[i] = "#"  # mark as #

# remove the marked strings
token_list = [token for token in token_list if token != "#"]

if token_list:
    return token_list
return [""]

"""

inputs:
- tag : str --> the tag obtained from POS tagging.

"""

"""

outputs:
- str --> Wordnet POS tag.

"""

def getWordnetPos(self, tag):
    """Map POS tag to WordNet POS tag"""
    if tag.startswith('J'):
        return wordnet.ADJ
    elif tag.startswith('V'):
        return wordnet.VERB
    elif tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN  # solves as noun by default.

"""

inputs:
- doclist : list<str> --> list of doc/sentences.

```

```

- isProcessed : bool --> has it already been preprocessed?
Defaults to True.

"""
"""

outputs:
- df_tfidf : Dataframe --> the TFIDF matrix in df form.
- matrix : matrix --> the TFIDF matrix purely. mainly for
LDA purposes.

"""

def GetTFIDF(self, doclist: list, isPreprocessed=True):
    if not isPreprocessed:
        doclist = [self.PreprocessDocument(
            doc, isLemma=True, isStopWords=True) for doc in
        doclist]
    # else:
    #     # just tokenize the thing
    #     doclist = [nltk.word_tokenize(doc) for doc in
    doclist]
    # i think the thing has already been tokenized. That's
the problem.

    flat_doclist = [' '.join(doc)
                    for doc in doclist] # turn into one big
corpus

    vectorizer = TfidfVectorizer()
    matrix = vectorizer.fit_transform(flat_doclist)
    tfidf_keys = vectorizer.get_feature_names_out()
    df_tfidf = db.pd.DataFrame(matrix.toarray(),
columns=tfidf_keys)

    return df_tfidf, matrix

# input : list<str> : tokens of one document/sentence
# output : list<(str, list<int>[300])> : list of word-vector
pair for each word available on the model

```

```

def WordEmbed(self, document: list, model):
    word_embed_pairs = []
    for word in document:
        if word in model:
            word_embed_pairs.append((word, model[word]))
    return word_embed_pairs

'''

inputs :
- model           : -->word2vec model
- document        : list<str> --> a list of word tokens to
embed.
- maxLength       : int          --> the maximum length, to pad
the vector with if necessary.
- weighted        : bool         --> multiply each word with
respective TF-IDF score or not.
'''


def TinyWordEmbed(self, document: list, model, maxLength:
int):
    features = []
    # this creates a feature length of len(document)
    for word in document:
        if word in model:
            features.append(np.mean(model[word], axis=0))
    # if less than maxLength, pad with zeros.
    if len(features) < maxLength:
        padLength = maxLength - len(features)
        padding = np.zeros(padLength)
        features = np.concatenate((features, padding))

    # i remain pessimistic that this would work.
    return features

# ditto TinyWordEmbed but with weight

```

```

    def TinyWordEmbedWeighted(self, document: list, model,
maxLength: int, index: int):
    features = []
    for word in document:
        if word in model:
            weight = 0
            if word in self.tfidf_df:
                weight = self.tfidf_df[word][index]
            features.append(
                np.mean(model[word], axis=0) * weight)

    # if less than maxLength, pad with zeros.
    if len(features) < maxLength:
        padLength = maxLength - len(features)
        padding = np.zeros(padLength)
        features = np.concatenate((features, padding))

    # i remain pessimistic that this would work.
    return features

# input : list<(str, list<float>[300])>, str : word-vector
pair list and preferred agg method.
# output : list<float>[300] : 300-d vector that represents
an aggregated value of the input words

def SentenceEmbedUnweightedFunction(self,
word_embed_pair_list: list, aggregateMethod: str = "avg"):
    wvs = []
    for pair in word_embed_pair_list:
        wvs.append(pair[1])
    if aggregateMethod == "avg":
        return np.mean(wvs, axis=0)
    else:
        return np.sum(wvs, axis=0)

```

```

# input : list<list<(str, list<float>[300])>>, str : list
containing word-vector pairs and preferred agg method
# output : list<(str, list<int>[300])> : list containing
sentence-vector pairs.

def SentenceEmbedUnweighted(self, word_embedded_docs: list,
aggregateMethod: str = "avg"):
    sentence_embedded_docs = []
    for i in range(len(word_embedded_docs)):

sentence_embedded_docs.append(self.SentenceEmbedUnweightedFunc
tion(
            word_embedded_docs[i], aggregateMethod))
    return sentence_embedded_docs

'''  

input :
list<list<(str, list<float>[300])>> : word-vector pair list
matrix : tf-idf matrix for the corresponding doc
int : the row we want
str : preferred agg method
'''  

# output : list<float>[300] : 300-d vector that represents
an aggregated value of the input words

def SentenceEmbedWeightedFunction(self,
word_embed_pair_list: list, tfidf_matrix, index: int,
aggregateMethod: str = "avg"):
    weighted_wvs = []
    # multiplies each word with its TF-IDF value in the
corresponding row. Is 0 if word isn't found somehow.
    for pair in word_embed_pair_list:
        tfidf_weight = 0
        if pair[0] in tfidf_matrix:
            tfidf_weight = tfidf_matrix[pair[0]][index]
        weighted_wvs.append(pair[1] * tfidf_weight)

```

```

        # turn into array for fast aggregating
        weighted_wvs = np.array(weighted_wvs)
        if aggregateMethod == "avg":
            sentence_vector = np.mean(weighted_wvs, axis=0)
        else:
            sentence_vector = np.sum(weighted_wvs, axis=0)
        return sentence_vector

    # input : list<list<(str, list<float>[300])>>, str : list
    containing word-vector pairs, TF-IDF matrix of the corpus, and
    preferred agg method
    # output : list<(str, list<float>[300])> : list containing
    sentence-vector pairs.

    def SentenceEmbedWeighted(self, word_embedded_docs: list,
    tfidf_matrix, aggregateMethod="avg"):
        sentence_embedded_docs = []
        for i in range(len(word_embedded_docs)):

    sentence_embedded_docs.append(self.SentenceEmbedWeightedFuncti
    on(
                word_embedded_docs[i],         tfidf_matrix,       i,
    aggregateMethod))
        return sentence_embedded_docs

    """
    input:
    - doclist : list<list<str>> --> list of tokenized
    sentences/docs
    - topics : int --> number of inferred topics.
    - use_tfidf : bool --> use TFIDF or not? defaults to yes.
    """
    """
    output:
    - docFeatureList : list<list<float>> --> topic distribution
    for each sentence/doc

```

```

"""
def GetLDADistribution(self, doclist: list, topics: int =
5, use_tfidf: bool = True):
    new_corpus = []

    if use_tfidf:
        for i in range(len(doclist)):
            doc = [(j, self.tfidf_matrix[i, j])
                    for j in self.tfidf_matrix[i].indices]
            new_corpus.append(doc)
        gensim_dict = corpora.Dictionary.from_corpus(new_corpus)
    else:
        gensim_dict = corpora.Dictionary(doclist)
        new_corpus = [gensim_dict.doc2bow(doc) for doc in doclist]

    lda_model = gensim.models.LdaModel(
        new_corpus, num_topics=topics, id2word=gensim_dict)
    goofy_ahh_doc_topic_distributions = lda_model[new_corpus]

    docFeatureList = []
    for doc_topic_dist in goofy_ahh_doc_topic_distributions:
        featureList = [0.0 for i in range(0, topics)]
        for topic_dist in doc_topic_dist:
            featureList[topic_dist[0]] = topic_dist[1]
        docFeatureList.append(featureList)

    return docFeatureList

"""

inputs:

```

```

    - vectors : list<list<float>> --> list of features
corresponding to each doc/sentence

    - epsilon : float --> the radius within which points are
considered connected.

    - min : int --> minimum amount of connected points for a
point to be considered a core point of a cluster.

    '''

    '''

    output:

    clusters : list<int> --> a list of integers to assign each
data point to a cluster. -1 means outlier.

    '''

def GetDBSCANClusters(self, vectors, epsilon: float, min:
int):

    dbscan = DBSCAN(eps=epsilon, min_samples=min)
    clusters = dbscan.fit_predict(vectors)
    # plt.title("to the depths of depravity {} and the cusp
of blasphemy {}".format(epsilon, min))
    # plt.scatter(vectors[:, 0], vectors[:, 1], c=clusters)
    # plt.show()
    # print(clusters)
    return clusters

    '''

inputs:

    - vectors : list<list<float>> --> list of features
corresponding to each doc/sentence

    '''

def GetIFResults(self, vector):
    isolationForest = IsolationForest(n_estimators=500,
contamination=0.1)
    isolationForest.fit(vector)
    IFResults = isolationForest.decision_function(vector)

```

```

# minus values yield anomalies.
for i in range(len(IFResults)):
    if IFResults[i] >= 0:
        IFResults[i] = 0
    else:
        IFResults[i] = -1
return IFResults

'''

inputs :
- clusters : list<int> --> a list of clusters assigned to
each doc/sentence
- df : DataFrame --> the dataframe in question
- isReturnSeparate : bool --> split return or not. Defaults
to split (for some reason...)
'''

'''

outputs:
- dfOutliers : DataFrame --> the dataframe whose answers
have been marked as outliers.
- dfGoods : DataFrame --> the dataframe whose answers have
not been marked as outliers.

'''


def ReturnClusters(self, isReturnSeparate: bool = True):
    if isReturnSeparate:
        dfGoods           = self.df.loc[self.df["Cluster"
Assignment"] != -1]
        dfGoods.reset_index(inplace=True)
        dfOutliers       = self.df.loc[self.df["Cluster"
Assignment"] == -1]
        dfOutliers.reset_index(inplace=True)
        return dfOutliers, dfGoods
    else:

```

```

        if self.df.isnull().values.any():
            self.df.reset_index(inplace=True)
        return self.df

    """
    inputs:
    - method : str --> LDA or Embedding.
    - isWeighted : bool --> use weights or not
    - nTopics : int --> for LDA.
    """
    """
    -
    - outputs : none. This is an internal function
    """

def GetAnomalies(self, isReturnSeparate: bool = False):
    self.SetDocumentTokens() # set tokens in the DF
    if self.FeatureExtractionParams["method"] == "Embedding":
        self.SetEmbeddingResult()
    elif self.FeatureExtractionParams["method"] == "LDA":
        self.SetLDAResult()
    elif self.FeatureExtractionParams["method"] == "Tiny":
        self.SetTinyEmbeddingResult()

    if self.AnomalyDetectionParams["algorithm"] == "DBSCAN":
        self.SetDBSCANClusters(list(self.df["Document
Embed"]))
    elif self.AnomalyDetectionParams["algorithm"] == "LOF":
        self.SetLOFClusters(list(self.df["Document
Embed"]))
    elif self.AnomalyDetectionParams["algorithm"] == "IF":
        self.SetIFClusters(list(self.df["Document
Embed"]))

```

```

        return
self.ReturnClusters(isReturnSeparate=isReturnSeparate)

'''


inputs  : None (checks self.FeatureExtractionParams)
desc    : embeds each doc and put it in a new column "Document
Embed"
'''


def SetEmbeddingResult(self):
    # extract feature with embedding
    self.wordEmbeddedDocs = [self.WordEmbed(
        doc, self.model) for doc in self.preprocessedDocs]

    if "weighted" in self.FeatureExtractionParams and
self.FeatureExtractionParams["weighted"]:
        self.tfidf_df, self.tfidf_matrix = self.GetTFIDF(
            self.preprocessedDocs)
        self.doc_embeds = self.SentenceEmbedWeighted(
            self.wordEmbeddedDocs, self.tfidf_df,
self.FeatureExtractionParams["aggregate_method"])
    else:
        self.doc_embeds = self.SentenceEmbedUnweighted(
            self.wordEmbeddedDocs,
self.FeatureExtractionParams["aggregate_method"])

    self.df["Document Embed"] = self.doc_embeds

def SetTinyEmbeddingResult(self):
    # get max length
    maxdoc = max(self.preprocessedDocs, key=len)
    maxlen = len(maxdoc)
    # extract feature of each word with embedding
    self.doc_embeds = [self.TinyWordEmbed(

```

```

        doc,      self.model,      maxlen)      for      doc      in
self.preprocessedDocs]

        if    "weighted"    in    self.FeatureExtractionParams    and
self.FeatureExtractionParams["weighted"]:

            self.tfidf_df,           self.tfidf_matrix       =
self.GetTFIDF(self.preprocessedDocs)

            self.doc_embeds = [self.TinyWordEmbedWeighted(doc,
self.model,      maxlen,      i)      for      i,      doc      in
enumerate(self.preprocessedDocs) ]

            self.df["Document Embed"] = self.doc_embeds


def SetDefaultParams(self):

    # here we will put the default params
    self.SetFeatureExtractionParam("method", "Embedding")
    self.SetFeatureExtractionParam("weighted", True)
    self.SetFeatureExtractionParam("condense", False)
    self.SetFeatureExtractionParam("n_topics", 5)
    self.SetFeatureExtractionParam("aggregate_method",
"avg")

    self.SetAnomalyDetectionParam("algorithm", "DBSCAN")
    self.SetAnomalyDetectionParam("epsilon", 1.0)
    self.SetAnomalyDetectionParam("minsample", 2)
    self.SetAnomalyDetectionParam("epsilon", 1.0)
    self.SetAnomalyDetectionParam("algorithm", "IF")
    self.SetAnomalyDetectionParam("estimators", 500)
    self.SetAnomalyDetectionParam("contamination", 0.1)
    self.SetAnomalyDetectionParam("neighbors", 5)


# preprocess each doc/sentence
def SetDocumentTokens(self):

    self.preprocessedDocs = [self.PreprocessDocument(
        doc,  isLemma=True,  isStopWords=True)  for  doc  in
self.df["answer"]]

    self.df["Tokenized"] = self.preprocessedDocs

```

```

# if cut off data with less than x values
if "prune" in self.FeatureExtractionParams:
    mask = self.df['Embedded Docs'].apply(
        lambda x: len(x) >
self.FeatureExtractionParams["prune"])
    self.df = self.df[mask]

"""
inputs : None (checks self.FeatureExtractionParams)
desc   : assigns topic distribution for each document.
"""

def SetLDAResult(self):
    doclist = list(self.df["Tokenized"])
    new_corpus = []
    if self.FeatureExtractionParams["weighted"]:
        self.tfidf_df, self.tfidf_matrix = self.GetTFIDF(
            self.preprocessedDocs)
        for i in range(len(doclist)):
            doc = [(j, self.tfidf_matrix[i, j])
                    for j in self.tfidf_matrix[i].indices]
            new_corpus.append(doc)
        gensim_dict =
corpora.Dictionary.from_corpus(new_corpus)
    else:
        gensim_dict = corpora.Dictionary(doclist)
        new_corpus = [gensim_dict.doc2bow(doc) for doc in
doclist]

    lda_model = gensim.models.LdaModel(
        new_corpus,
num_topics=self.FeatureExtractionParams["n_topics"],
id2word=gensim_dict)
    goofy_ahh_doc_topic_distributions =
lda_model[new_corpus]

```

```

        docFeatureList = []
        for doc_topic_dist in
goofy_ahh_doc_topic_distributions:
            featureList = [0.0 for i in range(
                0, self.FeatureExtractionParams["n_topics"])]
            for topic_dist in doc_topic_dist:
                featureList[topic_dist[0]] = topic_dist[1]
            docFeatureList.append(featureList)

        self.df["Document Embed"] = docFeatureList

    """
    inputs :
    - vectors : list<list<float>> --> list of features
corresponding to each doc/sentence
    desc : assigns cluster via DBSCAN.
    """

def SetDBSCANClusters(self, vectors):
    dbscan = DBSCAN(
        eps=self.AnomalyDetectionParams["epsilon"],
min_samples=self.AnomalyDetectionParams["minsamp"])
        clusters = dbscan.fit_predict(vectors)
        self.df["Cluster Assignment"] = clusters

    """
    inputs:
    - vectors : list<list<float>> --> list of features
corresponding to each doc/sentence

    """

def SetIFClusters(self, vector):
    isolationForest = IsolationForest(

```

```

n_estimators=self.AnomalyDetectionParams["estimators"],
contamination=self.AnomalyDetectionParams["contamination"])
    isolationForest.fit(vector)
    IFResults = isolationForest.decision_function(vector)

    # minus values yield anomalies.
    for i in range(len(IFResults)):
        if IFResults[i] >= 0:
            IFResults[i] = 0
        else:
            IFResults[i] = -1
    self.df["Cluster Assignment"] = IFResults

    """
    inputs : vectors : list<list<float>> --> list of features
    for each doc/sentence
    """

def SetLOFClusters(self, vector):
    lof = LocalOutlierFactor(
        n_neighbors=self.AnomalyDetectionParams["neighbors"],
        contamination=self.AnomalyDetectionParams["contamination"])
    lof.fit(vector)
    LOFResults = lof.negative_outlier_

    # minus values yield anomalies
    if "lof_generalize" in self.AnomalyDetectionParams and
    self.AnomalyDetectionParams["lof_generalize"]:
        print("if minus we go bald")
        LOFResults[LOFResults >= 0] = 0
        LOFResults[LOFResults < 0] = -1
    else:

```

```

        # if not generalize, we assume outliers are within
        # the n-th percentile, with n = contamination rate.

        print(
            "taking           the           {}-th
percentile".format(self.AnomalyDetectionParams["contamination"]
] * 100))

        threshold = np.percentile(
            LOFResults,           100           *
self.AnomalyDetectionParams["contamination"])

        LOFResults[LOFResults >= threshold] = 0
        LOFResults[LOFResults < threshold] = -1
        self.df["Cluster Assignment"] = LOFResults

```

Implementasi *class* ini didesain untuk memudahkan eksperimen dan program, sehingga bisa mengatur pengambilan data, pengaturan parameter-parameter, dan eksekusi program dengan cepat. Parameter-parameter diatur dalam *dictionary* FeatureExtractionParams dan AnomalyDetectionParams. Digunakan fungsi SetFeatureExtractionParam() dan SetAnomalyDetectionParam() untuk menambahkan atau mengurangi parameter yang sudah ada. Beberapa fungsi get- yang telah didefinisikan dalam subbab-subbab sebelumnya digantikan oleh fungsi-fungsi set- dengan alur logika yang kongruen. Penggantian didasarkan konversi dari program awal yang terdiri dari fungsi-fungsi global menuju pemrograman berbasis obyek.

### **4.3. Implementasi Program**

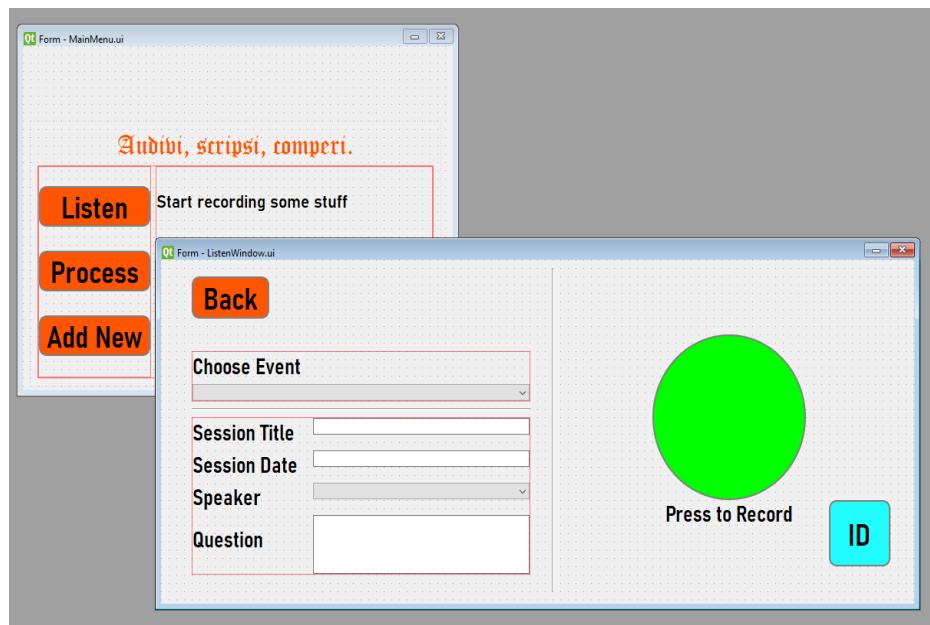
#### **4.3.1. Tools, IDE, dan Library**

Pengembangan program dilakukan dalam IDE Visual Studio Code menggunakan bahasa Python. Digunakan *library-library* Python yang diimpor dari PyPI untuk melaksanakan tugas-tugas yang berhubungan dengan perancangan GUI, mengadakan koneksi dengan *database*, serta melakukan *transcribing* dan *translating file* yang berupa rekaman suara menjadi teks.

#### **4.3.2. GUI**

Pengembangan GUI program dilakukan dengan software Qt Designer dan *library* Python-nya, PyQt6. Qt Designer digunakan untuk merancang desain layout untuk setiap window yang diperlukan, yang kemudian disimpan dalam sebuah file dengan extension *.ui*, yang berisi

data layout dalam format XML. Dengan command line `pyuic6`, file tersebut dikonversi menjadi code dalam bahasa Python dalam rangka mempermudah pemrograman fungsionalitas dengan library PyQt6.

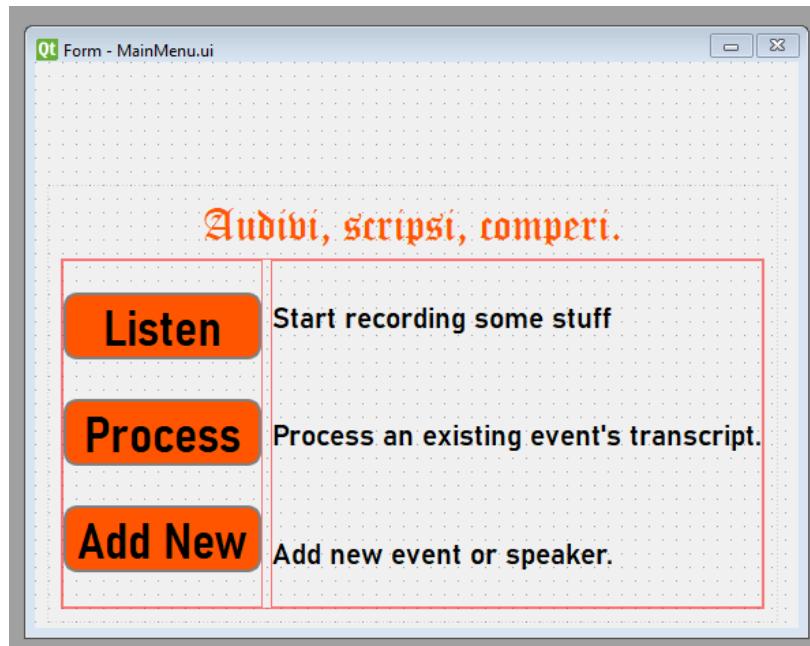


Gambar 4.1 Desain window MainMenu dan ListenWindow dalam program Qt Designer

Untuk program ini, dikembangkan lima Window utama, yaitu MainMenu, AddWindow, ListenWindow, ProcessWindow, dan PresentWindow.

#### 4.3.2.1. MainMenu

MainMenu adalah Window utama dari program. Window ini berisi menu untuk mengarahkan pengguna untuk menjalankan modul yang dikehendaki.



Gambar 4.2 Desain MainMenu dalam program Qt Designer

File MainMenu.ui yang dihasilkan dikonversi menjadi code dalam bahasa Python dengan perintah *command line* berikut ini :

Segmen Kode 4.16 Konversi MainMenu.ui ke mainmenu.py

```
pyuic6 ui/MainMenu.ui -x -o objects/mainmenu.py
```

Segmen Kode 4.17 mainmenu.py

```
# Form implementation generated from reading ui file
'MainMenu.ui'
#
# Created by: PyQt6 UI code generator 6.5.0
#
# WARNING: Any manual changes made to this file will be lost
when pyuic6 is
# run again. Do not edit this file unless you know what you are
doing.

from PyQt6 import QtCore, QtGui, QtWidgets
```

```

class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(560, 415)
        Form.setStyleSheet("QLabel#label_title{ \n"
"    font: 24pt \"Old English Text MT\";\n"
"    color:rgb(255, 85, 0);\n"
"}\n"
"\n"
"QPushButton{\n"
"    font: 63 28pt \"Bahnschrift SemiBold SemiConden\";\n"
"    background:rgb(255, 85, 0);\n"
"    border: 2px solid gray;\n"
"    border-radius: 10px;\n"
"    padding: 0 8px;\n"
"}\n"
"\n"
"QPushButton:hover{\n"
"    background:rgb(255, 140, 0);\n"
"}\n"
"\n"
"QPushButton:pressed{\n"
"    background: rgb(255, 170, 0);\n"
"}\n"
"\n"
"QLabel{\n"
"    \n"
"    font: 63 16pt \"Bahnschrift SemiBold\";\n"
"}\n"
""")
        self.frame = QtWidgets.QFrame(parent=Form)
        self.frame.setGeometry(QtCore.QRect(9, 90, 536, 321))

```

```

        self.frame.setStyleSheet("QLabel#label_title,
#label_subtitle{ \n"
"font: 24pt \"Old English Text MT\";\n"
"    color:rgb(255, 85, 0);\n"
"}\n"
"\n"
"QPushButton{\n"
"    font: 63 28pt \"Bahnschrift SemiBold SemiConden\";\n"
"    background:rgb(255, 85, 0);\n"
"    border: 2px solid gray;\n"
"    border-radius: 10px;\n"
"    padding: 0 8px;\n"
"}\n"
"\n"
"QPushButton:hover{\n"
"    background:rgb(255, 140, 0);\n"
"}\n"
"\n"
"QPushButton:pressed{\n"
"    background: rgb(255, 170, 0);\n"
"}\n"
"\n"
"QLabel{\n"
"    \n"
"    font: 63 16pt \"Bahnschrift SemiBold\";\n"
"}\n"
"")

self.frame.setFrameShape(QtWidgets.QFrame.Shape.StyledPanel)

self.frame.setFrameShadow(QtWidgets.QFrame.Shadow.Raised)
        self.frame.setObjectName("frame")
        self.verticalLayout_2
        QtWidgets.QVBoxLayout(self.frame)

```

```

self.verticalLayout_2.setObjectName("verticalLayout_2")
    self.label_title = QtWidgets.QLabel(parent=self.frame)
    font = QtGui.QFont()
    font.setFamily("Old English Text MT")
    font.setPointSize(24)
    font.setBold(False)
    font.setItalic(False)
    font.setWeight(50)
    self.label_title.setFont(font)

self.label_title.setAlignment(QtCore.Qt.AlignmentFlag.AlignCenter)
    self.label_title.setObjectName("label_title")
    self.verticalLayout_2.addWidget(self.label_title)
    self.horizontalLayout = QtWidgets.QHBoxLayout()

self.horizontalLayout.setObjectName("horizontalLayout")
    self.layout_buttons = QtWidgets.QVBoxLayout()
    self.layout_buttons.setObjectName("layout_buttons")
    self.button_listen = QtWidgets.QPushButton(parent=self.frame)
        self.button_listen.setSizePolicy(QtWidgets.QSizePolicy.Policy.Minimum,
                                         QtWidgets.QSizePolicy.Policy.Fixed)
        self.button_listen.setHorizontalStretch(0)
        self.button_listen.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.button_listen.sizePolicy().hasHeightForWidth())
    self.button_listen.setSizePolicy(sizePolicy)
    self.button_listen.setObjectName("button_listen")
    self.layout_buttons.addWidget(self.button_listen)
    self.button_process = QtWidgets.QPushButton(parent=self.frame)

```

```

        self.button_process.setObjectName("button_process")
        self.layout_buttons.addWidget(self.button_process)
        self.button_create = QtWidgets.QPushButton(parent=self.frame)
        self.button_create.setObjectName("button_create")
        self.layout_buttons.addWidget(self.button_create)
        self.horizontalLayout.addLayout(self.layout_buttons)
        self.layout_labels = QtWidgets.QVBoxLayout()
        self.layout_labels.setObjectName("layout_labels")
        self.label_listen = QtWidgets.QLabel(parent=self.frame)
        self.label_listen.setObjectName("label_listen")
        self.layout_labels.addWidget(self.label_listen)
        self.label_process = QtWidgets.QLabel(parent=self.frame)
        self.label_process.setObjectName("label_process")
        self.layout_labels.addWidget(self.label_process)
        self.label_add = QtWidgets.QLabel(parent=self.frame)
        self.label_add.setObjectName("label_add")
        self.layout_labels.addWidget(self.label_add)
        self.horizontalLayout.addLayout(self.layout_labels)
        self.horizontalLayout.setStretch(0, 1)
        self.horizontalLayout.setStretch(1, 2)
        self.verticalLayout_2.addLayout(self.horizontalLayout)
        self.verticalLayout_2.setStretch(1, 8)
        self.label_icon = QtWidgets.QLabel(parent=Form)
        self.label_icon.setGeometry(QtCore.QRect(220, 0, 100,
100))
        self.sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Policy.Fixed,
QtWidgets.QSizePolicy.Policy.Fixed)
        self.sizePolicy.setHorizontalStretch(0)
        self.sizePolicy.setVerticalStretch(0)

        self.sizePolicy.setHeightForWidth(self.label_icon.sizePolicy().hasHeightForWidth())

```

```

        self.label_icon.setSizePolicy(sizePolicy)
        self.label_icon.setText("")

self.label_icon.setPixmap(QtGui.QPixmap("asset/images/Solaire.
png"))
        self.label_icon.setScaledContents(True)
        self.label_icon.setObjectName("label_icon")

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)

def retranslateUi(self, Form):
    _translate = QtCore.QCoreApplication.translate
    Form.setWindowTitle(_translate("Form", "Form"))
    self.label_title.setText(_translate("Form", "Audivi,
scripsi, comperi."))
    self.button_listen.setText(_translate("Form",
"Listen"))
    self.button_process.setText(_translate("Form",
"Process"))
    self.button_create.setText(_translate("Form", "Add
New"))
    self.label_listen.setText(_translate("Form", "Start
recording some stuff"))
    self.label_process.setText(_translate("Form", "Process
an existing event's transcript."))
    self.label_add.setText(_translate("Form", "Add new
event or speaker."))

if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    Form = QtWidgets.QWidget()
    ui = Ui_Form()

```

```
ui.setupUi(Form)
Form.show()
sys.exit(app.exec())
```

File mainmenu.py yang dihasilkan digunakan sebagai *inheritance* untuk class MainMenu() yang ada di file utama app.py, dimana tombol-tombol yang telah didesain dihubungkan dengan fungsi-fungsi yang sesuai.

Segmen Kode 4.18 Class MainWindow

```
class MainWindow(qtw.QWidget):
    def __init__(self):
        super().__init__()
        # set stuff here
        self.ui = mm.Ui_Form()
        self.ui.setupUi(self)
        self.setWindowTitle(
            "I hear, I write, I discover")

    self.setWindowIcon(qtg.QIcon("asset/images/Solaire.png"))
        # thanks to Rueful on Discord for grammatical
correction.

        self.ui.label_title.setText("Audio, Scribo, Comperio")
        #
self.ui.label_icon.setPixmap(qtg.QPixmap("asset/images/Solaire
.png"))

        # connect stuff here

    self.ui.button_listen.clicked.connect(self.button_startListeni
ng)

    self.ui.button_process.clicked.connect(self.button_startProces
sing)
```

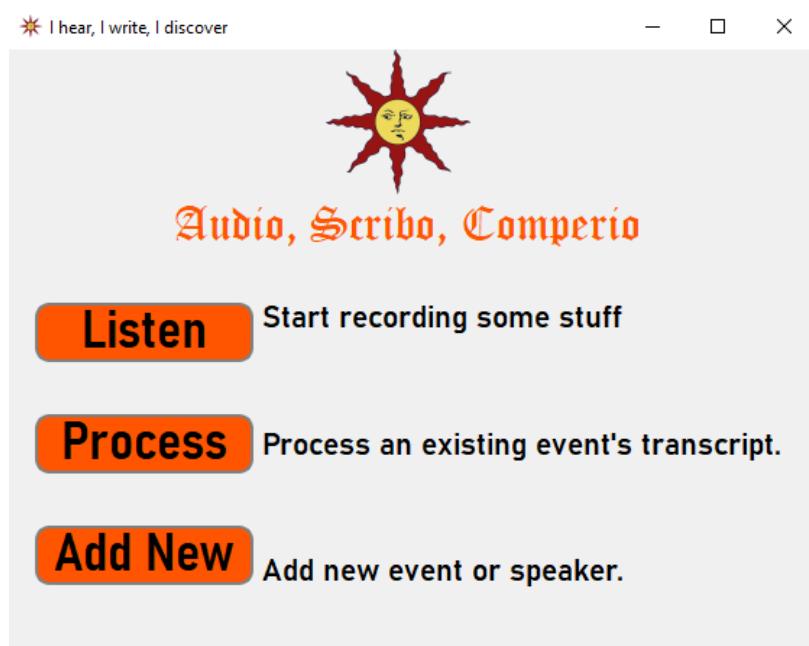
```
self.ui.button_create.clicked.connect(self.button_startCreating)

def button_startListening(self):
    self.listenWindow = ListenWindow(self)
    self.hide()
    self.listenWindow.show()

def button_startProcessing(self):
    self.processWindow = ProcessWindow(self)
    self.hide()
    self.processWindow.show()

def button_startCreating(self):
    self.addWindow = AddWindow(self)
    self.hide()
    self.addWindow.show()
```

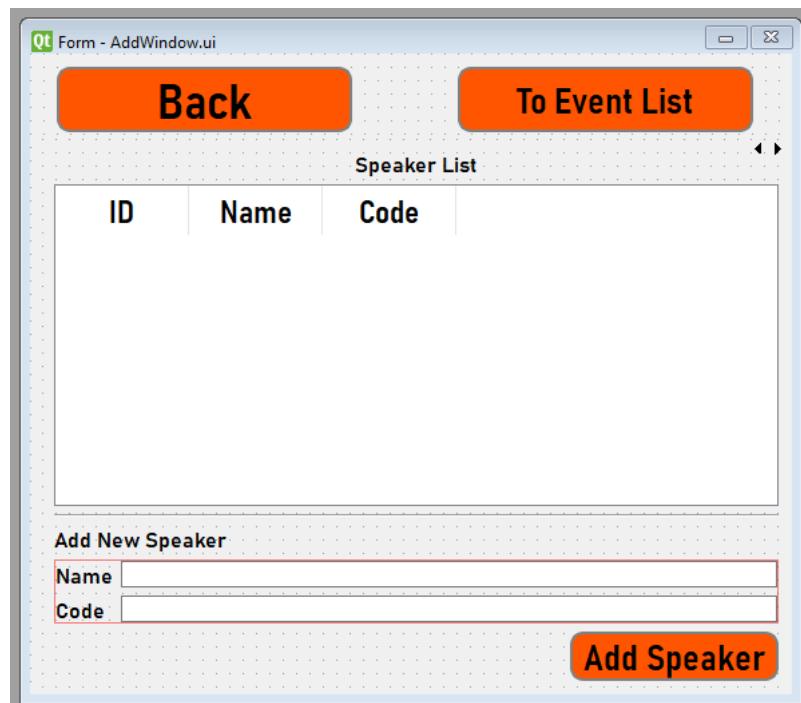
Dalam *class* ini, diatur fungsi setiap tombol yang ada dalam menu. Tombol “Listen” akan membuka ListenWindow, dimana pengguna bisa memulai proses rekaman dalam sebuah diskusi dan menyimpan datanya. Tombol “Process” akan membuka ProcessWindow, dimana pengguna bisa membuka data hasil rekaman sebuah diskusi, melakukan *transcribing*, lalu memerintahkan program untuk melakukan deteksi anomali pada data tersebut. Tombol “Add New” akan membuka AddWindow, dimana pengguna bisa melakukan penambahan data sesi diskusi, serta nama dan kode pembicara yang bisa digunakan dalam proses Listening berikutnya.



Gambar 4.3 Window MainMenu dalam program yang sudah dirancang dan dijalankan

#### 4.3.2.2. AddWindow

AddWindow adalah sebuah Window dimana pengguna dapat menambahkan data berupa *event* diskusi atau pembicara baru. Data-data ini digunakan sebagai metadata pelengkap untuk mencatat rekam pernyataan seorang peserta dalam sebuah sesi diskusi, menurut skema *database* yang telah ditentukan.



Gambar 4.4 Desain AddWindow dalam QtDesigner

File AddWindow.ui yang dihasilkan dikonversi menjadi code dalam bahasa Python dengan perintah *command line* berikut ini :

Segmen Kode 4.19 Konversi AddWindow.ui ke addwindow.py

```
pyuic6 ui/AddWindow.ui -x -o objects/addwindow.py
```

Segmen Kode 4.20 addwindow.py

```
#  Form implementation generated from reading ui file
'ui/AddWindow.ui'

#
# Created by: PyQt6 UI code generator 6.5.0
#
# WARNING: Any manual changes made to this file will be lost
when pyuic6 is
# run again. Do not edit this file unless you know what you are
doing.

from PyQt6 import QtCore, QtGui, QtWidgets


class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(570, 480)
        Form.setStyleSheet("QLabel#label_title,
#label_subtitle{ \n"
"    \n"
"    font: 63 28pt \"Bahnschrift SemiBold SemiConden\";\n"
"    color:rgb(255, 85, 0);\n"
"}\n"
"\n"
"QPushButton{\n"
"    font: 63 28pt \"Bahnschrift SemiBold SemiConden\";\n"
"    background:rgb(255, 85, 0); \n"
"
```

```

"    border: 2px solid gray;\n"
"    border-radius: 10px;\n"
"    padding: 0 8px;\n"
"}\n"
"\n"
"QPushButton#button_addSpeaker,     QPushButton#button_addEvent,
QPushButton#button_toggle_widget{\n"
"    font: 63 20pt \"Bahnschrift SemiBold SemiConden\";\n"
"}\n"
"QPushButton:hover{\n"
"    background:rgb(255, 140, 0);\n"
"}\n"
"\n"
"QPushButton:pressed{\n"
"    background: rgb(255, 170, 0);\n"
"}\n"
"\n"
"QLabel{\n"
"    font: 63 16pt \"Bahnschrift SemiBold\";\n"
"}\n"
"\n"
"QLabel{\n"
"    \n"
"    font: 63 12pt \"Bahnschrift SemiBold\";\n"
"}\n"
"\n"
"QTableWidget{\n"
"    font: 14pt \"Bahnschrift\";\n"
"}\n"
"\n"
"QHeaderView{\n"
"    font: 63 18pt \"Bahnschrift SemiBold SemiConden\";\n"
"}")
        self.button_toggle_widget = QtWidgets.QPushButton(parent=Form)
        self.button_toggle_widget.setEnabled(True)

```

```

self.button_toggle_widget.setGeometry(QtCore.QRect(320,      10,
221, 49))

    sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Policy.Maximum,
QtWidgets.QSizePolicy.Policy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.button_toggle_widget.sizePolicy().hasHeightForWidth())
    self.button_toggle_widget.setSizePolicy(sizePolicy)
    self.button_toggle_widget.setFlat(False)

self.button_toggle_widget.setObjectName("button_toggle_widget")
)

    self.stackedWidget = QtWidgets.QStackedWidget(parent=Form)
        self.stackedWidget.setGeometry(QtCore.QRect(9, 64, 560,
415))

        self.stackedWidget.setObjectName("stackedWidget")
        self.page_addSpeaker = QtWidgets.QWidget()
        self.page_addSpeaker.setObjectName("page_addSpeaker")
        self.verticalLayout = QtWidgets.QVBoxLayout(self.page_addSpeaker)
            self.verticalLayout.setObjectName("verticalLayout")
            self.label_speakerList = QtWidgets.QLabel(parent=self.page_addSpeaker)

self.label_speakerList.setAlignment(QtCore.Qt.AlignmentFlag.AlignCenter)

self.label_speakerList.setObjectName("label_speakerList")
    self.verticalLayout.addWidget(self.label_speakerList)

```

```

    self.table_speakers =  

QtWidgets.QTableWidget(parent=self.page_addSpeaker)  

    self.table_speakers.setShowGrid(True)  

    self.table_speakers.setColumnCount(3)  

    self.table_speakers.setObjectName("table_speakers")  

    self.table_speakers.setRowCount(0)  

    item = QtWidgets.QTableWidgetItem()  

    self.table_speakers.setHorizontalHeaderItem(0, item)  

    item = QtWidgets.QTableWidgetItem()  

    self.table_speakers.setHorizontalHeaderItem(1, item)  

    item = QtWidgets.QTableWidgetItem()  

    self.table_speakers.setHorizontalHeaderItem(2, item)  

    self.verticalLayout.addWidget(self.table_speakers)  

    self.line =  

QtWidgets.QFrame(parent=self.page_addSpeaker)  

    self.line.setFrameShape(QtWidgets.QFrame.Shape.HLine)  

self.line setFrameShadow(QtWidgets.QFrame.Shadow.Sunken)  

    self.line.setObjectName("line")  

    self.verticalLayout.addWidget(self.line)  

    self.label_addSpeaker =  

QtWidgets.QLabel(parent=self.page_addSpeaker)  

self.label_addSpeaker.setObjectName("label_addSpeaker")  

    self.verticalLayout.addWidget(self.label_addSpeaker)  

    self.formLayout = QtWidgets.QFormLayout()  

    self.formLayout.setObjectName("formLayout")  

    self.label_name =  

QtWidgets.QLabel(parent=self.page_addSpeaker)  

    font = QtGui.QFont()  

    font.setFamily("Bahnschrift SemiBold")  

    font.setPointSize(12)  

    font.setBold(False)  

    font.setItalic(False)  

    font.setWeight(7)

```

```

        self.label_name.setFont(font)
        self.label_name.setObjectName("label_name")
        self.formLayout.addWidget(0,
QtWidgets.QFormLayout.ItemRole.LabelRole, self.label_name)
        self.line_name =
QtWidgets.QLineEdit(parent=self.page_addSpeaker)
        self.line_name.setObjectName("line_name")
        self.formLayout.addWidget(0,
QtWidgets.QFormLayout.ItemRole.FieldRole, self.line_name)
        self.label_code =
QtWidgets.QLabel(parent=self.page_addSpeaker)
        self.label_code.setObjectName("label_code")
        self.formLayout.addWidget(1,
QtWidgets.QFormLayout.ItemRole.LabelRole, self.label_code)
        self.line_code =
QtWidgets.QLineEdit(parent=self.page_addSpeaker)
        self.line_code.setObjectName("line_code")
        self.formLayout.addWidget(1,
QtWidgets.QFormLayout.ItemRole.FieldRole, self.line_code)
        self.verticalLayout.addLayout(self.formLayout)
        self.button_addSpeaker =
QtWidgets.QPushButton(parent=self.page_addSpeaker)
        sizePolicy =
QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Policy.Maximum,
QtWidgets.QSizePolicy.Policy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.button_addSpeaker.sizePolicy
().hasHeightForWidth()))
        self.button_addSpeaker.setSizePolicy(sizePolicy)

self.button_addSpeaker.setLayoutDirection(QtCore.Qt.LayoutDire
ction.RightToLeft)

```

```

self.button_addSpeaker.setObjectName("button_addSpeaker")
    self.verticalLayout.addWidget(self.button_addSpeaker)
    self.stackedWidget.addWidget(self.page_addSpeaker)
    self.page_addEvent = QtWidgets.QWidget()
    self.page_addEvent.setObjectName("page_addEvent")
    self.verticalLayout_2 = QtWidgets.QVBoxLayout(self.page_addEvent)

self.verticalLayout_2.setObjectName("verticalLayout_2")
    self.label_eventList = QtWidgets.QLabel(parent=self.page_addEvent)

self.label_eventList.setAlignment(QtCore.Qt.AlignmentFlag.AlignCenter)
    self.label_eventList.setObjectName("label_eventList")
    self.verticalLayout_2.addWidget(self.label_eventList)
    self.table_events = QtWidgets.QTableWidget(parent=self.page_addEvent)
        self.table_events.setObjectName("table_events")
        self.table_events.setColumnCount(3)
        self.table_events.setRowCount(0)
        item = QtWidgets.QTableWidgetItem()
        self.table_events.setHorizontalHeaderItem(0, item)
        item = QtWidgets.QTableWidgetItem()
        self.table_events.setHorizontalHeaderItem(1, item)
        item = QtWidgets.QTableWidgetItem()
        self.table_events.setHorizontalHeaderItem(2, item)
        self.verticalLayout_2.addWidget(self.table_events)
    self.line_2 = QtWidgets.QFrame(parent=self.page_addEvent)

self.line_2.setFrameShape(QtWidgets.QFrame.Shape.HLine)

self.line_2 setFrameShadow(QtWidgets.QFrame.Shadow.Sunken)

```

```

        self.line_2.setObjectName("line_2")
        self.verticalLayout_2.addWidget(self.line_2)
        self.label_addEvent = =
QtWidgets.QLabel(parent=self.page_addEvent)
        self.label_addEvent.setObjectName("label_addEvent")
        self.verticalLayout_2.addWidget(self.label_addEvent)
        self.formLayout_2 = QtWidgets.QFormLayout()
        self.formLayout_2.setObjectName("formLayout_2")
        self.label_eventTitle = =
QtWidgets.QLabel(parent=self.page_addEvent)

self.label_eventTitle.setObjectName("label_eventTitle")
        self.formLayout_2.setWidget(0,
QtWidgets.QFormLayout.ItemRole.LabelRole,
self.label_eventTitle)
        self.label_2 = =
QtWidgets.QLabel(parent=self.page_addEvent)
        self.label_2.setObjectName("label_2")
        self.formLayout_2.setWidget(1,
QtWidgets.QFormLayout.ItemRole.LabelRole, self.label_2)
        self.line_title = =
QtWidgets.QLineEdit(parent=self.page_addEvent)
        self.line_title.setObjectName("line_title")
        self.formLayout_2.setWidget(0,
QtWidgets.QFormLayout.ItemRole.FieldRole, self.line_title)
        self.line_date = =
QtWidgets.QLineEdit(parent=self.page_addEvent)
        self.line_date.setObjectName("line_date")
        self.formLayout_2.setWidget(1,
QtWidgets.QFormLayout.ItemRole.FieldRole, self.line_date)
        self.verticalLayout_2.setLayout(self.formLayout_2)
        self.button_addEvent = =
QtWidgets.QPushButton(parent=self.page_addEvent)

```

```

        sizePolicy =  

QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Policy.Maximum,  

QtWidgets.QSizePolicy.Policy.Fixed)  

        sizePolicy.setHorizontalStretch(0)  

        sizePolicy.setVerticalStretch(0)  
  

sizePolicy.setHeightForWidth(self.button_addEvent.sizePolicy()  
.hasHeightForWidth())  

        self.button_addEvent.setSizePolicy(sizePolicy)  
  

self.button_addEvent.setLayoutDirection(QtCore.Qt.LayoutDirection.RightToLeft)  

        self.button_addEvent.setDefault(False)  

        self.button_addEvent.setObjectName("button_addEvent")  

        self.verticalLayout_2.addWidget(self.button_addEvent)  

        self.stackedWidget.addWidget(self.page_addEvent)  

        self.button_back = QtWidgets.QPushButton(parent=Form)  

        self.button_back.setEnabled(True)  

        self.button_back.setGeometry(QtCore.QRect(20, 10, 221,  
49))  

        sizePolicy =  

QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Policy.Maximum,  

QtWidgets.QSizePolicy.Policy.Fixed)  

        sizePolicy.setHorizontalStretch(0)  

        sizePolicy.setVerticalStretch(0)  
  

sizePolicy.setHeightForWidth(self.button_back.sizePolicy().has  
HeightForWidth())  

        self.button_back.setSizePolicy(sizePolicy)  

        self.button_back.setFlat(False)  

        self.button_back.setObjectName("button_back")  
  

        self.retranslateUi(Form)  

        self.stackedWidget.setCurrentIndex(0)  

QtCore.QMetaObject.connectSlotsByName(Form)

```

```

def retranslateUi(self, Form):
    _translate = QtCore.QCoreApplication.translate
    Form.setWindowTitle(_translate("Form", "Form"))
    self.button_toggle_widget.setText(_translate("Form",
"To Event List"))

    self.label_speakerList.setText(_translate("Form",
"Speaker List"))

    item = self.table_speakers.horizontalHeaderItem(0)
    item.setText(_translate("Form", "ID"))
    item = self.table_speakers.horizontalHeaderItem(1)
    item.setText(_translate("Form", "Name"))
    item = self.table_speakers.horizontalHeaderItem(2)
    item.setText(_translate("Form", "Code"))
    self.label_addSpeaker.setText(_translate("Form", "Add
New Speaker"))

    self.label_name.setText(_translate("Form", "Name"))
    self.label_code.setText(_translate("Form", "Code"))
    self.button_addSpeaker.setText(_translate("Form", "Add
Speaker"))

    self.label_eventList.setText(_translate("Form", "Event
List"))

    item = self.table_events.horizontalHeaderItem(0)
    item.setText(_translate("Form", "ID"))
    item = self.table_events.horizontalHeaderItem(1)
    item.setText(_translate("Form", "Title"))
    item = self.table_events.horizontalHeaderItem(2)
    item.setText(_translate("Form", "Date"))
    self.label_addEvent.setText(_translate("Form", "Add New
Event"))

    self.label_eventTitle.setText(_translate("Form",
>Title"))

    self.label_2.setText(_translate("Form", "Date"))
    self.button_addEvent.setText(_translate("Form", "Add
event"))

```

```

        self.button_back.setText(_translate("Form", "Back"))

if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    Form = QtWidgets.QWidget()
    ui = Ui_Form()
    ui.setupUi(Form)
    Form.show()
    sys.exit(app.exec())

```

*File addwindow.py* digunakan sebagai *inheritance* untuk *class* baru AddWindow() yang dibuat di app.py. Elemen-elemen yang telah didesain dihubungkan dengan fungsi-fungsi yang berfungsi untuk menambahkan data pada tabel-tabel speakers dan events pada *database*.

Segmen Kode 4.21 Class AddWindow

```

class AddWindow(qtw.QWidget):
    def __init__(self, mainwindow):
        super().__init__()

        # set mainwindow
        self.mainwindow = mainwindow

        # set stuff
        self.ui = aw.Ui_Form()
        self.ui.setupUi(self)
        # whosoever has hands with which to write, let him write!
        self.setWindowTitle("qui      habet      manus      scriendum,
scribat!")

        self.setWindowIcon(qtg.QIcon("asset/images/Solaire.png"))
        self.LoadTables(self.ui.table_speakers)
        self.LoadTables(self.ui.table_events)

```

```

        self.msgBox = qtw.QMessageBox()

self.msgBox.setStandardButtons(qtw.QMessageBox.StandardButton.
Ok)

self.msgBox.setWindowIcon(qtg.QIcon("asset/images/Solaire.png"
))

        # connect stuff here
        self.ui.button_back.clicked.connect(self.GoBack)

self.ui.button_toggle_widget.clicked.connect(self.ToggleWidget
)
        self.ui.button_addEvent.clicked.connect(self.AddEvent)

self.ui.button_addSpeaker.clicked.connect(self.AddSpeaker)

def GoBack(self):
        self.hide()
        self.mainwindow.show()

def ToggleWidget(self):
        if self.ui.stackedWidget.currentIndex() == 0:
            self.ui.stackedWidget.setCurrentIndex(1)
            self.ui.button_toggle_widget.setText("To Speaker
List")
        else:
            self.ui.stackedWidget.setCurrentIndex(0)
            self.ui.button_toggle_widget.setText("To Event
List")

def LoadTables(self, table):
        # fetch data
        dataList = []
        if table.objectName() == "table_speakers":

```

```

        dataList = cursor.list_speakers()
    elif table.objectName() == "table_events":
        dataList = cursor.list_events()
    # put in table
    for i in range(len(dataList)):
        table.insertRow(i)
        for j in range(len(dataList[i])):
            table.setItem(
                i,
                j,
                qtw.QTableWidgetItem(str(dataList[i][j])))

    # resize
    header = table.horizontalHeader()
    header.setSectionResizeMode(
        0, qtw.QHeaderView.ResizeMode.ResizeToContents)
    header.setSectionResizeMode(1,
                               qtw.QHeaderView.ResizeMode.Stretch)
    header.setSectionResizeMode(
        2, qtw.QHeaderView.ResizeMode.ResizeToContents)

def MessageSuccess(self, txt):
    # success
    self.msgBox.setIcon(qtw.QMessageBox.Icon.Information)
    self.msgBox.setText("You dun did it\n{}".format(txt))
    self.msgBox.show()

def MessageFail(self, txt):
    self.msgBox.setIcon(qtw.QMessageBox.Icon.Warning)
    self.msgBox.setText("You dun goofed\n{}".format(txt))
    self.msgBox.show()

def AddEvent(self):
    if self.ui.line_title.text() != "":
        and
self.ui.line_date.text() != "":
        cursor.create_event(self.ui.line_title.text(),

```

```

        self.ui.line_date.text()

    # success
    self.MessageSuccess("Event created")
    # clear lines
    self.ui.line_title.clear()
    self.ui.line_date.clear()
    # reload table
    self.ui.table_events.setRowCount(0)
    self.LoadTables(self.ui.table_events)

else:
    self.MessageFail("Event not created")

def AddSpeaker(self):
    if      self.ui.line_name.text()      !=      ""      and
self.ui.line_code.text()  !=  "":
        cursor.create_speaker(self.ui.line_name.text(),
                               self.ui.line_code.text())
        # success
        self.MessageSuccess("Speaker created")
        # clear lines
        self.ui.line_name.clear()
        self.ui.line_code.clear()
        # reload table
        self.ui.table_speakers.setRowCount(0)
        self.LoadTables(self.ui.table_speakers)
    else:
        self.MessageFail("Speaker not created")

```

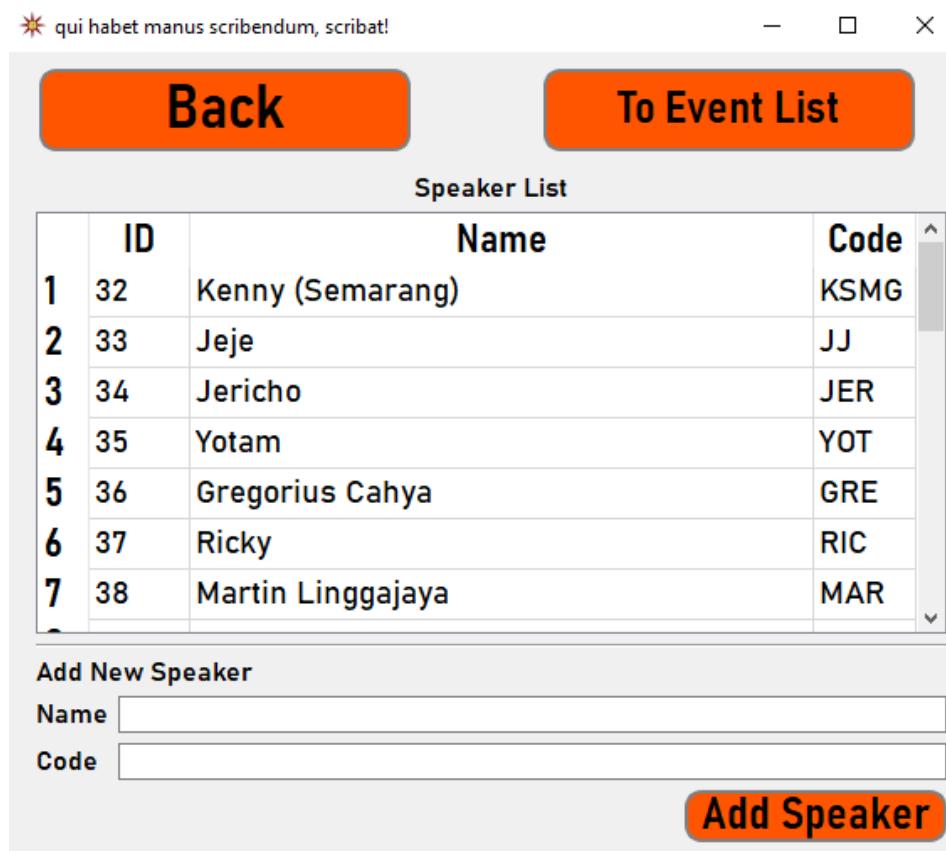
Berikut tabel fungsi-fungsi yang ada dalam class AddWindow dan fungsinya.

Tabel 4.2 Mapping fungsi class AddWindow

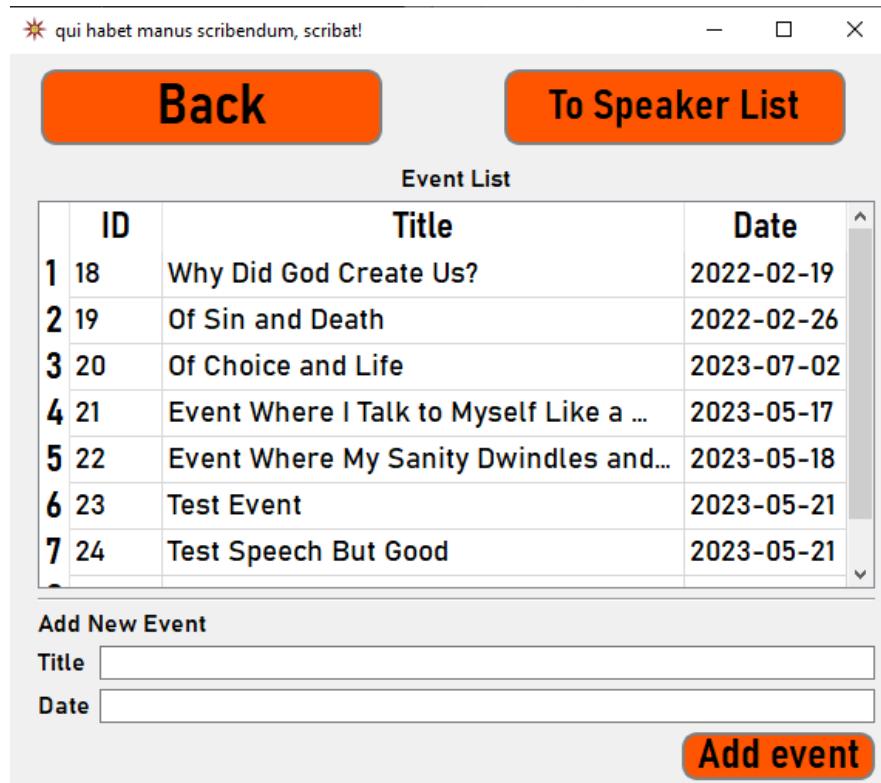
Class	Parameter	Keterangan
-------	-----------	------------

GoBack()	-	Dihubungkan pada tombol "Back". Berfungsi untuk kembali ke MainMenuItem.
ToggleWidget()	-	Untuk mengganti pilihan menu, melakukan toggle antara menambahkan pembicara ("speaker") atau sesi diskusi ("event") baru.
LoadTables()	Obyek tabel yang ada dalam addwindow.py	Memerintahkan obyek DatabaseHandler() untuk mengambil seluruh data pembicara dari tabel speakers atau sesi diskusi dari tabel events, tergantung <i>widget</i> yang sedang aktif sekarang.
MessageSuccess()	Pesan berupa <i>string</i>	Menampilkan <i>message box</i> yang akan dipanggil jika penambahan data berhasil.
MessageFail()	Pesan berupa <i>string</i>	Menampilkan message box yang akan dipanggil jika penambahan data gagal.
AddEvent()	-	Memerintahkan obyek DatabaseHandler() untuk membuat sebuah rekor data baru pada tabel events sesuai dengan data yang dimasukkan pengguna.
AddSpeaker()	-	Memerintahkan obyek DatabaseHandler() untuk membuat sebuah rekor data baru pada tabel

		speakers sesuai dengan masukan pengguna.
--	--	--



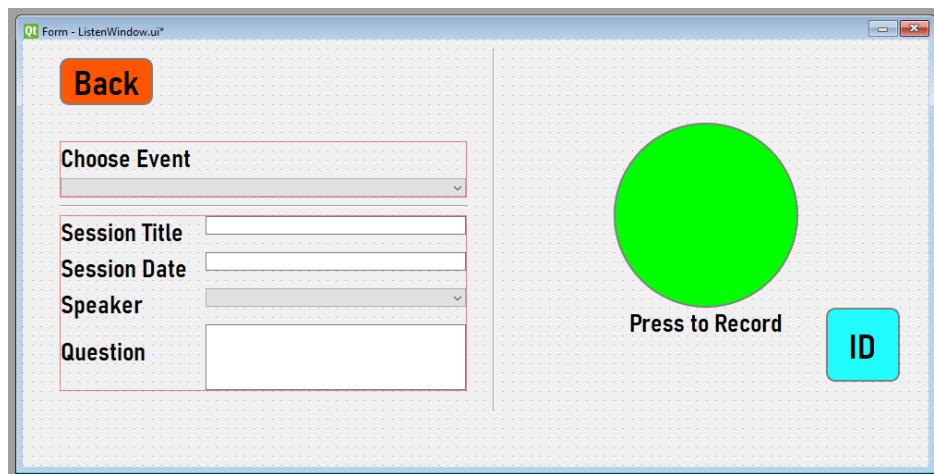
Gambar 4.5 AddWindow untuk menambahkan pembicara baru.



Gambar 4.6 AddWindow untuk menambahkan sesi diskusi baru.

#### 4.3.2.3. ListenWindow

ListenWindow digunakan sebagai sarana pengumpulan rekaman pendapat pembicara dalam sebuah sesi diskusi. Pengguna bisa memilih sesi dan pembicara yang telah terdaftarkan sebagai metadata, dan mulai atau menghentikan rekaman sesuai dengan kebutuhan.



Gambar 4.7 Desain ListenWindow pada Qt Designer

File ListenWindow.ui yang dihasilkan dikonversi menjadi code dalam bahasa Python dengan perintah *command line* berikut ini :

Segmen Kode 4.22 Konversi ListenWindow.ui ke listenwindow.py

```
pyuic6 ui/ListenWindow.ui -x -o objects/listenwindow.py
```

Segmen Kode 4.23 listenwindow.py

```
# Form implementation generated from reading ui file
'ui/ListenWindow.ui'

#
# Created by: PyQt6 UI code generator 6.5.0
#
# WARNING: Any manual changes made to this file will be lost
when pyuic6 is
# run again. Do not edit this file unless you know what you are
doing.

from PyQt6 import QtCore, QtGui, QtWidgets


class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(981, 415)
        Form.setStyleSheet("QLabel#label_title,
#label_subtitle{ \n"
"    \n"
"    font: 63 28pt \"Bahnschrift SemiBold SemiConden\"; \n"
"    color:rgb(255, 85, 0); \n"
"} \n"
"\n"
"QLabel{\n"
"    font: 63 20pt \"Bahnschrift SemiBold SemiConden\"; \n"
"} \n"
"\n"
"QPushButton{\n"
```

```

"      font: 63 28pt \"Bahnschrift SemiBold SemiConden\";\n"
"      background:rgb(255, 85, 0);\n"
"      border: 2px solid gray;\n"
"      border-radius:10px;\n"
"      padding: 0 8px;\n"
"}\n"
"\n"
"QPushButton:hover{\n"
"      background:rgb(255, 140, 0);\n"
"}\n"
"\n"
"QPushButton:pressed{\n"
"      background: rgb(255, 170, 0);\n"
"}\n"
"\n"
"QPushButton#button_record{\n"
"      background:rgb(0, 255, 0);\n"
"      border-radius: 100px;\n"
"}\n"
"\n"
"QPushButton#button_record:checked{\n"
"      background:rgb(255, 0, 0);\n"
"}\n"
"\n"
"QPushButton#button_record:pressed:hover{\n"
"      background:rgb(0, 170, 0);\n"
"}\n"
"\n"
"QPushButton#button_record:checked:pressed:hover{\n"
"      background:rgb(255, 35, 28);\n"
"}\n"
"\n"
"QPushButton#button_toggleLang:checked{\n"
"      \n"
"      background: rgb(32, 252, 255);\n"
"
```

```

"}\n"
"\n"
"QPushButton#button_toggleLang:checked:hover{\n"
"    \n"
"    background: rgb(0, 170, 255);\n"
"}\n"
"\n"
"QPushButton#button_toggleLang:pressed{\n"
"    \n"
"    background: rgb(85, 0, 255);\n"
"}\n"
"\n"
"QPushButton#button_toggleLang{\n"
"    \n"
"    background: rgb(255, 42, 88);\n"
"}\n"
"\n"
"QPushButton#button_toggleLang:hover{\n"
"    \n"
"    background: rgb(255, 0, 0);\n"
"}")\n\n        self.button_record = QtWidgets.QPushButton(parent=Form)\n        self.button_record.setGeometry(QtCore.QRect(640,      90,\n200, 200))\n\n        font = QtGui.QFont()\n        font.setFamily("Bahnschrift SemiBold SemiConden")\n        font.setPointSize(28)\n        font.setBold(False)\n        font.setItalic(False)\n        font.setWeight(7)\n\n        self.button_record.setFont(font)\n        self.button_record.setText("")\n\n        icon = QtGui.QIcon()

```

```

icon.addPixmap(QtGui.QPixmap("ui\\asset/images/mic.png"),
QtGui.QIcon.Mode.Normal, QtGui.QIcon.State.Off)
    self.button_record.setIcon(icon)
    self.button_record.setIconSize(QtCore.QSize(100, 100))
    self.button_record.setCheckable(True)
    self.button_record.setChecked(False)
    self.button_record.setObjectName("button_record")
    self.formLayoutWidget_2 = QtWidgets.QWidget(parent=Form)
        self.formLayoutWidget_2.setGeometry(QtCore.QRect(40,
190, 441, 190))

self.formLayoutWidget_2.setObjectName("formLayoutWidget_2")
    self.formLayout_2 = QtWidgets.QFormLayout(self.formLayoutWidget_2)
        self.formLayout_2.setContentsMargins(0, 0, 0, 0)
        self.formLayout_2.setHorizontalSpacing(20)
        self.formLayout_2.setObjectName("formLayout_2")
        self.label_session = QtWidgets.QLabel(parent=self.formLayoutWidget_2)
            font = QtGui.QFont()
            font.setFamily("Bahnschrift SemiBold SemiConden")
            font.setPointSize(20)
            font.setBold(False)
            font.setItalic(False)
            font.setWeight(7)
            self.label_session.setFont(font)
            self.label_session.setObjectName("label_session")
            self.formLayout_2.setWidget(0,
QtWidgets.QFormLayout.ItemRole.LabelRole, self.label_session)
            self.line_title = QtWidgets.QLineEdit(parent=self.formLayoutWidget_2)
                self.line_title.setReadOnly(True)
                self.line_title.setObjectName("line_title")

```

```

        self.formLayout_2.setWidget(0,
QtWidgets.QFormLayout.ItemRole.FieldRole, self.line_title)
            self.label_date =
QtWidgets.QLabel(parent=self.formLayoutWidget_2)
            font = QtGui.QFont()
            font.setFamily("Bahnschrift SemiBold SemiConden")
            font.setPointSize(20)
            font.setBold(False)
            font.setItalic(False)
            font.setWeight(7)
            self.label_date.setFont(font)
            self.label_date.setObjectName("label_date")
            self.formLayout_2.setWidget(1,
QtWidgets.QFormLayout.ItemRole.LabelRole, self.label_date)
            self.line_date =
QtWidgets.QLineEdit(parent=self.formLayoutWidget_2)
            self.line_date.setReadOnly(True)
            self.line_date.setObjectName("line_date")
            self.formLayout_2.setWidget(1,
QtWidgets.QFormLayout.ItemRole.FieldRole, self.line_date)
            self.label_speakers =
QtWidgets.QLabel(parent=self.formLayoutWidget_2)
            font = QtGui.QFont()
            font.setFamily("Bahnschrift SemiBold SemiConden")
            font.setPointSize(20)
            font.setBold(False)
            font.setItalic(False)
            font.setWeight(7)
            self.label_speakers.setFont(font)
            self.label_speakers.setObjectName("label_speakers")
            self.formLayout_2.setWidget(2,
QtWidgets.QFormLayout.ItemRole.LabelRole, self.label_speakers)
            self.combo_speakerList =
QtWidgets.QComboBox(parent=self.formLayoutWidget_2)

```

```

self.combo_speakerList.setObjectName("combo_speakerList")
    self.formLayout_2.setWidget(2,
QtWidgets.QFormLayout.ItemRole.FieldRole,
self.combo_speakerList)
    self.label_question = =
QtWidgets.QLabel(parent=self.formLayoutWidget_2)
    self.label_question.setObjectName("label_question")
    self.formLayout_2.setWidget(3,
QtWidgets.QFormLayout.ItemRole.LabelRole, self.label_question)
    self.text_question = =
QtWidgets.QTextEdit(parent=self.formLayoutWidget_2)
    self.text_question.setObjectName("text_question")
    self.formLayout_2.setWidget(3,
QtWidgets.QFormLayout.ItemRole.FieldRole, self.text_question)
    self.line = QtWidgets.QFrame(parent=Form)
    self.line.setGeometry(QtCore.QRect(40, 170, 441, 20))
    self.line.setFrameShape(QtWidgets.QFrame.Shape.HLine)

self.line setFrameShadow(QtWidgets.QFrame.Shadow.Sunken)
    self.line.setObjectName("line")
    self.line_2 = QtWidgets.QFrame(parent=Form)
    self.line_2.setGeometry(QtCore.QRect(500, 10, 20, 391))

self.line_2 setFrameShape(QtWidgets.QFrame.Shape.VLine)

self.line_2 setFrameShadow(QtWidgets.QFrame.Shadow.Sunken)
    self.line_2.setObjectName("line_2")
    self.label_record = QtWidgets.QLabel(parent=Form)
    self.label_record.setGeometry(QtCore.QRect(640, 290,
201, 31))

self.label_record.setAlignment(QtCore.Qt.AlignmentFlag.AlignCe
nter)
    self.label_record.setObjectName("label_record")

```

```

        self.button_back = QtWidgets.QPushButton(parent=Form)
        self.button_back.setGeometry(QtCore.QRect(40, 20, 101,
51))

        self.button_back.setObjectName("button_back")
        self.layoutWidget = QtWidgets.QWidget(parent=Form)
        self.layoutWidget.setGeometry(QtCore.QRect(40, 110,
441, 61))

        self.layoutWidget.setObjectName("layoutWidget")
        self.verticalLayout =
QtWidgets.QVBoxLayout(self.layoutWidget)

        self.verticalLayout.setContentsMargins(0, 0, 0, 0)
        self.verticalLayout.setObjectName("verticalLayout")
        self.label_chooseEvent =
QtWidgets.QLabel(parent=self.layoutWidget)

self.label_chooseEvent.setObjectName("label_chooseEvent")
        self.verticalLayout.addWidget(self.label_chooseEvent)
        self.combo_eventList =
QtWidgets.QComboBox(parent=self.layoutWidget)

        self.combo_eventList.setObjectName("combo_eventList")
        self.verticalLayout.addWidget(self.combo_eventList)
        self.button_toggleLang =
QtWidgets.QPushButton(parent=Form)

        self.button_toggleLang.setGeometry(QtCore.QRect(870,
290, 80, 80))

self.button_toggleLang.setStyleSheet("QPushButton#button_toggleLang{\n"
"    \n"
"    font: 22pt \"Algerian\";\n"
"}")

        self.button_toggleLang.setCheckable(True)
        self.button_toggleLang.setChecked(True)

self.button_toggleLang.setObjectName("button_toggleLang")

```

```

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)

    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Form"))
        self.label_session.setText(_translate("Form", "Session Title"))
        self.label_date.setText(_translate("Form", "Session Date"))
        self.label_speakers.setText(_translate("Form", "Speaker"))
        self.label_question.setText(_translate("Form", "Question"))
        self.label_record.setText(_translate("Form", "Press to Record"))
        self.button_back.setText(_translate("Form", "Back"))
        self.label_chooseEvent.setText(_translate("Form", "Choose Event"))
        self.button_toggleLang.setText(_translate("Form", "ID"))

    if __name__ == "__main__":
        import sys
        app = QtWidgets.QApplication(sys.argv)
        Form = QtWidgets.QWidget()
        ui = Ui_Form()
        ui.setupUi(Form)
        Form.show()
        sys.exit(app.exec())

```

Class ListenWindow yang ada di app.py mengimplementasikan listenwindow.py dan menghubungkan fungsi-fungsinya dengan elemen-elemen UI.

#### Segmen Kode 4.24 Class ListenWindow

```
class ListenWindow(qtw.QWidget):
    def __init__(self, mainwindow):
        super().__init__()

        # the mainwindow
        self.mainwindow = mainwindow

        # set stuff here
        self.ui = lw.Ui_Form()
        self.ui.setupUi(self)
        # whosoever has ears for hearing, let him hear!
        self.setWindowTitle("qui habet aurem audendi, audiat!")

        self.setWindowIcon(qtg.QIcon("asset/images/Solaire.png"))
        self.eventID = 0
        self.speakerID = 0
        self.eventList = cursor.list_events()
        self.speakerList = cursor.list_speakers()
        self.LoadComboBox(self.ui.combo_eventList)
        self.LoadComboBox(self.ui.combo_speakerList)
        self.msgBox = qtw.QMessageBox()

        self.msgBox.setStandardButtons(qtw.QMessageBox.StandardButton.Ok)

        self.msgBox.setWindowIcon(qtg.QIcon("asset/images/Solaire.png"))

        self.ui.button_record.setIcon(qtg.QIcon("asset/images/mic.png"))
        self.ui.button_toggleLang.setText("ID")

        # connect stuff here
        self.ui.button_back.clicked.connect(self.GoBack)
```

```

self.ui.button_record.clicked.connect(self.ToggleRecord)

self.ui.combo_eventList.currentIndexChanged.connect(self.ChangeEvent)
    self.ui.combo_speakerList.currentIndexChanged.connect(
        self.ChangeSpeaker)

self.ui.button_toggleLang.clicked.connect(self.ToggleLang)

def GoBack(self):
    self.hide()
    self.mainwindow.show()

def ToggleRecord(self):
    if self.ValidateRecord():
        if self.ui.button_record.isChecked():
            # file path
            filePath = "records/event{}/{}.wav".format(self.ui.combo_eventList.currentData(),
                                                       self.ui.combo_speakerList.currentText().split(" | ")[0] +
                                                       dt.now().strftime("%d%m%Y%H%M%S"))
            # begin record thread
            self.ui.rt = RecorderThread(filePath)
            self.ui.rt.start()
            # change label
            self.ui.label_record.setText("Recording...")
            self.ui.button_toggleLang.setDisabled(True) # disable toggle language
    else:
        # save in database

self.ui.rt.toggle_signal.connect(self.SaveRecording)

```

```

        # stop thread
        self.ui.rt.stop()

        # change label
        self.ui.label_record.setText("Press to Record")

        # success
        self.MessageSuccess("Recording finished.")
        self.ui.button_toggleLang.setDisabled(
            False) # re-enable toggle language

    else:
        self.MessageFail("Fill the speaker and event data
first.")

        self.ui.button_record.setChecked(False)

def ToggleLang(self):
    sender = self.sender()
    if sender.isChecked():
        sender.setText("ID")
    else:
        sender.setText("EN")

def SaveRecording(self, fileName):

cursor.create_record_audio(self.ui.combo_eventList.currentData
(
    ), self.ui.combo_speakerList.currentData(),
self.ui.text_question.toPlainText(), fileName,
self.ui.button_toggleLang.text())

def LoadComboBox(self, comboBox):
    dataList = []
    if comboBox.objectName() == "combo_eventList":
        comboBox.setPlaceholderText("Select Event")
        dataList = self.eventList

```

```

        elif comboBox.objectName() == "combo_speakerList":
            comboBox.setPlaceholderText("Select Speaker")
            dataList = self.speakerList

        for item in dataList:
            if comboBox.objectName() == "combo_eventList":
                comboBox.addItem(item[1], item[0])
            elif comboBox.objectName() == "combo_speakerList":
                comboBox.addItem("{} | {}".format(item[2],
item[1]), item[0])

    def ChangeEvent(self):
        self.eventID = self.ui.combo_eventList.itemData(
            self.ui.combo_eventList.currentIndex())
        event = cursor.get_event("id", self.eventID)
        self.ui.line_title.setText(event[1])
        self.ui.line_date.setText(event[2])

    def ChangeSpeaker(self):
        self.speakerID = self.ui.combo_speakerList.itemData(
            self.ui.combo_speakerList.currentIndex())

    def ValidateRecord(self):
        if self.eventID > 0 and self.speakerID > 0:
            return True
        return False

    def MessageSuccess(self, txt):
        # success
        self.msgBox.setWindowTitle("Good Job!")
        self.msgBox.setIcon(qtw.QMessageBox.Icon.Information)
        self.msgBox.setText("You dun did it\n{}".format(txt))
        self.msgBox.show()

    def MessageFail(self, txt):

```

```

        self.msgBox.setWindowTitle("What the hell, oh my god,
no way")

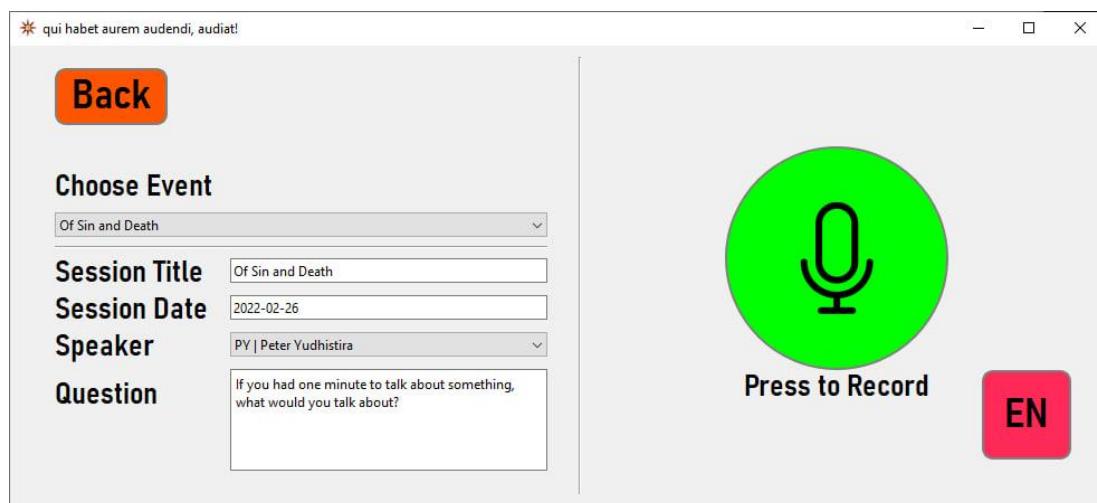
        self.msgBox.setIcon(qtw.QMessageBox.Icon.Warning)
        self.msgBox.setText("You dun goofed\n{}".format(txt))
        self.msgBox.show()
    
```

Berikut tabel fungsi-fungsi yang ada dalam class ListenWindow dan fungsinya.

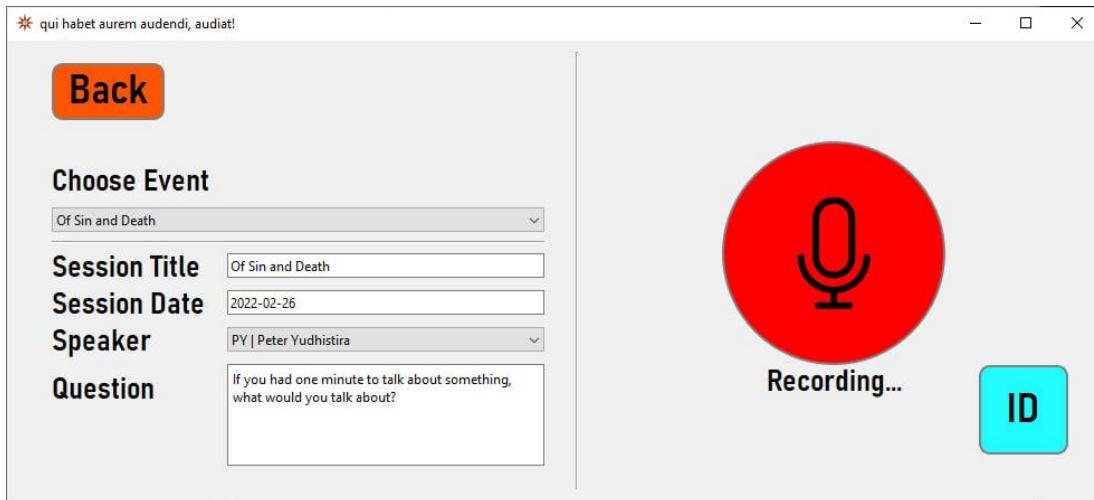
Tabel 4.3 Mapping fungsi class ListenWindow

Fungsi	Input	Keterangan
ToggleRecord()	-	<p>Memulai atau menghentikan rekaman. Jika memulai rekaman, program akan membuat sebuah <i>file</i> dalam direktori sesi diskusi untuk memuat data suara yang sedang direkam. Setelah itu, program akan memulai <i>thread</i> RecorderThread baru untuk merekam suara yang berjalan paralel dengan GUI program.</p> <p>Jika menghentikan rekaman, <i>thread</i> perekam suara akan mengirimkan sinyal berhenti pada program utama, melalui self.SaveRecording(self, fileName:str). Rekor data rekaman akan disimpan dalam <i>database</i> dengan memperhatikan pilihan sesi dan pembicara yang diatur lewat GUI program.</p>
ToggleLang()	-	Mengatur keterangan bahasa yang akan digunakan dalam proses <i>transcribing</i> . Pilihan “EN” berarti

		<p><i>file</i> akan ditranskripsikan dalam Bahasa Inggris, sedangkan pilihan “ID” berarti <i>file</i> akan ditranskripsikan dalam Bahasa Indonesia, lalu diterjemahkan dalam Bahasa Inggris.</p>
SaveRecording()	<i>File path</i> tujuan dalam bentuk <i>string</i> .	Fungsi yang dipanggil ketika proses rekaman yang <i>asynchronous</i> dihentikan. Setelah rekaman selesai, data rekaman akan ditulis dalam <i>file path</i> yang ditentukan, lalu rekord data akan disimpan dalam <i>database</i> oleh obyek DatabaseHandler.



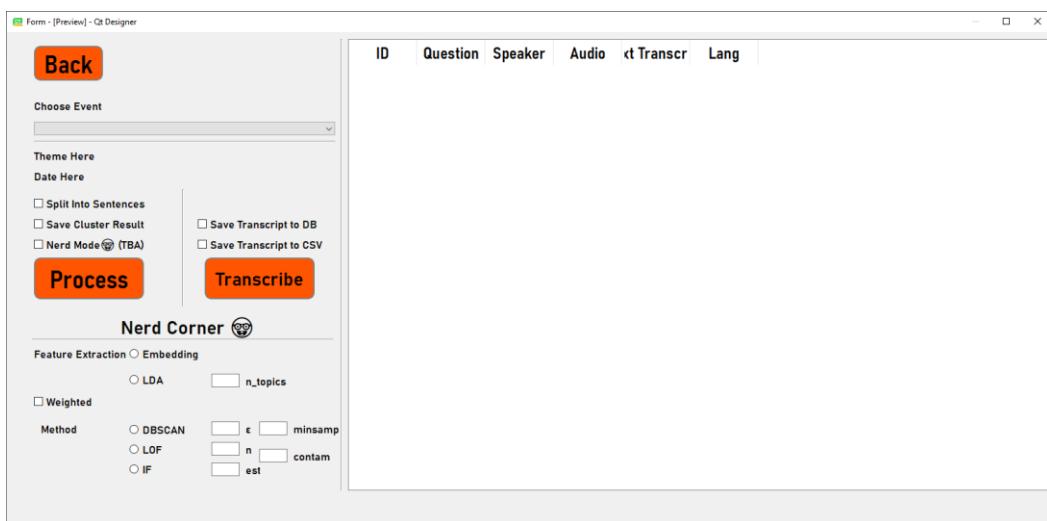
Gambar 4.8 ListenWindow dengan data sesi, pembicara, dan pertanyaan yang sudah diisi



Gambar 4.9 ListenWindow dalam proses rekaman menggunakan pilihan Bahasa Indonesia

#### 4.3.2.4. ProcessWindow

ProcessWindow adalah sebuah *interface* dimana pengguna bisa melihat rekord-rekord data yang telah direkam dalam sebuah sesi. Pada *interface* ini terdapat pilihan untuk melakukan *playback* rekord data audio, melakukan proses *transcribing* atas semua data audio dalam sebuah sesi, dan melakukan deteksi anomali dengan parameter-parameter yang ada. Pilihan “Nerd Mode” diadakan untuk kepentingan penelitian, agar memudahkan penggantian kombinasi parameter untuk mendeteksi anomali sesuai keinginan. Jika pilihan “Nerd Mode” dinonaktifkan, deteksi anomali dilakukan dengan parameter *default* yang akan ditentukan dalam akhir penelitian.



Gambar 4.10 Desain ProcessWindow dalam Qt Designer

File ListenWindow.ui yang dihasilkan dikonversi menjadi code dalam bahasa Python dengan perintah *command line* berikut ini :

Segmen Kode 4.25 Konversi ProcessWindow.ui ke processwindow.py

```
pyuic6 ui/ProcessWindow.ui -x -o objects/processwindow.py
```

```
#  Form implementation generated from reading ui file
'ui/ProcessWindow.ui'

#
# Created by: PyQt6 UI code generator 6.5.0
#
# WARNING: Any manual changes made to this file will be lost
when pyuic6 is
# run again. Do not edit this file unless you know what you are
doing.

from PyQt6 import QtCore, QtGui, QtWidgets


class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(1636, 707)
        Form.setStyleSheet("QLabel#label_title,
#label_subtitle{ \n"
"    \n"
"    font: 63 28pt \"Bahnschrift SemiBold SemiConden\";\n"
"    color:rgb(255, 85, 0);\n"
"}\n"
"\n"
"QPushButton{\n"
"    font: 63 28pt \"Bahnschrift SemiBold SemiConden\";\n"
"    background:rgb(255, 85, 0);\n"
"    border: 2px solid gray;\n"
"    border-radius: 10px;\n"
"    padding: 0 8px;\n"
"}\n"
```

```
"\n"
"QPushButton#button_addSpeaker,      QPushButton#button_addEvent,
QPushButton#button_toggle_widget{\n"
"    font: 63 20pt \"Bahnschrift SemiBold SemiConden\";\n"
"}\n"
"QPushButton:hover{\n"
"    background:rgb(255, 140, 0);\n"
"}\n"
"\n"
"QPushButton:pressed{\n"
"    background: rgb(255, 170, 0);\n"
"}\n"
"\n"
"QPushButton:disabled{\n"
"    background: rgb(150, 150, 150);\n"
"}\n"
"\n"
"QLabel{\n"
"    font: 63 16pt \"Bahnschrift SemiBold\";\n"
"}\n"
"\n"
"QLabel{\n"
"    \n"
"    font: 63 12pt \"Bahnschrift SemiBold\";\n"
"}\n"
"\n"
"QCheckBox{\n"
"    font: 63 12pt \"Bahnschrift SemiBold\";\n"
"}\n"
"\n"
"QTableWidget{\n"
"    font: 14pt \"Bahnschrift\";\n"
"}\n"
"\n"
"QHeaderView{\n"
```

```

"      font: 63 18pt \"Bahnschrift SemiBold SemiConden\";\n"
"}\n"
"\n"
"QLabel#label_NerdMode{\n"
"      font: 63 20pt \"Bahnschrift SemiBold\";\n"
"}\n"
"\n"
"QRadioButton{\n"
"      font: 63 12pt \"Bahnschrift SemiBold\";\n"
"} ")
      self.button_back = QtWidgets.QPushButton(parent=Form)
      self.button_back.setGeometry(QtCore.QRect(40, 20, 101,
51))
      self.button_back.setObjectName("button_back")
      self.layoutWidget = QtWidgets.QWidget(parent=Form)
      self.layoutWidget.setGeometry(QtCore.QRect(40, 90, 441,
61))
      self.layoutWidget.setObjectName("layoutWidget")
      self.verticalLayout =
QtWidgets.QVBoxLayout(self.layoutWidget)
      self.verticalLayout.setContentsMargins(0, 0, 0, 0)
      self.verticalLayout.setObjectName("verticalLayout")
      self.label_chooseEvent =
QtWidgets.QLabel(parent=self.layoutWidget)

self.label_chooseEvent.setObjectName("label_chooseEvent")
      self.verticalLayout.addWidget(self.label_chooseEvent)
      self.combo_eventList =
QtWidgets.QComboBox(parent=self.layoutWidget)
      self.combo_eventList.setObjectName("combo_eventList")
      self.verticalLayout.addWidget(self.combo_eventList)
      self.line = QtWidgets.QFrame(parent=Form)
      self.line.setGeometry(QtCore.QRect(40, 150, 441, 20))
      self.line.setFrameShape(QtWidgets.QFrame.Shape.HLine)

```

```

self.line.setFrameShadow(QtWidgets.QFrame.Shadow.Sunken)
    self.line.setObjectName("line")
    self.line_2 = QtWidgets.QFrame(parent=Form)
    self.line_2.setGeometry(QtCore.QRect(480, 10, 20, 661))

self.line_2.setFrameShape(QtWidgets.QFrame.Shape.VLine)

self.line_2.setFrameShadow(QtWidgets.QFrame.Shadow.Sunken)
    self.line_2.setObjectName("line_2")
    self.table_recordData =
QtWidgets.QTableWidget(parent=Form)
    self.table_recordData.setGeometry(QtCore.QRect(500,
10, 1121, 661))
    sizePolicy =
QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Policy.Expanding,
QtWidgets.QSizePolicy.Policy.Expanding)
    sizePolicy.setHorizontalStretch(0)
    sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.table_recordData.sizePolicy(
).hasHeightForWidth()))
    self.table_recordData.setSizePolicy(sizePolicy)

self.table_recordData.setObjectName("table_recordData")
    self.table_recordData.setColumnCount(6)
    self.table_recordData.setRowCount(0)
    item = QtWidgets.QTableWidgetItem()
    self.table_recordData.setHorizontalHeaderItem(0, item)
    item = QtWidgets.QTableWidgetItem()
    self.table_recordData.setHorizontalHeaderItem(1, item)
    item = QtWidgets.QTableWidgetItem()
    self.table_recordData.setHorizontalHeaderItem(2, item)
    item = QtWidgets.QTableWidgetItem()
    self.table_recordData.setHorizontalHeaderItem(3, item)

```

```

        item = QtWidgets.QTableWidgetItem()
        self.table_recordData.setHorizontalHeaderItem(4, item)
        item = QtWidgets.QTableWidgetItem()
        self.table_recordData.setHorizontalHeaderItem(5, item)
        self.label_theme = QtWidgets.QLabel(parent=Form)
        self.label_theme.setGeometry(QtCore.QRect(40, 170, 441,
21))
        self.label_theme.setObjectName("label_theme")
        self.label_date = QtWidgets.QLabel(parent=Form)
        self.label_date.setGeometry(QtCore.QRect(40, 200, 441,
21))
        self.label_date.setObjectName("label_date")
        self.button_process =
QtWidgets.QPushButton(parent=Form)
        self.button_process.setGeometry(QtCore.QRect(40, 330,
161, 61))
        self.button_process.setObjectName("button_process")
        self.line_3 = QtWidgets.QFrame(parent=Form)
        self.line_3.setGeometry(QtCore.QRect(250, 230, 16,
171))

self.line_3.setFrameShape(QtWidgets.QFrame.Shape.VLine)

self.line_3.setFrameShadow(QtWidgets.QFrame.Shadow.Sunken)
        self.line_3.setObjectName("line_3")
        self.check_isSplit = QtWidgets.QCheckBox(parent=Form)
        self.check_isSplit.setGeometry(QtCore.QRect(40, 240,
181, 21))
        self.check_isSplit.setObjectName("check_isSplit")
        self.check_saveResult =
QtWidgets.QCheckBox(parent=Form)
        self.check_saveResult.setGeometry(QtCore.QRect(40,
270, 181, 21))

self.check_saveResult.setObjectName("check_saveResult")

```

```

        self.check_nerdMode = QtWidgets.QCheckBox(parent=Form)
        self.check_nerdMode.setEnabled(True)
        self.check_nerdMode.setGeometry(QtCore.QRect(40,    300,
181, 21))
            self.check_nerdMode.setObjectName("check_nerdMode")
            self.button_transcribe_all =
QtWidgets.QPushButton(parent=Form)

self.button_transcribe_all.setGeometry(QtCore.QRect(290,    330,
161, 61))

font = QtGui.QFont()
font.setFamily("Bahnschrift SemiBold")
font.setPointSize(20)
font.setBold(False)
font.setItalic(False)
font.setWeight(7)
self.button_transcribe_all.setFont(font)
self.button_transcribe_all.setStyleSheet("font: 63 20pt
\"Bahnschrift SemiBold\";")

self.button_transcribe_all.setObjectName("button_transcribe_all")
            self.check_saveTranscriptToCSV =
QtWidgets.QCheckBox(parent=Form)

self.check_saveTranscriptToCSV.setGeometry(QtCore.QRect(280,
300, 191, 21))

self.check_saveTranscriptToCSV.setObjectName("check_saveTranscriptToCSV")
            self.check_saveTranscriptToDatabase =
QtWidgets.QCheckBox(parent=Form)

self.check_saveTranscriptToDatabase.setGeometry(QtCore.QRect(2
80, 270, 191, 21))

```

```

self.check_saveTranscriptToDatabase.setObjectName("check_saveT
ranscriptToDatabase")

    self.label_NerdMode = QtWidgets.QLabel(parent=Form)
    self.label_NerdMode.setGeometry(QtCore.QRect(50, 410,
431, 41))

self.label_NerdMode.setAlignment(QtCore.Qt.AlignmentFlag.Align
Center)

    self.label_NerdMode.setObjectName("label_NerdMode")
    self.line_4 = QtWidgets.QFrame(parent=Form)
    self.line_4.setGeometry(QtCore.QRect(37, 440, 441, 20))

self.line_4.setFrameShape(QtWidgets.QFrame.Shape.HLine)

self.line_4.setFrameShadow(QtWidgets.QFrame.Shadow.Sunken)
    self.line_4.setObjectName("line_4")
    self.label_FeatureExtraction =
QtWidgets.QLabel(parent=Form)

self.label_FeatureExtraction.setGeometry(QtCore.QRect(40, 460,
141, 21))

self.label_FeatureExtraction.setAlignment(QtCore.Qt.AlignmentF
lag.AlignLeading|QtCore.Qt.AlignmentFlag.AlignLeft|QtCore.Qt.A
lignmentFlag.AlignVCenter)

self.label_FeatureExtraction.setObjectName("label_FeatureExtra
ction")

    self.label_Method = QtWidgets.QLabel(parent=Form)
    self.label_Method.setGeometry(QtCore.QRect(50, 570,
131, 21))

self.label_Method.setAlignment(QtCore.Qt.AlignmentFlag.AlignLe

```

```

ading|QtCore.Qt.AlignmentFlag.AlignLeft|QtCore.Qt.AlignmentFlag.AlignVCenter)
        self.label_Method.setObjectName("label_Method")
        self.check_isWeighted = QtWidgets.QCheckBox(parent=Form)
        self.check_isWeighted.setGeometry(QtCore.QRect(40,
530, 131, 21))

self.check_isWeighted.setObjectName("check_isWeighted")
        self.line_nTopics = QtWidgets.QLineEdit(parent=Form)
        self.line_nTopics.setGeometry(QtCore.QRect(300,      500,
41, 20))
        self.line_nTopics.setObjectName("line_nTopics")
        self.line_epsilon = QtWidgets.QLineEdit(parent=Form)
        self.line_epsilon.setGeometry(QtCore.QRect(300,      570,
41, 20))
        self.line_epsilon.setObjectName("line_epsilon")
        self.line_minSamp = QtWidgets.QLineEdit(parent=Form)
        self.line_minSamp.setGeometry(QtCore.QRect(370,      570,
41, 20))
        self.line_minSamp.setObjectName("line_minSamp")
        self.label = QtWidgets.QLabel(parent=Form)
        self.label.setGeometry(QtCore.QRect(350, 500, 81, 21))
        self.label.setObjectName("label")
        self.label_2 = QtWidgets.QLabel(parent=Form)
        self.label_2.setGeometry(QtCore.QRect(350,      570,      21,
21))
        self.label_2.setObjectName("label_2")
        self.label_3 = QtWidgets.QLabel(parent=Form)
        self.label_3.setGeometry(QtCore.QRect(420,      570,      71,
21))
        self.label_3.setObjectName("label_3")
        self.verticalLayoutWidget =
QtWidgets.QWidget(parent=Form)

```

```
self.verticalLayoutWidget.setGeometry(QtCore.QRect(180,      450,
121, 80))

self.verticalLayoutWidget.setObjectName("verticalLayoutWidget")
)
    self.verticalLayout_2 = QtWidgets.QVBoxLayout(self.verticalLayoutWidget)
    self.verticalLayout_2.setContentsMargins(0, 0, 0, 0)

self.verticalLayout_2.setObjectName("verticalLayout_2")
    self.radio_EMBED =
QtWidgets.QRadioButton(parent=self.verticalLayoutWidget)
    self.radio_EMBED.setObjectName("radio_EMBED")
    self.verticalLayout_2.addWidget(self.radio_EMBED)
    self.radio_LDA =
QtWidgets.QRadioButton(parent=self.verticalLayoutWidget)
    self.radio_LDA.setObjectName("radio_LDA")
    self.verticalLayout_2.addWidget(self.radio_LDA)
    self.verticalLayoutWidget_2 =
QtWidgets.QWidget(parent=Form)

self.verticalLayoutWidget_2.setGeometry(QtCore.QRect(180, 570,
141, 83))

self.verticalLayoutWidget_2.setObjectName("verticalLayoutWidget_2")
    self.verticalLayout_4 =
QtWidgets.QVBoxLayout(self.verticalLayoutWidget_2)
    self.verticalLayout_4.setContentsMargins(0, 0, 0, 0)

self.verticalLayout_4.setObjectName("verticalLayout_4")
    self.radio_DBSCAN =
QtWidgets.QRadioButton(parent=self.verticalLayoutWidget_2)
    self.radio_DBSCAN.setObjectName("radio_DBSCAN")
```

```

        self.verticalLayout_4.addWidget(self.radio_DBSCAN)
        self.radio_LOF = QtWidgets.QRadioButton(parent=self.verticalLayoutWidget_2)
        self.radio_LOF.setObjectName("radio_LOF")
        self.verticalLayout_4.addWidget(self.radio_LOF)
        self.radio_IF = QtWidgets.QRadioButton(parent=self.verticalLayoutWidget_2)
        self.radio_IF.setObjectName("radio_IF")
        self.verticalLayout_4.addWidget(self.radio_IF)
        self.line_neighbors = QtWidgets.QLineEdit(parent=Form)
        self.line_neighbors.setGeometry(QtCore.QRect(300, 600, 41, 20))
        self.line_neighbors.setObjectName("line_neighbors")
        self.line_estimators = QtWidgets.QLineEdit(parent=Form)
        self.line_estimators.setGeometry(QtCore.QRect(300, 630, 41, 20))
        self.line_estimators.setObjectName("line_estimators")
        self.label_4 = QtWidgets.QLabel(parent=Form)
        self.label_4.setGeometry(QtCore.QRect(350, 600, 16, 21))
        self.label_4.setObjectName("label_4")
        self.label_5 = QtWidgets.QLabel(parent=Form)
        self.label_5.setGeometry(QtCore.QRect(350, 630, 21, 21))
        self.label_5.setObjectName("label_5")
        self.line_contamination = QtWidgets.QLineEdit(parent=Form)
        self.line_contamination.setGeometry(QtCore.QRect(370, 610, 41, 20))

self.line_contamination.setObjectName("line_contamination")
        self.label_6 = QtWidgets.QLabel(parent=Form)
        self.label_6.setGeometry(QtCore.QRect(420, 610, 71, 21))
        self.label_6.setObjectName("label_6")

```

```

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)

    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Form"))
        self.button_back.setText(_translate("Form", "Back"))
        self.label_chooseEvent.setText(_translate("Form",
"Choose Event"))

        item = self.table_recordData.horizontalHeaderItem(0)
        item.setText(_translate("Form", "ID"))
        item = self.table_recordData.horizontalHeaderItem(1)
        item.setText(_translate("Form", "Question"))
        item = self.table_recordData.horizontalHeaderItem(2)
        item.setText(_translate("Form", "Speaker"))
        item = self.table_recordData.horizontalHeaderItem(3)
        item.setText(_translate("Form", "Audio"))
        item = self.table_recordData.horizontalHeaderItem(4)
        item.setText(_translate("Form", "Text Transcript"))
        item = self.table_recordData.horizontalHeaderItem(5)
        item.setText(_translate("Form", "Lang"))
        self.label_theme.setText(_translate("Form", "Theme
Here"))
        self.label_date.setText(_translate("Form", "Date
Here"))
        self.button_process.setText(_translate("Form",
"Process"))
        self.check_isSplit.setText(_translate("Form", "Split
Into Sentences"))
        self.check_saveResult.setText(_translate("Form", "Save
Cluster Result"))
        self.check_nerdMode.setText(_translate("Form", "Nerd
Mode (TBA)"))

```

```

        self.button_transcribe_all.setText(_translate("Form",
"Transcribe"))

self.check_saveTranscriptToCSV.setText(_translate("Form",
"Save Transcript to CSV"))

self.check_saveTranscriptToDatabase.setText(_translate("Form",
"Save Transcript to DB"))

    self.label_NerdMode.setText(_translate("Form", "Nerd
Corner 😎"))

self.label_FeatureExtraction.setText(_translate("Form",
"Feature Extraction"))

    self.label_Method.setText(_translate("Form",
"Method"))

    self.check_isWeighted.setText(_translate("Form",
"Weighted"))

        self.label.setText(_translate("Form", "n_topics"))
        self.label_2.setText(_translate("Form", "\u03b5"))
        self.label_3.setText(_translate("Form", "minsamp"))
        self.radio_EMBED.setText(_translate("Form",
"Embedding"))

        self.radio_LDA.setText(_translate("Form", "LDA"))
        self.radio_DBSCAN.setText(_translate("Form",
"DBSCAN"))

        self.radio_LOF.setText(_translate("Form", "LOF"))
        self.radio_IF.setText(_translate("Form", "IF"))
        self.label_4.setText(_translate("Form", "n"))
        self.label_5.setText(_translate("Form", "est"))
        self.label_6.setText(_translate("Form", "contam"))

if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)

```

```

Form = QtWidgets.QWidget()
ui = Ui_Form()
ui.setupUi(Form)
Form.show()
sys.exit(app.exec())

```

Class ProcessWindow yang ada di app.py mengimplementasikan processwindow.py dan menghubungkan fungsi-fungsinya dengan elemen-elemen UI.

#### Segmen Kode 4.26 Class ProcessWindow

```

class ProcessWindow(qtw.QWidget):
    def __init__(self, mainwindow) -> None:
        super().__init__()

        # set mainwindow
        self.mainwindow = mainwindow

        # set stuff
        self.ui = pw.Ui_Form()
        self.ui.setupUi(self)
        self.eventList = cursor.list_events()
        self.audioPlayer = qtm.QMediaPlayer()
        self.audioOutput = qtm.QAudioOutput()
        self.audioPlayer.setAudioOutput(self.audioOutput)
        self.audioOutput.setVolume(100)
        self.presentWindow = None # Funni little workaround
        # whosoever has power over the machine, let him
discover!
        self.setWindowTitle("qui habet potentia super machina,
comperiat!")

        self.setWindowIcon(qtg.QIcon("asset/images/Solaire.png"))
        self.LoadComboBox(self.ui.combo_eventList)

```

```

self.ui.check_saveTranscriptToDatabase.setChecked(True)
    # we disable this for now.

self.ui.check_saveTranscriptToDatabase.setDisabled(True)
    self.ui.button_process.setDisabled(True)  # will only
enable after loading a table.
    self.ui.button_transcribe_all.setDisabled(True)
    self.ui.check_nerdMode.setChecked(True)
    self.ui.featureExtractionGroup = qtw.QButtonGroup(self)

self.ui.featureExtractionGroup.addButton(self.ui.radio_EMBED)

self.ui.featureExtractionGroup.addButton(self.ui.radio_LDA)
    self.ui.anomalyDetectionGroup = qtw.QButtonGroup(self)

self.ui.anomalyDetectionGroup.addButton(self.ui.radio_DBSCAN)

self.ui.anomalyDetectionGroup.addButton(self.ui.radio_LOF)

self.ui.anomalyDetectionGroup.addButton(self.ui.radio_IF)
    # for now, disable LOF and IF
    # self.ui.radio_LOF.setDisabled(True)
    # self.ui.radio_IF.setDisabled(True)
    self.msgBox = qtw.QMessageBox()

self.msgBox.setStandardButtons(qtw.QMessageBox.StandardButton.
Ok)

self.msgBox.setWindowIcon(qtg.QIcon("asset/images/Solaire.png"))
    self.buttonPlayList = []

    # connect stuff here
    self.ui.button_back.clicked.connect(self.GoBack)

```

```

self.ui.combo_eventList.currentIndexChanged.connect(self.ChangeEvent)

    # i forgot the existence of lambda ENTIRELY.
    self.ui.button_process.clicked.connect(
        lambda: self.StartProcessing())

self.ui.button_transcribe_all.clicked.connect(self.TranscribeAll)

self.ui.check_nerdMode.toggled.connect(self.ToggleNerdMode)

def GoBack(self):
    self.hide()
    self.mainwindow.show()

def LoadComboBox(self, comboBox):
    comboBox.setPlaceholderText("Select Event")
    for item in self.eventList:
        comboBox.addItem(item[1], item[0])

def ChangeEvent(self):
    eventID = self.ui.combo_eventList.currentData()
    currentEvent = cursor.get_event("id", eventID)
    self.LoadTables(self.ui.table_recordData, eventID)
    self.ui.label_theme.setText(str(currentEvent[1]))
    self.ui.label_date.setText(str(currentEvent[2]))

def LoadTables(self, table, eventID):
    # fetch data
    self.dataList = cursor.get_recordDataJoined("event_id",
eventID)
    # clear table first
    table.clearContents()
    table.setRowCount(0)

```

```

# put in table

for i in range(len(self.dataList)):
    table.insertRow(i)
    table.setItem(i, 0, qtw.QTableWidgetItem(
        str(self.dataList[i][0]))) # id
    table.setItem(i, 1, qtw.QTableWidgetItem(
        str(self.dataList[i][4]))) # question
    table.setItem(i, 2, qtw.QTableWidgetItem(
        str(self.dataList[i][3]))) # speaker

    # make audio button. if audio path doesn't exist,
    disable it.

    self.button_play = qtw.QPushButton("▶".format(i),
self)

    self.button_play.setObjectName(str(self.dataList[i][5]))
    self.button_play.setCheckable(True)
    self.button_play.clicked.connect(
        lambda
            ch,
filePath=self.button_play.objectName():
self.TogglePlayRecord(filePath))

    if self.dataList[i][5] == "":
        self.button_play.setDisabled(True)
        self.button_play.setText("🔇")
    self.buttonPlayList.append(self.button_play)
    table.setCellWidget(i, 3, self.button_play)

    table.setItem(i, 4, qtw.QTableWidgetItem(
        str(self.dataList[i][6]))) # text
    table.setItem(i, 5, qtw.QTableWidgetItem(
        str(self.dataList[i][7]))) # language code

# resize

table.horizontalHeader().setSectionResizeMode(
    1, qtw.QHeaderView.ResizeMode.Stretch)
table.horizontalHeader().setSectionResizeMode(

```

```

        4, qtw.QHeaderView.ResizeMode.Stretch)

# do this AFTER stretching.

self.ui.table_recordData.resizeRowsToContents()

# enable the buttons

self.ui.button_process.setDisabled(False)
self.ui.button_transcribe_all.setDisabled(False)

# toggle the params. I will surely regret this later.

def ToggleNerdMode(self):
    sender = self.sender()
    if sender.isChecked():
        self.ui.radio_EMBED.setDisabled(False)
        self.ui.radio_LDA.setDisabled(False)
        self.ui.radio_DBSCAN.setDisabled(False)
        self.ui.radio_LOF.setDisabled(False)
        self.ui.radio_IF.setDisabled(False)
        self.ui.check_isWeighted.setDisabled(False)
        self.ui.line_epsilon.setDisabled(False)
        self.ui.line_minSamp.setDisabled(False)
        self.ui.line_nTopics.setDisabled(False)
    else:
        self.ui.radio_EMBED.setDisabled(True)
        self.ui.radio_LDA.setDisabled(True)
        self.ui.radio_DBSCAN.setDisabled(True)
        self.ui.radio_LOF.setDisabled(True)
        self.ui.radio_IF.setDisabled(True)
        self.ui.check_isWeighted.setDisabled(True)
        self.ui.line_epsilon.setDisabled(True)
        self.ui.line_minSamp.setDisabled(True)
        self.ui.line_nTopics.setDisabled(True)

def StartProcessing(self):
    currentEventID = self.ui.combo_eventList.currentData()
    ad.SetDFFromDB(eventID=currentEventID,

```

```

splitBySentences=self.ui.check_isSplit.isChecked()

    # i'll implement nerd mode tonight... today.

    nerdOK = True

    if self.ui.check_nerdMode.isChecked():

        if self.ValidateNerdModeParams():

            featureExtractor = self.ui.featureExtractionGroup.checkedButton().text()

                ad.SetFeatureExtractionParam("method",
featureExtractor)

                    anomalyDetector = self.ui.anomalyDetectionGroup.checkedButton().text()

                        ad.SetAnomalyDetectionParam("algorithm",
anomalyDetector)

                    isWeighted = self.ui.check_isWeighted.isChecked()

                        ad.SetFeatureExtractionParam("weighted",
isWeighted)

                    if self.ui.line_epsilon.text() != '':
                        epsilon = float(self.ui.line_epsilon.text())

                            ad.SetAnomalyDetectionParam("epsilon",
epsilon)

                    if self.ui.line_minSamp.text() != '':
                        minSamp = int(self.ui.line_minSamp.text())
                        ad.SetAnomalyDetectionParam("minsamp",
minSamp)

                    if self.ui.line_neighbors.text() != '':
                        nNeighbors = int(self.ui.line_neighbors.text())

```

```

        ad.SetAnomalyDetectionParam("neighbors",
nNeighbors)

        if self.ui.line_contamination.text() != '':
            contaminationRate =
float(self.ui.line_contamination.text())

ad.SetAnomalyDetectionParam("contamination",
contaminationRate)

        if self.ui.line_estimators.text() != '':
            nEstimators =
int(self.ui.line_estimators.text())
            ad.SetAnomalyDetectionParam("estimators",
nEstimators)

        if self.ui.line_nTopics.text() != '':
            LDAnTopic =
int(self.ui.line_nTopics.text())
            ad.SetFeatureExtractionParam("n_topics",
LDAnTopic)

            print(ad.FeatureExtractionParams)
            print(ad.AnomalyDetectionParams)
            myResult =
ad.GetAnomalies(isReturnSeparate=False)

        else:
            nerdOK = False

        else:
            print(ad.FeatureExtractionParams)
            print(ad.AnomalyDetectionParams)
            myResult = ad.GetAnomalies(isReturnSeparate=False)

if nerdOK:
    self.presentWindow = PresentWindow(
        myResult, self.ui.label_theme.text(), self)
    self.presentWindow.show()

```

```

def ValidateNerdModeParams(self):
    # we will use the elimination method as we have learned
    # in that god-forsaken place.
    if not (self.ui.featureExtractionGroup.checkedButton()
            and self.ui.anomalyDetectionGroup.checkedButton()):
        self.MessageFail("one or both of the radio button
                          groups are unchecked.")
    return False
else:
    # in the case of LDA, make sure the nTopics is
    # filled.
    if
self.ui.featureExtractionGroup.checkedButton().text() == "LDA":
        if not self.ui.line_nTopics.text():
            self.MessageFail("You doing LDA without the
                             number of topics")
        return False
    else:
        # in the case of DBSCAN
        if
self.ui.anomalyDetectionGroup.checkedButton().text() == "DBSCAN":
            # make sure that epsilon and minsamp is
            # filled.
            if not (self.ui.line_epsilon.text() and
                    self.ui.line_minSamp.text()):
                self.MessageFail("You      doing      DBSCAN
                                 without the epsilon and/or minsamp")
            return False
        # in the case of LOF
        elif
self.ui.anomalyDetectionGroup.checkedButton().text() == "LOF":
            if not (self.ui.line_contamination.text()
                    and self.ui.line_neighbors.text()):

```

```

                self.MessageFail("You doing LOF without
the contamination and/or n_neighbors")
                    return False
                # in the case of IF
                elif
self.ui.anomalyDetectionGroup.checkedButton().text() == "IF":
                    if      not(self.ui.line_contamination.text()
and self.ui.line_estimators.text()):
                        self.MessageFail("You doing IF without
the contamination and/or n_estimators")
                        return False
                    return True

def MessageFail(self, txt):
    self.msgBox.setIcon(qtw.QMessageBox.Icon.Warning)
    self.msgBox.setText("You dun goofed\n{}".format(txt))
    self.msgBox.show()

def TranscribeAll(self):
    tttt = []  # i am funny
    for row in range(self.ui.table_recordData.rowCount()):
        button  = self.ui.table_recordData.cellWidget(row,
3)
        if button.objectName() != "":  # goofy condition to
stand-in for "does this object have an audio file path"
            # i'm about to get even funnier
            tttt.append((row, button.objectName(),
self.ui.table_recordData.item(row, 5).text()))
            print("the list contains : {}".format(tttt))
            # pass the value over to a TTTThread then let it run.
            self.tttt = TurnToTextinatorThread()
            self.tttt.setFilePath(tttt)

self.tttt.transcribe_signal.connect(self.GetTranscript)

```

```

        self.tttt.start()
        # to prevent unwanted consequences...
        self.ui.button_back.setDisabled(True)
        self.ui.button_process.setDisabled(True)
        self.ui.button_transcribe_all.setDisabled(True)

        # TranscribeAll's thread will run this after transcription
        is finished.

        def GetTranscript(self, transcriptResult: list):
            print("im here now??")
            print(transcriptResult)
            # display it and write to DB and/or CSV if needed.
            for row in range(len(transcriptResult)):
                self.ui.table_recordData.setItem(
                    int(transcriptResult[row][0]),           4,
                qtw.QTableWidgetItem(transcriptResult[row][2]))
                if
            self.ui.check_saveTranscriptToDatabase.isChecked():
                print("we are updating the db with : {} ||"
{ }).format(
                    self.ui.table_recordData.item(row,
0).text(), transcriptResult[row][2]))

            cursor.update_recordData_text(ID=int(self.ui.table_recordData.
item(transcriptResult[row][0], 0).text()), text=str(
                transcriptResult[row][2])) # at times like
this i miss GORM.

            self.ui.table_recordData.resizeRowsToContents()

            self.ui.table_recordData.horizontalHeader().setSectionResizeMo
de(
                1, qtw.QHeaderView.ResizeMode.Stretch)

```

```

self.ui.table_recordData.horizontalHeader().setSectionResizeMode (
    4, qtw.QHeaderView.ResizeMode.Stretch)
self.ui.button_back.setDisabled(False) # unfreeze
self.ui.button_process.setDisabled(False)
self.ui.button_transcribe_all.setDisabled(False)

def TogglePlayRecord(self, filePath):
    sender = self.sender()
    if sender.isChecked():

        self.audioPlayer.setSource(qtc.QUrl.fromLocalFile(filePath))
        self.audioPlayer.play()
        sender.setText("■")
    else:
        self.audioPlayer.stop()
        sender.setText("▶")

def closeEvent(self, event):
    if self.presentWindow:
        self.presentWindow.close()

```

Berikut tabel yang berisi fungsi-fungsi yang ada dalam Class ProcessWindow beserta fungsinya.

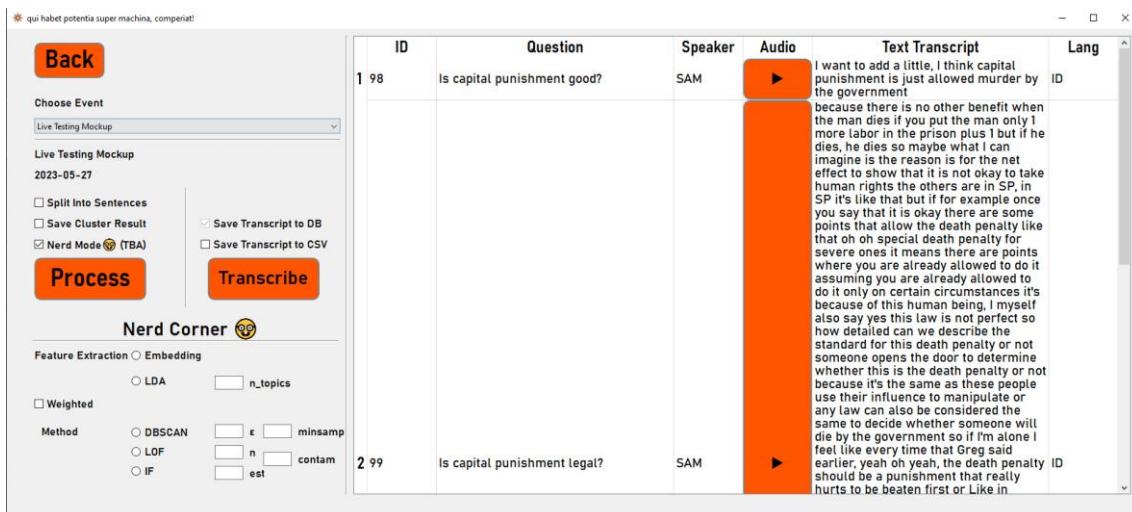
Tabel 4.4 Mapping fungsi class ProcessWindow

Fungsi	Input	Keterangan
GoBack()	-	Dihubungkan dengan tombol “Back”. Berfungsi untuk kembali ke MainMenuWindow.
LoadComboBox()	Obyek self.QComboBox	Mengambil data <i>event</i> dari <i>database</i> dan

		memasukkannya dalam <i>combo box</i> yang bisa dipilih oleh pengguna.
LoadTables()	Obyek <code>self.QTableView, ID event</code> yang didapatkan dari <i>combo box</i> berupa <i>integer</i> .	Mengambil <i>ID</i> yang didapatkan dari <i>combo box</i> , dan memerintahkan <i>DatabaseHandler</i> untuk melakukan <i>query</i> dan mendapatkan tabel yang berisi rekord-rekor data dalam sesi yang ditentukan oleh <i>event ID</i> .
ChangeEvent()	-	Mengganti data <i>title</i> dan <i>date</i> sesi yang sedang dipilih.
ToggleNerdMode()	-	Menyalakan atau mematikan parameter-parameter <i>Nerd Mode</i> .
StartProcessing()	-	Memulai proses <i>anomaly detection</i> pada rekord-rekor data yang dipilih, dengan komponen <i>object</i> dari <i>class AnomalyDetector</i> yang telah dibuat. Jika pilihan “Nerd Mode” aktif, parameter-parameter yang diisi dalam bagian “Nerd Corner” akan dijadikan parameter <i>anomaly detection</i> . Jika pilihan tidak aktif, akan dilakukan <i>anomaly detection</i> dengan parameter default.

ValidateNerdModeParams()	-	Melakukan validasi pada parameter-parameter yang ada di “Nerd Corner” untuk keperluan penelitian.
MessageFail()	<i>error message</i> dalam bentuk string	Menunjukkan error message ketika ValidateNerdModeParams() menemukan sebuah error.
TranscribeAll()	-	Melakukan proses transcribing dan translating pada seluruh rekor data yang dipilih. Proses <i>transcribing</i> dijalankan secara <i>asynchronous</i> oleh obyek TurnToTextinatorThread dan akan memanggil fungsi self.GetTranscript(self, transcriptResult:list) ketika proses <i>transcribing</i> selesai.
GetTranscript()	Hasil dari transkripsi yang dilakukan oleh obyek TurnToTextinatorThread, berupa <i>list of strings</i> .	Fungsi yang dipanggil ketika proses <i>transcribing</i> selesai. Melakukan <i>update</i> pada setiap rekor data untuk mengisi transkrip teks-nya, dan memasukkan data transkrip teks dalam tabel yang sudah disediakan.
TogglePlayRecord()	<i>File path</i> dari transkrip suara yang akan dimainkan atau dihentikan.	Setiap rekor data pada tabel dihubungkan pada fungsi ini. Ketika tombol dalam sebuah baris tabel ditekan, program

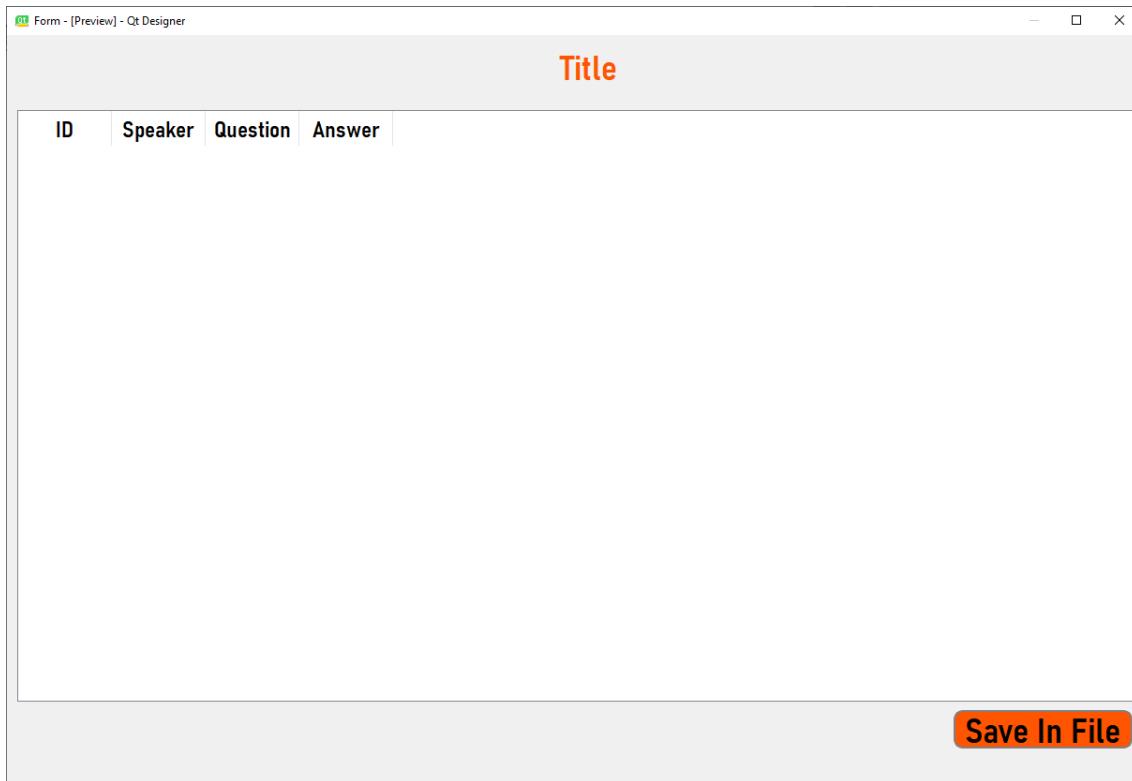
		akan melakukan <i>playback file</i> rekaman yang terkait.
closeEvent()	-	Menutup <i>window</i> ini jika <i>PresentWindow</i> ditutup, untuk menghindari desinkronisasi.



Gambar 4.11 PresentWindow yang sedang menampilkan data sebuah sesi

#### 4.3.2.5. PresentWindow

PresentWindow adalah Window dimana hasil *anomaly detection* ditampilkan. Pengguna dapat melihat hasil proses *anomaly detection* dengan parameter yang ditentukan, berupa tabel rekor data. Rekor data yang terdeteksi sebagai anomali akan diberikan warna merah. Terdapat juga fungsi untuk menyimpan hasil deteksi anomali ke dalam sebuah *file*.



Gambar 4.12 Desain PresentWindow pada Qt Designer

File PresentWindow.ui yang dihasilkan dikonversi menjadi code dalam bahasa Python dengan perintah *command line* berikut ini :

Segmen Kode 4.27 Konversi PresentWindow.ui ke presentwindow.py

```
pyuic6 ui/PresentWindow.ui -x -o objects/presentwindow.py
```

Segmen Kode 4.28 presentwindow.py

```
# Form implementation generated from reading ui file
'ui/PresentWindow.ui'
#
# Created by: PyQt6 UI code generator 6.5.0
#
# WARNING: Any manual changes made to this file will be lost
when pyuic6 is
# run again. Do not edit this file unless you know what you are
doing.
```

```

from PyQt6 import QtCore, QtGui, QtWidgets


class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(1210, 800)
        Form.setStyleSheet("QLabel#label_title,
#label_subtitle{ \n"
"    \n"
"    font: 63 28pt \\"Bahnschrift SemiBold SemiConden\\";\n"
"    color:rgb(255, 85, 0);\n"
"}\n"
"\n"
"QPushButton{\n"
"    font: 63 28pt \\"Bahnschrift SemiBold SemiConden\\";\n"
"    background:rgb(255, 85, 0);\n"
"    border: 2px solid gray;\n"
"    border-radius: 10px;\n"
"    padding: 0 8px;\n"
"}\n"
"\n"
"QPushButton#button_addSpeaker,     QPushButton#button_addEvent,
QPushButton#button_toggle_widget{\n"
"    font: 63 20pt \\"Bahnschrift SemiBold SemiConden\\";\n"
"}\n"
"\n"
"QPushButton:hover{\n"
"    background:rgb(255, 140, 0);\n"
"}\n"
"\n"
"QPushButton:pressed{\n"
"    background: rgb(255, 170, 0);\n"
"}\n"
"\n"
"QPushButton:disabled{\n"

```

```

"      background: rgb(150, 150, 150);\n"
"}\n"
"\n"
"QLabel{\n"
"    font: 63 16pt \"Bahnschrift SemiBold\";\n"
"}\n"
"\n"
"QLabel{\n"
"    \n"
"    font: 63 12pt \"Bahnschrift SemiBold\";\n"
"}\n"
"\n"
"QCheckBox{\n"
"    font: 63 12pt \"Bahnschrift SemiBold\";\n"
"}\n"
"\n"
"QTableWidget{\n"
"    font: 14pt \"Bahnschrift\";\n"
"}\n"
"\n"
"QHeaderView{\n"
"    font: 63 18pt \"Bahnschrift SemiBold SemiConden\";\n"
"}")
        self.button_saveInFile = QtWidgets.QPushButton(parent=Form)
        self.button_saveInFile.setGeometry(QtCore.QRect(1010,
720, 191, 41))

self.button_saveInFile.setObjectName("button_saveInFile")
        self.label_title = QtWidgets.QLabel(parent=Form)
        self.label_title.setGeometry(QtCore.QRect(21, 11, 1200,
45))
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Policy.Preferred,
QtWidgets.QSizePolicy.Policy.Minimum)

```

```

        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.label_title.sizePolicy().has
HeightForWidth())
        self.label_title.setSizePolicy(sizePolicy)

self.label_title.setAlignment(QtCore.Qt.AlignmentFlag.AlignCen
ter)
        self.label_title.setObjectName("label_title")
        self.tableWidget = QtWidgets.QTableWidget(parent=Form)
        self.tableWidget.setGeometry(QtCore.QRect(11, 80, 1191,
631))
        sizePolicy =
QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Policy.Minimum,
QtWidgets.QSizePolicy.Policy.Minimum)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)

sizePolicy.setHeightForWidth(self.tableWidget.sizePolicy().has
HeightForWidth())
        self.tableWidget.setSizePolicy(sizePolicy)
        self.tableWidget.setObjectName("tableWidget")
        self.tableWidget.setColumnCount(4)
        self.tableWidget.setRowCount(0)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget.setHorizontalHeaderItem(0, item)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget.setHorizontalHeaderItem(1, item)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget.setHorizontalHeaderItem(2, item)
        item = QtWidgets.QTableWidgetItem()
        self.tableWidget.setHorizontalHeaderItem(3, item)

        self.retranslateUi(Form)

```

```

QtCore.QMetaObject.connectSlotsByName(Form)

def retranslateUi(self, Form):
    _translate = QtCore.QCoreApplication.translate
    Form.setWindowTitle(_translate("Form", "Form"))
    self.button_saveInFile.setText(_translate("Form",
    "Save In File"))

    self.label_title.setText(_translate("Form", "Title"))
    item = self.tableWidget.horizontalHeaderItem(0)
    item.setText(_translate("Form", "ID"))
    item = self.tableWidget.horizontalHeaderItem(1)
    item.setText(_translate("Form", "Speaker"))
    item = self.tableWidget.horizontalHeaderItem(2)
    item.setText(_translate("Form", "Question"))
    item = self.tableWidget.horizontalHeaderItem(3)
    item.setText(_translate("Form", "Answer"))

if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    Form = QtWidgets.QWidget()
    ui = Ui_Form()
    ui.setupUi(Form)
    Form.show()
    sys.exit(app.exec())

```

Class PresentWindow pada app.py merupakan *inheritance* dari file presentwindow.py, dimana dibuat fungsi-fungsi yang berinteraksi dengan elemen-elemen GUI.

Segmen Kode 4.29 Class PresentWindow

```

class PresentWindow(qtw.QWidget):
    def __init__(self, df: db.pd.DataFrame, eventTitle: str,
processWindow: ProcessWindow) -> None:
        super().__init__()
        # the processwindow here

```

```

        self.processWindow = processWindow

        # set stuff here
        self.ui = prw.Ui_Form()
        self.ui.setupUi(self)
        self.df = df
        self.ui.label_title.setText(eventTitle)

        self.setWindowIcon(qtg.QIcon("asset/images/Solaire.png"))
        self.setWindowTitle("Qui      habet      oculi      vidiendi,
videat!")
        # connect stuff here
        self.LoadTable(self.df)

        self.ui.button_saveInFile.clicked.connect(self.SaveInFile)

        # functions
        def SaveInFile(self):
            print("saving to file.")

        def LoadTable(self, df: db.pd.DataFrame):

            self.ui.tableWidget.verticalScrollBar().setSingleStep(1)
            self.ui.tableWidget.setRowCount(0)
            for i in range(len(self.df.index)):
                self.ui.tableWidget.insertRow(i)
                self.ui.tableWidget.setItem(
                    i, 0, qtw.QTableWidgetItem(str(self.df.loc[i]["id"])))
                self.ui.tableWidget.setItem(
                    i, 1, qtw.QTableWidgetItem(str(self.df.loc[i]["speaker"])))
                self.ui.tableWidget.setItem(
                    i, 2, qtw.QTableWidgetItem(str(self.df.loc[i]["question"])))

```

```

        self.ui.tableWidget.setItem(
            i,
            3,
            qtw.QTableWidgetItem(str(self.df.loc[i]["answer"])))
            # if outlier, color red
            if self.df.loc[i]["Cluster Assignment"] == -1:
                for j in range(self.ui.tableWidget.columnCount()):
                    self.ui.tableWidget.item(i,
j).setBackground(
                        qtg.QColor(232, 52, 39, 255))
            # resize

self.ui.tableWidget.horizontalHeader().setSectionResizeMode(
    3, qtw.QHeaderView.ResizeMode.Stretch)

self.ui.tableWidget.horizontalHeader().setSectionResizeMode(
    2, qtw.QHeaderView.ResizeMode.Stretch)
    self.ui.tableWidget.resizeRowsToContents()    # do this
AFTER stretching.

    # do this AFTER stretching.
    self.ui.tableWidget.resizeColumnsToContents()
    # fix the scrollbar
    self.ui.tableWidget.setVerticalScrollMode(
        qtw.QAbstractItemView.ScrollMode.ScrollPerPixel)

self.ui.tableWidget.verticalScrollBar().setSingleStep(5)
    # self.ui.tableWidget()

```

Berikut tabel fungsi-fungsi yang ada dalam class PresentWindow dan fungsinya.

Tabel 4.5 Mapping fungsi class PresentWindow

Fungsi	Input	Keterangan
SaveInFile()	-	Menyimpan data <i>outlier</i> dalam sebuah <i>file CSV</i> .

LoadTable()	Rekor data dan keterangan anomalinya dalam bentuk DataFrame.	Menyajikan data dalam tabel yang telah disediakan. Rekor data yang menjadi anomali diberi warna merah.
-------------	--	--

Qui habet oculi vidiendi, videat!

### Of Choice and Life

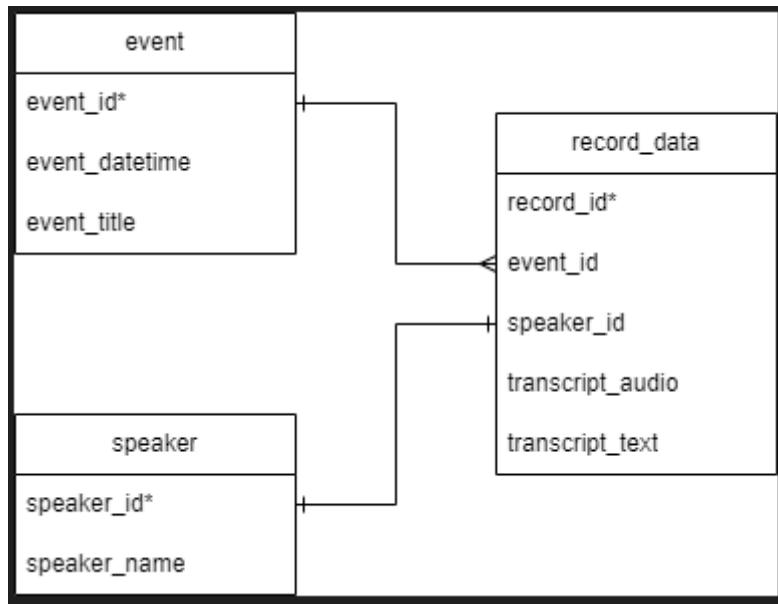
ID	Speaker	Question	Answer
1 55	KSMG	Pro life or pro choice?	I am pro choice. I feel bad for the baby. People have abortions for a reason. If it is forced, it will grow imperfectly. Born into the world in a state that is not optimal. With prochoice, her fate is more manageable.
2 56	JJ	Pro life or pro choice?	Logically, both make sense. Conflicted between the two because pro life - you remove their chance of being born, pro choice - there are events that make the baby if born not 100 percent healthy. Also if the parents are not ready, financially etc. Life is a life - no matter what form it is
3 57	JER	Pro life or pro choice?	I'm pro life. Because in my belief, if a fetus is conceived, there is already life, there is already consciousness. And if you abort that it has a life collector. That thing that doesn't know anything gotta follow you anywhere. Maybe in the next time this couple will have a planned child. It would be even more difficult, what with birth defects etc. Contraception isn't good either - like pills. Sexual education is important, but sometimes lifestyle can't lie.
4 58	YOT	Pro life or pro choice?	Prochoice. If she was a victim of rape, etc, she doesn't want to keep the child, at least she has the option to choose. The unwanted child.
5 59	GRE	Pro life or pro choice?	Prolife for religious reasons. Being religious is good. But most prolife people reject abortion altogether meaning that if there are defects etc, it is also not allowed. The birth is full of suffering. In the US it's a big deal because in Texas it's not allowed at all even because of incest.
6 60	RIC	Pro life or pro choice?	Prochoice. Unless the baby is normal.
7 61	GRE	Do you think abortion should be legal?	Legal - not really legal - legal for special cases. Like for sexual assault, rape, etc. But at least 6 months - you can go over the pregnancy period. But really it should be all legal, because orphanages are still not 100% ideal. So it

**Save In File**

Gambar 4.13 Tampilan PresentWindow dengan satu *outlier* dalam data (merah)

#### 4.3.3. Skema Database

Untuk menyimpan data rekaman, pembicara, dan sesi diskusi, dibuat skema *database* berikut:



Gambar 4.14 Skema database untuk program

Skema database ini digunakan baik dalam proses eksperimen maupun program. Database dibuat dengan `sqlite3`, library Python yang memungkinkan koneksi dan operasi-operasi data dengan database SQLite. Untuk kepentingan eksperimen dan program, dibuat sebuah *class* untuk *database handling*, yang dinamakan `DatabaseHandler`. *Class* ini dimuat dalam file `db.py`.

Segmen Kode 4.30 Class DatabaseHandler dalam file db.py

```

import sqlite3
import pandas as pd
import os

class DatabaseHandler():
    def __init__(self, dbName, keepSchema: bool = True):
        self.cursor = sqlite3.connect(dbName)
        if not keepSchema:
            print('ya schema empty, yeet')
            self.create_tables()

    # create
    def create_tables(self):
        self.cursor.execute("""

```

```

CREATE TABLE IF NOT EXISTS speakers (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    name TEXT NOT NULL,
    code TEXT NOT NULL
) ;
""")

self.cursor.execute("""
    CREATE TABLE IF NOT EXISTS events (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        title TEXT NOT NULL,
        datetime TEXT NOT NULL
    );
""")
# self.cursor.execute("DROP TABLE record_data") # the
sacred seal of the nine tails. Do not open
self.cursor.execute("""
    CREATE TABLE IF NOT EXISTS record_data (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        event_id INTEGER,
        speaker_id INTEGER,
        question_text NOT NULL,
        transcript_audio TEXT NOT NULL,
        transcript_text TEXT NOT NULL,
        transcribe_lang TEXT NOT NULL
    );
""")
"""

def create_speaker(self, speakerName, speakerCode):
    self.cursor.execute("""
        INSERT INTO speakers (name, code)
        VALUES (?, ?)
    """, (speakerName, speakerCode))
    self.cursor.commit()

```

```

def create_event(self, eventTitle, eventDate):
    id = self.cursor.execute("""
        INSERT INTO events (title, datetime)
        VALUES (?, ?)
    """, (eventTitle, eventDate)).lastrowid
    self.cursor.commit()
    print("Current event is at {}".format(id))

    # create the directory.
    os.mkdir("records/event{}".format(id))

    def create_record_audio(self, event_id, speaker_id,
question, audio_path, transcribe_lang):
        self.cursor.execute("""
            INSERT INTO record_data (event_id, speaker_id,
question_text, transcript_audio, transcript_text,
transcribe_lang)
            VALUES (?, ?, ?, ?, "", ?)
        """, (event_id, speaker_id, question, audio_path,
transcribe_lang))
        self.cursor.commit()

    def create_record_text(self, event_id, speaker_id,
question, text):
        self.cursor.execute("""
            INSERT INTO record_data (event_id, speaker_id,
question_text, transcript_audio, transcript_text,
transcribe_lang)
            VALUES (?, ?, ?, "", ?, "")
        """, (event_id, speaker_id, question, text))
        self.cursor.commit()

    # read
    def get_speaker(self, selector, ID):
        return self.cursor.execute("""

```

```

        SELECT * FROM speakers
        WHERE {} = ?
    """".format(selector), (ID,)).fetchone()

def get_event(self, selector, ID):
    return self.cursor.execute("""
        SELECT * FROM events
        WHERE {} = ?
    """".format(selector), (ID,)).fetchone()

def get_recordData(self, selector, ID):
    query = """
        SELECT * FROM record_data
        WHERE {} = ?
    """".format(selector)
    self.cursor.execute(query, (ID,)).fetchall()

# list

def list_speakers(self):
    return self.cursor.execute("""
        SELECT * FROM speakers
    """").fetchall()

def list_events(self):
    return self.cursor.execute("""
        SELECT * FROM events
    """").fetchall()

def list_recordData(self):
    return self.cursor.execute("""
        SELECT * FROM record_data
    """").fetchall()

# definitive record data

```

```

def list_recordData_complete(self):
    pass

# update
def update_recordData_text(self, ID, text):
    self.cursor.execute("""
        UPDATE record_data
        SET transcript_text = ?
        WHERE id = ?
    """, (text, ID))
    self.cursor.commit()

# delete
def clear_table(self, table):
    self.cursor.execute("""
        DELETE FROM {};
    """.format(table))
    self.cursor.commit()

def delete_by_id(self, table:str, id:int):
    self.cursor.execute("""
        DELETE FROM {}
        WHERE ID = ?
    """.format(table), (id))
    self.cursor.commit()

# goofy ahh functions
def get_recordCountByEventID(self, eventID):
    return len(self.get_recordData("event_id", eventID))

def loop_create_speaker(self):
    # # create speakers
    while True:
        name = input("Name : ")
        if name == "stop":

```

```

        break

    code = input("Code : ")
    self.create_speaker(name, code)

def create_recordDataFromExcel(self, filePath, sheetName,
eventID):
    df = pd.read_excel(filePath, sheet_name=sheetName)
    for i in df.index:
        print("-" * 80)
        speakerCode = df.loc[i]["Speaker"]
        print(speakerCode)
        speakerData = self.get_speaker("code", speakerCode)
        speakerID = speakerData[0]

        self.create_record_text(event_id=eventID,
speaker_id=speakerID,
question=df.loc[i]["Question"], text=df.loc[i]["Answer"])
        print("-" * 80)

def get_recordDataJoined(self, selector, ID):
    query = """
        SELECT record_data.id as id, events.title as
event_title, events.datetime as date, speakers.code as
speaker_code, question_text, transcript_audio, transcript_text,
transcribe_lang
        FROM record_data
        INNER JOIN events
        ON record_data.event_id = events.id
        INNER JOIN speakers
        ON record_data.speaker_id = speakers.id
        WHERE record_data.{} = ?
    """.format(selector)
    return self.cursor.execute(query, (ID,)).fetchall()

```

```

def get_recordDataJoinedDF(self, selector, ID):
    query = """
        SELECT record_data.id as id, events.title as
event_title, speakers.code as speaker, question_text as
question, transcript_text as answer
        FROM record_data
        INNER JOIN events
        ON record_data.event_id = events.id
        INNER JOIN speakers
        ON record_data.speaker_id = speakers.id
        WHERE record_data.{ } = { }
    """.format(selector, ID)
    df = pd.read_sql_query(query, self.cursor)
    return df

```

Berikut fungsi-fungsi yang ada dalam class DatabaseHandler dan kegunaannya.

Segmen Kode 4.31 *Mapping* fungsi class DatabaseHandler

Fungsi	Input	Keterangan
create_tables()	-	Membuat skema database sebagaimana ditentukan oleh ERD sebagai inisialisasi.
create_speaker()	Nama dan kode pembicara, dalam bentuk <i>string</i> .	Membuat rekor data pembicara pada tabel speakers.
create_event()	Nama dan tanggal event, dalam format <i>string</i> .	Membuat rekor data sesi diskusi pada tabel events.
create_record_audio()	- ID Event : integer - ID pembicara : integer	Membuat rekor data rekaman, dengan isi data berupa <i>file path</i> rekaman audio. File audio akan direkam dan diisi oleh program dengan fungsi

	<ul style="list-style-type: none"> <li>- Pertanyaan : string</li> <li>- File path audio : string</li> <li>- Bahasa transkripsi : string</li> </ul>	<p>ListenWindow.ToggleRecord().</p> <p>Parameter transcribeLang menyatakan bahasa transkripsi, untuk menjaga kualitas transkripsi jika menggunakan bahasa Inggris saja.</p>
create_record_text	<ul style="list-style-type: none"> <li>- ID Event : integer</li> <li>- ID pembicara : integer</li> <li>- Pertanyaan : string</li> <li>- Teks : string</li> <li>- Bahasa transkripsi : string</li> </ul>	Membuat rekor data rekaman dengan isi data berupa teks secara langsung. Digunakan untuk mengimpor data dari file .xlsx.
get_speaker()	<ul style="list-style-type: none"> <li>- ID</li> <li>- Selector</li> </ul>	Mendapatkan data pembicara dengan <i>selector</i> tertentu.
get_event()	<ul style="list-style-type: none"> <li>- ID</li> <li>- Selector</li> </ul>	Mendapatkan data sesi diskusi dengan <i>selector</i> tertentu.
get_recordData()	<ul style="list-style-type: none"> <li>- ID</li> <li>- Selector</li> </ul>	Mendapatkan data rekaman dengan <i>selector</i> tertentu.
list_speakers()	-	Mendapatkan seluruh tabel speakers.
list_events()	-	Mendapatkan seluruh tabel events.

list_recordData()	-	Mendapatkan seluruh tabel record_data.
update_recordData_text() ()	<ul style="list-style-type: none"> <li>- ID : integer</li> <li>- Text : string</li> </ul>	Melakukan update pada sebuah data rekaman di tabel record_data. Digunakan ketika sudah mendapatkan hasil transkripsi dan akan menyimpannya dalam rekordata yang bersangkutan.
get_recordDataJoinedDF() ()	<ul style="list-style-type: none"> <li>- ID</li> <li>- Selector</li> </ul>	Mendapatkan rekordata dalam sesi yang sama (eventID tertentu), dan melakukan join dengan tabel speakers dan events untuk mendapatkan nama sesi dan kode pembicara. Data akan dikonversi dalam bentuk pandas.DataFrame untuk mempermudah operasi-operasi deteksi anomali.

#### 4.3.4. Komponen Transcriber dan Translator

Untuk mengubah data rekaman suara pembicara menjadi teks yang bisa diolah, digunakan komponen-komponen *transcriber* dan *translator*. Komponen *transcriber* yang digunakan adalah *library Python whisper*, yang merupakan produk whisper.ai. Komponen ini menerima input berupa *file* audio dan parameter berupa bahasa transkripsi, dan memberikan *output* transkrip dalam bentuk teks. Jika transkrip berbahasa Indonesia, teks transkrip akan diterjemahkan oleh komponen *translator*, yaitu *library Python googletrans*. Implementasi komponen-komponen ini diwujudkan dalam *class TurnToTextinator* dalam *file turntotextinator.py*

Segmen Kode 4.32 Class TurnToTextinator dalam file turntotextinator.py

```
import whisper
import os
import googletrans as gt
```

```

# behold the most critical non-core component: the Turn-to-
textinator

# it transcribes an audio file and translates to English if it
isn't already.

# takes a .wav and returns a string. The saving will be done in
the main app, invoking the db function.

class TurnToTextinator():

    def __init__(self) -> None:
        self.whisperModel = whisper.load_model("large")
        self.translator = gt.Translator()

    def TranslateText(self, text, src="auto", dest="en"):
        return self.translator.translate(text, src=src,
dest=dest).text

    def TranscribeText(self, fileName, transcribeLang="id"):
        filenamePath = os.path.join(os.getcwd(), fileName)
        result = self.whisperModel.transcribe(filenamePath,
fp16=False, language=transcribeLang)
        return result["text"]

    def TwoForOneSpecial(self, fileName, transcribeLang="id",
translateLang="en"):
        text = self.TranscribeText(fileName, transcribeLang)
        return self.TranslateText(text, transcribeLang,
translateLang)

```

Tabel 4.6 *Mapping* fungsi class TranslateText

Fungsi	Input	Keterangan
TranslateText()	Input berupa teks, bahasa asal dan bahasa tujuan dalam <i>string</i> .	Menggunakan <i>library</i> googletrans untuk menerjemahkan teks dari

		satu bahasa ke bahasa lainnya. Digunakan untuk menerjemahkan teks hasil transkrip bahasa Indonesia ke bahasa Inggris.
TranscribeText ()	<i>File path</i> audio dan bahasa transkrip dalam bentuk <i>string</i>	Menggunakan <i>library</i> <i>whisper</i> untuk melakukan transkripsi audio ke teks. Jika input dalam bahasa Inggris, akan langsung dilakukan transkripsi dalam bahasa Inggris.
TwoForOneSpecial ()	<i>File path</i> audio, bahasa asal dan bahasa tujuan dalam <i>string</i> .	Melakukan transkripsi dalam bahasa Indonesia dengan fungsi <i>TranscribeText()</i> , lalu menerjemahkan hasil transkripsi ke bahasa Inggris dengan <i>TranslateText()</i> .

Class *TurnToTextinator* dimanfaatkan dalam program untuk melakukan transkripsi *file* audio. Fungsi ini harus dijalankan secara *asynchronous* agar *loop* UI program tidak terinterupsi. Fungsi *asynchronous* diimplementasikan dengan cara menggunakan *class* *QThread* yang disediakan oleh *library* *PyQt6*. Oleh karena itu, *class* ini diimplementasikan sebagai *inheritance* dari *class* *TurnToTextinatorThread*.

Segmen Kode 4.33 *Class TurnToTextinatorThread* sebagai implementasi *class* *TurnToTextinator* yang memanfaatkan *multithreading* dengan *QThread*

```
class TurnToTextinatorThread(qtc.QThread): # tttt for short
    transcribe_signal = qtc.pyqtSignal(list)

    def __init__(self, filePathList: list = []) -> None:
        super().__init__()
```

```

        self.filePathList = filePathList

    def setFilePath(self, filePathList: list):
        self.filePathList = filePathList

    def run(self):
        filePath_transcriptList = []
        for row in range(len(self.filePathList)):
            # conditions for transcribing
            if self.filePathList[row][2].lower() == "id":
                # if indonesian, hit it with the two for one
                # special (transcribe as ID, then translate ID to EN)
                transcript = ttt.TwoForOneSpecial(
                    fileName=self.filePathList[row][1],
                    transcribeLang="id", translateLang="en")
            elif self.filePathList[row][2].lower() == "en":
                # if english, only return the transcription
                transcript = ttt.TranscribeText(
                    fileName=self.filePathList[row][1],
                    transcribeLang="en")
            filePath_transcriptList.append(
                (self.filePathList[row][0],
                 self.filePathList[row][1], transcript))
        # returns a list of (row, filePath, transcript)
        self.transcribe_signal.emit(filePath_transcriptList)

```

*Object* dari *class* ini akan dibuat ketika tombol “Transcribe” di *ProcessWindow* ditekan. Semua rekord data pada tabel yang memiliki *file* audio akan *di-passing* ke *object* *TurnToTextinatorThread* yang telah dibuat. Variabel *transcribe\_signal* dalam *object* *TurnToTextinatorThread* akan mengirimkan sinyal ketika proses transkripsi selesai; sinyal ini dihubungkan pada fungsi *ProcessWindow.GetTranscript()* untuk kemudian disajikan dalam tabel dan disimpan dalam *database*.

#### 4.3.5. Audio Recorder

Untuk merekam suara peserta yang sedang berbicara, digunakan *library* pyaudio yang langsung diimplementasikan dalam *class* RecorderThread yang merupakan *inheritance* dari QThread. Proses rekaman suara merupakan proses *asynchronous* dan harus berjalan secara paralel dengan GUI program.

Segmen Kode 4.34 Class RecorderThread

```
class RecorderThread(qtc.QThread) :  
  
    toggle_signal = qtc.pyqtSignal(str)  
  
    def __init__(self, fileName=None):  
        super().__init__()  
        self.fileName = fileName  
        self.is_running = True # for toggling  
  
        # this function will be called when QThread::start() is  
        # called.  
    def run(self):  
        audio = pa.PyAudio()  
        frames = []  
        stream = audio.open(format=pa.paInt16, channels=1,  
                            rate=44100, input=True,  
                            frames_per_buffer=1024)  
        # apparently the point is making a separate loop while  
        # keeping the GUI loop running.  
        while self.is_running:  
            data = stream.read(1024)  
            frames.append(data)  
        stream.close()  
        wf = wave.open(self.fileName, 'wb')  
        wf.setnchannels(1)  
        wf.setsampwidth(audio.get_sample_size(pa.paInt16))  
        wf.setframerate(44100)  
        wf.writeframes(b''.join(frames))
```

```

wf.close()

# this here is what to do upon stopping
def stop(self):
    self.is_running = False
    self.toggle_signal.emit(self.fileName)

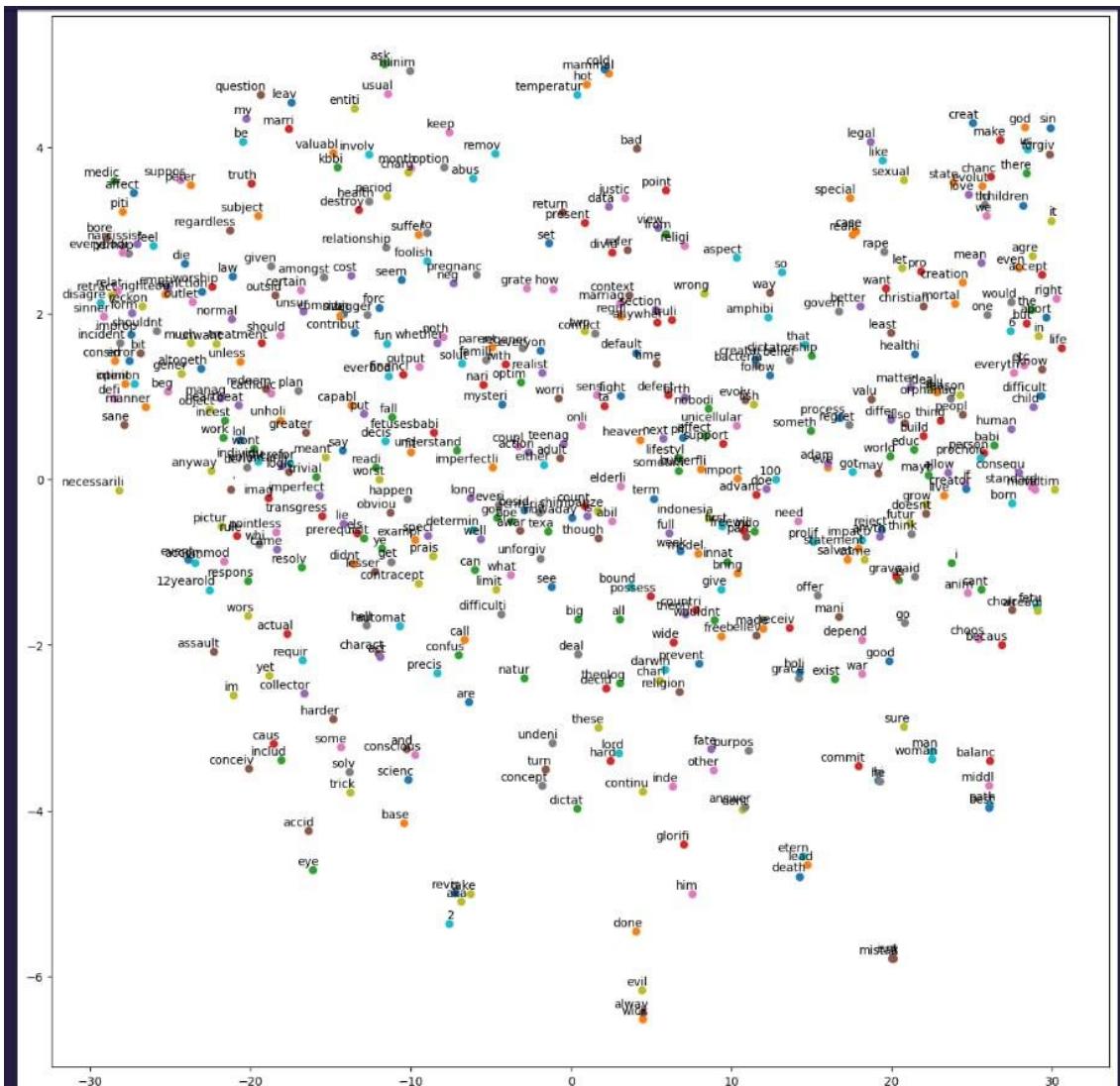
```

Ketika fungsi `ListenWindow.ToggleRecord()` dijalankan, program akan memulai rekaman dengan cara membuat sebuah object `RecorderThread`. `RecorderThread` akan terus merekam audio sampai fungsi `ListenWindow.ToggleRecord()` dijalankan lagi, dimana program akan mengirimkan sinyal `stop()` pada object `RecorderThread`, dan variabel `toggle_signal` akan mengirimkan sinyal yang dihubungkan pada fungsi `ListenWindow.SaveRecording()` untuk menyimpan *file* hasil rekaman di *directory* yang telah ditentukan.

## 5. ANALISIS HASIL PENGUJIAN

### 5.1. Text Preprocessing dan Tokenization

Dalam fungsi `PreprocessDocument()` diberikan beberapa parameter, yaitu penggunaan *lemmatizing*, *inflection*, *stop-word removal*, dan *filtering* untuk angka. *Text preprocessing* dilakukan untuk menjaga kualitas data yang akan menjadi basis *feature extraction*. Fokus terutama adalah mengurangi *noise* atau data dengan relevansi rendah dan frekuensi tinggi, serta menghilangkan data berupa angka dan tanda baca, agar proses *word-tokenization* menghasilkan *word-tokens* yang baik dan dikenali oleh model yang digunakan.



Gambar 5.1 Hasil *word-embedding* dengan model yang dibuat sendiri

Pada awal penelitian, penggunaan algoritma *stemming* PorterStemmer() sempat dicoba. Namun, banyak token yang dihasilkan oleh algoritma yang tidak dikenali oleh model-model *pre-trained* yang digunakan, karena *word-token* yang dihasilkan bukan merupakan kosakata Bahasa Inggris yang sering digunakan. Contohnya adalah hasil stemming dari kata-kata Bahasa Inggris “accident”, “accidental”, dan “accidentally” yang menjadi token “accid” setelah melewati proses *stemming*. Token-token yang dihasilkan oleh algoritma *stemming* hanya bisa di-train sebagai model *word-embedding* sendiri, namun terdapat perbedaan kualitas yang sangat besar dengan model-model *pre-trained*, karena jumlah dataset *training* yang jauh berbeda.

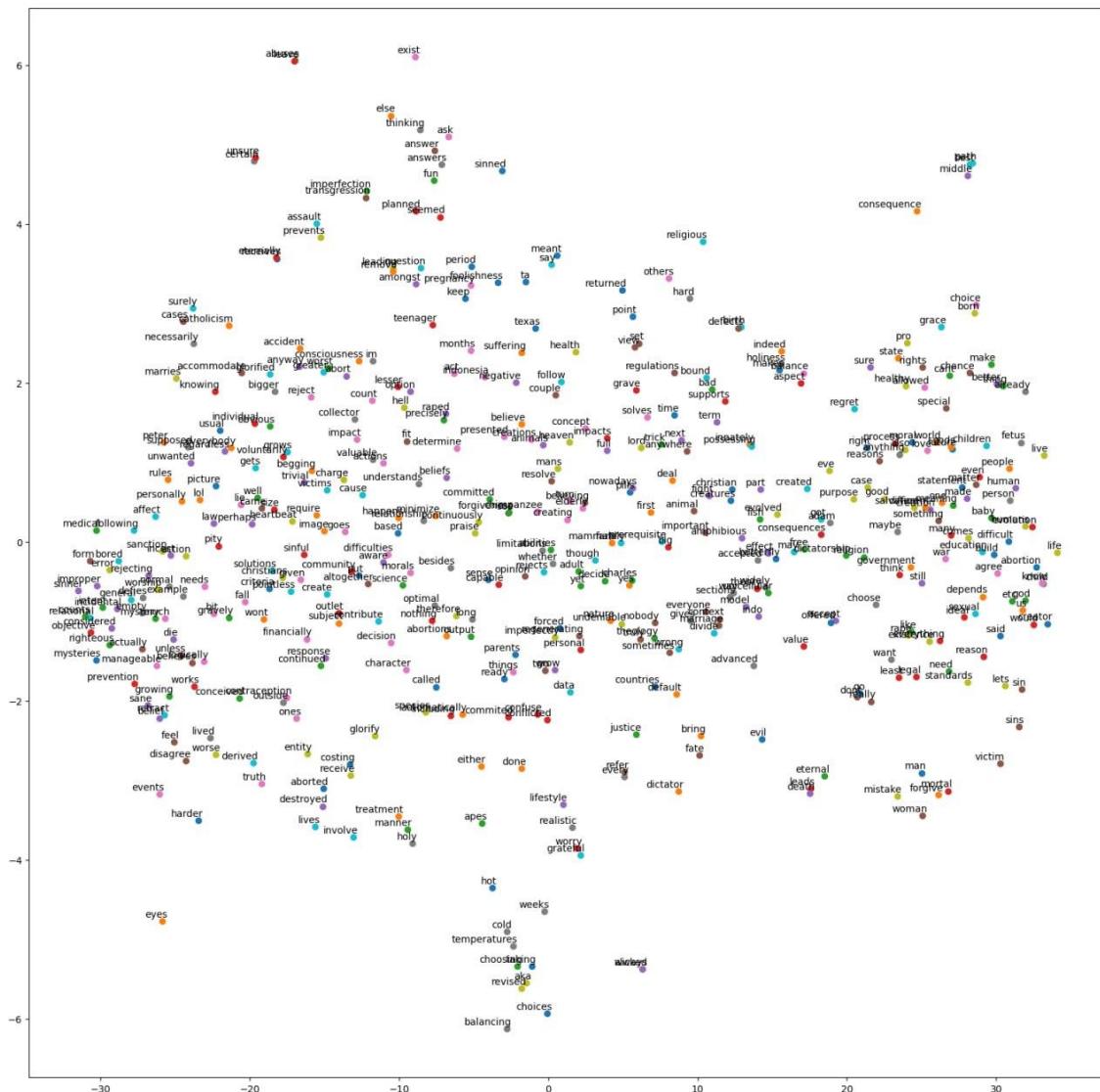
Oleh karena itu, digunakan *lemmatizer* WordNetLemmatizer() untuk melakukan *lemmatizing*, yang berupaya mengurangi jumlah token data unik dengan menentukan tag POS-

nya dengan algoritma POSTagging(). Dengan algoritma ini, token-token yang dihasilkan berupa kata-kata Bahasa Inggris yang ada dalam *vocabulary* model-model *pre-trained* yang digunakan. Maka, proses *lemmatizing* digunakan dalam setiap kombinasi eksperimen tanpa terkecuali.

Terakhir, tahap *stop-word removal* dan *number filtering*. *Stop-word removal* digunakan untuk mengurangi word-token yang tergolong stop-words, yaitu kata-kata yang berfrekuensi tinggi, namun tidak memberikan makna semantik yang signifikan, seperti *pronouns* (*he*, *she*, *they*, *them*, dan lain-lain) dan *prepositions* (*in*, *on*, *at*, dan lain-lain). Dengan dikeluarkannya token-token ini dari dokumen, didapatkan *word-tokens* yang relevan dalam sebuah dokumen. Selain itu, penerapan *number filtering* yang menganggap digit-digit angka sebagai *stop-word* pada dataset yang digunakan berhasil mengurangi jumlah token-token yang ringan makna, jika dibandingkan dengan *inflection* (mengubah digit-digit angka menjadi bentuk terbilangnya). Oleh karena itu, secara *default*, fungsi PreprocessDocs menggunakan *lemmatizing*, *stop-word removal*, dan *number filtering*. Terbentuknya *word-tokens* yang sarat makna dalam setiap dokumen memastikan *word-embedding* dan *topic modelling* yang diterapkan berhasil menangkap makna semantik yang lebih baik.

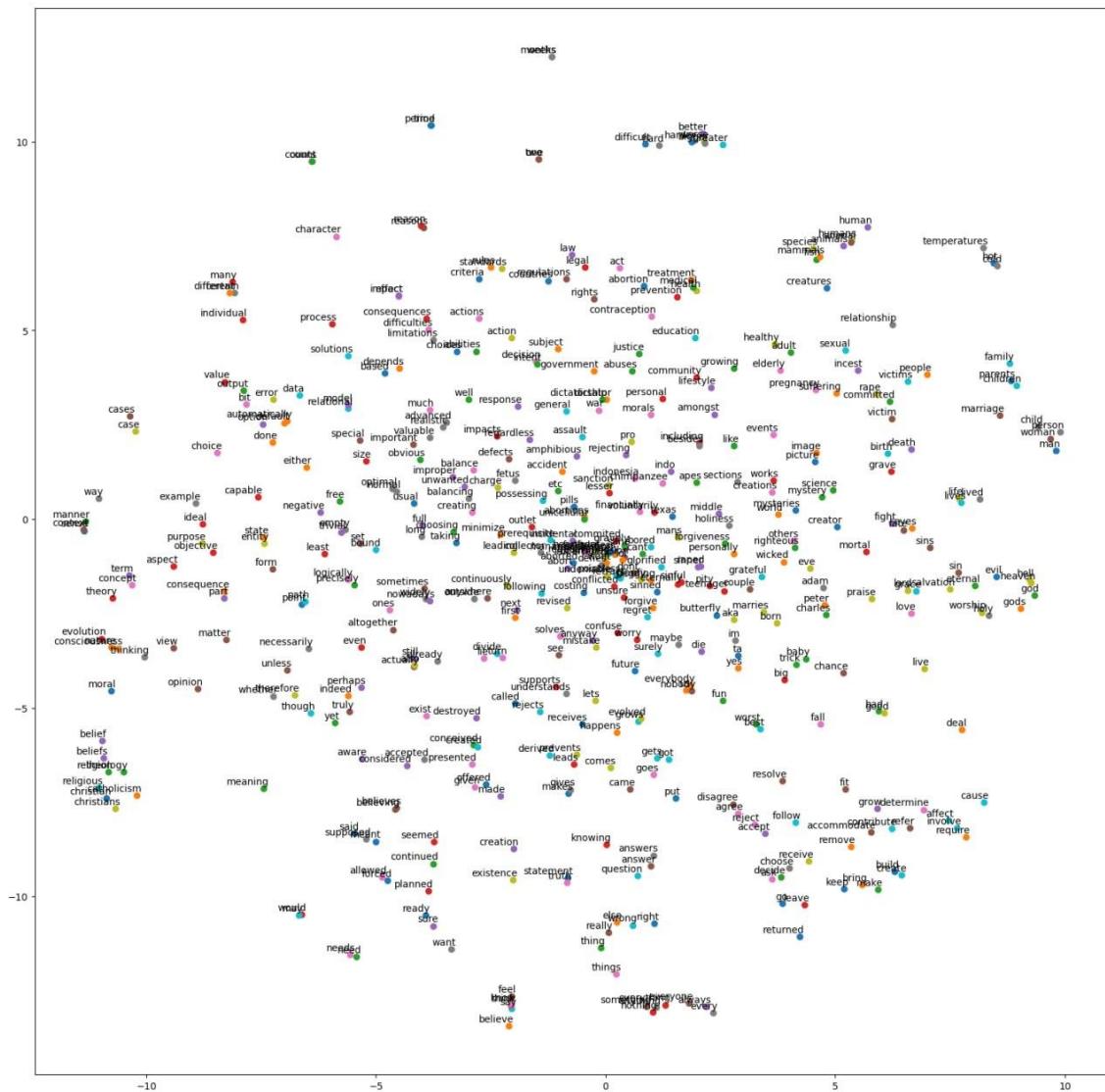
## 5.2. Word Embedding

Dilakukan evaluasi terlebih dahulu terhadap model-model *word-embedding* yang digunakan, untuk menentukan kualitas semantik yang didapatkan dari *embedding* tersebut. Evaluasi ini dilakukan secara visual dengan *dimensionality reduction* dengan *library* sklearn.manifold.TSNE dan *scatter plot* dengan matplotlib.pyplot. Beberapa model yang digunakan pada awal penelitian termasuk model yang di-train sendiri berdasarkan hasil *append 3* dataset teks yang tersedia, serta pre-trained model text8 yang disediakan oleh gensim.downloader.



Gambar 5.2 Model *word2vec* yang di-train sendiri

Tidak terbentuk makna semantik yang baik dalam model word2vec yang di-train dari dataset sendiri, karena ukuran dataset yang terlalu kecil untuk menangkap pola-pola semantik yang luas. Dibandingkan dengan `word2vec-google-news-300` yang di-train dengan dataset dengan jumlah rekord data mencapai milyaran, dataset yang dikumpulkan untuk penelitian ini jauh lebih kecil dengan jumlah rekord data dibawah 200 rekord. Oleh karena itu, model yang di-*train* sendiri tidak akan digunakan sebagai dasar *feature extraction* dengan pendekatan *embedding*.



Gambar 5.3 Model text8

Dibandingkan dengan model yang di-train sendiri, `text8` diamati dapat menangkap makna semantik dengan lebih koheren, seperti yang dapat diamati dalam token-kata “*belief*”, “*religious*”, “*christian*”, dan “*catholicism*”. Namun, banyak word-token, terutama yang didapatkan dari hasil transkripsi klip suara, yang tidak terdapat dalam *vocabulary* dalam model ini. Model `text8` tidak digunakan dalam eksperimen kombinasi parameter. Pengujian selanjutnya dilakukan dengan model-model `word2vec-google-news-300`, `glove-wiki-gigaword-50`, `glove-wiki-gigaword-100`, dan `glove-wiki-gigaword-300`.

Segmen Kode 5.1 Mendapatkan *scatter plot* untuk setiap *pre-trained* model.

```
from sklearn.manifold import TSNE  
import matplotlib.pyplot as plt
```

```

modelList = ["word2vec-google-news-300",
             "glove-wiki-gigaword-50",
             "glove-wiki-gigaword-100",
             "glove-wiki-gigaword-300"]

for model in modelList:

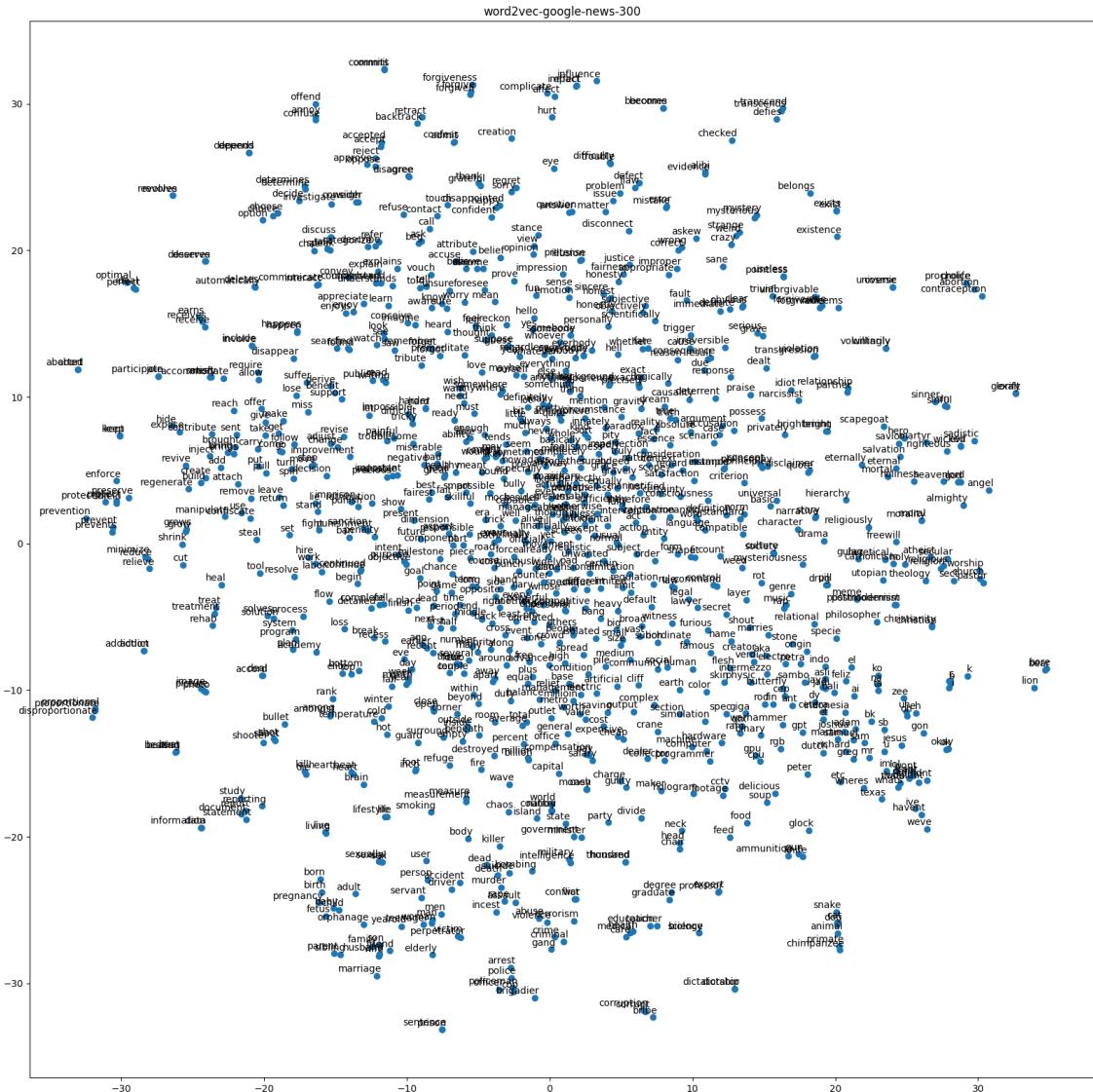
    WEModel = api.load(model)
    embedded_words = WordEmbed(mySet, WEModel)
    words, vectors = zip(*embedded_words)
    vectors = np.array(vectors)
    tsne = TSNE(n_components=2, random_state=69)
    reduced_vectors = tsne.fit_transform(vectors)

    plt.figure(figsize=(20,20))
    plt.scatter(reduced_vectors[:, 0], reduced_vectors[:, 1],
cmap="viridis")
    for i, word in enumerate(words):
        plt.annotate(word,
                     xy=(reduced_vectors[i, 0],
reduced_vectors[i, 1]),
                     xytext=(5,2),
                     textcoords="offset points",
                     ha="right",
                     va="bottom"
                     )
    plt.title(model)
    plt.show()

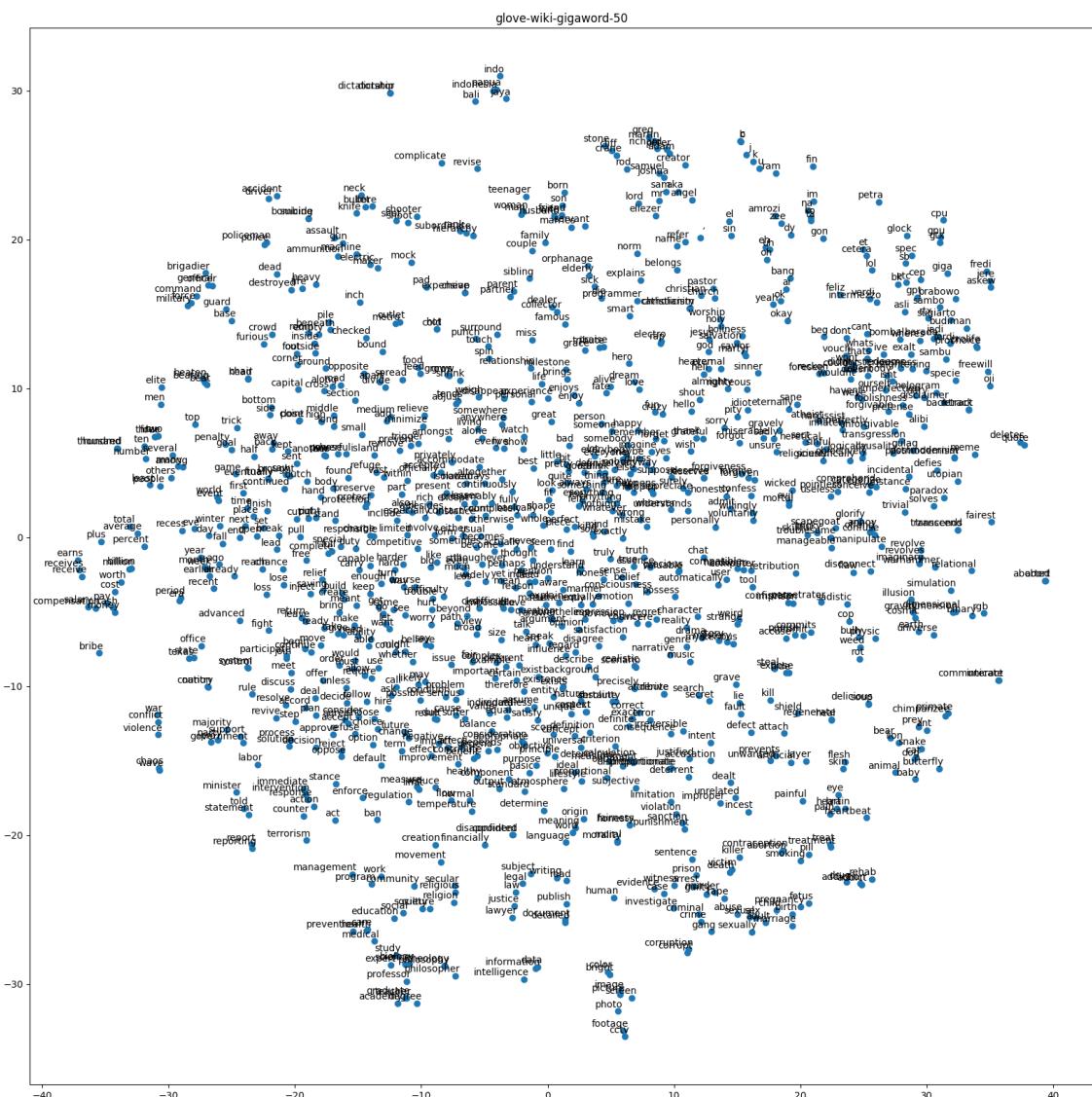
```

Dari *dataset-dataset* yang dipakai, diekstraksi setiap *word-token* yang unik dan dimasukkan ke dalam sebuah set, untuk menghindari duplikat. Setelah itu, fitur-fitur yang didapatkan akan direduksi secara dimensi menggunakan TSNE dengan `n_components = 2` untuk mengubahnya menjadi 2 dimensi agar bisa diilustrasikan dalam sebuah *scatter plot* untuk

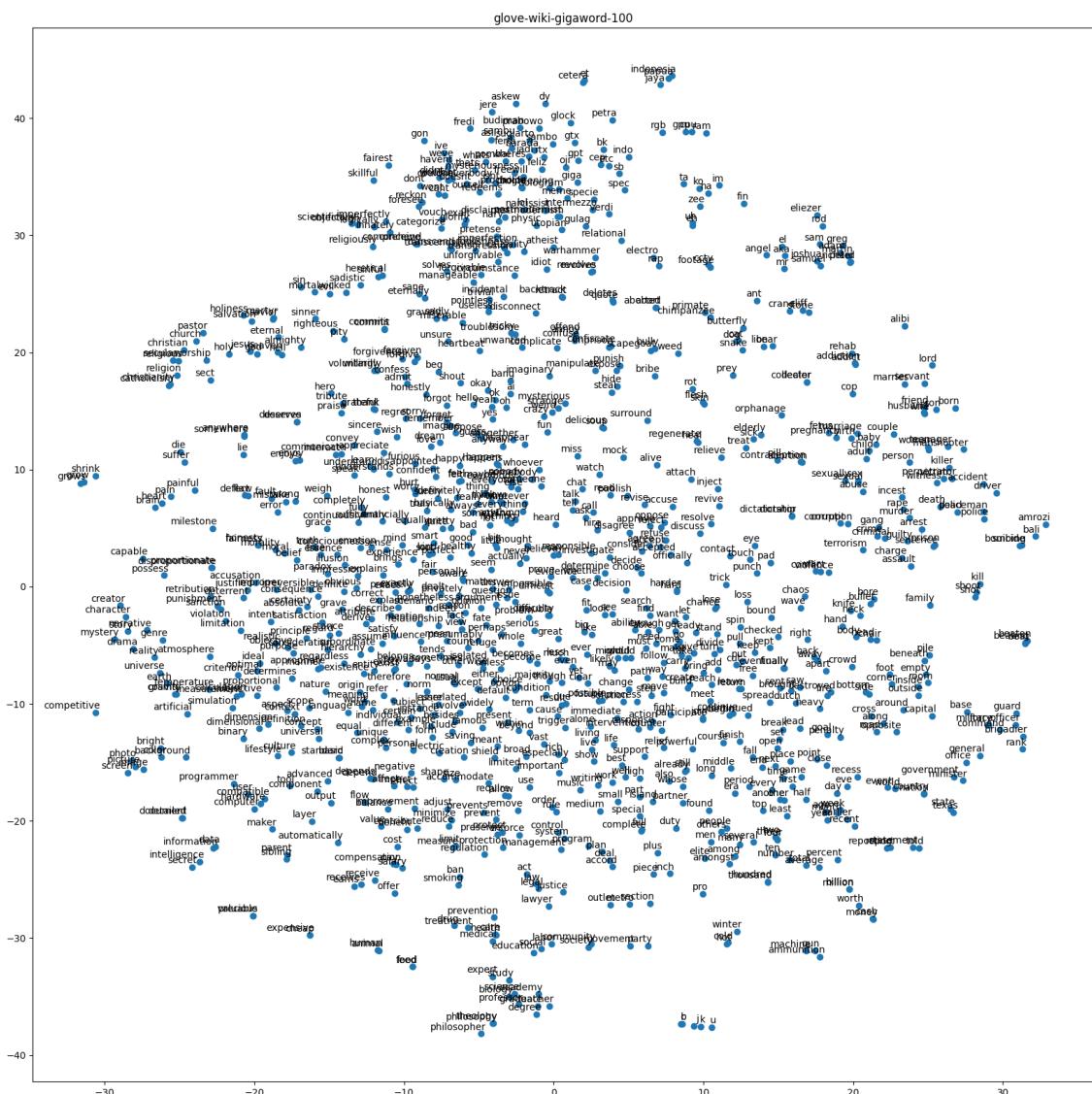
mengevaluasi kedekatan makna semantiknya. Nilai-nilai yang dihasilkan divisualisasikan dengan plt.scatter() untuk menghasilkan gambar-gambar berikut.



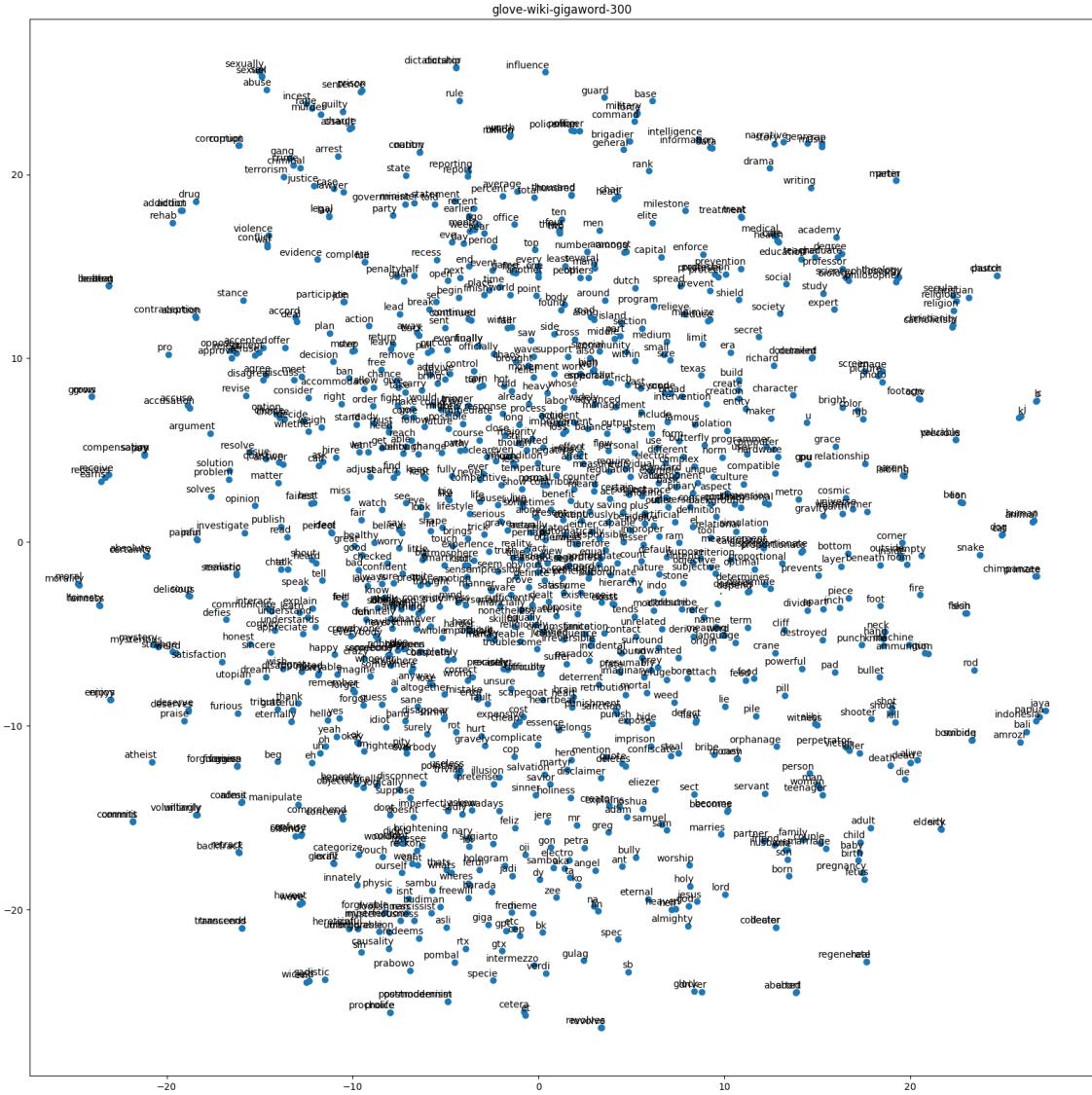
Gambar 5.4 Model word2vec-google-news-300



Gambar 5.5 Model glove-wiki-gigaword-50



Gambar 5.6 Model glove-wiki-gigaword-100



Gambar 5.7 Model glove-wiki-gigaword-300

Hasil percobaan dengan keempat model diatas menunjukkan penangkapan makna semantik yang lebih baik dari `text8` dan model yang di-train sendiri. Selain itu, lebih sedikit *word-token* yang tidak ada dalam *vocabulary* model, sehingga mengurangi *data loss* yang terjadi dalam upaya menangkap fitur dokumen.

### 5.3. Sentence Embedding

Dalam penelitian ini, *sentence-embedding* dan *document-embedding* adalah istilah yang bisa saling ditukar karena mengikuti prinsip-prinsip yang sama. Sebuah *sentence-embedding* dari sebuah dokumen terbuat dari agregasi seluruh *word-vector* yang ada dalam dokumen tersebut yang didapatkan dari model *word-embedding* yang digunakan. Agregasi yang dicoba adalah *averaging* dan *sum*, yang akan menghasilkan sebuah *sentence-vector* dengan dimensi yang sama

dengan dimensi *word-vector* yang didapatkan. Selain itu, diadakan pula opsi untuk melakukan *weighting* yang didapatkan dari TF-IDF *score* tiap kata dalam tiap dokumen. Dilakukan perbandingan sebelum kombinasi parameter untuk mengevaluasi relevansi *weighting* dengan TF-IDF.

Segmen Kode 5.2 Visualisasi kinerja fungsi agregat dan *weighting* pada *sentence-embedding*

```
from sklearn.manifold import TSNE

# returns a tsne shrinkage also...
def plot_documents(df):
    labels = np.array(df["No"])
    values = list(df["Document Embed"]) # don't forget to list
it first, then np array it later.

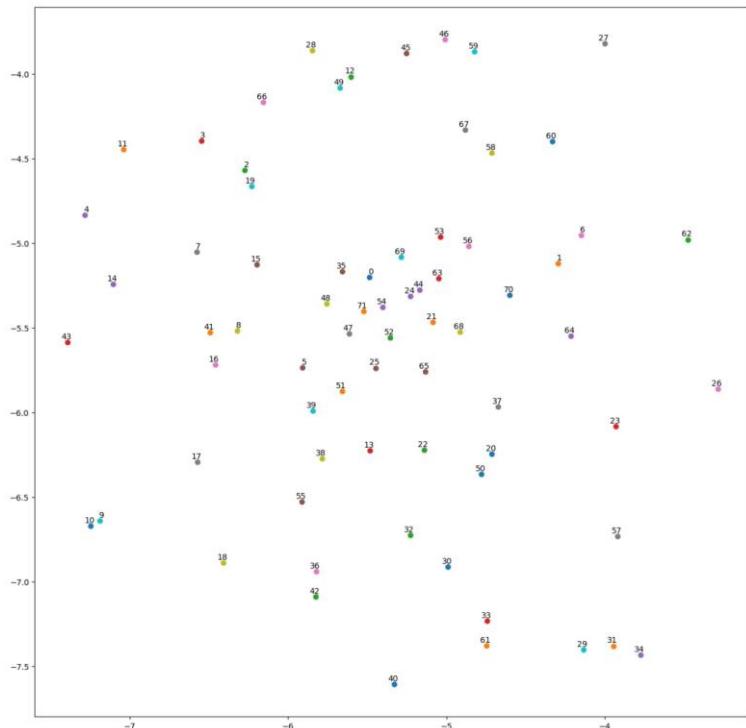
    # train model
    tsne_model      =     TSNE(perplexity=40,      n_components=2,
init='pca', n_iter=2500, random_state=23)
    new_values = tsne_model.fit_transform(np.array(values))

    # plot
    x = []
    y = []
    for value in new_values:
        x.append(value[0])
        y.append(value[1])

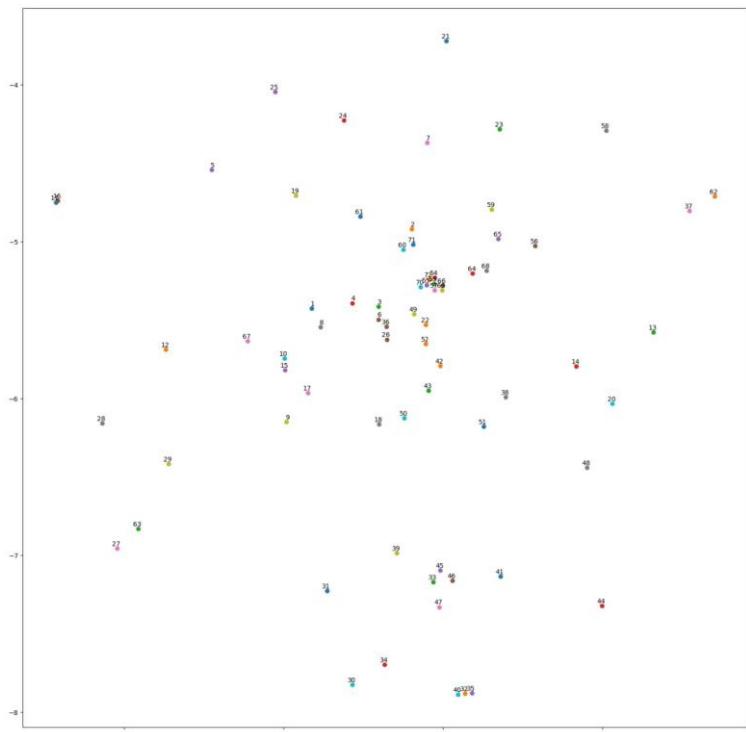
    plt.figure(figsize=(20, 20))
    for i in range(len(x)):
        plt.scatter(x[i],y[i])
        plt.annotate(labels[i],
                    xy=(x[i], y[i]),
                    xytext=(5, 2),
                    textcoords='offset points',
                    ha='right',
                    va='bottom')

    plt.show()
```

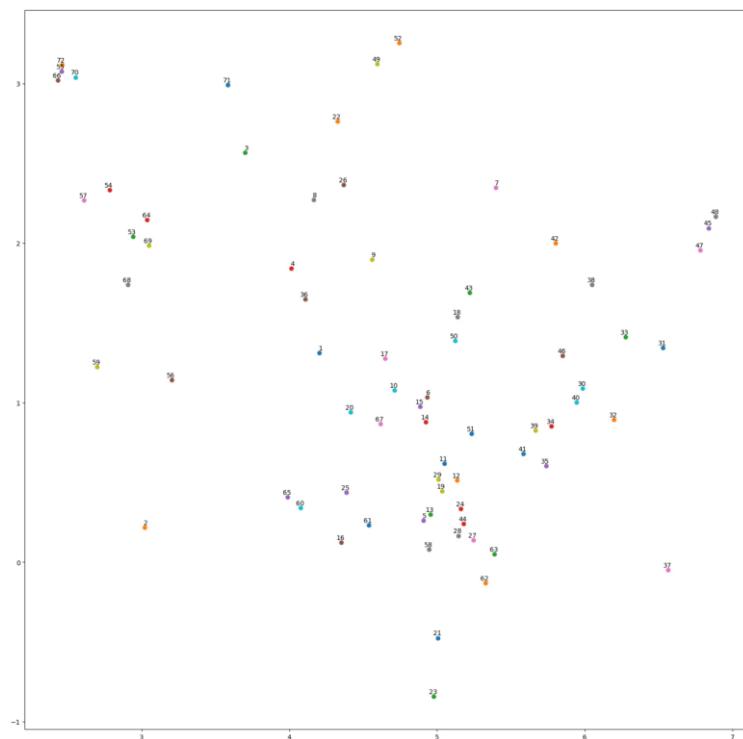
```
# use the thing to find new clusters.  
return new_values
```



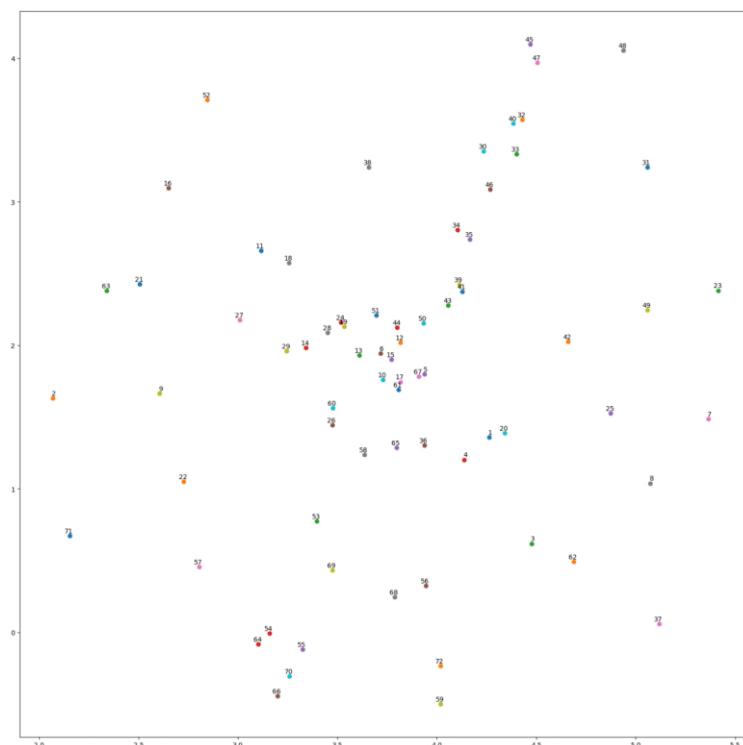
Gambar 5.8 Sentence-embedding dengan averaging, tanpa weighting



Gambar 5.9 Sentence-embedding dengan averaging dan weighting dengan TF-IDF score



Gambar 5.10 Sentence-embedding dengan sum, tanpa weighting



Gambar 5.11 Sentence-embedding dengan sum dan weighting dengan TF-IDF score

Dari percobaan-percobaan yang dilakukan untuk membandingkan metode agregat dan *weighting* yang digunakan, ditemukan bahwa adanya *weighting* setiap *word-token* dengan TF-

IDF sebelum agregasi menyebabkan *cluster-cluster* yang bisa diidentifikasi secara visual. Digunakan parameter *weight* dalam pengujian untuk melihat pengaruh *weighting* dalam pembuatan fitur-fitur data yang menjadi input algoritma *anomaly detection* yang digunakan.

#### 5.4. Tiny Word-Embedding

*Tiny word-embedding* adalah istilah yang mengacu pada agregasi semua dimensi dalam *word-token* yang digunakan menjadi satu angka yang merepresentasikan *word-token* tersebut. Dengan pendekatan ini, didapatkan vektor n-dimensi untuk setiap dokumen, dimana n adalah jumlah *word-token* yang ada dalam dokumen tersebut. Agregasi ini dimaksudkan untuk memberikan fokus lebih pada komposisi dokumen dan mempermudah perhitungan.

Segmen Kode 5.3 *Tiny word-embedding*

```
def TinyWordEmbed(self, document: list, model, maxLength: int):
    features = []
    # this creates a feature length of len(document)
    for word in document:
        if word in model:
            features.append(np.mean(model[word], axis=0))
    # if less than maxLength, pad with zeros.
    if len(features) < maxLength:
        padLength = maxLength - len(features)
        padding = np.zeros(padLength)
        features = np.concatenate((features, padding))
    return features
```

Sesuai dengan contoh kode diatas, *tiny word-embedding* dijalankan dengan terlebih dahulu diketahui jumlah token terbanyak dalam sebuah dokumen dalam dataset yang dijalankan. Setiap *word-token* akan diubah menjadi satu angka *word-value*, yang berfungsi sebagai satu dimensi dalam sebuah dokumen. Jika jumlah dimensi kurang dari panjang maksimum dokumen, akan dilakukan *padding* dengan angka 0. Secara umum, hasil *feature extraction* dengan metode ini tidak lebih baik dari teknik *sentence-embedding* yang dilakukan, karena terlalu banyak informasi yang hilang dalam agregasi 50, 100, atau 300 dimensi kedalam satu angka.

#### 5.5. Weighting dengan TF-IDF

*Weighting* dengan TF-IDF dilakukan dengan harapan bahwa TF-IDF score sebuah *word-token* dalam sebuah dokumen meningkatkan atau mengurangi kontribusi word-token tersebut dalam dokumennya berdasarkan relevansinya terhadap token-token lain dalam dokumen yang sama, dan token-token yang sama dalam dokumen yang lain. Oleh karena itu, *stop-word removal* pada *text preprocessing* sangat berpengaruh pada hasil TF-IDF yang baik. Untuk tujuan ini, *text preprocessing* dilakukan dengan tujuan mengurangi *stop-words* sebanyak mungkin sehingga hanya token-token inti yang unik yang tersisa dalam setiap dokumen yang ada. *Weighting* dilakukan pada metode *feature extraction* *word-embedding* dan *topic modelling* untuk tujuan yang sama.

Segmen Kode 5.4 Weighting pada *sentence-embedding*

```
"""
input :
list<list<(str, list<float>[300])>> : word-vector pair list
matrix : tf-idf matrix for the corresponding doc
int : the row we want
str : preferred agg method
"""

# output : list<float>[300] : 300-d vector that represents an
aggregated value of the input words

def SentenceEmbedWeightedFunction(self, word_embed_pair_list:
list, tfidf_matrix, index: int, aggregateMethod: str = "avg"):
    weighted_wvs = []
    # multiplies each word with its TF-IDF value in the
corresponding row. Is 0 if word isn't found somehow.
    for pair in word_embed_pair_list:
        tfidf_weight = 0
        if pair[0] in tfidf_matrix:
            tfidf_weight = tfidf_matrix[pair[0]][index]
        weighted_wvs.append(pair[1] * tfidf_weight)
    # turn into array for fast aggregating
    weighted_wvs = np.array(weighted_wvs)
    if aggregateMethod == "avg":
        sentence_vector = np.mean(weighted_wvs, axis=0)
```

```

else:
    sentence_vector = np.sum(weighted_wvs, axis=0)
    return sentence_vector

# input : list<list<(str, list<float>[300])>>, str : list
containing word-vector pairs, TF-IDF matrix of the corpus, and
preferred agg method
# output : list<(str, list<float>[300])> : list containing
sentence-vector pairs.

def SentenceEmbedWeighted(self, word_embedded_docs: list,
tfidf_matrix, aggregateMethod="avg"):
    sentence_embedded_docs = []
    for i in range(len(word_embedded_docs)):

        sentence_embedded_docs.append(self.SentenceEmbedWeightedFunction(
            word_embedded_docs[i], tfidf_matrix, i,
            aggregateMethod))

    return sentence_embedded_docs

```

Segmen Kode 5.5 *Tiny Word-embedding* dengan *weight*

```

def TinyWordEmbedWeighted(self, document: list, model,
maxLength: int, index: int):
    features = []
    for word in document:
        if word in model:
            weight = 0
            if word in self.tfidf_df:
                weight = self.tfidf_df[word][index]
            features.append(
                np.mean(model[word], axis=0) * weight)

    # if less than maxLength, pad with zeros.
    if len(features) < maxLength:
        padLength = maxLength - len(features)

```

```

padding = np.zeros(padLength)
features = np.concatenate((features, padding))

# i still remain pessimistic that this would work.

return features

```

## 5.6. Topic Modelling dengan LDA

LDA sebagai metode *topic modelling* stokastik digunakan karena karakteristiknya yang mengandalkan probabilitas. Distribusi-distribusi yang diambil oleh LDA diambil dari pendekatan *bag-of-words*, yang berarti frekuensi word-token dalam dokumen-dokumen yang menjadi korpus sangat mempengaruhi probabilitas distribusi *word-token* per topik, dan distribusi topik-topik yang ada per dokumen yang ada. Adapun jumlah topik yang ada dalam sebuah iterasi LDA ditentukan oleh pengguna. Dalam penelitian ini, karena jumlah dokumen dalam sebuah dataset kecil, digunakan jumlah topik yang kecil pula.

Jika pilihan *weighting* diberlakukan, LDA akan menggunakan matriks TF-IDF sebagai ganti distribusi *bag-of-words*. Probabilitas terpilihnya tiap *word-token* yang di-*assign* dalam sebuah topik tidak hanya dipengaruhi oleh frekuensi kata tersebut dalam dataset secara keseluruhan, namun juga komposisi dan relevansi *word-token* tersebut dalam dokumen yang bersangkutan. Oleh karena itu, kombinasi parameter akan menyertakan *weighted* LDA untuk menentukan apakah pendekatan ini lebih baik daripada hanya menghitung frekuensi *word-token*.

## 5.7. Hasil Kuesioner

Validasi hasil eksperimen akan diuji berdasarkan kemiripan hasil-hasil eksperimen dengan *ground truth* yang disediakan oleh para penyelenggara diskusi, yang merupakan pengguna program dan *domain expert*. *Ground truth* yang diperoleh berupa *majority vote* untuk setiap rekor data dalam kelima dataset yang digunakan, yang menentukan apakah rekor data tersebut koheren dengan pertanyaan diskusi atau tidak. Terdapat 7 orang penyelenggara diskusi yang mengambil kuesioner.

## Who has responded?

### Email

christiancep96@gmail.com

xoffel2000@gmail.com

achandra10@student.ciputra.ac.id

jeremia.daniel98@gmail.com

sam.teonata@gmail.com

martin.alinggajaya@gmail.com

c14190067@john.petra.ac.id

Gambar 5.12 Para penyelenggara diskusi yang mengambil kuesioner

Data *ground truth* disimpan dalam format .csv dalam format *boolean*, dengan 1 menandakan rekor data yang dianggap anomali dan 0 menandakan rekor data yang normal. *Ground truth* untuk sebuah rekor data diambil dari majority *vote* responden kuesioner. Data yang didapatkan adalah sebagai berikut.

Segmen Kode 5.6 *Ground truth* untuk event dengan ID 18

```
id,anomaly  
1,0  
2,0  
3,0  
4,0  
5,1  
6,0  
7,1  
8,1  
9,0  
10,0  
11,0  
12,1  
13,1  
14,0  
15,0  
16,0  
17,1  
18,1  
19,0  
20,0  
21,0  
22,0  
23,1  
24,0  
25,0  
26,0  
27,1  
28,1  
29,1
```

Segmen Kode 5.7 *Ground truth* untuk event dengan ID 19

```
id,anomaly
```

```
30,0  
31,0  
32,0  
33,0  
34,0  
35,0  
36,0  
37,1  
38,0  
39,0  
40,0  
41,0  
42,0  
43,0  
44,0  
45,1  
46,0  
47,1  
48,0  
49,0  
50,1  
51,1  
52,1  
53,1  
54,1
```

Segmen Kode 5.8 *Ground truth* untuk *event* dengan ID 20

```
id,anomaly  
55,0  
56,0  
57,0  
58,0  
59,0  
60,0  
61,0
```

```
62,1  
63,0  
64,1  
65,1  
66,1  
67,1  
68,1  
69,1  
70,0  
71,0  
72,0  
73,0  
74,0
```

Segmen Kode 5.9 *Ground truth* untuk *event* dengan ID 27

```
id,anomaly  
118,0  
119,0  
120,1  
121,0  
122,0  
123,1  
124,0  
125,1
```

Segmen Kode 5.10 *Ground truth* untuk *event* dengan ID 28

```
id,anomaly  
100,0  
101,0  
102,1  
103,0  
104,0  
105,0
```

106,1
107,0
108,0
109,1
110,1
111,0
112,1
113,1
114,0
116,0
117,0

### 5.8. Kombinasi-Kombinasi Parameter

Kombinasi parameter-parameter percobaan adalah sebagai berikut.

Parameter	Nilai	Keterangan
WordEmbeddingModel	[ "word2vec-google-news-300", "glove-wiki-gigaword-50", "glove-wiki-gigaword-100", "glove-wiki-gigaword-300" ]	Model <i>word-embedding</i> yang digunakan. Keempat pilihan merupakan <i>pre-trained</i> model yang menghasilkan kinerja terbaik dari model-model yang sudah diuji.
FeatureExtraction	[ "LDA", "Embedding", "Tiny" ]	Metode <i>feature extraction</i> yang digunakan. <i>Embedding</i> merujuk pada pendekatan <i>sentence-embedding</i> , dan <i>Tiny</i> merujuk pada pendekatan <i>tiny word-embedding</i> .
IsWeighted	[ True, False ]	Parameter weighting untuk parameter FeatureExtraction dengan TF-IDF score. Penerapan weighting telah

		dibahas dalam subbab-subbab 5.3, 5.4, 5.5, dan 5.6.
AnomalyDetection	["DBSCAN", "IF", "LOF"]	Metode <i>anomaly detection</i> yang digunakan. LOF mengacu pada Local Outlier Factor dan IF mengacu pada Isolation Forest.
LDATopic	[5, 10, 20]	Parameter <code>n_topics</code> pada LDA, untuk menentukan berapa jumlah topik yang diasumsikan.
epsilon	$\text{epsilon} \in [0.1, 2.0] \cap (0.1 * \mathbb{Z})$	Parameter untuk DBSCAN, menyatakan radius dimana poin-poin data dianggap berdekatan dengan poin data yang sedang dievaluasi.
minsamp	$2 \leq \text{minsamp} \leq 5$	Parameter untuk DBSCAN, menyatakan jumlah poin-poin yang berdekatan untuk suatu poin data bisa dianggap poin data inti.
contamination	$\text{contamination} \in [0.1, 0.5] \cap (0.1 * \mathbb{Z})$	Parameter untuk LOF dan IF, menyatakan asumsi bagian dari dataset yang dianggap anomali.
neighbor	$2 \leq \text{neighbor} \leq 5$	Parameter untuk LOF, menandakan jumlah <i>neighbor</i> yang ditentukan.
estimator	[50, 100, 500]	Parameter untuk IF, menandakan jumlah <i>tree estimator</i> yang akan dipakai.

Pengujian parameter dilakukan dengan mendapatkan kombinasi parameter yang ada untuk setiap dataset, lalu menjalankan fungsi `AnomalyDetector.GetAnomalies()` pada setiap kombinasi parameter dalam setiap dataset tersebut. Hasil *anomaly detection* di-*encode* secara biner dengan nilai 1 menandakan rekor data yang merupakan anomali, dan nilai 0 menandakan rekor data yang bukan anomali. Setelah itu hasil *encode* akan dicari Jaccard Index-nya dengan *ground truth* untuk dataset yang bersangkutan.

Segmen Kode 5.11 Fungsi untuk mencari kombinasi-kombinasi parameter.

```
EventIDList = [18, 19, 20, 27, 28] # add later
def GetCombo(currentEventID:int):
    FeatureExtractionList = ["LDA", "Embedding", "Tiny"]
    IsWeightedList = [True, False]
    AnomalyDetectionList = ["DBSCAN", "LOF", "IF"]
    LDATopicList = [5, 10, 20]
    WordEmbeddingModelList = ["word2vec-google-news-300",
                                "glove-wiki-gigaword-50",
                                "glove-wiki-gigaword-100",
                                "glove-wiki-gigaword-300"]
    epsilonRange = np.floor(np.arange(0.1, 2.1, 0.1) * 10)/10
    minsampRange = np.arange(2, 6)
    contaminationRange      = np.floor(np.arange(0.1, 0.6,
0.1)*10)/10
    neighborRange = np.arange(2, 6)
    estimatorList = [50, 100, 500]

    combinations = []
    for WordEmbeddingModel in WordEmbeddingModelList:
        for FeatureExtraction in FeatureExtractionList:
            for IsWeighted in IsWeightedList:
                for AnomalyDetection in AnomalyDetectionList:
                    if AnomalyDetection == "DBSCAN":
                        for minsamp in minsampRange:
                            for epsilon in epsilonRange:
                                if FeatureExtraction == "LDA":
```

```

for      LDATopic      in
LDATopicList:
    combination = {
        "EventID"      :
currentEventID,
        "method"       :
FeatureExtraction,
        "weighted"     :
IsWeighted,
        "algorithm"    :
AnomalyDetection,
        "n_topics"     :
LDATopic,
        "epsilon"      :
epsilon,
        "minsample"   : minsamp
    }

combinations.append(combination)
elif      FeatureExtraction == "Embedding" or FeatureExtraction == "Tiny":
    combination = {
        "EventID"      :
currentEventID,
        "method"       :
FeatureExtraction,
        "weighted"     :
IsWeighted,
        "algorithm"    :
AnomalyDetection,
        "Model"        :
WordEmbeddingModel,
        "epsilon"      : epsilon,
        "minsample"   : minsamp
    }

```

```

combinations.append(combination)
        elif AnomalyDetection == "LOF":
            for contamination in
contaminationRange:
                for neighbor in neighborRange:
                    if FeatureExtraction == "LDA":
                        for LDATopic in
LDATopicList:
                            combination = {
                                "EventID" :
currentEventID,
                                "method" :
FeatureExtraction,
                                "weighted" :
IsWeighted,
                                "algorithm" :
AnomalyDetection,
                                "n_topics" :
LDATopic,
                                "contamination":-
contamination,
                                "neighbors":-
neighbor
                            }
combinations.append(combination)
        elif FeatureExtraction ==
"Embedding" or FeatureExtraction == "Tiny":
            combination = {
                "EventID" :
currentEventID,
                "method" :
FeatureExtraction,

```

```

        "weighted" : IsWeighted,
        "algorithm" : AnomalyDetection,
        "Model" : WordEmbeddingModel,
        "contamination": contamination,
        "neighbors": neighbor
    }

combinations.append(combination)
elif AnomalyDetection == "IF":
    for contamination in contaminationRange:
        for estimator in estimatorList:
            if FeatureExtraction == "LDA":
                for LDATopic in LDATopicList:
                    combination = {
                        "EventID" : currentEventID,
                        "method" : "weighted",
                        "IsWeighted" : IsWeighted,
                        "algorithm" : AnomalyDetection,
                        "n_topics" : LDATopic,
                        "contamination": contamination,
                        "estimators": estimator
                    }

```

```

combinations.append(combination)

        elif FeatureExtraction == "Embedding" or FeatureExtraction == "Tiny":
            combination = {
                "EventID" : currentEventID,
                "method" : FeatureExtraction,
                "weighted" : IsWeighted,
                "algorithm" : AnomalyDetection,
                "Model" : WordEmbeddingModel,
                "contamination": contamination,
                "estimators": estimator
            }

combinations.append(combination)
return combinations

```

Segmen Kode 5.12 Fungsi untuk menjalankan *anomaly detection* untuk setiap kombinasi parameter

```

def RunCombo(currentCombo):
    currentEventID = 0
    currentModel = "word2vec-google-news-300"
    resultDict = {}
    for combo in currentCombo:
        # set eventID
        if combo["EventID"] != currentEventID:
            ad.SetDFFromDB(eventID=combo ["EventID"])

```

```

currentEventID = combo["EventID"]

# set model
if "Model" in combo and combo["Model"] != currentModel:
    print("current model : ", combo["Model"])
    ad.SetModel(combo["Model"])
    currentModel = combo["Model"]
    print("changed to : ", combo["Model"])

# add keys
for key in ad.AnomalyDetectionParams.keys():
    if key in combo:
        ad.AnomalyDetectionParams[key] = combo[key]
for key in ad.FeatureExtractionParams.keys():
    if key in combo:
        ad.FeatureExtractionParams[key] = combo[key]

# change model name
if "Model" in combo:
    if combo["Model"] == "word2vec-google-news-300":
        modelShort = "n300"
    elif combo["Model"] == "glove-wiki-gigaword-50":
        modelShort = "g50"
    elif combo["Model"] == "glove-wiki-gigaword-100":
        modelShort = "g100"
    elif combo["Model"] == "glove-wiki-gigaword-300":
        modelShort = "g300"

# change parameters
if combo["algorithm"] == "DBSCAN":
    if combo["method"] == "LDA":
        filePath =
f'{combo["EventID"]}_{ad.FeatureExtractionParams["method"]}_{a
d.FeatureExtractionParams["n_topics"]}_{ad.FeatureExtractionPa
rams["weighted"]}_{ad.AnomalyDetectionParams["algorithm"]}_{ad

```

```

.ad.AnomalyDetectionParams["epsilon"] }_{ad.AnomalyDetectionParams
["minSamp"] }.csv'
        elif      combo["method"]      ==      "Embedding"      or
combo["method"] == "Tiny":
            filePath
            =
f'{combo["EventID"] }_{ad.FeatureExtractionParams["method"] }_{m
odelShort}_{ad.FeatureExtractionParams["weighted"] }_{ad.Anomal
yDetectionParams["algorithm"] }_{ad.AnomalyDetectionParams["eps
ilon"] }_{ad.AnomalyDetectionParams["minSamp"] }.csv'
        elif  combo["algorithm"] == "LOF":
            if  combo["method"] == "LDA":
                filePath
                =
f'{combo["EventID"] }_{ad.FeatureExtractionParams["method"] }_{a
d.FeatureExtractionParams["n_topics"] }_{ad.FeatureExtractionPa
ram["weighted"] }_{ad.AnomalyDetectionParams["algorithm"] }_{ad
.AnomalyDetectionParams["contamination"] }_{ad.AnomalyDetection
Params["neighbors"] }.csv'
            elif      combo["method"]      ==      "Embedding"      or
combo["method"] == "Tiny":
                filePath
                =
f'{combo["EventID"] }_{ad.FeatureExtractionParams["method"] }_{m
odelShort}_{ad.FeatureExtractionParams["weighted"] }_{ad.Anomal
yDetectionParams["algorithm"] }_{ad.AnomalyDetectionParams["con
tamination"] }_{ad.AnomalyDetectionParams["neighbors"] }.csv'
            elif  combo["algorithm"] == "IF":
                if  combo["method"] == "LDA":
                    filePath
                    =
f'{combo["EventID"] }_{ad.FeatureExtractionParams["method"] }_{a
d.FeatureExtractionParams["n_topics"] }_{ad.FeatureExtractionPa
ram["weighted"] }_{ad.AnomalyDetectionParams["algorithm"] }_{ad
.AnomalyDetectionParams["contamination"] }_{ad.AnomalyDetection
Params["estimators"] }.csv'
                elif      combo["method"]      ==      "Embedding"      or
combo["method"] == "Tiny":

```

```

        filePath
        =
f' {combo["EventID"]} {_ad.FeatureExtractionParams["method"]} {_modelShort} {_ad.FeatureExtractionParams["weighted"]} {_ad.AnomalyDetectionParams["algorithm"]} {_ad.AnomalyDetectionParams["contamination"]} {_ad.AnomalyDetectionParams["estimators"]}.csv'

        print(combo["EventID"], " | ", filePath, " | " current
model : currentModel )
        result = ad.GetAnomalies()
        resultDict[filePath] = GetVerdict(combo["EventID"],
result)
        return resultDict

```

Segmen Kode 5.13 Fungsi untuk mendapatkan Jaccard Similarity antara *ground truth* dan hasil *anomaly detection*

```

from sklearn.metrics import jaccard_score

def GetJaccardScore(ground, result):
    outliers = np.where((ground == 1) | (result == 1))
    ground_outliers = ground[outliers]
    result_outliers = result[outliers]
    return jaccard_score(ground_outliers, result_outliers)

def GetVerdict(id, result):
    ground_labels
    =
np.genfromtxt('hasilsurvey/{}.csv'.format(id), delimiter=',',
skip_header=1, usecols=1)
    result_labels = result["Cluster Assignment"].to_numpy()
    return GetJaccardScore(ground_labels, result_labels)

```

## 5.9. Analisa Kesuksesan Algoritma

Kelima dataset yang dikumpulkan dapat dibedakan menjadi dua kategori: dataset dengan ID 18, 19, dan 20 merupakan dataset yang didapatkan dari notulen yang dicatat secara

manual oleh moderator diskusi, sehingga setiap rekord data cenderung pendek dan sarat dengan parafrase. Oleh karena itu, *word-tokens* yang didapatkan setelah *text preprocessing* cenderung pendek dan informasi yang disimpan tidak banyak. Dataset dengan ID 27 dan 28 merupakan dataset yang didapatkan dari hasil transkripsi pendapat setiap peserta diskusi, sehingga dokumen yang didapatkan lebih panjang. Setelah *preprocessing*, didapatkan jumlah *word-tokens* yang lebih banyak daripada dataset berupa notulen.

Terdapat 2530 jenis kombinasi untuk setiap dataset, sehingga mencapai 12650 total pemanggilan fungsi `AnomalyDetector.GetAnomaly()` untuk mendapatkan Jaccard Similarity Score yang terbaik. Untuk setiap dataset, diambil 10 kombinasi parameter yang mencapai Similarity Score tertinggi.

Tabel 5.1 10 Kombinasi parameter dengan indeks Jaccard terbaik untuk dataset dengan ID 18

Parameter Encoding	Similarity
18_EMBEDDING_g50_False_IF_0.5_100	0.5625
18_EMBEDDING_g50_False_LOF_0.5_5	0.5625
18_LDA_20_False_IF_0.5_50	0.5625
18_EMBEDDING_g50_False_IF_0.5_500	0.5625
18_EMBEDDING_g50_False_LOF_0.3_4	0.538462
18_EMBEDDING_g50_True_IF_0.3_50	0.538462
18_EMBEDDING_g300_False_LOF_0.3_4	0.538462
18_EMBEDDING_g300_False_LOF_0.3_3	0.538462
18_EMBEDDING_g300_False_LOF_0.3_2	0.538462
18_EMBEDDING_n300_False_LOF_0.3_5	0.538462

Tabel 5.2 10 Kombinasi parameter dengan indeks Jaccard terbaik untuk dataset dengan ID 19

Parameter Encoding	Similarity
19_LDA_20_False_DBSCAN_0.4_2	0.583333
19_LDA_10_False_DBSCAN_0.5_3	0.555556
19_LDA_20_True_DBSCAN_0.9_3	0.5
19_LDA_5_False_DBSCAN_0.8_5	0.5
19_EMBEDDING_g50_False_IF_0.4_50	0.5
19_Tiny_g50_False_DBSCAN_0.1_2	0.470588
19_LDA_20_False_LOF_0.5_2	0.461538
19_LDA_10_False_DBSCAN_1.1_5	0.454545
19_LDA_5_True_LOF_0.3_3	0.454545
19_LDA_20_True_DBSCAN_0.2_2	0.444444

Tabel 5.3 10 Kombinasi parameter dengan indeks Jaccard terbaik untuk dataset dengan ID 20

Parameter Encoding	Similarity
20_LDA_10_True_DBSCAN_0.7_4	0.636364
20_LDA_5_True_LOF_0.3_4	0.625
20_EMBEDDING_g300_True_DBSCAN_0.5_2	0.545455
20_EMBEDDING_g100_True_LOF_0.5_3	0.545455
20_EMBEDDING_g100_True_LOF_0.5_4	0.545455
20_EMBEDDING_g100_True_LOF_0.5_5	0.545455
20_EMBEDDING_g300_True_LOF_0.5_5	0.545455
20_EMBEDDING_n300_True_IF_0.5_50	0.545455
20_EMBEDDING_n300_True_LOF_0.5_5	0.545455
20_EMBEDDING_g300_True_LOF_0.5_2	0.545455

Tabel 5.4 10 Kombinasi parameter dengan indeks Jaccard terbaik untuk dataset dengan ID 27

Parameter Encoding	Similarity
27_Tiny_g100_False_LOF_0.3_3	1
27_Tiny_g50_False_LOF_0.4_4	1
27_Tiny_g50_False_IF_0.3_500	1
27_Tiny_g50_False_IF_0.4_50	1
27_Tiny_g50_False_IF_0.4_100	1
27_Tiny_g50_False_IF_0.4_500	1
27_Tiny_g100_False_IF_0.4_500	1
27_Tiny_g100_False_IF_0.4_100	1
27_Tiny_g100_False_IF_0.4_50	1
27_Tiny_g100_False_IF_0.3_500	1

Tabel 5.5 10 Kombinasi parameter dengan indeks Jaccard terbaik untuk dataset dengan ID 28

Parameter Encoding	Similarity
28_LDA_5_False_LOF_0.3_3.csv	0.571429
28_LDA_10_False_DBSCAN_0.7_4.csv	0.555556
28_EMBEDDING_g300_False_IF_0.5_50.csv	0.555556
28_LDA_10_False_DBSCAN_0.6_5.csv	0.555556
28_LDA_20_False_DBSCAN_1.0_3.csv	0.545455
28_LDA_10_True_LOF_0.4_3.csv	0.5
28_LDA_10_True_DBSCAN_0.7_3.csv	0.5
28_LDA_10_False_DBSCAN_0.8_3.csv	0.454545
28_EMBEDDING_g50_True_LOF_0.4_2.csv	0.444444
28_LDA_5_True_LOF_0.4_3.csv	0.444444

## 6. KESIMPULAN

Berdasarkan eksperimen-eksperimen yang telah dilakukan, pencarian kombinasi parameter, serta evaluasi komponen-komponen yang ada, diperoleh kesimpulan-kesimpulan berikut:

- Performa *anomaly detection* dengan metode-metode yang diajukan secara umum tidak bekerja dengan baik karena kriteria anomali yang telah ditentukan *domain experts* tidak sepenuhnya sesuai dengan asumsi awal. Kriteria anomali dalam data transkrip diskusi Komunitas X didasarkan pada informasi-informasi implisit yang tidak tercatat dalam transkrip diskusi maupun notulen yang telah tersedia. Informasi yang ditangkap oleh metode-metode *feature extraction*, walaupun koheren secara semantik, tidak bisa mendapatkan informasi yang implisit yang tidak tercatat dalam teks tersebut.
- Oleh karena informasi-informasi implisit tersebut, pendekatan *unsupervised learning* umumnya gagal untuk menggambarkan fitur secara akurat. Anomali memiliki kemungkinan paling besar terdeteksi pada rekord-rekord data yang mempunyai *word-tokens* yang unik diantara rekord-rekord data lainnya dalam dataset yang bersangkutan, dengan asumsi proporsi data yang kurang lebih sama (*tidak ada imbalance*)
- Pendekatan yang ideal adalah *supervised learning* dengan klasifikasi, namun pendekatan ini membutuhkan jumlah dataset yang besar dengan anotasi yang

baik, serta *feature extraction* yang menggambarkan makna semantik tiap dokumen dengan koheren.

## DAFTAR PUSTAKA

- "Forum". (2022). Retrieved from Oxford Learner's Dictionaries.: <https://www.oxfordlearnersdictionaries.com/definition/english/forum?q=forum>
- Aghammazada, E. (2020). *kaggle.com*. Retrieved from Stemming vs Lemmatization NLP [Image]: <https://www.kaggle.com/getting-started/186152>
- Blei, D., Ng, A., & Jordan, M. (2001). Latent Dirichlet Allocation. *Advances in Neural Information Processing Systems 14 (NIPS 2001)*. Vancouver. Retrieved from <https://proceedings.neurips.cc/paper/2001/file/296472c9542ad4d4788d543508116cb> c-Paper.pdf
- Brownlee, J. (2020, April 6). *10 Clustering Algorithms With Python*. Retrieved from machinelearningmastery.com: <https://machinelearningmastery.com/clustering-algorithms-with-python/>
- Chandola, V., Banerjee, A., & Kumar, V. (2009, July). Anomaly Detection: A Survey. *ACM Computing Surveys*, Vol 41, No. 3, Article 15. Retrieved from <https://dl.acm.org/doi/10.1145/1541880.1541882>
- Chowdhary, K. (2020). Fundamentals of Artificial Intelligence. Springer. Retrieved from <https://dokumen.pub/qdownload/fundamentals-of-artificial-intelligence-1nbsped-8132239709-9788132239703.html>
- Curiskis, S., & al., e. (2019). An evaluation of document clustering and topic modelling in two online social networks: Twitter and Reddit. *Information Processing and Management* 57(2). Retrieved from <https://doi.org/10.1016/j.ipm.2019.04.002>
- Danziger, S., Levav, J., & Avnaim-Pesso, L. (2011). Extraneous factors in judicial decisions. *Proceedings of the National Academy of Sciences*, (pp. 6889-6892). doi:10.1073/pnas.1018033108
- Dictionary, C. (n.d.). *Lemmatisation*. Retrieved from <https://www.collinsdictionary.com/dictionary/english/lemmatize>
- Ester, M., Kriegel, H., Sander, J., & Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *KDD-96* (pp. 226-231). Munich: AAAI. Retrieved from <https://dl.acm.org/doi/10.5555/3001460.3001507>

- Firdaus, A. (2019, Aug 13). *Cara Kerja Word2Vec*. Retrieved from <https://medium.com/@afrizalfir/mengenal-word2vec-af4758da6b5d>
- Gautam, H. (2020, Mar 1). *Word Embedding: Basics*. Retrieved from <https://medium.com/@hari4om/word-embedding-d816f643140>
- GeekForGeeks. (2022, December 11). *Removing stop words with NLTK in Python*. Retrieved from [www.geeksforgeeks.org: https://www.geeksforgeeks.org/removing-stop-words-nltk-python/](https://www.geeksforgeeks.org/removing-stop-words-nltk-python/)
- Google. (2022, July 19). *Clustering Algorithms*. Retrieved from <https://developers.google.com/>: <https://developers.google.com/machine-learning/clustering/clustering-algorithms>
- Hvitfeld, E., & Silge, J. (2022, May 11). *Supervised Machine Learning for Text Analysis in R*. Retrieved from <https://smltar.com: https://smltar.com/tokenization.html>
- Jacobi, C., van Atteveldt, W., & Welbers, K. (2015). Quantitative analysis of large amounts of journalistic texts using topic modelling. *Digital Journalism*, 89–106. Retrieved from <https://doi.org/10.1080/21670811.2015.1093271>
- Kalepalli, Y., Tasneem, S., Phani Teja, P. D., & Manne, S. (2020). Control Systems (ICICCS). *Effective Comparison of LDA with LSA for Topic Modelling*. doi:10.1109/iciccs48265.2020.9120888
- Kannan, S., Gurusamy, V., Vijayarani, S., Ilamathi, J., & Nithya, M. (2014). Preprocessing Techniques for Text Mining. *International Journal of Computer Science & Communication Networks*, 7-16. Retrieved from [https://www.academia.edu/35015140/Preprocessing\\_Techniques\\_for\\_Text\\_Mining](https://www.academia.edu/35015140/Preprocessing_Techniques_for_Text_Mining)
- Katryn, R. (2020, September 28). *Text Preprocessing: Tahap Awal dalam Natural Language Processing (NLP)*. Retrieved from <https://medium.com/: https://medium.com/mandiri-engineering/text-preprocessing-tahap-awal-dalam-natural-language-processing-nlp-bc5fbb6606a>
- Kienle, A., & Ritterskamp, C. (2007). Facilitating asynchronous discussions in learning communities: The impact of moderation strategies. *Behaviour and Information Technology*, 26(1), 73-80. doi:10.1080/01449290600811594
- Kumar, E. (2013). *Natural Language Processing*. Retrieved from [https://books.google.com/books/about/Natural\\_Language\\_Processing.html?id=FpUBFNFuKWgC](https://books.google.com/books/about/Natural_Language_Processing.html?id=FpUBFNFuKWgC)
- Lai, S., Liu, K., He, S., & Zhao, J. (2016). How to Generate a Good Word Embedding. *IEEE Intelligent Systems*, 5-14. Retrieved from <https://arxiv.org/abs/1507.05523>

- Lee, J., & et.al. (2018). Ensemble Modeling for Sustainable Technology Transfer. *Sustainability*, X(7), 2278-. doi:10.3390/su10072278
- Mohammed, S., & Al-augby, S. (2020). LSA & LDA Topic Modeling Classification : Comparison study on E-books. *Indonesian Journal of Electrical Engineering and Computer Science*, 353-362. Retrieved from <https://ijeeecs.iaescore.com/index.php/IJEECS/article/view/20547>
- NLTK. (2022, December 12). Documentation. Retrieved from <https://www.nltk.org/>: <https://www.nltk.org/howto/stem.html>
- Pai, A. (2020, May 26). *What is Tokenization in NLP? Here's All You Need To Know*. Retrieved from [www.analyticsvidhya.com:](https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/) <https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/>
- Porter, M. F. (1980). An algorithm for suffix stripping. *Program: electronic library and information systems*, 130-137. doi:10.1108/eb046814
- Ribeiro, J. (2021, Feb 18). *What is Natural Language Processing (NLP), and why does it matter to you?* Retrieved from <https://towardsdatascience.com/what-is-natural-language-processing-nlp-and-why-does-it-matter-to-you-3cc4fb003940>
- Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, XXIV(5), 513-523. doi:doi:10.1016/0306-4573(88)90021-0
- Scikit Learn. (n.d.). *Demo of DBSCAN clustering algorithm*. Retrieved from scikit-learn.org: [https://scikit-learn.org/stable/auto\\_examples/cluster/plot\\_cluster\\_comparison.html](https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html)
- Williams, T., & Betak, J. (2019). A Comparison of LSA and LDA for the Analysis of Railroad Accident. *Journal of Ubiquitous Systems & Pervasive Networks*, 11-15. Retrieved from <https://iasks.org/articles/juspn-v11-i1-pp-11-15.pdf>
- Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). *Data Mining: Practical Machine Learning Tools and Techniques (Morgan Kaufmann Series in Data Management Systems)*. Massachusetts: Morgan Kaupfmann.
- Xu, D., & Tian, Y. (2015). A Comprehensive Survey of Clustering Algorithms. *Annals of Data Science*, 165-193. Retrieved from <https://link.springer.com/article/10.1007/s40745-015-0040-1>