

VOICE CLONING

VOICE CLONING

Clone Your Voice in 5 Seconds

Dinda Majesty
Politeknik Pos Indonesia



Kreatif Industri Nusantara

Penulis:

Dinda Majesty

ISBN : xxx-xxx-xxxxx-x-x

Editor:

Rolly Maulana Awangga

Penyunting:

Syafrial Fachrie Pane

Desain sampul dan Tata letak:

Dinda Majesty

Penerbit:

Kreatif Industri Nusantara

Redaksi:

Jl. Ligar Nyawang No. 2

Bandung 40191

Tel. 022 2045-8529

Email : awangga@kreatif.co.id

Distributor:

Informatics Research Center

Jl. Sariasisih No. 54

Bandung 40151

Email : irc@poltekpos.ac.id

Cetakan Pertama, 2021

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan dengan cara
apapun tanpa ijin tertulis dari penerbit

*'Jika Kamu tidak dapat
menahan lelahnya
belajar, Maka kamu harus
sanggup menahan
perihnya Kebodohan.'*

Imam Syafi'i

CONTRIBUTORS

DINDA MAJESTY, Informatics Research Center., Politeknik Pos Indonesia, Bandung,
Indonesia

ROLLY MAULANA AWANGGA, Informatics Research Center., Politeknik Pos Indone-
sia, Bandung, Indonesia

CONTENTS IN BRIEF

1 Sistem Text to Speech (TTS)	1
2 Neural Network	7
3 Sequence to Sequence (Seq2Seq)	13
4 Speech Synthesis	23
5 Voice Cloning	33
6 Tacotron-2	43
7 WaveNet	51
8 Single-Speaker Voice Cloning	63
9 Multi-Speaker Voice Cloning	69
10 Tutorial Pembuatan Multi-Speaker Voice Cloning	75
11 Tutorial Pembuatan Project Menggunakan Google Colab	125

DAFTAR ISI

Daftar Gambar	xiii
Daftar Tabel	xxi
Foreword	xxv
Kata Pengantar	xxvii
Acknowledgments	xxix
Acronyms	xxxi
Glossary	xxxiii
List of Symbols	xxxv
Introduction	xxxvii
<i>Dinda Majesty</i>	
1 Sistem Text to Speech (TTS)	1
1.1 Text to Speech (TTS)	1
1.1.1 Artificial Intelligence	2
1.1.2 Machine Learning	3
1.1.3 Deep Learning	3

1.2 Cara Kerja TTS	4
2 Neural Network	7
2.1 Neural Network	7
2.1.1 Sejarah Neural Network	9
2.1.2 Struktur Neuron Pada Otak Manusia	10
2.1.3 Struktur Neural Network	11
2.1.4 Cara Kerja Neural Network	11
3 Sequence to Sequence (Seq2Seq)	13
3.1 Model Sequence to Sequence	13
3.1.1 Arsitektur Encoder-Decoder	15
3.1.2 Kekurangan Model Encoder-Decoder	19
3.1.3 Sequence to Sequence dengan Attention	20
3.1.4 Rumusan Masalah untuk Pemodelan Sequence to Sequence	21
4 Speech Synthesis	23
4.1 Speech Synthesis	23
4.1.1 Sejarah Speech Synthesis	25
4.1.2 Perangkat Speech Synthesis	26
4.1.3 Teknologi Speech Synthesis	27
5 Voice Cloning	33
5.1 Voice Cloning	33
5.1.1 Manfaat Voice Cloning	34
5.1.2 Aplikasi Voice Cloning Terbaik	35
5.1.3 Pembuatan Voice Cloning	36
5.1.4 Dampak Negatif Voice Cloning	37
5.1.5 Dampak Positif Voice Cloning	39
5.1.6 Mendeteksi Deepfake Voice	40
6 Tacotron-2	43
6.1 MODEL BERBASIS PERHATIAN UNTUK PENGENALAN Pidato	45
6.1.1 Decoder	45
6.1.2 Loss Function	45
7 WaveNet	51

7.1	WaveNet	51
7.1.1	Sejarah Speech Synthesis	55
7.1.2	Perangkat Speech Synthesis	56
7.1.3	Teknologi Speech Synthesis	56
8	Single-Speaker Voice Cloning	63
8.1	Model SV2TTS	63
8.1.1	Speaker Encoder	64
8.1.2	Synthesizer	65
8.1.3	Vocoder	66
8.1.4	Alur Model SV2TTS	66
8.1.5	Mengukur Kualitas Hasil	67
9	Multi-Speaker Voice Cloning	69
9.1	Model SV2TTS	69
9.1.1	Speaker Encoder	70
9.1.2	Synthesizer	71
9.1.3	Vocoder	72
9.1.4	Alur Model SV2TTS	72
9.1.5	Mengukur Kualitas Hasil	73
10	Tutorial Pembuatan Multi-Speaker Voice Cloning	75
10.1	Pengenalan Anaconda	75
10.1.1	Instalasi Anaconda 3 Windows 10 x64	75
10.1.2	Update Anaconda dan Spyder	82
10.1.3	Instalasi Anaconda Ubuntu 19.04	82
10.1.4	Konfigurasi <i>Python</i>	86
10.2	Instalasi Pycharm	87
10.3	Membuat Akun Github	90
10.4	Download dan Install Git	94
10.5	Konfigurasi Git	101
10.6	Fork dan Clone Voice Cloning Repository	104
10.7	Menambahkan Interpreter ke dalam Voice Cloning Project	108
10.7.1	Download dan Install Python 3.7	108
10.7.2	Download dan Install FFmpeg	110
10.7.3	Tutorial Membuat Virtual Environment	116
10.7.4	Tutorial Membuat Conda Environment	116
10.7.5	Install Requirements	116

10.7.6	Download and install Pytorch dengan Anaconda	117
10.7.7	Download and install Pytorch dengan Pip	120
11	Tutorial Pembuatan Project Menggunakan Google Colab	125
11.1	Membuat Akun Google	126
11.2	Membuat New Project Pada Google Colab	131

DAFTAR GAMBAR

1.1	Korelasi antara Deep Learning, Machine Learning, dan Artificial Intelligence	2
1.2	Perbedaan Machine Learning dan Deep Learning	4
1.3	Cara Kerja Text to Speech (TTS)	5
2.1	Sel Saraf (Neuron)	8
2.2	McCulloch dan Pitts, penemu pertama Neural Network	9
3.1	Contoh Penggunaan Model Sequence to Sequence	14
3.2	Google Translate	14
3.3	Speech Recognition	15
3.4	Video Captioning	15
3.5	Rumus Hidden State	16
3.6	Encoder	16
3.7	Rumus Hidden State	17

3.8	Rumus Hidden State	17
3.9	Decoder	18
3.10	Arsitektur Encoder-Decoder Secara Keseluruhan	18
3.11	Chatbot dengan Seq2Seq Model	19
3.12	Gambar to Text	21
3.13	Mekanisme Attention pada Gambar to Text	21
4.1	Speech Synthesis	24
4.2	Stephen Hawking adalah salah satu orang paling terkenal yang menggunakan sintesis ucapan untuk berkomunikasi	26
5.1	Voice Cloning	34
5.2	Contoh Aplikasi Voice Cloning (SV2TTS Toolbox)	35
5.3	Spoofing Voice Biometrics	38
5.4	Voice Phising, Sumber: https://id.pinterest.com/pin/195765915039905230/	38
5.5	Smart Assistant, Sumber: https://blog.evolvemachinelearners.com/voice-assistant-its-history-twist-and-turn/	40
6.1	Blok Diagram dari Arsitektur Tacotron 2	44
6.2	Spektogram mel yang diprediksi : Seperti yang dapat Anda bandingkan dengan wilayah atas, ia memiliki banyak celah dan masih membutuhkan banyak pelatihan. Sisi kanan (hijau solid) hanya padding dalam satu batch.	46
6.3	Target mel-spektrogram	47
6.4	Perhatian (Seperti yang Anda lihat di sisi kiri bawah, sepertinya sedang belajar untuk menyelaraskan tetapi masih membutuhkan sekitar satu minggu pelatihan untuk mendapatkan diagonal yang sempurna untuk perhatian)	48
7.1	Speech Synthesis	54
8.1	Arsitektur SV2TTS umum Sumber: Jia, Zhang, dan Weiss et al.	63
8.2	Representasi visual dari embeddings, setiap warna adalah speaker yang berbeda Sumber: Jia, Zhang, dan Weiss et al.	64
8.3	Model Arsitektur Multi-Speaker Voice Cloning Sumber: Jia, Zhang, dan Weiss et al.	65
8.4	Training Synthesizer Sumber: Jia, Zhang, dan Weiss et al.	66

8.5	Vocoder Sumber: Jia, Zhang, dan Weiss et al.	66
8.6	Pengukuran Kualitas Model Sumber: Jia, Zhang, dan Weiss et al.	67
8.7	MOS Sumber: Jia, Zhang, dan Weiss et al.	68
8.8	Speech Similarity Sumber: Jia, Zhang, dan Weiss et al.	68
9.1	Arsitektur SV2TTS umum Sumber: Jia, Zhang, dan Weiss et al.	69
9.2	Representasi visual dari embeddings, setiap warna adalah speaker yang berbeda Sumber: Jia, Zhang, dan Weiss et al.	70
9.3	Model Arsitektur Multi-Speaker Voice Cloning Sumber: Jia, Zhang, dan Weiss et al.	71
9.4	Training Synthesizer Sumber: Jia, Zhang, dan Weiss et al.	72
9.5	Vocoder Sumber: Jia, Zhang, dan Weiss et al.	72
9.6	Pengukuran Kualitas Model Sumber: Jia, Zhang, dan Weiss et al.	73
9.7	MOS Sumber: Jia, Zhang, dan Weiss et al.	74
9.8	Speech Similarity Sumber: Jia, Zhang, dan Weiss et al.	74
10.1	Run Setup Anaconda	76
10.2	Setup Loading	76
10.3	Welcome to Anaconda Setup	77
10.4	<i>License Agreement</i>	77
10.5	<i>Just Me(recomended)</i>	78
10.6	<i>Pilih lokasi</i>	79
10.7	<i>Centang Anaconda to my PATH</i>	79
10.8	<i>Waiting Installation Complete</i>	80
10.9	<i>Installation Complete</i>	80
10.10	<i>Anaconda+JetBrains</i>	81
10.11	<i>Thanks for install Anaconda</i>	82
10.12	Gambar halaman download	83
10.13	Gambar install anaconda	83
10.14	Gambar eksekusi anaconda	84

10.15	Gambar anaconda license agreement	84
10.16	Gambar perintah yes or no	85
10.17	Gambar path anaconda	85
10.18	Gambar perintah install spyder3	86
10.19	Gambar setpath	87
10.20	Klik Next untuk Install Pycharm	88
10.21	Menentukan lokasi folder dan klik next	88
10.22	Add launchers dir to PATH	89
10.23	Klik Install	89
10.24	Reboot Now	90
10.25	Sign Up sumber: https://omcyber.com/cara-membuat-akun-github/	91
10.26	Form Daftar	91
10.27	Free Plan	92
10.28	Verifikasi Email	92
10.29	Verifikasi Email Address	93
10.30	Verifikasi Email Berhasil	93
10.31	Install Git	94
10.32	Lokasi Instalasi Git	95
10.33	Komponen Tambahan	95
10.34	Select Start Menu Folder	96
10.35	Choose Default Editor	96
10.36	PATH Environment	97
10.37	Choose SSH	97
10.38	Configuring the line ending	98
10.39	Configuring the terminal emulator	98
10.40	Configuring extra options	99
10.41	Install	99
10.42	Proses Install	100

10.43	Command Prompt (CMD)	100
10.44	Versi Git	100
10.45	Konfigurasi Username Git	101
10.46	Konfigurasi Email Git	101
10.47	Generate SSH key	102
10.48	Direktori SSH	102
10.49	SSH Key	103
10.50	Menu SSH dan GPG keys	103
10.51	New SSH key	103
10.52	Add SSH key	104
10.53	Fork Repotori	104
10.54	Klik Tombol Code	105
10.55	Copy Url	105
10.56	Git Clone Repotori	106
10.57	Proses Clone Repotori	106
10.58	Pycharm	107
10.59	Lokasi Folder Repotori	107
10.60	Pop Up Create Virtual Environment	108
10.61	Real-Time-Voice-Cloning project	108
10.62	Download Python 3.7	109
10.63	Add Python to PATH	109
10.64	Success Install Python 3.7	110
10.65	Check Python Version	110
10.66	Download FFmpeg	111
10.67	Download FFmpeg for Windows	111
10.68	Extract File Instalasi	112
10.69	Ganti Nama Folder	112
10.70	Move Folder	113

10.71	Move Folder	113
10.72	Environment Variables	114
10.73	Add FFmpeg to PATH	114
10.74	Save Changes	115
10.75	Check FFmpeg Version	115
10.76	Select Pytorch with Conda	116
10.77	Select Pytorch with Conda	116
10.78	Select Pytorch with Conda	117
10.79	Select Pytorch with Conda	117
10.80	Select Pytorch with Conda	118
10.81	Select Pytorch with Pip	119
10.82	Select Pytorch with Pip	119
10.83	Select Pytorch with Pip	119
10.84	Select Pytorch with Pip	120
10.85	Select Pytorch with Pip	120
10.86	Select Pytorch with Pip	121
10.87	Select Pytorch with Pip	121
10.88	Select Pytorch with Pip	122
10.89	Select Pytorch with Pip	122
10.90	Select Pytorch with Pip	122
10.91	Select Pytorch with Pip	123
11.1	<i>Create Google Account</i>	126
11.2	Isi form pendaftaran	127
11.3	Isikan Nomor Handphon	127
11.4	<i>Verify Phone Number</i>	128
11.5	<i>Form Personal Data</i>	129
11.6	<i>Get more from your number</i>	130
11.7	<i>Create Google Account</i>	130

11.8	<i>Create Google Account</i>	131
11.9	<i>Create New Project</i>	132
11.10	Rename File	132
11.11	Script Mounting Google Drive	133
11.12	<i>Add Text to Project</i>	133
11.13	<i>Change Directory</i>	134
11.14	<i>Script Clone Repository</i>	135
11.15	<i>Check Folder</i>	136
11.16	<i>Struktur Dataset</i>	140
11.17	<i>Preprocessing Speaker Encoder Model</i>	141

DAFTAR TABEL

Listings

11.1 Mounting Google Drive	132
11.2 Clone Repository	134
11.3 Config Dataset	136
11.4 Preprocessing Function	137
11.5 Preprocessing Encoder Model	138
11.6 Script to Run Preprocessing Speaker Encoder Model	140

FOREWORD

Sepatah kata dari Kaprodi, Kabag Kemahasiswaan dan Mahasiswa

KATA PENGANTAR

Buku ini berisi penjelasan teknologi serta tata cara yang harus dilakukan dalam pembuatan *voice cloning* berbahasa indonesia. Buku ini diharapkan dapat membantu orang-orang memahami teknologi terkini terkait *voice cloning* dan cara kerjanya, hal yang dibutuhkan dalam pembuatan *voice cloning* serta cara membuat voice cloning.

TIM PENULIS

Bandung, Jawa Barat

Desember, 2021

ACKNOWLEDGMENTS

Terima kasih atas semua masukan dari para dosen D4 Teknik Informatika Politeknik Pos Indonesia agar bisa membuat buku ini lebih baik dan lebih mudah dimengerti.

Terima kasih ini juga ditujukan khusus untuk dosen pembimbing yang telah membimbing dan membantu saya menyelesaikan pembuatan buku ini sebagai syarat kelulusan Internship.

D. M.

ACRONYMS

ACGIH	American Conference of Governmental Industrial Hygienists
AEC	Atomic Energy Commission
OSHA	Occupational Health and Safety Commission
SAMA	Scientific Apparatus Makers Association

GLOSSARY

chatbot	Merupakan sebuah teknologi layanan obrolan yang ditangani oleh sistem atau robot.
bash	Merupakan bahasa sistem operasi berbasiskan *NIX.
linux	Sistem operasi berbasis sumber kode terbuka yang dibuat oleh Linus Torvald

SYMBOLS

A Amplitude

$\&$ Propositional logic symbol

a Filter Coefficient

B Number of Beats

INTRODUCTION

DINDA MAJESTY

Informatics Research Center
Bandung, Jawa Barat, Indonesia

Perkembangan teknologi yang semakin maju dalam sistem *Text to Speech (TTS)* memungkinkan manusia membuat kloningan suara hanya dengan beberapa detik sampel suara. Hasil kloningan suara dapat dimodifikasi sesuai dengan inputan teks, sehingga kloningan suara tersebut akan mengucapkan kalimat yang sesuai dengan teks yang diinputkan. Suara yang dihasilkanpun terdengar alami mirip seperti suara manusia aslinya.

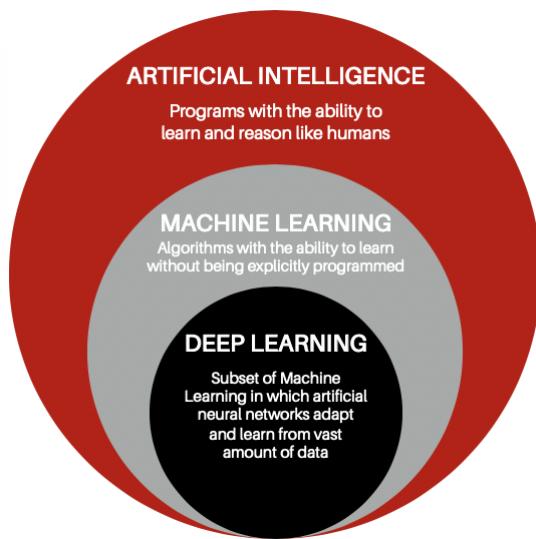
BAB 1

SISTEM TEXT TO SPEECH (TTS)

Sebelum mengetahui teknologi apa saja yang digunakan serta tata cara pembuatan sistem voice cloning ini, akan lebih baik apabila kita memahami terlebih dahulu mengenai sistem *Text to Speech (TTS)*.

1.1 Text to Speech (TTS)

Text to Speech merupakan proses pembuatan suara melalui inputan berupa text. mungkin teman-teman bingung, bagaimana bisa text berubah menjadi suara, semestinya text itu hanya dapat kita lihat menggunakan panca indra kita yaitu mata sedangkan suara merupakan sesuatu yang tidak bisa kita lihat menggunakan mata tetapi bisa kita dengar menggunakan telingga. Text to Speech ini ada untuk menjawab kebingungan teman-teman tersebut. Di zaman sekarang ini, dimana teknologi sudah semakin canggih, mengubah teks menjadi suara bukanlah hal yang sulit untuk dilakukan, apalagi setelah teman-teman mengenal deep learning, machine learning, dan artificial intelligence. Bagi teman-teman yang masih belum kenal, yuk kenalan dulu.



Gambar 1.1 Korelasi antara Deep Learning, Machine Learning, dan Artificial Intelligence

1.1.1 Artificial Intelligence

Artificial Intelligence (AI) merupakan kecerdasan yang dibuat dan diberikan atau diterapkan pada program komputer yang diharapkan memiliki kecerdasan seperti manusia sehingga memiliki kemampuan memahami, merencanakan, mengambil keputusan, dan problem solving. Dalam pembuatannya, setiap algoritma memungkinkan mesin untuk meniru, mengembangkan, dan berprilaku seperti manusia. Sederhananya AI itu merupakan sebuah teknologi yang memungkinkan kita menciptakan sebuah mesin yang memiliki kecerdasan berdasarkan pada perintah atau algoritma yang kita berikan pada program tersebut. algoritma berupa langkah-langkah yang kita lakukan dalam menyelesaikan suatu permasalahan. Misalnya kita ingin menciptakan mesin yang dapat menjawab semua pertanyaan kita seperti chatbot, maka kita membutuhkan algoritma atau langkah-langkah apa yang dilakukan oleh chatbot untuk memahami dan mampu memberikan respon secara cepat dan tepat kepada kita.

Artificial Intelligence dibuat dengan 3 tujuan, diantaranya:

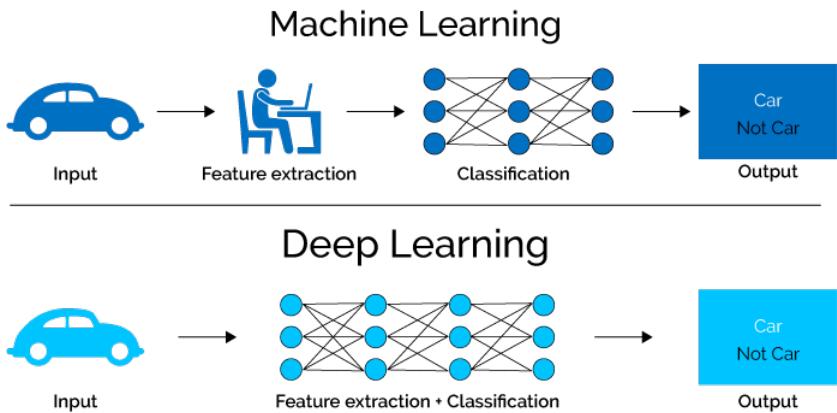
1. tujuan utama, Membuat mesin menjadi lebih pintar dan terus mengalami peningkatan seiring dengan perkembangan teknologi
2. tujuan ilmiah, Memahami apa itu kecerdasan buatan sehingga dapat dimanfaatkan untuk membuat mesin yang lebih cerdas dalam membantu dan memecahkan masalah secara efektif dan efisien.
3. tujuan entrepreneurial, Membuat mesin lebih bermanfaat sehingga memudahkan manusia dalam melakukan perkerjaan dan memecahkan masalah secara cepat, tepat, dan lebih teliti.

1.1.2 Machine Learning

Machine Learning (ML) merupakan bagian dari Artificial Intelligence yang ditambahkan dengan metode-metode statistika dengan tujuan agar mesin dapat meningkatkan pengalamannya berdasarkan kepada data-data yang ada dan dilatih pada mesin tersebut. Dengan begitu mesin dapat melakukan tugas tertentu berdasarkan data dan algoritma yang diterapkan padanya. Data dan Algoritma yang dirancang membuat mesin terus mengalami peningkatan dari waktu ke waktu. Sebagai contoh, pada pembuatan sistem untuk memprediksi tweet yang bersifat positif, negatif, atau netral kita akan membutuhkan data tweet dari para pengguna twitter, kita bisa dapatkan melalui API yang telah disediakan oleh twitter. pada tahap pertama, kita mengambil 100 data tweet yang terdiri dari tweet positif, negatif, dan netral. Lalu kita melatih mesin menggunakan 100 data tersebut dan menggunakan algoritma naive bayes untuk mengetahui apakah data hasil prediksi kita sama dengan data tweet aslinya dan menghitung akurasi dari algoritma yang telah kita buat. Pada tahap pertama akurasi menunjukkan angka 90% dan ketika kita melakukan training kembali kepada algoritma atau model yang telah kita buat dengan menggunakan data yang lebih banyak lagi maka mesin akan menjadi lebih terlatih dan kemungkinan kesalahan prediksi menjadi sangat kecil bahkan akurasi yang didapatkan dari model bisa mencapai 99% atau bahkan kita bisa mengganti algoritma yang kita gunakan menggunakan algoritma lainnya yang menghasilkan akurasi yang lebih baik dari model atau algoritma kita sebelumnya. hal inilah yang dimaksud dengan machine learning, ada data dan algorithma serta perhitungan-perhitungan yang sering kita jumpai dalam statistika seperti perhitungan akurasi, rata-rata (mean), median, recall, precision, dll.

1.1.3 Deep Learning

Setelah mengenal AI dan ML saatnya kita mengenal Deep Learning (DL) yang merupakan bagian dari Machine Learning yang berkaitan dengan algoritma yang berdasarkan pada struktur dan fungsi otak atau sering dikenal sebagai jaringan saraf tiruan. Pada dasarnya Deep Learning memiliki konsep seperti Machine Learning hanya saja dalam konteks yang lebih dalam. Contohnya, pada deep learning kita ingin membuat mesin dapat mengetahui apakah hewan yang ada digambar merupakan seekor anjing atau kucing, maka kita akan membuat algoritma untuk melakukan pengecekan seperti melakukan pengecekan pada bentuk ekor, bentuk telinga, apakah memiliki kumis atau tidak, dan ciri-ciri lainnya yang membuat kucing dan anjing terlihat berbeda tanpa kita perlu memberikan fitur pembedanya atau fitur mana yang lebih penting untuk dapat mengidentifikasi hewan tersebut secara manual. Deep Learning dapat mengetahui fitur tersebut tanpa kita beritahu secara manual seperti pada Machine Learning seperti terlihat pada gambar 1.2. Oleh karena itu Deep Learning dianggap sebagai otak utama yang menciptakan kecerdasan buatan yang lebih manusiawi.



Gambar 1.2 Perbedaan Machine Learning dan Deep Learning

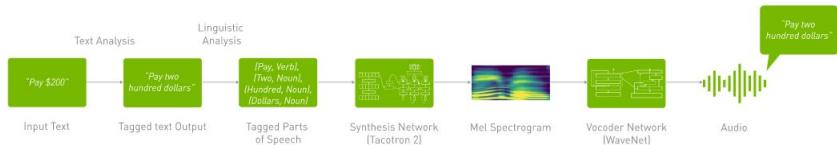
Text to Speech termasuk bagian dari Deep Learning lebih tepatnya Neural Network. Neural Network merupakan cabang ilmu yang mengadopsi cara berpikir dan cara bekerja otak manusia yang memberikan rangsangan/stimulus (input), melakukan proses, dan menghasilkan sesuatu (output). Output dihasilkan dari inputan yang diberikan dan proses yang dilakukan oleh otak manusia. Kemampuan otak untuk memproses inputan atau informasi ini yang akan membuat otak manusia selalu mempelajari hal-hal baru dan terus berkembang, begitupun dengan mesin yang diberikan kecerdasan buatan atau AI. Contohnya saja ketika kita berusia 5 tahun dan otak kita belum memahami informasi-informasi yang sulit seperti pelajaran matematika dasar, menambah dan mengurangi ataupun membagi dan mengalikan angka. Setelah kita mempelajarinya selama 6 tahun di Sekolah Dasar maka otak kita sekarang bisa memahaminya bahkan hal tersebut merupakan hal yang sangat gampang. Pada dasarnya konsep AI sama dengan otak manusia yang selalu berkembang dan mempelajari hal-hal yang baru berdasarkan inputan atau informasi apa yang diberikan kepada otak dan otak akan memproses inputan tersebut hingga dapat memahami dan mengeluarkan hasil dari proses tersebut (output). Dalam pembuatan voice cloning ini kita akan mengajarkan kepada mesin cara membaca text dan mengucapkannya dengan menggunakan suara seperti yang kita contohkan, Oleh karena itu kita membutuhkan sistem text to speech.

1.2 Cara Kerja TTS

Cara kerja TTS terdiri dari 2 proses, dapat dilihat pada gambar 1.3:

1. Model Teks ke Spektogram, model ini mengubah teks menjadi fitur yang selaras dengan waktu seperti spektogram, mel-spektogram, atau frekuensi F0 dan fitur linguistik lainnya. ada beberapa model yang bisa digunakan untuk mengubah teks menjadi spektogram, diantaranya yaitu Tacotron, Tacotron2, Deep Voice 3, dll. Model ini juga disebut sebagai model encoder-decoder.

2. Model spektrogram ke audio, model ini akan mengonversi spektrogram yang dihasilkan menjadi audio. Model ini disebut sebagai vocoder. beberapa model yang banyak digunakan untuk membuat vocoder yaitu WaveGlow, Griffin-Lim Algorithm, WaveNet, dll. Model Tacotron-2 sebagai model encoder-decoder dan model WaveNet yang dimodifikasi sebagai vocoder akan dibahas secara mendalam pada buku ini.



Gambar 1.3 Cara Kerja Text to Speech (TTS)

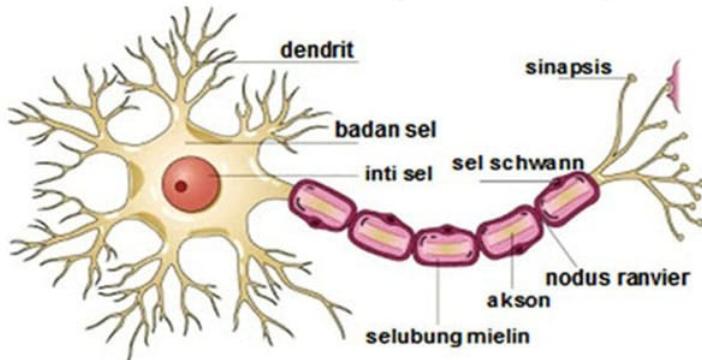
BAB 2

NEURAL NETWORK

2.1 Neural Network

Neural Network terinspirasi dari cara kerja neuron yang ada pada otak manusia, neuron bertugas sebagai penerima stimulus/rangsangan dan pengirim informasi yang akan diolah atau diproses oleh otak menjadi suatu output. Neuron merupakan sistem saraf pusat dan terdapat sekitar 100 miliar neuron yang ada pada tubuh manusia. Perhatikan gambar 2.1

Sel Saraf (Neuron)

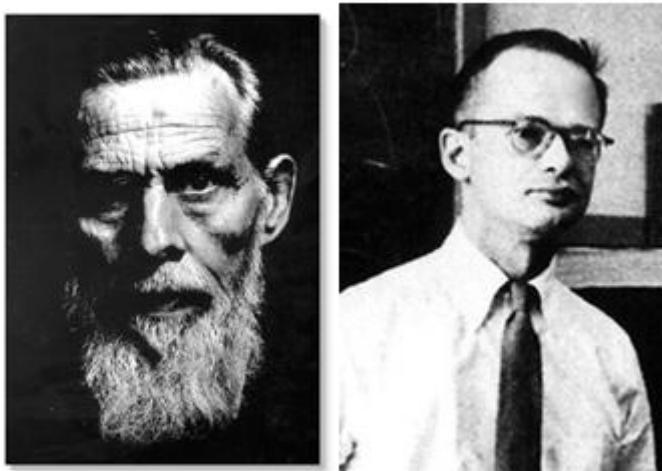


Gambar 2.1 Sel Saraf (Neuron)

Neuron terdiri dari 3 bagian yaitu akson (akar), soma (batang), dan dendrite (cabang). Neuron juga dibedakan menjadi 3 yaitu neuron sensorik atau sel saraf indera karena fungsinya yang berhubungan dengan penerima (indra) dan saraf pusat (otak dan sumsum tulang belakang). Neuron motorik atau sel saraf penggerak yang berfungsi membawa rangsangan dari saraf pusat (otak dan sumsum tulang belakang) ke otot. dan yang terakhir yaitu neuron asosiasi atau sel saraf penghubung, sel ini menghubungkan atau meneruskan rangsangan dari sel saraf sensorik ke sel saraf motorik.

Tiap neuron yang ada dalam otak kita saling terhubung dan berkirim informasi atau rangsangan berupa neurotransmitter, neuron yang saling berinteraksi dengan mengirimkan rangsangan ini akan menghasilkan kemampuan tertentu pada kerja otak kita. Contohnya, ketika kita bertemu dengan seorang kenalan yang menyapa kita maka otak akan berkerja sehingga dapat mengenali orang tersebut dan akan menghasilkan respon atau output, outputnya bisa berupa sapaan, lambaian tangan, obrolan, dan hal-hal lainnya yang biasa kita lakukan apabila bertemu dengan seseorang yang kita kenal. Jadi secara ilmiahnya, inputan atau rangsangan yang diterima yaitu berupa sapaan dari orang yang dikenali. Rangsangan ini akan diterima oleh alat indra (mata, telinga, hidung, lidah, dan kulit) melalui sel reseptör, kemudian akan diteruskan dalam bentuk impuls berupa arus listrik yang diteruskan ke sel saraf sensorik melalui sinapsis, lalu melewati sel saraf koneksi menuju otak. Pada otak informasi akan diolah terlebih dahulu kemudian dikirimkan ke sel saraf motorik dan memberikan atau menghasilkan reaksi berupa sapaan, gerakan berupa lambaian tangan dll. Inilah proses kerja otak manusia begitupun dengan neural network yang terinspirasi dari cara neuron bekerja pada otak manusia. Siapa sangka mesin yang merupakan benda mati akan bisa berprilaku dan berkerja seperti layaknya manusia. Hingga saat ini sangat banyak diminati dan terus dikembangkan dengan tujuan dapat membantu memudahkan manusia dalam bekerja.

2.1.1 Sejarah Neural Network



Gambar 2.2 McCulloch dan Pitts, penemu pertama Neural Network

Neural Network bermula ketika Warren McMulloch yang merupakan seorang neurofisiologi dan Walter Pitts yang merupakan seorang ahli matematika menulis makalah tentang cara kerja dari neuron pada tahun 1943. kemudian diperkuatnya konsep neuron dalam buku yang ditulis oleh Donald Hebb pada tahun 1949 yang berjudul *The Organization of Behavior* yang menunjukkan bahwa jaringan syaraf akan bertambah kuat setiap kali digunakan. kemudian dilanjutkan dengan penelitian di IBM untuk mensimulasikan neural network tahun 1950 yang dipimpin oleh Nathania Rochester.

Dengan diadakannya konferensi Dartmouth pada tahun 1956 yang membahas tentang penelitian neural network oleh John McCarthy memperkuat konsep mengenai neural network. Lalu pada tahun 1957, John Von Neumann menyarankan untuk meniru fungsi neuron menggunakan relay telegraf atau tabung vakum. Dengan adanya penemuan two-layer-network yang dikenal dengan perceptron. Perceptron berguna untuk menghitung jumlah input, mengurangi treshold, dan meneruskan salah satu dari dua nilai yang mungkin keluar sebagai hasil, penemuan ini ditemukan oleh Frank Rosenblatt pada tahun 1958.

Pada tahun 1959, diperkenalkan model neural network pertama yang dikenal dengan ADALINE (Adaptive Linear Elements) dan MEDALINE (Multiple Adaptive Linear Elements). Model ini merupakan model pertama yang diterapkan pada permasalahan yang ada di dunia nyata. Model ini berfungsi untuk menghilangkan gema pada saluran telepon. Pengembangan model ini dilakukan oleh Bernard Widrow dan Marcian Hoff dari Stanford.

Pada tahun 1982 dalam makalah yang dipresentasikan pada National Academy of Sciences tentang pendekatan untuk menciptakan perangkat yang berguna, menyenangkan, pandai berbicara dan kharismatik, makalah ini dipresentasikan oleh John

Hopfield. Lalu konferensi International Institute of Electrical and Electronics (IEEE) mengadakan konferensi mengenai Neural Network yang dihadiri oleh lebih dari 1.800 peserta pada tahun 1987. Sekarang, Neural Network telah diterapkan pada classification, approximation, prediction, recognition, memory simulation, clustering, dll.

2.1.2 Struktur Neuron Pada Otak Manusia

Ide dasar dari pembuatan Neural Network dimulai dari cara kerja otak manusia dalam belajar dan struktur otak manusia yang terdiri dari neuron yang saling terhubung, mengirim, dan menerima informasi. Satu neuron memiliki satu akson dan minimal satu dendrit. Setiap sel saraf terhubung dengan sel saraf lain yang saling berinteraksi dan menghasilkan kemampuan tertentu pada kerja otak manusia sehingga otak menjadi semakin pintar atau memiliki banyak kemampuan serta pengetahuan. Misalnya kemampuan otak ketika kita masih kecil dengan kemampuan otak kita saat dewasa tidak akan sama, karena otak kita selalu belajar hingga kita dapat memahami hal-hal yang baru dan kemampuan otak kita semakin berkembang.

Perhatikan gambar 2.1, pada gambar neuron terdiri dari:

1. Dendrit (Dendrites) berfungsi untuk mengirimkan impuls yang diterima berupa arus listrik ke badan sel saraf.
2. Akson (Axon) berfungsi untuk mengirimkan impuls dari badan sel ke jaringan lain.
3. Sinapsis berfungsi sebagai unit fungsional (penghubung) di antara dua sel saraf.
4. Selubung mielin
5. Nodus Ranvier
6. Nukleus
7. Soma
8. Sel Schwann

Berikut cara kerja otak manusia:

Ketika sebuah neuron menerima impuls dari neuron lain melalui dendrit, maka dendrit akan mengirimkan sinyal tersebut ke badan sel saraf. Selanjutnya Akson akan menerima sinyal dari badan sel dan mengirimkannya ke sel saraf lain. Akson dari sel saraf ini bercabang-cabang, akson sel saraf satu berhubungan dengan akson sel saraf dua melalui sinapsis. Sinapsis adalah unit fungsional antara 2 buah sel saraf, misal sel A dan sel B, akson sel A akan berhubungan dengan dendrit sel B melalui sinapsis. Kekuatan sinapsis bisa menurun/meningkat tergantung tingkat propagasi sinyal yang diterimanya. Impuls-impuls sinyal (informasi) akan diterima oleh neuron lain jika memenuhi batasan tertentu, dikenal dengan nilai ambang (threshold).

2.1.3 Struktur Neural Network

Dari struktur neuron pada otak manusia, dan proses kerja yang dijelaskan di atas, maka konsep dasar pembangunan neural network buatan (Artificial Neural Network) terbentuk. Ide mendasar dari Artificial Neural Network (ANN) adalah mengadopsi mekanisme berpikir sebuah sistem atau aplikasi yang menyerupai otak manusia, baik untuk pemrosesan berbagai sinyal elemen yang diterima, toleransi terhadap kesalahan/error, dan juga parallel processing.

Karakteristik dari ANN dilihat dari pola hubungan antar neuron, metode penentuan bobot dari tiap koneksi, dan fungsi aktivasinya. Gambar di atas menjelaskan struktur ANN secara mendasar, yang dalam kenyataannya tidak hanya sederhana seperti itu.

Input, berfungsi seperti dendrite Output, berfungsi seperti akson Fungsi aktivasi, berfungsi seperti sinapsis Neural network dibangun dari banyak node/unit yang dihubungkan oleh link secara langsung. Link dari unit yang satu ke unit yang lainnya digunakan untuk melakukan propagasi aktivasi dari unit pertama ke unit selanjutnya. Setiap link memiliki bobot numerik. Bobot ini menentukan kekuatan serta penanda dari sebuah koneksi.

Proses pada ANN dimulai dari input yang diterima oleh neuron beserta dengan nilai bobot dari tiap-tiap input yang ada. Setelah masuk ke dalam neuron, nilai input yang ada akan dijumlahkan oleh suatu fungsi perambatan (summing function), yang bisa dilihat seperti pada diagram dengan lambang sigma. Hasil penjumlahan akan diproses oleh fungsi aktivasi setiap neuron, disini akan dibandingkan hasil penjumlahan dengan threshold (nilai ambang) tertentu. Jika nilai melebihi threshold, maka aktivasi neuron akan dibatalkan, sebaliknya, jika masih dibawah nilai threshold, neuron akan diaktifkan. Setelah aktif, neuron akan mengirimkan nilai output melalui bobot-bobot outputnya ke semua neuron yang berhubungan dengannya. Proses ini akan terus berulang pada input-input selanjutnya.

ANN terdiri dari banyak neuron di dalamnya. Neuron-neuron ini akan dikelompokkan ke dalam beberapa layer. Neuron yang terdapat pada tiap layer dihubungkan dengan neuron pada layer lainnya. Hal ini tentunya tidak berlaku pada layer input dan output, tapi hanya layer yang berada di antaranya. Informasi yang diterima di layer input dilanjutkan ke layer-layer dalam ANN secara satu persatu hingga mencapai layer terakhir/layer output. Layer yang terletak di antara input dan output disebut sebagai hidden layer. Namun, tidak semua ANN memiliki hidden layer, ada juga yang hanya terdapat layer input dan output saja.

2.1.4 Cara Kerja Neural Network

Cara kerja Neural Network sama halnya dengan proses belajar yang dilakukan oleh otak manusia dengan menggunakan contoh atau disebut juga supervised learning. Neural network dikonfigurasikan pada aplikasi tertentu seperti pengklasifikasian data atau pengenalan pola. Neural Network memproses informasi seperti cara kerja otak manusia yang terdiri dari sejumlah besar elemen pemrosesan yang saling berhubungan dan berkerja secara paralel dalam menyelesaikan masalah. Neural Network da-

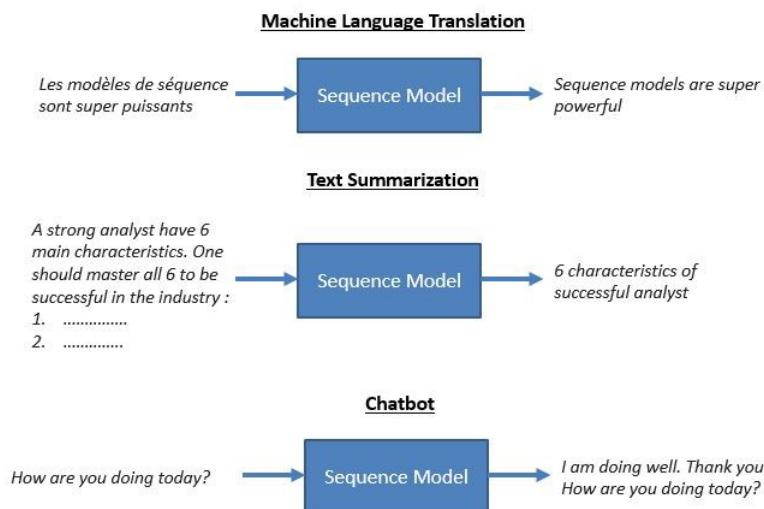
pat digunakan untuk memperoleh informasi atau pengetahuan dari data-data yang rumit, mengekstrak pola, mendeteksi tren yang memiliki kompleksitas yang rumit untuk dipelajari manusia ataupun teknik komputasi lainnya. Oleh karena itu, neural network yang telah dilatih hingga dapat menganalisis dan memproses informasi dari data dapat dikategorikan sebagai ahli. Neural Network sangat cocok untuk menyelesaikan beberapa masalah terkait prediksi yang membutuhkan pemahaman dan analisis yang baik contohnya prediksi pergerakan data time-series. Sedangkan algoritma komputer konvensional lebih cocok untuk menyelesaikan masalah terkait operasi aritmatika. Namun, pada beberapa masalah Neural Network dan Algoritma Komputer Konvensional dikombinasikan agar dapat memberikan kinerja maksimum.

BAB 3

SEQUENCE TO SEQUENCE (SEQ2SEQ)

3.1 Model Sequence to Sequence

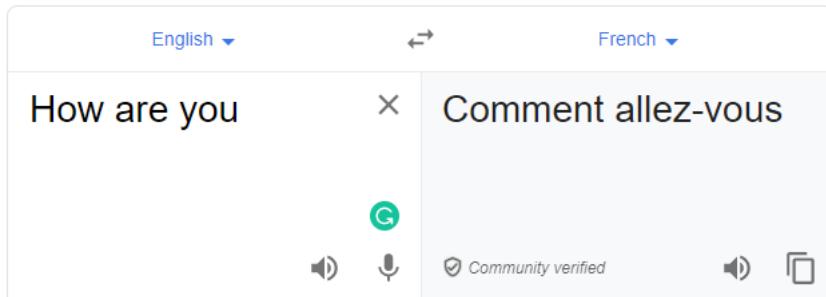
Model Sequence to Sequence (seq2seq) merupakan arsitektur Jaringan Saraf Berulang yang biasa digunakan untuk memecahkan masalah bahasa yang kompleks seperti Terjemahan, Menjawab Pertanyaan, membuat Chatbot, Peringkasan Teks, dll. Contoh penggunaan model seq2seq dapat dilihat pada gambar 3.1



Gambar 3.1 Contoh Penggunaan Model Sequence to Sequence

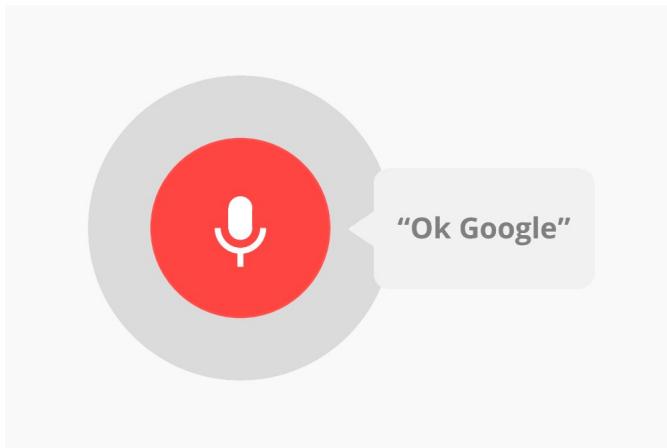
Model seq2seq sering dijumpai pada sistem yang sering kita gunakan sehari-hari. Model seq2seq mendukung aplikasi seperti Google Terjemahan, perangkat yang mendukung suara, voice assistant, chatbot, dll . Berikut ini adalah beberapa aplikasinya:

1. Google Translate, makalah tahun 2016 dari Google menunjukkan bagaimana kualitas terjemahan model seq2seq yang mendekati atau bahkan melampaui semua hasil yang dipublikasikan saat ini.



Gambar 3.2 Google Translate

2. Speech Recognition, makalah Google yang membandingkan model seq2seq yang ada pada pengenalan suara.



Gambar 3.3 Speech Recognition

3. Video Captioning – Secara otomatis membuat subtitle video untuk setiap frame, termasuk deskripsi isyarat suara (seperti mesin yang dinyalakan, orang yang tertawa di latar belakang, dll.).



Gambar 3.4 Video Captioning

Beberapa aplikasi tersebut membuat model seq2seq dipandang sebagai solusi terbaik. Model ini dapat digunakan sebagai solusi untuk setiap masalah berbasis urutan, terutama yang input dan outputnya memiliki ukuran dan kategori yang berbeda.

3.1.1 Arsitektur Encoder-Decoder

Arsitektur yang paling umum digunakan untuk membangun model Seq2Seq adalah arsitektur Encoder-Decoder.

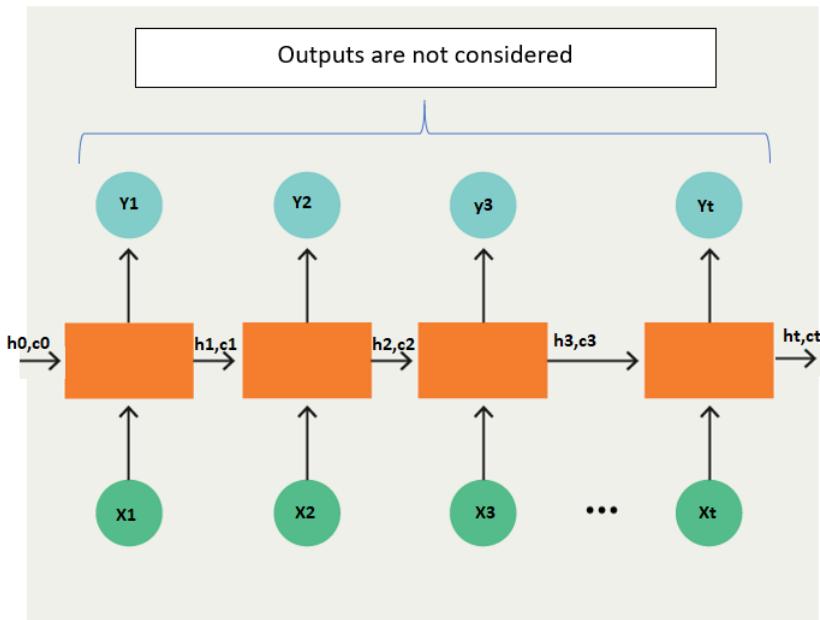
Encoder:

1. Encoder dan decoder adalah model LSTM (atau terkadang model GRU)
2. Encoder membaca urutan input dan merangkum informasi dalam sesuatu yang disebut internal state vectors atau context vector (dalam kasus LSTM ini disebut hidden state dan cell state vectors). Kami membuang output encoder dan hanya mempertahankan status internal. Vektor konteks ini bertujuan untuk merangkum informasi untuk semua elemen input untuk membantu dekoder membuat prediksi yang akurat.
3. Hidden State h_i dihitung menggunakan rumus pada gambar 3.5:

$$h_t = f(W^{(hh)} h_{t-1} + W^{(hx)} x_t)$$

Gambar 3.5 Rumus Hidden State

Perhatikan gambar 9.2 LSTM membaca data dari satu urutan secara berurutan. Jadi jika inputnya adalah barisan dengan panjang ' t ', kita katakan bahwa LSTM membacanya dalam langkah waktu ' t '.



Gambar 3.6 Encoder

1. X_i = Urutan input pada langkah waktu i.
2. h_i dan c_i = LSTM mempertahankan dua status ('h' untuk status tersembunyi dan 'c' untuk status sel) pada setiap langkah waktu. h dan c yang dikombinasikan bersama-sama merupakan keadaan internal LSTM pada langkah waktu i.
3. Y_i = Urutan keluaran pada langkah waktu i. Y_i sebenarnya adalah distribusi probabilitas atas seluruh kosakata yang dihasilkan dengan menggunakan aktivasi softmax. Jadi setiap Y_i adalah vektor dengan ukuran "vocab_size" yang mewakili distribusi probabilitas.

Dekoder:

1. Decoder adalah LSTM yang status awalnya diinisialisasi ke final state Encoder LSTM, yaitu context vector dari final cell encoder dimasukkan ke first cell pada jaringan decoder. Dengan menggunakan initial states ini, dekoder mulai menghasilkan output sequence, dan output ini juga dipertimbangkan untuk output selanjutnya.
2. Tumpukan beberapa unit LSTM di mana masing-masing memprediksi output y_{-t} pada langkah waktu t.
3. Setiap unit berulang menerima hidden state dari unit sebelumnya menghasilkan dan mengeluarkan hidden statenya sendiri.
4. Setiap hidden state h_{-i} dihitung menggunakan rumus:

$$h_t = f(W^{(hh)} h_{t-1})$$

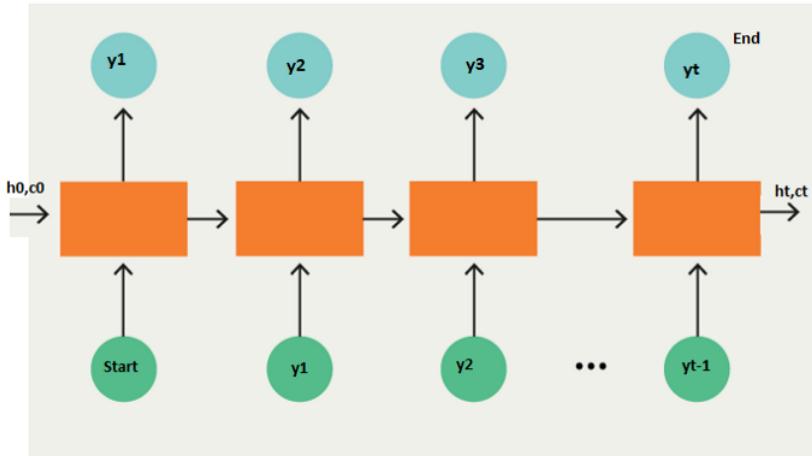
Gambar 3.7 Rumus Hidden State

5. Output y_{-t} pada langkah waktu t dihitung dengan menggunakan rumus:

$$y_t = \text{softmax}(W^S h_t)$$

Gambar 3.8 Rumus Hidden State

Menghitung output dilakukan dengan menggunakan hidden state pada langkah waktu saat ini bersama-sama dengan respective weight $W(S)$. Softmax digunakan untuk membuat vektor probabilitas yang akan membantu kita menentukan hasil akhir (misalnya kata dalam masalah tanya jawab).

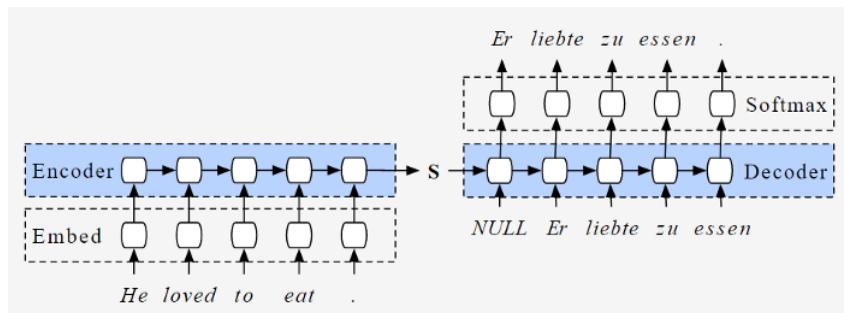


Gambar 3.9 Decoder

Sebagai contoh kita akan menambahkan dua token dalam urutan output sebagai berikut:

Contoh: “ START_John pekerja keras _END ”.

Poin yang paling penting adalah bahwa first state (h_0, c_0) dari dekoder diatur ke final state dari encoder. Ini secara intuitif berarti bahwa dekoder dilatih untuk mulai menghasilkan urutan output tergantung pada informasi yang dikodekan oleh encoder. Sehingga loss dihitung pada output yang diprediksi dari setiap langkah waktu dan error disebarluaskan kembali melalui waktu untuk memperbarui parameter jaringan. Training network dalam jangka waktu yang lebih lama dengan jumlah data yang cukup besar menghasilkan prediksi yang cukup bagus.

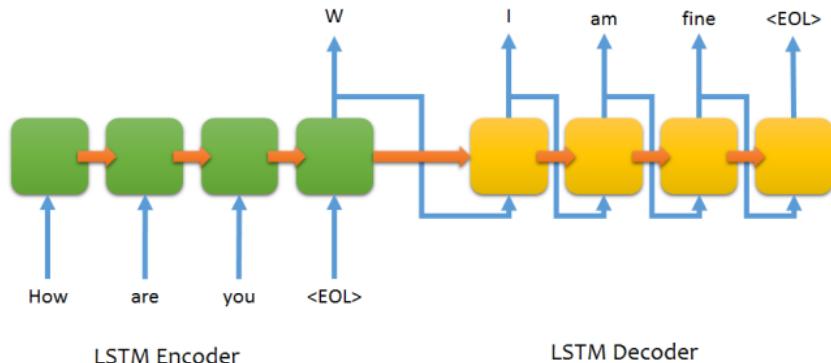


Gambar 3.10 Arsitektur Encoder-Decoder Secara Keseluruhan

1. Selama inferensi dihasilkan satu kata pada satu waktu.
2. Initial state dekoder diatur ke final state encoder.

3. initial input ke dekoder selalu berupa token START.
4. Pada setiap time step kita harus mempertahankan status dekoder dan menetapkannya sebagai status awal untuk time step berikutnya.
5. Pada setiap time step, output yang diprediksi diumpulkan sebagai input pada time step berikutnya.
6. loop akan dihentikan ketika decoder memprediksi token END.

Sebuah sequence untuk model urutan memiliki dua bagian - encoder dan decoder. Kedua bagian itu praktis adalah dua model jaringan saraf yang berbeda digabungkan menjadi satu jaringan raksasa. Secara umum, tugas jaringan encoder adalah memahami urutan input, dan membuat representasi dimensi yang lebih kecil darinya. Representasi ini kemudian diteruskan ke jaringan decoder yang menghasilkan urutannya sendiri yang mewakili output. Mari kita ambil contoh agen percakapan untuk memahami konsepnya.



Gambar 3.11 Chatbot dengan Seq2Seq Model

Pada gambar 3.11, urutan input adalah "Bagaimana kabarmu". Jadi ketika urutan input seperti itu dilewatkan melalui jaringan encoder-decoder yang terdiri dari blok LSTM (sejenis arsitektur RNN), decoder menghasilkan kata-kata satu per satu di setiap langkah waktu iterasi dekoder. Setelah satu iterasi penuh, urutan output yang dihasilkan adalah "Saya baik-baik saja".

3.1.2 Kekurangan Model Encoder-Decoder

Ada dua kelemahan utama arsitektur ini, keduanya terkait dengan panjang.

Pertama, seperti halnya manusia, arsitektur ini memiliki memori yang sangat terbatas. Keadaan tersembunyi terakhir dari LSTM, yang kami sebut S atau W , adalah saat Anda mencoba menjelaskan keseluruhan kalimat yang harus Anda terjemahkan.

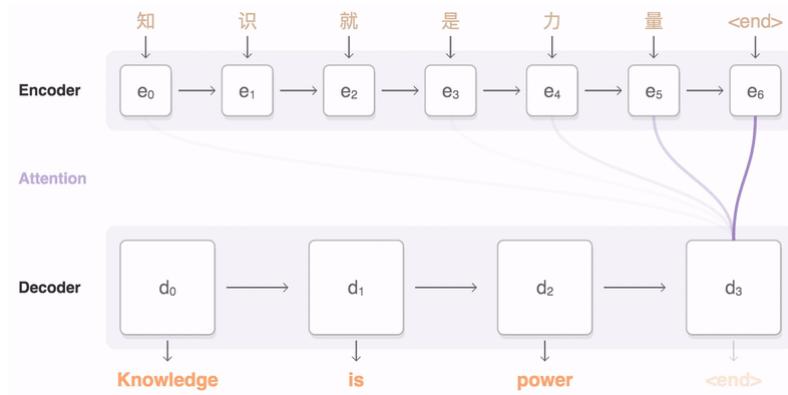
S atau W biasanya hanya beberapa ratus unit (baca: bilangan floating-point) semakin Anda mencoba untuk memaksa ke dalam vektor dimensi tetap ini, semakin besar kerugian jaringan saraf yang dipaksakan. Memikirkan jaringan saraf dalam hal "kompresi lossy" yang harus mereka lakukan terkadang cukup berguna. Kedua, sebagai aturan umum, semakin dalam jaringan saraf, semakin sulit untuk dilatih. Untuk jaringan saraf berulang, semakin panjang urutannya, semakin dalam jaringan saraf sepanjang dimensi waktu. Ini menghasilkan gradien yang hilang, di mana sinyal gradien dari tujuan yang dipelajari oleh jaringan saraf berulang menghilang saat bergerak mundur. Bahkan dengan RNN yang dibuat khusus untuk membantu mencegah gradien yang hilang, seperti LSTM, ini masih merupakan masalah men-dasar. Selanjutnya, untuk kalimat yang lebih kuat dan panjang, kami memiliki model seperti Attention Models dan Transformers .

3.1.3 Sequence to Sequence dengan Attention

Deep Learning dalam skala besar mengganggu banyak industri dengan membuat chatbot dan bot yang belum pernah ada sebelumnya. Di sisi lain, seseorang yang baru memulai Deep Learning akan membaca tentang Dasar-dasar Neural Networks dan berbagai arsitekturnya seperti CNN dan RNN. Tapi sepertinya ada lonjakan besar dari konsep sederhana ke aplikasi industri Deep Learning. Konsep-konsep seperti Batch Normalization, Dropout dan Attention hampir merupakan persyaratan untuk diketahui dalam membangun aplikasi deep learning. Berikut dua konsep penting yang digunakan dalam aplikasi terkini dalam Pengenalan Ucapan dan Pemrosesan Bahasa Alami – yaitu pemodelan Sequence to Sequence dan Attention Model. Sekadar memberikan gambaran tentang potensi penerapan kedua teknik ini Sistem AI Baidu menggunakan untuk membuat kloningan suara manusia. Ini mereplikasi suara seseorang dengan memahami suaranya hanya dalam tiga detik pelatihan. Kita dapat melihat beberapa sampel audio yang disediakan oleh Tim Riset Baidu yang terdiri dari suara asli dan suara yang disintesis. Ketika manusia mencoba memahami sebuah gambar, ia memfokuskan pada bagian-bagian tertentu dari gambar untuk mendapatkan keseluruhan esensi dari gambar tersebut. Dengan cara yang sama, kita dapat melatih sistem buatan untuk fokus pada elemen tertentu dari gambar untuk mendapatkan keseluruhan "gambar". Ini pada dasarnya adalah bagaimana mekanisme perhatian bekerja. Mari kita ambil contoh masalah teks gambar, di mana sistem harus menghasilkan teks yang sesuai untuk gambar. Dalam skenario ini, untuk menghasilkan keterangan, mekanisme perhatian membantu model untuk memahami bagian-bagian individu dari gambar yang paling penting pada contoh tertentu. Perhatikan gambar 3.12

**Gambar 3.12** Gambar to Text

Untuk menerapkan mekanisme attention, kami mengambil input dari setiap langkah waktu encoder tetapi memberi bobot pada langkah waktu. Pembobotan tergantung pada pentingnya langkah waktu tersebut bagi dekoder untuk menghasilkan kata berikutnya secara optimal dalam urutan, seperti yang ditunjukkan pada gambar 3.13

**Gambar 3.13** Mekanisme Attention pada Gambar to Text

3.1.4 Rumusan Masalah untuk Pemodelan Sequence to Sequence

Kita tahu bahwa untuk memecahkan masalah pemodelan sequence, Jaringan Syaraf Tiruan adalah arsitektur terbaik yang dapat kita pilih. Mari kita ambil contoh Sistem Penjawab Pertanyaan untuk memahami seperti apa masalah pemodelan urutan. Misalkan terdapat serangkaian pernyataan sebagai berikut:

Jo pergi ke dapur. Fred pergi ke dapur. Joe mengambil susu itu.

Joe pergi ke kantor. Joe meninggalkan susu. Jo pergi ke kamar mandi.

Pertanyaannya dimana joe sebelum pergi ke kantor? Jawaban yang tepat adalah "dapur". Pandangan sekilas membuat ini tampak seperti masalah sederhana. Tetapi untuk memahami kompleksitasnya terdapat dua dimensi yang harus dipahami oleh sistem:

1. Pemahaman dalam bahasa Inggris dan urutan karakter/kata yang membentuk kalimat.
2. Urutan peristiwa yang berkisar pada orang-orang yang disebutkan dalam pernyataan.

Ini dapat dianggap sebagai masalah pemodelan urutan, karena memahami urutan itu penting untuk membuat prediksi apa pun di sekitarnya. Ada banyak skenario masalah pemodelan urutan seperti itu, yang dirangkum dalam gambar di bawah ini. Contoh yang diberikan di atas adalah masalah banyak inputan dengan satu output (Jika Anda menganggap sebuah kata sebagai output tunggal).

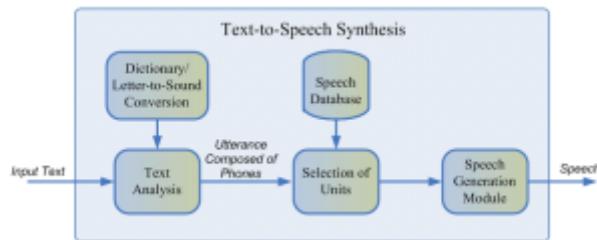
BAB 4

SPEECH SYNTHESIS

4.1 Speech Synthesis

Di berbagai media, Anda mungkin pernah menyaksikan Stephen Hawking berbicara di depan mahasiswanya. Fisikawan yang terkenal dengan teori black hole-nya ini sudah tidak mampu lagi mengeluarkan suara dari lisannya, namun berkat teknologi speech synthesizer, dia masih bisa bercakap-cakap. Mesin speech synthesizer Hawking memang cukup kompleks. Alat ini tidak hanya memproduksi suara, tetapi juga menangkap input dari gerakan mata sang doktor. Demikian pula, misalnya, dengan aplikasi voice command yang banyak tertanam di smartphone mutakhir yang memadukan speech recognizer dengan speech synthesizer.

Aplikasi speech synthesizer yang paling sederhana sebenarnya ada pada setiap PC ber-OS Windows. Bila anda menekan tuts Winkey + U di keyboard, Windows akan mengaktifkan Utility Manager, yang di dalamnya terdapat aplikasi Microsoft Narrator. Aplikasi ini akan membaca setiap jendela yang anda aktifkan, termasuk tombol-tombol di dalamnya. Atau, mungkin anda pernah menginstal aplikasi microsoft reader di PC. Aplikasi yang diperuntukkan bagi file .LTT ini pun dilengkapi dengan kemampuan menerjemahkan teks menjadi suara (text to speech) yang merupakan contoh teknologi speech synthesizer.



Gambar 4.1 Speech Synthesis

Speech synthesis adalah sebuah kemampuan bicara manusia yang dibuat oleh manusia (artificial). Sebuah sistem komputer digunakan untuk tujuan ini yang disebut sebagai speech synthesizer, dan dapat diimplementasikan ke dalam software atau hardware. Sebagai contoh sebuah sistem text-to-speech (TTS) yang dapat mengkonversikan teks dengan bahasa biasa menjadi suara.

Sebuah sistem komputer yang digunakan untuk tujuan ini disebut speech synthesizer, dan dapat diimplementasikan dalam perangkat lunak atau perangkat keras produk. Sebuah teks-to-speech (TTS) sistem mengkonversi teks bahasa normal menjadi berbicara; sistem lain membuat representasi linguistik simbolik seperti transkripsi fonetik ke dalam pidato, pidato disintesis dapat dibuat dengan menggabungkan potongan pidato direkam yang disimpan dalam database. Sistem berbeda dalam ukuran unit pidato disimpan, sebuah sistem yang menyimpan telepon atau diphones menyediakan berbagai keluaran terbesar, tapi mungkin kurang jelas.

Speech synthesis adalah transformasi dari teks ke arah suara (speech). Transformasi ini mengkonversi teks ke pemandu suara (speech synthesis) yang sebisa mungkin dibuat menyerupai suara nyata, disesuaikan dengan aturan – aturan pengucapan bahasa. TTS (text to speech) dimaksudkan untuk membaca teks elektronik dalam bentuk buku, dan juga untuk menyuarakan teks dengan menggunakan pemanduan suara. Sistem ini dapat digunakan sebagai sistem komunikasi, pada sistem informasi referral, dapat diterapkan untuk membantu orang-orang yang kehilangan kemampuan melihat dan membaca.

Synthesized speech dapat diciptakan dengan menggabungkan beberapa potongan-potongan dari pembicaraan/pidato yang sudah direkam dalam sebuah basis data. Kualitas dari sebuah speech synthesizer dilihat dari kemiripannya dengan suara manusia dan kemampuannya untuk bisa dipahami. Program TTS yang jelas dapat membantu orang dengan gangguan visual atau ketidakmampuan membaca, untuk mendengarkan pada pekerjaan yang tertulis dalam komputer. Banyak Sistem Operasi komputer yang telah dimasukkan speech synthesizer sejak tahun 1980-an.

Sebuah sistem text-to-speech (atau “mesin”) terdiri dari dua bagian: front-end dan back-end . Front-end memiliki dua tugas utama. Pertama, mengubah teks mentah berisi simbol seperti angka dan singkatan menjadi setara dengan kata-kata tertulis-out. Proses ini sering disebut normalisasi teks, pra-pengolahan, atau tokenization . Front-end kemudian memberikan transkripsi fonetik untuk setiap kata, dan membagi dan menandai teks ke unit prosodi , seperti frase , klausa , dan kalimat . Proses mene-

tapkan transkripsi fonetik untuk kata-kata ini disebut teks-ke-fonem atau grafem konversi -untuk-fonem. Transkripsi fonetik dan informasi prosodi bersama-sama membentuk representasi linguistik simbolik yang output dengan front-end. Back-end sering disebut sebagai synthesizer-maka mengubah representasi linguistik simbolik menjadi suara. Dalam sistem tertentu, bagian ini meliputi perhitungan dari target prosodi (kontur pitch, durasi fonem), yang kemudian dikenakan pada pidato output.

4.1.1 Sejarah Speech Synthesis

Upaya yang paling awal untuk menghasilkan lahirnya pemandu suara, pada abad XVIII. Terlepas dari kenyataan bahwa upaya pertama adalah bentuk mesin mekanis, kita dapat mengatakan hari ini bahwa synthesizer sudah berkualitas tinggi. Pada tahun 1779 di

St Petersburg, Rusia Profesor Kratzenshtein Kristen fisiologis menjelaskan perbedaan antara lima vokal panjang (/ A /, / e /, / i /, / o /, dan / u /) dan membuat alat untuk menghasilkan mereka artifisial. Tahun 1791 di Wina, Wolfgang von Kempelen memperkenalkan nya “Akustik-Mekanik Mesin Speech”. Dalam sekitar pertengahan 1800-an Charles Wheatstone dibangun terkenal versi mesin berbicara von Kempenlen’s.

Generasi dari sistem pemanduan suara ini dapat dibagi ke dalam 3 masa, yaitu:

1. Generasi pertama (1962-1977). Format sintesis dari fonem adalah teknologi dominan. Teknologi ini memanfaatkan aturan berdasarkan penguraian fonetik pada kalimat untuk kontur frekuensi forman. Beberapa sintesis masih miskin atau kurang dalam kejelasan dan kealamiannya.
2. Generasi kedua (1977-1992). Metode pemandu suara adalah diphone diwakilkan dengan parameter LPC. Hal tersebut menunjukkan bahwa kejelasan yang baik pada pemandu suara dapat diperoleh dengan andal dari input teks dengan menggabungkan diphone yang sesuai dengan unit. Kejelasan meningkat selama sintesis forman, tetapi kealamian dari pemandu suara masih tetap rendah.
3. Generasi ketiga (1992-sekarang). Generasi ini ditandai dengan metode ‘ sintesis pemilihan unit’ yang diperkenalkan dan disempurnakan oleh Sagisaka di Labs ATR Kyoto. Hasil dari pemandu suara pada periode ini sangat mendekati human-generated speech pada bagian kejelasan dan kealamian.

Teknologi pemandu suara modern melibatkan metode dan algoritma yang canggih dan rumit. alat pemandu suara dari keluarga “Infovox” mungkin menjadi salah satu multi bahasa TTS yang paling dikenal saat ini. Versi komersial pertamanya, Infovox-SA 101, dikembangkan pada tahun 1982 di Institute Teknologi Royal, Swedia dan didasarkan pada sintesis forman.

AT & T Bell Laboratories (Lucent Technologies) juga memiliki tradisi yang sangat panjang tentang pemandu suara (speech synthesis). TTS lengkap yang pertama didemostrasikan di Boston pada tahun 1972 dan diliris pada tahun 1973. Hal ini didasarkan pada model artikulatoris yang dikembangkan oleh Ceceil Coker (Klatt

1987). Pengembangan proses dari sistem penggabungan sintesis ini dimulai oleh Joseph Olive pada pertengahan tahun 1970-an (Bell Labs 1997). Sistem ini sekarang sudah tersedia untuk bahasa Inggris, Perancis, Spanyol, Italia, Jerman, Rusia, Rumania, Cina, dan Jepang (Mcbius et al 1996).



Gambar 4.2 Stephen Hawking adalah salah satu orang paling terkenal yang menggunakan sintesis ucapan untuk berkomunikasi

4.1.2 Perangkat Speech Synthesis

Pidato sistem sintesis berbasis komputer pertama diciptakan pada akhir 1950-an. Pertama umum Inggris sistem text-to-speech dikembangkan oleh Noriko Umeda et al. Pada tahun 1968 di Laboratorium Elektroteknik, Jepang. Pada tahun 1961, fisikawan John Larry Kelly, Jr dan Louis rekan Gerstman menggunakan IBM 704 komputer untuk mensintesis pidato, acara yang paling menonjol dalam sejarah Bell Labs . Kelly perekam suara synthesizer (vocoder) ulang lagu " Daisy Bell ", dengan irungan musik dari Max Mathews . Kebetulan, Arthur C. Clarke mengunjungi teman dan kolega John Pierce di fasilitas Bell Labs Murray Hill. Clarke begitu terkesan

oleh demonstrasi bahwa ia digunakan dalam adegan klimaks dari skenario-Nya untuk novel nya 2001: A Space Odyssey, di mana HAL 9000 komputer menyanyikan lagu yang sama seperti yang sedang ditidurkan oleh astronot Dave Bowman. Meskipun keberhasilan pidato sintesis murni elektronik, penelitian masih terus dilakukan ke synthesizer pidato mekanis.

Handheld elektronik menampilkan sintesis pidato mulai muncul pada 1970-an. Salah satu yang pertama adalah Telesensory Systems Inc(TSI) Pidato + kalkulator portabel untuk orang buta pada tahun 1976. Perangkat lain yang diproduksi terutama untuk tujuan pendidikan, seperti Bicara & Eja , yang diproduksi oleh Texas Instruments pada tahun 1978. Fidelity merilis versi berbicara komputer catur elektronik pada tahun 1979. Yang pertama video game yang memiliki fitur sintesis pidato adalah 1.980 shoot 'em up arcade game , Stratovox , dari Sun Electronics. Contoh lain awal adalah versi arcade dari Berzerk , dirilis pada tahun yang sama.Pertama multi-player permainan elektronik menggunakan sintesis suara adalah Milton dari Milton Bradley Company , yang memproduksi perangkat di tahun 1980.

4.1.3 Teknologi Speech Synthesis

Yang paling penting dalam kualitas sistem speech synthesis adalah kealamian dan kejelasannya. Kealamia menjelaskan bagaimana dekatnya suara output dengan suara manusia, sementara kejelasan adalah dengan kemudahan di mana output tersebut dapat dipahami. Speech synthesizer yang ideal adalah yang alami dan jelas. Sistem speech synthesis biasanya mencoba untuk memaksimalkan kedua karakteristik.

Kualitas terpenting dari sebuah aplikasi speech synthesizer adalah seberapa alami dan inteligibel output yang dihasilkannya. Alami, artinya seberapa dekat suara yang dihasilkan aplikasi speech synthesizer dengan suara manusia. Sedangkan inteligibel adalah seberapa mudah output tersebut dipahami oleh manusia. Semua aplikasi speech synthesizer berusaha untuk menghasilkan output yang alami dan inteligibel sekaligus.

Sampai saat ini, ada banyak teknologi untuk meng-generate gelombang suara sintetis ini. Dua teknologi yang paling banyak digunakan adalah concatenative synthesis dan formant synthesis. Keduanya memiliki keunggulan dan kekurangan sendiri-sendiri.

Teknologi pertama, concatenative synthesis, berbasis pada rangkaian (atau merangkai bersama) segmen-semen dari suara yang direkam. Umumnya, teknologi ini menghasilkan suara sintesis yang terdengar paling alami.Namun, perbedaan antara suara alami yang direkam dengan segmentasi gelombang bunyi kadang menghasilkan suara yang mengganggu. Mirip seperti suara pemberitahuan nomor antrean di bank atau suara call center operator ponsel yang menyebutkan sisa pulsa dan masa berlaku kartu ponsel anda.

Teknologi kedua, formant synthesis, tidak menggunakan sampel suara manusia melainkan membuat suara sintesi menggunakan model akustik. Parameter-parameter seperti frekuensi dasar, alunan suara, dan tingkat kebisingan bervariasi dari waktu ke waktu untuk menciptakan gelombang suara buatan.

Kebanyakan aplikasi berbasis teknologi ini menghasilkan suara buatan (tidak alami) seperti suara robot. Melihat keterbatasan kedua teknologi ini dalam menghasilkan suara buatan, seperti kita harus sabar menunggu pengembangannya lebih lanjut dalam beberapa tahun atau dekade ke depan.

Kualitas yang paling penting dari sebuah sistem sintesis pidato kewajaran dan dimengerti. Kealamian menjelaskan seberapa dekat output terdengar seperti suara manusia, sementara kejelasan adalah kemudahan yang output dipahami. Speech synthesizer yang ideal adalah baik alam dan dimengerti. Sistem sintesis pidato biasanya mencoba untuk memaksimalkan kedua karakteristik.

Dua teknologi utama dalam pembuatan gelombang suara synthetic speech adalah Concatenative Synthesis dan Formant Synthesis. Setiap teknologi mempunyai kekuatan dan kelemahannya, dan penggunaan yang ditujukan dari sistem synthesis akan menentukkan pendekatan mana yang digunakan.

1. Concatenative Synthesis Concatenative synthesis didasarkan dengan penggabungan dari segmen-semen dari pembicaraan yang sudah direkam. Secara umum, concatenative synthesis memproduksi synthesized speech dengan suara yang paling alami. Tetapi, perbedaan antara variasi alami dalam pembicaraan dan sifat dari teknik otomasi untuk pensegmentasi gelombang suara terkadang menghasilkan kesalahan suara dalam output. Namun, perbedaan antara variasi alami dalam pidato dan sifat teknik otomatis untuk membagi bentuk gelombang kadang-kadang menyebabkan gangguan terdengar pada output. Ada tiga sub-jenis utama dari sintesis concatenative.
 - (a) Sintesis Pemilihan unit Sintesis Pemilihan unit menggunakan besar database pidato direkam. Selama pembuatan database, setiap ucapan tercatat tersegmentasi ke dalam beberapa atau semua hal berikut: individu telepon , di-phones , setengah-telepon, suku kata , morfem , kata , frase , dan kalimat . Biasanya, pembagian ke dalam segmen dilakukan dengan menggunakan dimodifikasi khusus recognizer pidato disetel ke “keselarasan dipaksa” mode dengan beberapa koreksi manual setelah itu, dengan menggunakan representasi visual seperti yang gelombang dan spektrogram . Sebuah indeks unit dalam database pidato kemudian dibuat berdasarkan segmentasi dan parameter akustik seperti frekuensi dasar (lapangan), durasi, posisi dalam suku kata, dan telepon tetangga. Pada waktu berjalan , target ucapan yang diinginkan dibuat dengan menentukan rantai terbaik unit calon dari database (pemilihan unit). Proses ini biasanya dicapai dengan menggunakan khusus tertimbang pohon keputusan.

Pemilihan unit menyediakan kealamian terbesar, karena hanya berlaku sedikit pemrosesan sinyal digital (DSP) untuk pidato direkam. DSP sering membuat pidato yang direkam terdengar kurang alami, meskipun beberapa sistem menggunakan sejumlah kecil pengolahan sinyal pada titik Rangkaian untuk menghaluskan bentuk gelombang. Output dari yang terbaik units seleksi sistem sering dibedakan dari suara manusia nyata, terutama dalam konteks dimana sistem TTS telah disetel. Namun, kealamian maksimum biasanya membutuhkan unit-pilihan database pidato menjadi sangat besar,

dalam beberapa sistem mulai ke gigabyte data dicatat, mewakili puluhan jam berbicara. Juga, pilihan algoritma Unit telah dikenal untuk memilih segmen dari Tempat yang menghasilkan kurang dari sintesis ideal (misalnya kata-kata kecil menjadi tidak jelas) bahkan ketika pilihan yang lebih baik ada dalam database. Baru-baru ini, peneliti telah mengusulkan berbagai metode otomatis untuk mendeteksi segmen alami di unit-pilihan sistem sintesis pidato.

- (b) Sintesis diphone Sintesis diphone menggunakan database pidato minimal berisi semua diphones (suara-to-suara transisi) yang terjadi dalam suatu bahasa. Jumlah diphones tergantung pada fonotaktik bahasa: misalnya, Spanyol memiliki sekitar 800 diphones, dan Jerman sekitar 2500. Dalam sintesis diphone, hanya satu contoh dari setiap diphone terkandung dalam database pidato. Pada saat runtime, target prosodi kalimat ditumpangkan pada unit-unit minimal dengan cara pemrosesan sinyal digital teknik seperti linear predictive coding ,PSOLA atau MBROLA. Diphone sintesis menderita gangguan sonik sintesis concatenative dan robot-terdengar sifat sintesis forman, dan memiliki beberapa keuntungan baik pendekatan lain dari ukuran kecil. Dengan demikian, penggunaannya dalam aplikasi komersial menu-run, meskipun terus digunakan dalam penelitian karena ada beberapa implementasi perangkat lunak tersedia secara bebas.
- (c) Domain-spesifik sintesis Domain-spesifik sintesis concatenates direkam sebelumnya kata dan frase untuk menciptakan ucapan-ucapan yang lengkap. Hal ini digunakan dalam aplikasi di mana berbagai teks output sistem akan terbatas pada domain tertentu, seperti jadwal angkutan pengumuman atau laporan cuaca. Teknologi ini sangat sederhana untuk menerapkan, dan telah digunakan secara komersial untuk waktu yang lama , dalam perangkat seperti berbicara jam dan kalkulator. Tingkat kealamian sistem ini bisa sangat tinggi karena berbagai jenis kalimat terbatas, dan mereka cocok dengan prosodi dan intonasi dari rekaman asli.

Karena sistem ini dibatasi oleh kata-kata dan frasa dalam database mereka, mereka tidak tujuan umum dan hanya dapat mensintesis kombinasi kata dan frase yang mereka telah terprogram. Campuran kata-kata dalam bahasa alami diucapkan namun masih dapat menyebabkan masalah kecuali banyak variasi diperhitungkan. Misalnya, dalam non-rhotic dialek dari bahasa Inggris “r” dalam kata-kata seperti “jelas” biasanya hanya diucapkan ketika kata berikut memiliki vokal sebagai huruf pertama (misalnya”membersihkan” direalisasikan sebagai). Demikian juga di Perancis , banyak konsonan akhir menjadi tidak lagi diam jika diikuti oleh sebuah kata yang dimulai dengan vokal, efek yang disebut penghubung . Ini pergantian tidak bisa direproduksi oleh sistem kata-Rangkaian sederhana, yang akan membutuhkan kompleksitas tambahan untuk konteks-sensitif .

2. Formant Synthesis Formant synthesis tidak menggunakan pembicaraan manusia sebagai sample pada runtime. Daripada itu, synthesized speech yang di-

hasilkan dibuat dengan additive synthesis dan sebuah model akustik (physical modelling synthesis).

Parameter seperti frekuensi dasar, penyuaran, dan tingkat kebisingan di variasikan dari waktu ke waktu untuk menciptakan gelombang buatan (artificial) dari sebuah pembicaraan. Banyak sistem yang berdasarkan formant synthesis menciptakan pembicaraan yang seperti robot yang tidak mungkin dapat dikenal sebagai suara manusia. Tetapi, kealamian maksimum bukan selalu tujuan dari sebuah sistem speech synthesis, dan sistem formant synthesis mempunyai keuntungan dari sistem concatenative. Pembicaraan yang di-formant synthesis-kan dapat menjadi sangat jelas, bahkan dalam kecepatan yang tinggi, sehingga menghindari kesalahan suara yang sering dialami sistem concatenative.

Formant synthesis biasanya program yang lebih kecil dari concatenative sistem karena ia tidak menggunakan basis data dari sampel-sampel pembicaraan. Oleh karena itu formant synthesis dapat ditanamkan dalam sistem yang mempunyai memory dan mikroprosesor yang terbatas. Karena sistem yang berdasarkan formant mempunyai kendali penuh dari sluruh aspek dari hasil pembicaraan, variasi yang luas dari prosodi dan intonasi dapat dihasilkan, menyampaikan tidak hanya pertanyaan dan pernyataan tetapi juga emosi dan nada suara.

Formant sintesis tidak menggunakan sampel suara manusia pada saat runtime. Sebaliknya, keluaran suara yang disintesis dibuat menggunakan aditif sintesis dan model akustik (sintesis pemodelan fisik). Parameter seperti frekuensi dasar , menyuarakan , dan kebisingan tingkat yang bervariasi dari waktu ke waktu untuk membuat gelombang pidato buatan. Metode ini kadang-kadang disebut aturan berbasis sintesis; Namun, banyak sistem concatenative juga memiliki aturan berbasis komponen. Banyak sistem yang didasarkan pada teknologi sintesis forman menghasilkan buatan, robot yang terdengar pidato yang tidak akan pernah salah untuk pidato manusia. Namun, kealamian maksimum tidak selalu tujuan sistem sintesis pidato, dan sistem sintesis forman memiliki keunggulan dibandingkan sistem concatenative. Pidato forman-disintesis dapat diandalkan dimengerti, bahkan pada kecepatan yang sangat tinggi, menghindari Glitches akustik yang biasanya wabah sistem concatenative. Kecepatan tinggi disintesis pidato digunakan oleh tunanetra untuk navigasi cepat komputer menggunakan pembaca layar . Synthesizer forman adalah program biasanya lebih kecil dibandingkan dengan sistem concatenative karena mereka tidak memiliki database contoh pidato. Karena itu mereka dapat digunakan dalam embedded system , di mana memori dan mikroprosesor daya terutama terbatas. Karena sistem berbasis forman memiliki kontrol penuh dari semua aspek pidato output, berbagai prosodies dan intonasi dapat menjadi output, tidak hanya menyampaikan pertanyaan dan pernyataan, tetapi berbagai emosi dan nada suara.

Contoh non-real-time tapi sangat akurat kontrol intonasi dalam sintesis forman meliputi pekerjaan yang dilakukan pada akhir tahun 1970 untuk Texas Instruments mainan Bicara & Eja , dan pada awal tahun 1980 Sega arcade mesin dan dalam banyak Atari, Inc. game arcade menggunakan TMS5220 LPC Chips .

Menciptakan intonasi yang tepat untuk proyek ini adalah telaten, dan hasilnya masih harus dicocokkan dengan real-time text-to-speech interface.

3. Sintesis artikulatoris Sintesis artikulatoris mengacu pada teknik komputasi untuk sintesis pidato berdasarkan model manusia saluran vokal dan artikulasi proses yang terjadi di sana. Synthesizer artikulatoris pertama teratur digunakan untuk percobaan laboratorium dikembangkan di Haskins Laboratories di pertengahan 1970-an oleh Philip Rubin, Tom Baer, dan Paul Mermelstein. Synthesizer ini, yang dikenal sebagai ASY, didasarkan pada model saluran vokal dikembangkan di Bell Laboratories pada tahun 1960 dan 1970-an oleh Paul Mermelstein, Cecil Coker, dan rekan.

Sampai saat ini, model sintesis artikulatoris belum dimasukkan ke dalam sistem sintesis pidato komersial. Sebuah pengecualian adalah NeXT sistem berbasis awalnya dikembangkan dan dipasarkan oleh Trillium Suara Research, sebuah perusahaan spin-off dari University of Calgary , di mana banyak riset asli dilakukan. Setelah runtuhan berbagai inkarnasi NeXT (dimulai oleh Steve Jobs pada akhir tahun 1980 dan bergabung dengan Apple Computer pada tahun 1997), perangkat lunak TRILLIUM diterbitkan di bawah GNU General Public License , dengan bekerja terus sebagai gnuSpeech . Sistem, pertama kali dipasarkan pada tahun 1994, memberikan penuh text-to-speech konversi berbasis artikulatoris menggunakan Waveguide atau transmisi-line analog dari saluran mulut dan hidung manusia dikendalikan oleh Carré ini “model daerah khas”.

4. Sintesis berbasis HMM Sintesis berbasis HMM adalah metode sintesis berdasarkan model Markov tersembunyi , juga disebut statistik Parametrik Sintesis. Dalam sistem ini, spektrum frekuensi (vokal),frekuensi dasar (sumber vokal), dan durasi (prosodi) berbicara dimodelkan secara bersamaan oleh HMMs. Pidato bentuk gelombang yang dihasilkan dari HMMs sendiri berdasarkan maksimum kriteria.
5. Sintesis Sinewave Sintesis sinewave adalah teknik untuk sintesis pidato dengan mengganti forman (band utama energi) dengan peluit nada murni.

Ada beberapa masalah yang terdapat pada pemanfaatan suara, yaitu:

- (a) User sangat sensitif terhadap variasi dan informasi suara. Oleh sebab itu, mereka tidak dapat memberikan toleransi atas ketidak sempurnaan pemanfaatan suara.
- (b) Output dalam bentuk suara tidak dapat diulang atau dicari dengan mudah.
- (c) Meningkatkan keberisikan pada lingkungan kantor atau jika menggunakan handphone, maka akan meningkatkan biaya pengeluaran.

BAB 5

VOICE CLONING

5.1 Voice Cloning

Kloning suara sering diombang-ambingkan dengan istilah lain, seperti suara deepfake, sintesis ucapan, dan suara sintetis, yang memiliki arti yang sedikit berbeda. Kloning suara adalah proses di mana seseorang menggunakan komputer untuk menghasilkan ucapan individu nyata, menciptakan tiruan dari suara mereka yang spesifik dan unik menggunakan kecerdasan buatan (AI). Sistem text-to-speech (TTS), yang dapat mengambil bahasa tertulis dan mengubahnya menjadi komunikasi lisan, tidak sama dengan kloning suara. Sistem TTS jauh lebih terbatas dari output yang mereka hasilkan dibandingkan dengan teknologi kloning suara, yang sebenarnya lebih merupakan proses kustom. Dengan sistem TTS, data pelatihan, komponen kunci untuk setiap media yang dibuat secara sintetis, menginformasikan produksi keluaran suara. Dengan kata lain, suara yang Anda dengar adalah suara yang diberikan dalam kumpulan data. Sekarang, dengan diperkenalkannya teknologi AI kloning suara, itu berubah. Metode telah diterapkan untuk memberikan analisis yang lebih dalam dan ekstraksi karakteristik suara target. Atribut-atribut ini kemudian dapat diterapkan pada bentuk gelombang ucapan yang berbeda, memungkinkan seseorang untuk mengubah keluaran suara dari satu suara ke suara lainnya.



Gambar 5.1 Voice Cloning

Berkat kemajuan dalam kecerdasan buatan (AI), khususnya pembelajaran mendalam, bagian dari pembelajaran mesin di bawah payung AI, kami telah mampu menghasilkan replika suara yang akurat. Tapi ini hanya dimungkinkan oleh dua hal:

1. Perangkat keras yang kuat dengan kemampuan komputasi awan untuk memproses dan merender secara tepat waktu dan efisien
2. Data pelatihan ekstensif dari suara yang ditargetkan dari mana model dapat memanfaatkan untuk membuat klon suara yang akurat

Dengan AI yang tepat dan keahlian dan alat pengembangan, itu benar-benar tergantung pada yang terakhir. Anda memerlukan sejumlah besar ucapan yang direkam untuk memiliki data yang cukup untuk melatih model suara. Informasi seputar suara disimpan dalam embedding , ruang berdimensi cukup rendah di mana Anda dapat menerjemahkan variabel diskrit menjadi vektor berdimensi tinggi. Dengan kata lain, lebih mudah untuk bekerja dengan input besar dengan model pembelajaran mesin. Agar tidak terlalu teknis, kami akan berhenti di situ, tetapi jangan ragu untuk menyelemah lebih dalam ke subjek jika itu menarik minat Anda.

5.1.1 Manfaat Voice Cloning

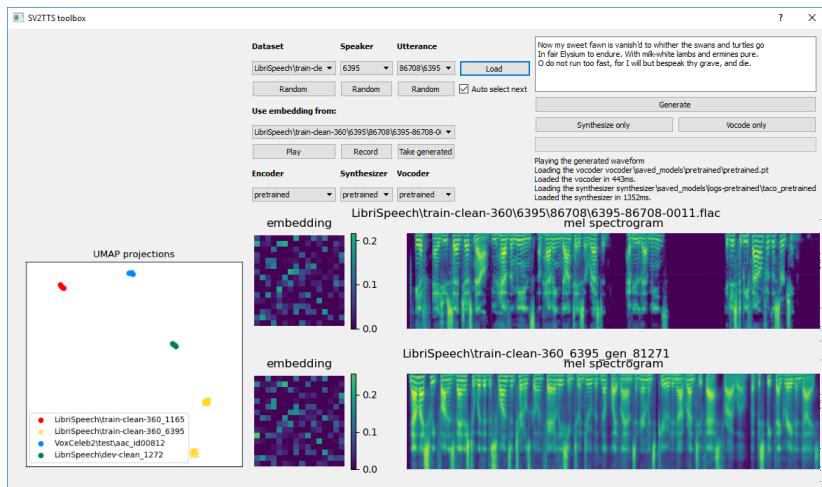
Mari kita mulai dengan yang baik. Ada banyak kasus penggunaan potensial untuk kloning suara yang sering kali dibayangi oleh penggunaan negatif, yang akan kita bahas sebentar lagi. Beberapa aplikasi positif dari teknologi meliputi:

1. Meningkatkan peluang iklan dan sponsor untuk pengisi suara, selebritas, dan influencer
2. Bantu perusahaan bekerja dengan talenta selama waktu tersibuk mereka dalam setahun, seperti musim sepak bola untuk pemain atau pelatih

3. Menghidupkan kembali suara-suara dari masa lalu untuk digunakan dalam hiburan guna membantu menceritakan kisah dalam dokumenter, film, dan acara TV
4. Diversifikasi konten siaran untuk konten berulang seperti laporan cuaca atau pembaruan olahraga
5. Lokalkan konten sehingga dapat didengar dalam suara pembawa acara atau narator dalam bahasa lain

Ini hanyalah beberapa kegunaan positif dari kloning suara, dan seiring dengan berkembangnya teknologi, lebih banyak lagi yang akan muncul. Tapi tentu saja, semuanya bergantung pada etika penggunaan suara seseorang. Itulah mengapa perlunya gerakan menuju standarisasi proses persetujuan sangat penting untuk melindungi suara semua orang dan memastikan mereka memiliki kendali penuh atas cara penggunaannya.

5.1.2 Aplikasi Voice Cloning Terbaik



Gambar 5.2 Contoh Aplikasi Voice Cloning (SV2TTS Toolbox)

Untuk mempersempit pencarian Anda untuk menemukan aplikasi kloning suara terbaik, Anda harus terlebih dahulu menentukan apa yang Anda cari. Apakah Anda memerlukan sesuatu yang lebih untuk output text-to-speech? Atau apakah Anda membutuhkan sesuatu yang lebih khusus?

Setelah Anda mengetahui mengapa Anda memerlukan aplikasi kloning suara, Anda harus mengasah tiga kriteria utama:

1. Kualitas keluaran: Anda pasti ingin memastikan bahwa keluaran terdengar otentik dan memenuhi kebutuhan yang ditentukan. Biasanya, mereka akan memi-

liki sampel tentang apa yang dapat dilakukan produk tersebut. Jika tidak, Anda harus mempertimbangkan untuk meminta demo, jika tersedia, untuk menentukan seberapa manusiawi produk mereka.

2. Antarmuka intuitif: seberapa mudah menggunakan aplikasi? Apakah sulit untuk menemukan sesuatu saat Anda berada di aplikasi atau dapatkah Anda menavigasi dan menggunakannya untuk memenuhi kebutuhan Anda? Sekali lagi, ini dapat ditentukan oleh video produk, konten pemasaran, dan demo.
3. Perlindungan suara: Anda pasti ingin memastikan bahwa perusahaan mengikuti penggunaan suara yang etis. Jika itu adalah layanan kustom yang memerlukan data pelatihan, maka penting untuk menanyakan tentang perlindungan data dan bagaimana suara, saat dibuat, tidak akan digunakan secara tidak semestinya.

Implikasi etis seputar kloning suara adalah hubungan dari Veritone MARVEL.ai, aplikasi suara sebagai layanan kami . Dibangun dalam kerangka aplikasi adalah tuas untuk memberi pengguna kendali atas suara mereka, memungkinkan perlindungan yang tepat sehingga mereka memutuskan siapa yang dapat menggunakan suara mereka. Ini membantu kami memberikan solusi voice-as-a-service khusus kami untuk memungkinkan pengalaman sarung tangan putih lengkap untuk talenta yang bekerja dengan kami.

5.1.3 Pembuatan Voice Cloning

Perangkat lunak kloning suara AI online dimulai dengan menggunakan komputer untuk mensintesis suara. Text-to-Speech (TTS) adalah teknologi berusia puluhan tahun yang mengubah teks menjadi ucapan sintetis, memungkinkan suara digunakan untuk interaksi komputer-manusia.

Di masa lalu ada dua pendekatan untuk TTS. Yang pertama, TTS Concatenative , menggunakan rekaman audio untuk membuat perpustakaan kata dan satuan suara (fonem) yang dapat dirangkai menjadi kalimat. Meskipun keluarannya berkualitas tinggi dan dapat dipahami, ia tidak memiliki emosi dan infleksi yang ditemukan dalam ucapan manusia yang alami. Saat menggunakan TTS Concatenative, setiap gaya bicara atau bahasa baru memerlukan database audio baru. Dan tentu saja upaya untuk mengkloning suara individu menggunakan metode ini membutuhkan investasi yang sangat besar, biasanya hanya dilakukan untuk mendukung suara bermerek.

Pendekatan kedua adalah Parametrik TTS , metode yang menggunakan model statistik ucapan untuk menyederhanakan pembuatan suara, mengurangi biaya dan upaya dibandingkan dengan Penggabungan. Namun, upaya untuk menciptakan satu suara apa pun secara historis mahal, dan hasilnya jelas bukan manusia.

Saat ini, Kecerdasan Buatan (AI) dan kemajuan dalam Pembelajaran Mendalam meningkatkan kualitas ucapan sintetis. Pengajuan TTS sekarang sudah lumrah. Setiap orang yang telah berinteraksi dengan sistem Respon Suara Interaktif berbasis telepon, Siri Apple, Amazon Alexa, sistem navigasi mobil, atau banyak antarmuka suara lainnya, telah mengalami ucapan sintetis.

Jika Anda terbiasa dengan konsep video deepfake, perangkat lunak kloning suara AI online adalah setara dengan ucapan. Hanya dengan beberapa menit rekaman ucapan, pengembang dapat membangun kumpulan data audio dan menggunakannya untuk melatih model suara AI yang dapat membaca teks apa pun dalam suara target.

Tugas sekarang secara signifikan lebih mudah berkat berbagai alat bertenaga jaringan saraf seperti Tacotron dan Wavenet atau Lyrebird Google, yang memungkinkan hampir semua suara direplikasi dan digunakan untuk "membaca" input teks. Kualitas output terus meningkat, seperti yang ditunjukkan oleh tiruan suara podcaster Joe Rogan ini. Insinyur pembelajaran mendalam di Dessa yang membuat klon juga menyajikan kuis. Ambil untuk melihat apakah Anda dapat melihat Rogan palsu.

Model TTS berbasis jaringan saraf meniru cara otak beroperasi dan sangat efisien dalam mempelajari pola dalam data. Meskipun ada pendekatan berbeda untuk penggunaan pembelajaran mendalam dalam suara sintetis, sebagian besar menghasilkan pengucapan kata yang lebih baik, serta menangkap kehalusan seperti kecepatan dan intonasi untuk menciptakan ucapan yang lebih mirip manusia.

Penting untuk dicatat bahwa alat yang disebutkan di atas dan alat lain seperti ini tidak dibuat untuk tujuan penipuan atau penipuan. Tetapi kenyataannya adalah bahwa bisnis dan konsumen perlu mewaspadai ancaman baru yang terkait dengan perangkat lunak kloning suara AI online. Kami menjelajahi beberapa kegunaan — baik dan buruk — di bawah ini.

5.1.4 Dampak Negatif Voice Cloning

1. Voice adalah pengenal pribadi unik yang mudah diakses oleh penipu. Hal ini tentu berlaku bagi publik figur termasuk selebriti, politisi dan pemimpin bisnis, tetapi kenyataannya adalah siapa saja bisa menjadi sasaran. Video online, pidato, panggilan konferensi, percakapan telepon, dan posting media sosial semuanya dapat digunakan untuk mengumpulkan data yang diperlukan untuk melatih sistem untuk mengkloning suara.
2. Spoofing biometrik suara — Suara adalah pengidentifikasi unik dan ukuran yang andal untuk keamanan biometrik. Namun, penjahat dapat menggunakan serangan presentasi termasuk suara yang direkam, suara yang diubah komputer dan suara sintetis, atau kloning suara, untuk mengelabui sistem biometrik suara agar mengira mereka mendengar pengguna yang sebenarnya dan berwenang dan memberikan akses ke informasi dan akun sensitif.



Gambar 5.3 Spoofing Voice Biometrics

3. Penipuan phishing – Perangkat lunak kloning suara AI online juga memungkinkan jenis baru penipuan phishing yang mengeksplorasi fakta bahwa korban percaya bahwa mereka sedang berbicara dengan seseorang yang mereka percayai. Tahun lalu, seorang CEO yang berbasis di Inggris ditipu untuk mentransfer lebih dari \$240.000 berdasarkan panggilan telepon yang dia yakini berasal dari bosnya, CEO perusahaan induk organisasi Jerman.



Gambar 5.4 Voice Phising, Sumber: <https://id.pinterest.com/pin/195765915039905230/>

4. Penipuan seperti ini adalah evolusi penipuan email di mana email eksekutif dipalsukan dalam upaya agar penerima membocorkan nomor rekening bank, informasi kartu kredit, kata sandi, dan data sensitif lainnya. Sekarang scammers, dipersenjatai dengan klon suara, menggunakan panggilan telepon dan pesan suara. Dan serangan itu tidak hanya mengancam bisnis. Dalam generasi

baru penjahat "mother's deception" menyamar sebagai anggota keluarga yang membutuhkan dana darurat.

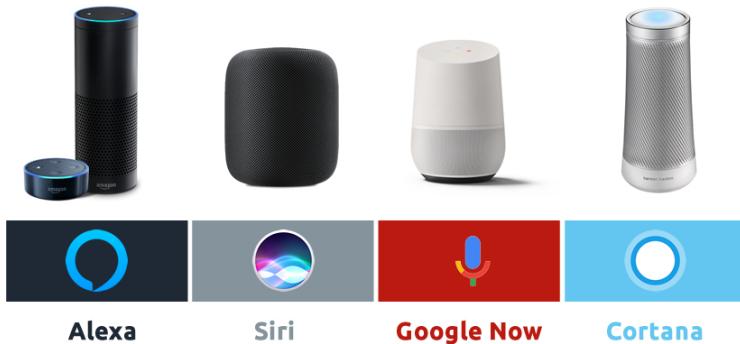
5. Misinformasi – Berita palsu dan bentuk misinformasi serupa merupakan ancaman serius. Banyak dari kita yang akrab dengan bagaimana video yang di-manipulasi berdampak pada lanskap politik. Misalnya, video populer menunjukkan Barack Obama memanggil Trump – yah, anggap saja itu tidak terlalu bagus. Itu juga tidak nyata. Teknologi perangkat lunak text-to-speech berbasis AI akan semakin mendorong upaya ini untuk mempengaruhi opini publik, menghidupkan sumbangsih kampanye palsu, mencemarkan nama baik tokoh masyarakat, dan banyak lagi. Di bidang bisnis, pertimbangkan bagaimana pernyataan eksekutif atau figur publik yang dimanipulasi dapat memengaruhi pasar saham.
6. Bukti – Suara sintetis dan deepfake lainnya dapat digunakan untuk membuat bukti palsu yang berdampak pada kasus kriminal. Meskipun ada pemeriksaan untuk memvalidasi bukti audio dan video yang disajikan di pengadilan, mencegah taktik ini memengaruhi kesaksian berdasarkan apa yang orang yakini mereka lihat atau dengar mungkin merupakan tantangan.
7. Pemerasan dan intimidasi – Video dan audio yang dimanipulasi dari orang-orang yang melakukan atau mengatakan hal-hal yang tidak mereka katakan dapat digunakan untuk intimidasi online dan ancaman untuk mengekspos konten palsu dan memalukan jika korban menolak untuk membayar biaya.

5.1.5 Dampak Positif Voice Cloning

1. Pendidikan, Mengkloning suara tokoh sejarah menawarkan peluang baru untuk pengajaran interaktif dan penceritaan yang dinamis. Misalnya, pada 22 November 1963 Presiden Kennedy sedang dalam perjalanan untuk memberikan pidato di Dallas ketika dia dibunuh. Kita sekarang dapat mendengar pidato itu dengan kata-katanya sendiri menggunakan teknologi deepfake. Dalam penggunaan deepfake AI lainnya yang menakjubkan, pengunjung Museum Dalí di St. Petersburg akan disambut oleh Salvador Dalí sendiri. Dalí berinteraksi dengan tamu menggunakan kutipan aktual dan membuat komentar, dan bahkan berfoto selfie dengan mereka. Lihat video dan pelajari lebih lanjut tentang bagaimana Dalí dihidupkan kembali melalui kekuatan AI.
2. Audiobooks, Menggunakan perangkat lunak kloning suara AI, suara selebriti dapat digunakan untuk menceritakan buku, otobiografi dapat dibaca oleh penulis, dan tokoh sejarah dapat menceritakan kisah mereka dengan suara mereka sendiri. Hasilnya adalah pengalaman mendengarkan yang imersif dan berkualitas tinggi.
3. Assistive Tech, Suara sintetis dapat digunakan untuk membantu penyandang disabilitas atau masalah kesehatan yang memengaruhi kemampuan bicara mereka.

Misalnya, orang yang tunarungu atau menderita gangguan seperti Penyakit Parkinson atau ALS dapat meningkatkan kemampuan mereka untuk berkomunikasi menggunakan versi sintetis dari suara dan TTS mereka.

4. Voice Branding, menggunakan suara pribadi dan khusus sebagai merek utama perusahaan Anda dalam asisten percakapan dan aktivitas pemasaran Anda.
5. Digital Dubbing and Animation, Kloning suara memungkinkan untuk membuat suara yang dapat diidentifikasi untuk karakter digital unik atau avatar dalam sistem interaksi manusia-komputer atau untuk produksi konten multimedia dan audiovisual.
6. Smart Assistant, Suara smart assistant terdengar semakin alami karena kemajuan di bidang AI teknologi text-to-speech. Kloning suara memungkinkan personalisasi mereka, melalui penggunaan suara tertentu atau favorit untuk mengembangkan asisten percakapan yang disesuaikan.



Gambar 5.5 Smart Assistant, Sumber: <https://blog.evolvemachinelearners.com/voice-assistant-its-history-twist-and-turn/>

5.1.6 Mendeteksi Deepfake Voice

Karena teknologi suara terus meningkat, memiliki teknologi yang dapat mengenali dan menghentikan penggunaan ucapan palsu untuk penipuan dan penipuan sangat penting.

Anti-spoofing suara, juga disebut deteksi keaktifan suara, adalah teknologi yang mampu membedakan antara suara langsung dan suara yang direkam, dimanipulasi, atau sintetis. Banyak pemalsuan saat ini tidak terlihat oleh telinga manusia, tetapi dapat dideteksi oleh perangkat lunak berbasis AI yang dilatih untuk mengidentifikasi artefak yang tidak ada dalam suara langsung.

Teknologi yang mendeteksi perangkat lunak kloning suara AI pada awalnya dibuat untuk memecahkan masalah spoofing biometrik suara. Di mana biometrik suara mencocokkan suara seseorang dengan templat suara pada file, teknologi anti-spoofing

memeriksa untuk memastikan suara itu hidup. Teknologi ini akan terus beradaptasi untuk mengatasi kasus penggunaan tambahan karena penipuan kloning suara menjadi lebih umum. Topik tersebut bahkan menjadi fokus lokakarya FTC baru -baru ini dengan peserta dari ID R&D DARPA, University of Florida, MIT Sloan School of Management, dan program Defending Democracy dari Microsoft.

BAB 6

TACOTRON-2

Tacotron 2 merupakan penelitian yang diteliti oleh Google pada bulan Desember 2016. Tacotron-2 merupakan implementasi jaringan saraf untuk Text-to-Speech Synthesis. Silahkan kunjungi website <https://google.github.io/tacotron/publications/tacotron2/> ini apabila teman-teman penasaran dengan penelitian tersebut. Apabila teman-teman perhatikan dan mencoba mendengarkan sampel suara yang ada di website tersebut bukankah sangat mirip dengan suara asli manusia? Setelah mendengarkan sampel suara tersebut saya menjadi sangat tertarik dengan penelitian tentang Arsitektur Tacotron-2 ini dan mencari tahu serta menerapkan model tersebut dalam project pembuatan voice cloning berbahasa indonesia ini. Cara kerja sistem dijelaskan oleh Jonathan Shen dan Ruoming Pang, Software Engineers, Google Brain and Machine Perception Teams.

"Singkatnya cara kerjanya seperti ini: Kami menggunakan model urutan-ke-urutan yang dioptimalkan untuk TTS untuk memetakan urutan huruf ke urutan fitur yang mengkodekan audio. Fitur-fitur ini, spektrogram audio 80-dimensi dengan bingkai yang dihitung setiap 12,5 milidetik, tidak hanya menangkap pengucapan kata-kata, tetapi juga berbagai seluk-beluk ucapan manusia, termasuk volume, kecepatan, dan intonasi. Akhirnya fitur ini diubah menjadi bentuk gelombang 24 kHz menggunakan arsitektur mirip WaveNet."

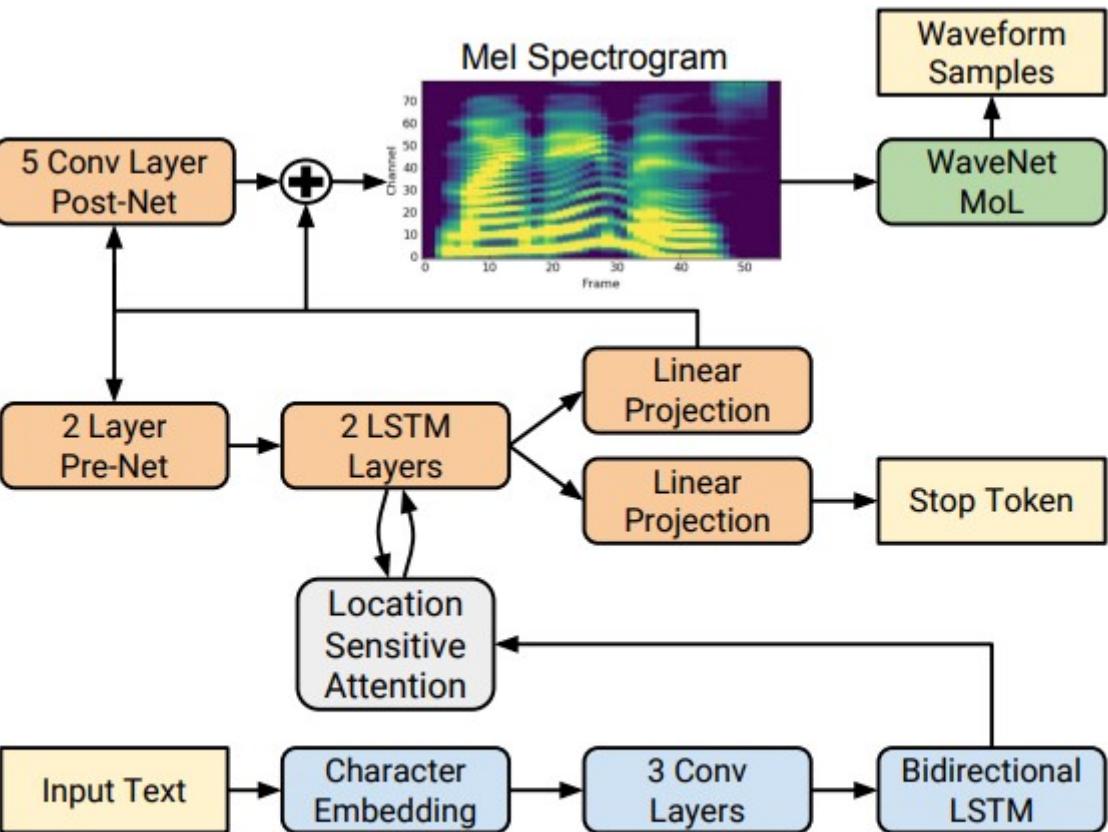


Fig. 1. Block diagram of the Tacotron 2 system architecture.

Gambar 6.1 Blok Diagram dari Arsitektur Tacotron 2

Bagian pertama dari model yaitu arsitektur Seq2Seq yang bertanggung jawab untuk mengubah teks menjadi mel-spektogram dan spektogram ini dimasukkan dalam model wave-net untuk menghasilkan bentuk gelombang audio. Satu hal yang menarik adalah kedua bagian dari arsitektur Tacotron (Seq2Seq dan vocoder Wavenet) ini dapat dilatih secara mandiri. Saya bekerja pada model Seq2Seq. Modelnya adalah penyiapan enkoder-perhatian-dekoder di mana mereka menggunakan 'Perhatian sensitif lokasi'. Bagian pertama adalah Encoder yang mengubah urutan karakter menjadi vektor penyisipan kata. Representasi ini kemudian dikonsumsi oleh Decoder untuk memprediksi spektogram. Karena saya menggunakan dataset Indonesia, saya memastikan bahwa ruang karakter saya memiliki abjad Indonesia.

Encoder terdiri dari 3 lapisan konvolusi yang masing-masing berisi 512 filter berbentuk 5×1 , diikuti oleh normalisasi batch dan aktivasi ReLU. Bagian selan-

jutnya adalah jaringan attention yang mengambil keluaran enkoder sebagai masukan dan mencoba meringkas urutan yang disandikan penuh sebagai vektor konteks dengan panjang yang tetap untuk setiap langkah keluaran dekoder. Output dari lapisan konvolusi akhir dilewatkan ke lapisan LSTM dua arah tunggal yang berisi 512 unit (256 di setiap arah) untuk menghasilkan fitur yang dikodekan.

6.1 MODEL BERBASIS PERHATIAN UNTUK PENGENALAN Pidato

Mekanisme attention yang digunakan di sini memperhitungkan lokasi fokus pada langkah sebelumnya dan fitur urutan input. Katakanlah kita memiliki data $x = x_1, x_2, x_3, \dots, x_N$. Kami meneruskan data ini ke encoder yang menghasilkan urutan keluaran yang dikodekan $h = h_1, h_2, h_3, \dots, h_N$. $A(i) = \text{Perhatian}(s(i-1), A(i-1), h)$ di mana $s(i-1)$ adalah status decoding sebelumnya dan $A(i-1)$ adalah perataan sebelumnya. $s(i-1)$ adalah 0 untuk iterasi pertama dari langkah pertama. Fungsi perhatian biasanya diimplementasikan dengan menilai setiap elemen dalam h secara terpisah dan kemudian menormalkan skor. $G(i) = A(i, 0) h(0) + A(i, 1) h(1) + \dots + A(i, N) h(N)$. $Y(i) = \text{Hasilkan}(s(i-1), G(i))$ di mana s adalah output decoding, $A(i)$ adalah vektor bobot perhatian yang disebut keselarasan. Akhirnya, $s(i) = \text{Pengulangan}(s(i-1), G(i), Y(i))$. Pengulangan biasanya LSTM.

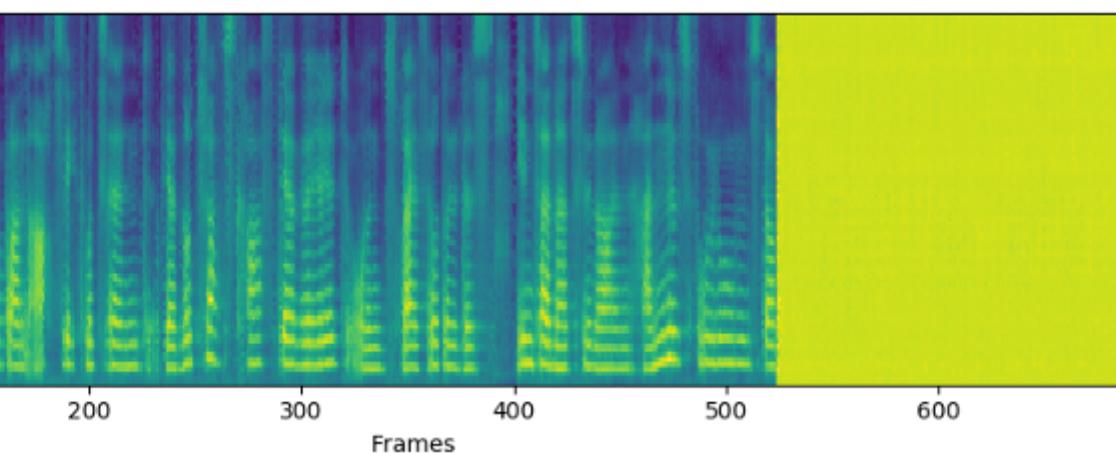
6.1.1 Decoder

Dekoder adalah jaringan saraf rekuren autoregresif yang memprediksi spektrogram mel dari urutan input yang disandikan satu frame pada satu waktu. Prediksi dari langkah waktu sebelumnya pertama-tama dilewatkan melalui pra-net kecil yang berisi 2 lapisan yang terhubung penuh dari 256 unit ReLU tersembunyi. Output prenet dan vektor konteks perhatian digabungkan dan dilewatkan melalui tumpukan 2 lapisan LSTM uni-directional dengan 1024 unit. Akhirnya, spektrogram mel yang diprediksi dilewatkan melalui post-net convolutional 5-lapisan yang memprediksi residi untuk ditambahkan ke prediksi untuk meningkatkan rekonstruksi keseluruhan. Setiap lapisan pasca jaring terdiri dari 512 filter dengan bentuk 5×1 dengan normalisasi batch, diikuti oleh aktivasi tanh pada semua kecuali lapisan terakhir.

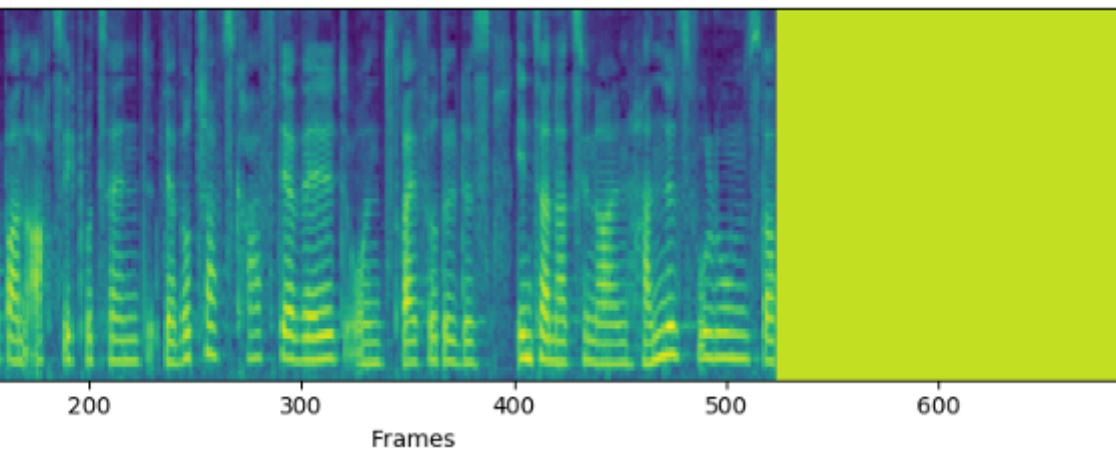
6.1.2 Loss Function

Summed mean squared error (MSE) Sejalan dengan prediksi bingkai spektogram, rangkaian keluaran LSTM dekoder dan konteks perhatian diproyeksikan ke skalar dan melewati aktivasi sigmoid untuk memprediksi kemungkinan bahwa urutan keluaran telah selesai. Prediksi "token berhenti" ini digunakan selama inferensi untuk memungkinkan model menentukan secara dinamis kapan harus menghentikan pembangkitan alih-alih selalu menghasilkan untuk durasi yang tetap. Saya memutuskan untuk menggunakan pytorch untuk implementasi saya, melacak pelatihan dengan tensorboard , menggunakan GPU gcloud Tesla K80, terhubung ke port server dengan 'ssh -NfL', dan lab jupyter yang banyak digunakan selama pengembangan. [per-

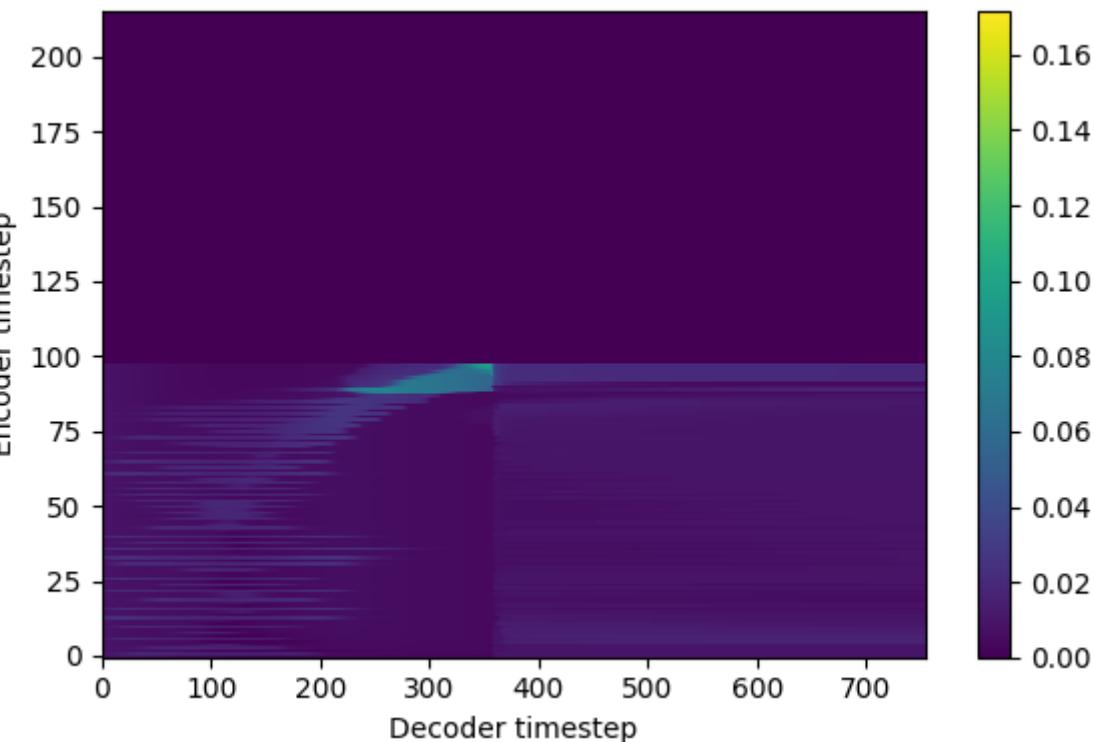
lengkapan penyelamat hidup] Saya mereferensikan berbagai repositori github [1 , 2] untuk memahami makalah, implementasi, mengoreksi bug dalam kode saya sendiri. Karena kompleksitas alami dari pernyataan masalah, saya tidak bisa mendapatkan hasil pidato yang menakjubkan seperti manusia tetapi saya belajar banyak hal tentang Text-to-speech dan itu adalah tujuan utama ketika saya mulai mengerjakan proyek ini. Beberapa hasil untuk referensi:



Gambar 6.2 Spektrogram mel yang diprediksi : Seperti yang dapat Anda bandingkan dengan wilayah atas, ia memiliki banyak celah dan masih membutuhkan banyak pelatihan. Sisi kanan (hijau solid) hanya padding dalam satu batch.



Gambar 6.3 Target mel-spektogram



Gambar 6.4 Perhatian (Seperti yang Anda lihat di sisi kiri bawah, sepertinya sedang belajar untuk menyelaraskan tetapi masih membutuhkan sekitar satu minggu pelatihan untuk mendapatkan diagonal yang sempurna untuk perhatian)

Semua gambar yang dihasilkan di atas adalah setelah iterasi 50K (1 iterasi = 1 batch) yaitu 3 hari pelatihan. Model ini membutuhkan sekitar 300 ribu iterasi untuk mendekati seperti manusia. Anda dapat melihat bahwa, spektrogram mel yang diprediksi terlihat cukup bagus bahkan ketika perhatian tidak dipelajari dengan benar. Selamatkan diri Anda dari jebakan dan pedulikan perhatian!

Menerapkan model dan pelatihan ternyata tidak sesederhana yang saya kira awalnya. Saya menemukan banyak masalah yang saya ingin kalian ketahui sebelumnya dan menghemat waktu di GPU Anda. Pelajari data Anda. Ini adalah bagian terpenting dari proyek. Dengarkan sampel data Anda, periksa panjang sampel teks, durasi sampel audio, dll. Anda dapat menghemat banyak waktu selama pelatihan, jika Anda mengetahui data Anda dengan baik. M-AILABS mengumumkan kumpulan data pidato besar mereka awal tahun ini. Mereka memiliki kumpulan data ucap yang sangat banyak dalam berbagai bahasa. Saya menggunakan data Angela Merkel dari bagian wanita Jerman, yang memiliki 12 jam pidato dari pidato publik dan wawancaranya. Dataset ini lebih rendah dibandingkan dengan LJSpeech

(dataset bahasa Inggris paling populer, 24 jam bicara). Saya menemukan ini hanya ketika saya mulai berlatih dan menghabiskan berhari-hari mengamati hasilnya. Jadi, bersiaplah! TTS sangat mahal secara komputasi . Sebagai mahasiswa, saya baru saja memiliki akses ke satu GPU (Nvidia Tesla K80) di Google cloud. Mengingat struktur dataset yang saya gunakan, GPU saya hanya mengizinkan ukuran batch 8 saat pelatihan. Google mengatakan, mereka melatihnya dengan ukuran batch 64. Saya pertama kali mencoba dengan ukuran batch 2 (karena memori GPU terbatas) dan ketika model gagal menunjukkan konvergensi setelah 2-3 hari pelatihan, saya mengurutkan data saya sesuai panjang teks dan durasi audio, dan mulai berlatih dengan ukuran batch 8. Meskipun demikian, saya tidak dapat mengoptimalkan lebih banyak dengan dataset dan GPU yang saya miliki. Jadi, rencanakan dengan tepat. Rasio pemaksaan guru. Dalam pelatihan paksa guru, model dibantu oleh label yang benar yaitu menggunakan kerangka Kebenaran Dasar saat ini untuk memprediksi langkah decoding berikutnya. Tidak jelas dalam makalah tentang rasio apa yang digunakan. Bahkan jika perhatiannya tidak dipelajari, model akan memprediksi kerangka yang baik untuk data pelatihan dalam mode paksaan guru tetapi dalam mode evaluasi itu tidak akan berhasil karena kita tidak memiliki kebenaran dasar (Mengira model itu bekerja sejak prediksi mels tampak bagus terlepas dari keberpihakan yang buruk). Saya melakukan pelatihan dengan 1.0, 0.75 dan 0.5 untuk membuat model belajar keberpihakan. Selama mode evaluasi, pemaksaan guru harus dimatikan. Dibutuhkan berhari-hari untuk melatih dan mendapatkan keberpihakan. Ini adalah proses yang sangat rumit untuk melatih sistem TTS. Mungkin perlu sekitar 7–10 hari untuk melatih model asalkan Anda memiliki dukungan GPU terbatas (Kami bukan Google). Dan kemudian, men-debug kode dengan model seperti itu, adalah cerita lain. Tuning hyperparameter adalah bagian yang sangat penting dari sistem Tacotron-2. Ukuran batch, tingkat pembelajaran, rasio guru-memaksa, panjang batch adalah beberapa parameter yang harus Anda perhatikan juga. Hal-hal bervariasi dengan dataset, jadi sangat sensitif!

Text-to-speech masih merupakan masalah penelitian yang sangat kompleks dan sangat menarik untuk dikerjakan. Pengalaman saya secara keseluruhan luar biasa dan saya belajar banyak hal tentang sistem TTS, bentuk gelombang audio, jaringan berulang, mel-spektrogram, mekanisme perhatian dan saya harap posting ini dapat membantu Anda dalam perjalanan Anda dengan sistem TTS. Di masa mendatang, saya ingin melihat versi model Tacotron-2 yang dioptimalkan, sesuatu yang lebih kuat di berbagai bahasa, lebih mudah dilatih, dan tidak terlalu berat secara komputasi. Jadi, saya hanya akan mengatakan, praproses data Anda dengan baik, sesuaikan hyperparameter Anda, catat semuanya di tensorboard dan mulai! Semua yang terbaik!

BAB 7

WAVENET

7.1 WaveNet

WaveNet adalah jaringan saraf dalam untuk menghasilkan audio mentah. Itu dibuat oleh para peneliti di perusahaan AI yang berbasis di London, DeepMind . Teknik yang diuraikan dalam makalah pada bulan September 2016, [1] mampu menghasilkan suara mirip manusia yang terdengar relatif realistik dengan secara langsung memodelkan bentuk gelombang menggunakan metode jaringan saraf yang dilatih dengan rekaman ucapan nyata. Pengujian dengan bahasa Inggris dan Mandarin AS dilaporkan menunjukkan bahwa sistem tersebut mengungguli sistem text-to-speech (TTS) terbaik yang ada di Google, meskipun pada 2016 sintesis text-to-speechnya masih kurang meyakinkan daripada ucapan manusia yang sebenarnya. [2] Kemampuan WaveNet untuk menghasilkan bentuk gelombang mentah berarti dapat memodelkan semua jenis audio, termasuk musik. [3] Menghasilkan ucapan dari teks adalah tugas yang semakin umum berkat popularitas perangkat lunak seperti Siri Apple, Cortana Microsoft , Amazon Alexa , dan Asisten Google . [4]

Sebagian besar sistem tersebut menggunakan variasi teknik yang melibatkan fragmen suara yang digabungkan bersama untuk membentuk suara dan kata yang dapat dikenali. [5] Yang paling umum disebut TTS gabungan. [6] Ini terdiri dari per-

pustakaan besar fragmen pidato, direkam dari satu pembicara yang kemudian digabungkan untuk menghasilkan kata-kata dan suara yang lengkap. Hasilnya terdengar tidak wajar, dengan irama dan nada yang aneh. [7] Ketergantungan pada perpustakaan yang direkam juga membuat sulit untuk memodifikasi atau mengubah suara. [8]

Teknik lain, yang dikenal sebagai TTS parametrik, [9] menggunakan model matematika untuk menciptakan kembali suara yang kemudian dirangkai menjadi kata dan kalimat. Informasi yang diperlukan untuk menghasilkan suara disimpan dalam parameter model. Karakteristik pidato keluaran dikendalikan melalui input ke model, sedangkan pidato biasanya dibuat menggunakan synthesizer suara yang dikenal sebagai vocoder . Ini juga dapat menghasilkan audio yang terdengar tidak wajar.

WaveNet adalah jenis feedforward neural network yang dikenal sebagai deep convolutional neural network (CNN). Di WaveNet, CNN mengambil sinyal mentah sebagai input dan mensintesis output satu sampel pada satu waktu. Ia melakukannya dengan mengambil sampel dari distribusi softmax (yaitu kategorikal) dari nilai sinyal yang dikodekan menggunakan transformasi -law companding dan dikuantisasi ke 256 nilai yang mungkin. [10]

Menurut makalah penelitian DeepMind September 2016 asli WaveNet: A Generative Model for Raw Audio , [11] jaringan tersebut diberi makan bentuk gelombang nyata dari pidato dalam bahasa Inggris dan Mandarin. Saat ini melewati jaringan, ia mempelajari seperangkat aturan untuk menggambarkan bagaimana bentuk gelombang audio berkembang dari waktu ke waktu. Jaringan terlatih kemudian dapat digunakan untuk membuat bentuk gelombang seperti ucapan baru pada 16.000 sampel per detik. Bentuk gelombang ini termasuk napas yang realistik dan pukulan bibir – tetapi tidak sesuai dengan bahasa apa pun. [12]

WaveNet mampu memodelkan suara yang berbeda secara akurat, dengan aksen dan nada input yang berkorelasi dengan output. Misalnya, jika dilatih dengan bahasa Jerman, ia menghasilkan pidato bahasa Jerman. [13] Kemampuan ini juga berarti bahwa jika WaveNet diberi masukan lain – seperti musik – keluarannya akan berupa musik. Pada saat dirilis, DeepMind menunjukkan bahwa WaveNet dapat menghasilkan bentuk gelombang yang terdengar seperti musik klasik . [14]

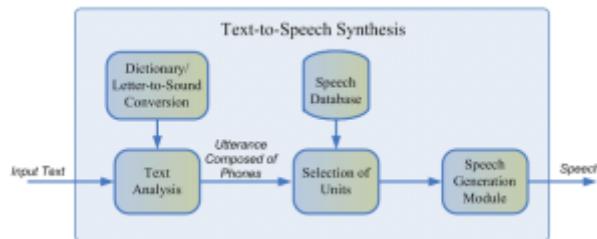
Menurut makalah Juni 2018 Disentangled Sequential Autoencoder , [15] Deep-Mind telah berhasil menggunakan WaveNet untuk "pertukaran konten" audio dan suara: jaringan dapat menukar suara pada rekaman audio dengan suara lain yang sudah ada sebelumnya sambil mempertahankan teks dan lainnya fitur dari rekaman asli. "Kami juga bereksperimen pada data urutan audio. Representasi kami yang tidak terjerat memungkinkan kami untuk mengubah identitas pembicara menjadi satu sama lain sambil mengkondisikan isi pidato." (hal. 5) "Untuk audio, ini memungkinkan kita untuk mengubah pembicara pria menjadi pembicara wanita dan sebaliknya [...]." (hal. 1) Menurut makalah tersebut, jumlah minimum dua digit jam (c. 50 jam) rekaman suara yang sudah ada sebelumnya dari suara sumber dan suara target diperlukan untuk dimasukkan ke WaveNet agar program dapat belajar fitur masing-masing sebelum dapat melakukan konversi dari satu suara ke suara lain dengan kualitas yang memuaskan. Penulis menekankan bahwa " [a] n keuntungan dari model ini adalah memisahkan fitur dinamis dari fitur statis [...] ." (hal. 8), yaitu WaveNet mampu

membedakan antara teks lisan dan mode pengiriman (modulasi, kecepatan, nada, suasana hati, dll.) untuk mempertahankan selama konversi dari satu suara ke suara lain di satu sisi, dan fitur dasar dari kedua sumber dan suara target yang diperlukan untuk bertukar di sisi lain.

Makalah tindak lanjut Januari 2019 Pembelajaran representasi ucapan tanpa pengawasan menggunakan autoencoder WaveNet [16] merinci metode untuk berhasil meningkatkan pengenalan otomatis yang tepat dan diskriminasi antara fitur dinamis dan statis untuk "pertukaran konten", terutama termasuk menukar suara pada rekaman audio yang ada, di untuk membuatnya lebih dapat diandalkan. Makalah tindak lanjut lainnya, Sample Efficient Adaptive Text-to-Speech , [17] tertanggal September 2018 (revisi terbaru Januari 2019), menyatakan bahwa DeepMind telah berhasil mengurangi jumlah minimum rekaman kehidupan nyata yang diperlukan untuk mengambil sampel suara yang ada melalui WaveNet untuk "hanya beberapa menit data audio" sambil mempertahankan hasil berkualitas tinggi.

Kemampuannya untuk mengkloning suara telah menimbulkan kekhawatiran etis tentang kemampuan WaveNet untuk meniru suara orang hidup dan mati. Menurut artikel BBC 2016 , perusahaan yang mengerjakan teknologi kloning suara serupa (seperti Adobe Voco) bermaksud untuk memasukkan tanda air yang tidak terdengar oleh manusia untuk mencegah pemalsuan, sambil mempertahankan kloning suara yang memuaskan, misalnya, kebutuhan tujuan industri hiburan akan memiliki kompleksitas yang jauh lebih rendah dan menggunakan metode yang berbeda dari yang diperlukan untuk menipu metode pembuktian forensik dan perangkat ID elektronik, sehingga suara alami dan suara yang dikloning untuk tujuan industri hiburan masih dapat dengan mudah dibedakan dengan analisis teknologi. [18]

Pada saat peluncurnya, DeepMind mengatakan bahwa WaveNet membutuhkan terlalu banyak kekuatan pemrosesan komputasi untuk digunakan dalam aplikasi dunia nyata. [19] Pada Oktober 2017, Google mengumumkan peningkatan kinerja 1.000 kali lipat bersama dengan kualitas suara yang lebih baik. WaveNet kemudian digunakan untuk menghasilkan suara Asisten Google untuk bahasa Inggris AS dan Jepang di semua platform Google. [20] Pada bulan November 2017, peneliti Deep-Mind merilis makalah penelitian yang merinci metode yang diusulkan untuk "menghasilkan sampel ucapan dengan ketelitian tinggi lebih dari 20 kali lebih cepat dari pada waktu nyata", yang disebut "Distilasi Kepadatan Probabilitas". [21] Pada konferensi pengembang I/O tahunan pada Mei 2018, diumumkan bahwa suara Asisten Google baru tersedia dan dimungkinkan oleh WaveNet; WaveNet sangat mengurangi jumlah rekaman audio yang diperlukan untuk membuat model suara dengan memodelkan audio mentah dari sampel aktor suara. [22]



Gambar 7.1 Speech Synthesis

Speech synthesis adalah sebuah kemampuan bicara manusia yang dibuat oleh manusia (artificial). Sebuah sistem komputer digunakan untuk tujuan ini yang disebut sebagai speech synthesizer, dan dapat diimplementasikan ke dalam software atau hardware. Sebagai contoh sebuah sistem text-to-speech (TTS) yang dapat mengkonversikan teks dengan bahasa biasa menjadi suara.

Sebuah sistem komputer yang digunakan untuk tujuan ini disebut speech synthesizer, dan dapat diimplementasikan dalam perangkat lunak atau perangkat keras produk. Sebuah teks-to-speech (TTS) sistem mengkonversi teks bahasa normal menjadi berbicara; sistem lain membuat representasi linguistik simbolik seperti transkripsi fonetik ke dalam pidato, pidato disintesis dapat dibuat dengan menggabungkan potongan pidato direkam yang disimpan dalam database. Sistem berbeda dalam ukuran unit pidato disimpan, sebuah sistem yang menyimpan telepon atau diphones menyediakan berbagai keluaran terbesar, tapi mungkin kurang jelas.

Speech synthesis adalah transformasi dari teks ke arah suara (speech). Transformasi ini mengkonversi teks ke pemandu suara (speech synthesis) yang sebisa mungkin dibuat menyerupai suara nyata, disesuaikan dengan aturan – aturan pengucapan bahasa. TTS (text to speech) dimaksudkan untuk membaca teks elektronik dalam bentuk buku, dan juga untuk menyuarakan teks dengan menggunakan pemanduan suara. Sistem ini dapat digunakan sebagai sistem komunikasi, pada sistem informasi referral, dapat diterapkan untuk membantu orang-orang yang kehilangan kemampuan melihat dan membaca.

Synthesized speech dapat diciptakan dengan menggabungkan beberapa potongan-potongan dari pembicaraan/pidato yang sudah direkam dalam sebuah basis data. Kualitas dari sebuah speech synthesizer dilihat dari kemiripannya dengan suara manusia dan kemampuannya untuk bisa dipahami. Program TTS yang jelas dapat membantu orang dengan gangguan visual atau ketidakmampuan membaca, untuk mendengarkan pada pekerjaan yang tertulis dalam komputer. Banyak Sistem Operasi komputer yang telah dimasukkan speech synthesizer sejak tahun 1980-an.

Sebuah sistem text-to-speech (atau “mesin”) terdiri dari dua bagian: front-end dan back-end . Front-end memiliki dua tugas utama. Pertama, mengubah teks mentah berisi simbol seperti angka dan singkatan menjadi setara dengan kata-kata tertulis-out. Proses ini sering disebut normalisasi teks, pra-pengolahan, atau tokenization . Front-end kemudian memberikan transkripsi fonetik untuk setiap kata, dan membagi dan menandai teks ke unit prosodi , seperti frase , klausa , dan kalimat . Proses mene-

tapkan transkripsi fonetik untuk kata-kata ini disebut teks-ke-fonem atau grafem konversi -untuk-fonem. Transkripsi fonetik dan informasi prosodi bersama-sama membentuk representasi linguistik simbolik yang output dengan front-end. Back-end sering disebut sebagai synthesizer-maka mengubah representasi linguistik simbolik menjadi suara. Dalam sistem tertentu, bagian ini meliputi perhitungan dari target prosodi (kontur pitch, durasi fonem), yang kemudian dikenakan pada pidato output.

7.1.1 Sejarah Speech Synthesis

Upaya yang paling awal untuk menghasilkan lahirnya pemandu suara, pada abad XVIII. Terlepas dari kenyataan bahwa upaya pertama adalah bentuk mesin mekanis, kita dapat mengatakan hari ini bahwa synthesizer sudah berkualitas tinggi. Pada tahun 1779 di

St Petersburg, Rusia Profesor Kratzenshtein Kristen fisiologis menjelaskan perbedaan antara lima vokal panjang (/ A /, / e /, / i /, / o /, dan / u /) dan membuat alat untuk menghasilkan mereka artifisial. Tahun 1791 di Wina, Wolfgang von Kempelen memperkenalkan nya “Akustik-Mekanik Mesin Speech”. Dalam sekitar pertengahan 1800-an Charles Wheatstone dibangun terkenal versi mesin berbicara von Kempenlen’s.

Generasi dari sistem pemanduan suara ini dapat dibagi ke dalam 3 masa, yaitu:

1. Generasi pertama (1962-1977). Format sintesis dari fonem adalah teknologi dominan. Teknologi ini memanfaatkan aturan berdasarkan penguraian fonetik pada kalimat untuk kontur frekuensi forman. Beberapa sintesis masih miskin atau kurang dalam kejelasan dan kealamiannya.
2. Generasi kedua (1977-1992). Metode pemandu suara adalah diphone diwakilkan dengan parameter LPC. Hal tersebut menunjukkan bahwa kejelasan yang baik pada pemandu suara dapat diperoleh dengan andal dari input teks dengan menggabungkan diphone yang sesuai dengan unit. Kejelasan meningkat selama sintesis forman, tetapi kealamian dari pemandu suara masih tetap rendah.
3. Generasi ketiga (1992-sekarang). Generasi ini ditandai dengan metode ‘ sintesis pemilihan unit’ yang diperkenalkan dan disempurnakan oleh Sagisaka di Labs ATR Kyoto. Hasil dari pemandu suara pada periode ini sangat mendekati human-generated speech pada bagian kejelasan dan kealamian.

Teknologi pemandu suara modern melibatkan metode dan algoritma yang canggih dan rumit. alat pemandu suara dari keluarga “Infovox” mungkin menjadi salah satu multi bahasa TTS yang paling dikenal saat ini. Versi komersial pertamanya, Infovox-SA 101, dikembangkan pada tahun 1982 di Institute Teknologi Royal, Swedia dan didasarkan pada sintesis forman.

AT & T Bell Laboratories (Lucent Technologies) juga memiliki tradisi yang sangat panjang tentang pemandu suara (speech synthesis). TTS lengkap yang pertama didemostrasikan di Boston pada tahun 1972 dan diliris pada tahun 1973. Hal ini didasarkan pada model artikulatoris yang dikembangkan oleh Ceceil Coker (Klatt

1987). Pengembangan proses dari sistem penggabungan sintesis ini dimulai oleh Joseph Olive pada pertengahan tahun 1970-an (Bell Labs 1997). Sistem ini sekarang sudah tersedia untuk bahasa Inggris, Perancis, Spanyol, Italia, Jerman, Rusia, Rumania, Cina, dan Jepang (McBius et al 1996).

7.1.2 Perangkat Speech Synthesis

Pidato sistem sintesis berbasis komputer pertama diciptakan pada akhir 1950-an. Pertama umum Inggris sistem text-to-speech dikembangkan oleh Noriko Umeda et al. Pada tahun 1968 di Laboratorium Elektroteknik, Jepang. Pada tahun 1961, fisikawan John Larry Kelly, Jr dan Louis rekan Gerstman menggunakan IBM 704 komputer untuk mensintesis pidato, acara yang paling menonjol dalam sejarah Bell Labs . Kelly perekam suara synthesizer (vocoder) ulang lagu " Daisy Bell ", dengan irungan musik dari Max Mathews .Kebetulan, Arthur C. Clarke mengunjungi teman dan kolega John Pierce di fasilitas Bell Labs Murray Hill. Clarke begitu terkesan oleh demonstrasi bahwa ia digunakan dalam adegan klimaks dari skenario-Nya untuk novel nya 2001: A Space Odyssey, di mana HAL 9000 komputer menyanyikan lagu yang sama seperti yang sedang ditidurkan oleh astronot Dave Bowman. Meskipun keberhasilan pidato sintesis murni elektronik, penelitian masih terus dilakukan ke synthesizer pidato mekanis.

Handheld elektronik menampilkan sintesis pidato mulai muncul pada 1970-an. Salah satu yang pertama adalah Telesensory Systems Inc(TSI) Pidato + kalkulator portabel untuk orang buta pada tahun 1976. Perangkat lain yang diproduksi terutama untuk tujuan pendidikan, seperti Bicara & Eja , yang diproduksi oleh Texas Instruments pada tahun 1978. Fidelity merilis versi berbicara komputer catur elektronik pada tahun 1979. Yang pertama video game yang memiliki fitur sintesis pidato adalah 1.980 shoot 'em up arcade game , Stratovox , dari Sun Electronics. Contoh lain awal adalah versi arcade dari Berzerk , dirilis pada tahun yang sama.Pertama multi-player permainan elektronik menggunakan sintesis suara adalah Milton dari Milton Bradley Company , yang memproduksi perangkat di tahun 1980.

7.1.3 Teknologi Speech Synthesis

Yang paling penting dalam kualitas sistem speech synthesis adalah kealamian dan kejelasannya. Kealamian menjelaskan bagaimana dekatnya suara output dengan suara manusia, sementara kejelasan adalah dengan kemudahan di mana output tersebut dapat dipahami. Speech synthesizer yang ideal adalah yang alami dan jelas. Sistem speech synthesis biasanya mencoba untuk memaksimalkan kedua karakteristik.

Kualitas terpenting dari sebuah aplikasi speech synthesizer adalah seberapa alami dan inteligibel output yang dihasilkannya. Alami, artinya seberapa dekat suara yang dihasilkan aplikasi speech synthesizer dengan suara manusia. Sedangkan inteligibel adalah seberapa mudah output tersebut dipahami oleh manusia. Semua aplikasi speech synthesizer berusaha untuk menghasilkan output yang alami dan inteligibel sekaligus.

Sampai saat ini, ada banyak teknologi untuk meng-generate gelombang suara sintetis ini. Dua teknologi yang paling banyak digunakan adalah concatenative synthesis dan formant synthesis. Keduanya memiliki keunggulan dan kekurangan sendiri-sendiri.

Teknologi pertama, concatenative synthesis, berbasis pada rangkaian (atau merangkai bersama) segmen-semen dari suara yang direkam. Umumnya, teknologi ini menghasilkan suara sintesis yang terdengar paling alami. Namun, perbedaan antara suara alami yang direkam dengan segmentasi gelombang bunyi kadang menghasilkan suara yang mengganggu. Mirip seperti suara pemberitahuan nomor antrean di bank atau suara call center operator ponsel yang menyebutkan sisa pulsa dan masa berlaku kartu ponsel anda.

Teknologi kedua, formant synthesis, tidak menggunakan sampel suara manusia melainkan membuat suara sintesi menggunakan model akustik. Parameter-parameter seperti frekuensi dasar, alunan suara, dan tingkat kebisingan bervariasi dari waktu ke waktu untuk menciptakan gelombang suara buatan.

Kebanyakan aplikasi berbasis teknologi ini menghasilkan suara buatan (tidak alami) seperti suara robot. Melihat keterbatasan kedua teknologi ini dalam menghasilkan suara buatan, seperti kita harus sabar menunggu pengembangannya lebih lanjut dalam beberapa tahun atau dekade ke depan.

Kualitas yang paling penting dari sebuah sistem sintesis pidato kewajaran dan dimengerti. Kealamian menjelaskan seberapa dekat output terdengar seperti suara manusia, sementara kejelasan adalah kemudahan yang output dipahami. Speech synthesizer yang ideal adalah baik alam dan dimengerti. Sistem sintesis pidato biasanya mencoba untuk memaksimalkan kedua karakteristik.

Dua teknologi utama dalam pembuatan gelombang suara synthetic speech adalah Concatenative Synthesis dan Formant Synthesis. Setiap teknologi mempunyai kelebihan dan kelemahannya, dan penggunaan yang ditujukan dari sistem synthesis akan menentukkan pendekatan mana yang digunakan.

1. Concatenative Synthesis. Concatenative synthesis didasarkan dengan penggabungan dari segmen-semen dari pembicaraan yang sudah direkam. Secara umum, concatenative synthesis memproduksi synthesized speech dengan suara yang paling alami. Tetapi, perbedaan antara variasi alami dalam pembicaraan dan sifat dari teknik otomasi untuk pensegmentasi gelombang suara terkadang menghasilkan kesalahan suara dalam output. Namun, perbedaan antara variasi alami dalam pidato dan sifat teknik otomatis untuk membagi bentuk gelombang kadang-kadang menyebabkan gangguan terdengar pada output. Ada tiga sub-jenis utama dari sintesis concatenative.
 - (a) Sintesis Pemilihan unit. Sintesis Pemilihan unit menggunakan besar database pidato direkam. Selama pembuatan database, setiap ucapan tercatat tersegmentasi ke dalam beberapa atau semua hal berikut: individu telepon, di-phones, setengah-telepon, suku kata, morfem, kata, frase, dan kalimat. Biasanya, pembagian ke dalam segmen dilakukan dengan menggunakan di-modifikasi khusus recognizer pidato disetel ke “keselarasan dipaksa” mode dengan beberapa koreksi manual setelah itu, dengan menggunakan repre-

sentasi visual seperti yang gelombang dan spektrogram . Sebuah indeks unit dalam database pidato kemudian dibuat berdasarkan segmentasi dan parameter akustik seperti frekuensi dasar (lapangan), durasi, posisi dalam suku kata, dan telepon tetangga. Pada waktu berjalan , target ucapan yang dinginkan dibuat dengan menentukan rantai terbaik unit calon dari database (pemilihan unit). Proses ini biasanya dicapai dengan menggunakan khusus tertimbang pohon keputusan.

Pemilihan unit menyediakan kealamian terbesar, karena hanya berlaku sedikit pemrosesan sinyal digital (DSP) untuk pidato direkam. DSP sering membuat pidato yang direkam terdengar kurang alami, meskipun beberapa sistem menggunakan sejumlah kecil pengolahan sinyal pada titik Rangkaian untuk menghaluskan bentuk gelombang. Output dari yang terbaik unit-seleksi sistem sering dibedakan dari suara manusia nyata, terutama dalam konteks dimana sistem TTS telah disetel. Namun, kealamian maksimum biasanya membutuhkan unit-pilihan database pidato menjadi sangat besar, dalam beberapa sistem mulai ke gigabyte data dicatat, mewakili puluhan jam berbicara. Juga, pilihan algoritma Unit telah dikenal untuk memilih segmen dari Tempat yang menghasilkan kurang dari sintesis ideal (misalnya kata-kata kecil menjadi tidak jelas) bahkan ketika pilihan yang lebih baik ada dalam database. Baru-baru ini, peneliti telah mengusulkan berbagai metode otomatis untuk mendekripsi segmen alami di unit-pilihan sistem sintesis pidato.

- (b) Sintesis diphone Sintesis diphone menggunakan database pidato minimal berisi semua diphones (suara-to-suara transisi) yang terjadi dalam suatu bahasa. Jumlah diphones tergantung pada fonotaktik bahasa: misalnya, Spanyol memiliki sekitar 800 diphones, dan Jerman sekitar 2500. Dalam sintesis diphone, hanya satu contoh dari setiap diphone terkandung dalam database pidato. Pada saat runtime, target prosodi kalimat ditumpangkan pada unit-unit minimal dengan cara pemrosesan sinyal digital teknik seperti linear predictive coding ,PSOLA atau MBROLA. Diphone sintesis menderita gangguan sonik sintesis concatenative dan robot-terdengar sifat sintesis forman, dan memiliki beberapa keuntungan baik pendekatan lain dari ukuran kecil. Dengan demikian, penggunaannya dalam aplikasi komersial menuju, meskipun terus digunakan dalam penelitian karena ada beberapa implementasi perangkat lunak tersedia secara bebas.
- (c) Domain-spesifik sintesis Domain-spesifik sintesis concatenates direkam sebelumnya kata dan frase untuk menciptakan ucapan-ucapan yang lengkap. Hal ini digunakan dalam aplikasi di mana berbagai teks output sistem akan terbatas pada domain tertentu, seperti jadwal angkutan pengumuman atau laporan cuaca. Teknologi ini sangat sederhana untuk menerapkan, dan telah digunakan secara komersial untuk waktu yang lama , dalam perangkat seperti berbicara jam dan kalkulator. Tingkat kealamian sistem ini bisa sangat tinggi karena berbagai jenis kalimat terbatas, dan mereka cocok dengan prosodi dan intonasi dari rekaman asli.

Karena sistem ini dibatasi oleh kata-kata dan frasa dalam database mereka, mereka tidak tujuan umum dan hanya dapat mensintesis kombinasi kata dan frase yang mereka telah terprogram. Campuran kata-kata dalam bahasa alami diucapkan namun masih dapat menyebabkan masalah kecuali banyak variasi diperhitungkan. Misalnya, dalam non-rhotic dialek dari bahasa Inggris “r” dalam kata-kata seperti “jelas” biasanya hanya diucapkan ketika kata berikut memiliki vokal sebagai huruf pertama (misalnya “membersihkan” direalisasikan sebagai). Demikian juga di Perancis , banyak konsonan akhir menjadi tidak lagi diam jika diikuti oleh sebuah kata yang dimulai dengan vokal, efek yang disebut penghubung . Ini pergantian tidak bisa direproduksi oleh sistem kata-Rangkaian sederhana, yang akan membutuhkan kompleksitas tambahan untuk konteks-sensitif .

2. Formant Synthesis Formant synthesis tidak menggunakan pembicaraan manusia sebagai sample pada runtime. Daripada itu, synthesized speech yang dihasilkan dibuat dengan additive synthesis dan sebuah model akustik (physical modelling synthesis).

Parameter seperti frekuensi dasar, penyuaraan, dan tingkat kebisingan di variasikan dari waktu ke waktu untuk menciptakan gelombang buatan (artificial) dari sebuah pembicaraan. Banyak sistem yang berdasarkan formant synthesis menciptakan pembicaraan yang seperti robot yang tidak mungkin dapat dikepalai sebagai suara manusia. Tetapi, kealamian maksimum bukan selalu tujuan dari sebuah sistem speech synthesis, dan sistem formant synthesis mempunyai keuntungan dari sistem concatenative. Pembicaraan yang di-formant synthesis-kan dapat menjadi sangat jelas, bahkan dalam kecepatan yang tinggi, sehingga menghindari kesalahan suara yang sering dialami sistem concatenative.

Formant synthesis biasanya program yang lebih kecil dari concatenative sistem karena ia tidak menggunakan basis data dari sampel-sampel pembicaraan. Oleh karena itu formant synthesis dapat ditanamkan dalam sistem yang mempunyai memory dan mikroprosesor yang terbatas. Karena sistem yang berdasarkan formant mempunyai kendali penuh dari sluruh aspek dari hasil pembicaraan, variasi yang luas dari prosodi dan intonasi dapat dihasilkan, menyampaikan tidak hanya pertanyaan dan pernyataan tetapi juga emosi dan nada suara.

Formant sintesis tidak menggunakan sampel suara manusia pada saat runtime. Sebaliknya, keluaran suara yang disintesis dibuat menggunakan aditif sintesis dan model akustik (sintesis pemodelan fisik). Parameter seperti frekuensi dasar , menyuarakan , dan kebisingan tingkat yang bervariasi dari waktu ke waktu untuk membuat gelombang pidato buatan. Metode ini kadang-kadang disebut aturan berbasis sintesis; Namun, banyak sistem concatenative juga memiliki aturan berbasis komponen. Banyak sistem yang didasarkan pada teknologi sintesis forman menghasilkan buatan, robot yang terdengar pidato yang tidak akan pernah salah untuk pidato manusia. Namun, kealamian maksimum tidak selalu tujuan sistem sintesis pidato, dan sistem sintesis forman memiliki keunggulan dibandingkan sistem concatenative. Pidato forman-disintesis dapat diandalkan

dimengerti, bahkan pada kecepatan yang sangat tinggi, menghindari Glitches akustik yang biasanya wabah sistem concatenative. Kecepatan tinggi disintesis pidato digunakan oleh tunanetra untuk navigasi cepat komputer menggunakan pembaca layar . Synthesizer forman adalah program biasanya lebih kecil dibandingkan dengan sistem concatenative karena mereka tidak memiliki database contoh pidato. Karena itu mereka dapat digunakan dalam embedded system , di mana memori dan mikroprosesor daya terutama terbatas. Karena sistem berbasis forman memiliki kontrol penuh dari semua aspek pidato output, berbagai prosodies dan intonasi dapat menjadi output, tidak hanya menyampaikan pertanyaan dan pernyataan, tetapi berbagai emosi dan nada suara.

Contoh non-real-time tapi sangat akurat kontrol intonasi dalam sintesis forman meliputi pekerjaan yang dilakukan pada akhir tahun 1970 untuk Texas Instruments mainan Bicara & Eja , dan pada awal tahun 1980 Sega arcade mesin dan dalam banyak Atari, Inc. game arcade menggunakan TMS5220 LPC Chips . Menciptakan intonasi yang tepat untuk proyek ini adalah telaten, dan hasilnya masih harus dicocokkan dengan real-time text-to-speech interface.

3. Sintesis artikulatoris Sintesis artikulatoris mengacu pada teknik komputasi untuk sintesis pidato berdasarkan model manusia saluran vokal dan artikulasi proses yang terjadi di sana. Synthesizer artikulatoris pertama teratur digunakan untuk percobaan laboratorium dikembangkan di Haskins Laboratories di pertengahan 1970-an oleh Philip Rubin, Tom Baer, dan Paul Mermelstein. Synthesizer ini, yang dikenal sebagai ASY, didasarkan pada model saluran vokal dikembangkan di Bell Laboratories pada tahun 1960 dan 1970-an oleh Paul Mermelstein, Cecil Coker, dan rekan.

Sampai saat ini, model sintesis artikulatoris belum dimasukkan ke dalam sistem sintesis pidato komersial. Sebuah pengecualian adalah NeXT sistem berbasis awalnya dikembangkan dan dipasarkan oleh Trillium Suara Research, sebuah perusahaan spin-off dari University of Calgary , di mana banyak riset asli dilakukan. Setelah runtuhan berbagai inkarnasi NeXT (dimulai oleh Steve Jobs pada akhir tahun 1980 dan bergabung dengan Apple Computer pada tahun 1997), perangkat lunak TRILLIUM diterbitkan di bawah GNU General Public License , dengan bekerja terus sebagai gnuSpeech . Sistem, pertama kali dipasarkan pada tahun 1994, memberikan penuh text-to-speech konversi berbasis artikulatoris menggunakan Waveguide atau transmisi-line analog dari saluran mulut dan hidung manusia dikendalikan oleh Carré ini “model daerah khas”.

4. Sintesis berbasis HMM Sintesis berbasis HMM adalah metode sintesis berdasarkan model Markov tersembunyi , juga disebut statistik Parametrik Sintesis. Dalam sistem ini, spektrum frekuensi (vokal),frekuensi dasar (sumber vokal), dan durasi (prosodi) berbicara dimodelkan secara bersamaan oleh HMMs. Pidato bentuk gelombang yang dihasilkan dari HMMs sendiri berdasarkan maksimum kriteria.
5. Sintesis Sinewave Sintesis sinewave adalah teknik untuk sintesis pidato dengan mengganti forman (band utama energi) dengan peluit nada murni.

Ada beberapa masalah yang terdapat pada pemanduan suara, yaitu:

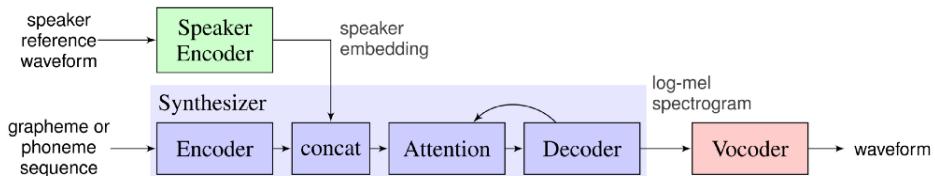
- (a) User sangat sensitif terhadap variasi dan informasi suara. Oleh sebab itu, mereka tidak dapat memberikan toleransi atas ketidaksempurnaan pemandu suara.
- (b) Output dalam bentuk suara tidak dapat diulang atau dicari dengan mudah.
- (c) Meningkatkan keberisikan pada lingkungan kantor atau jika menggunakan handphone, maka akan meningkatkan biaya pengeluaran.

BAB 8

SINGLE-SPEAKER VOICE CLONING

8.1 Model SV2TTS

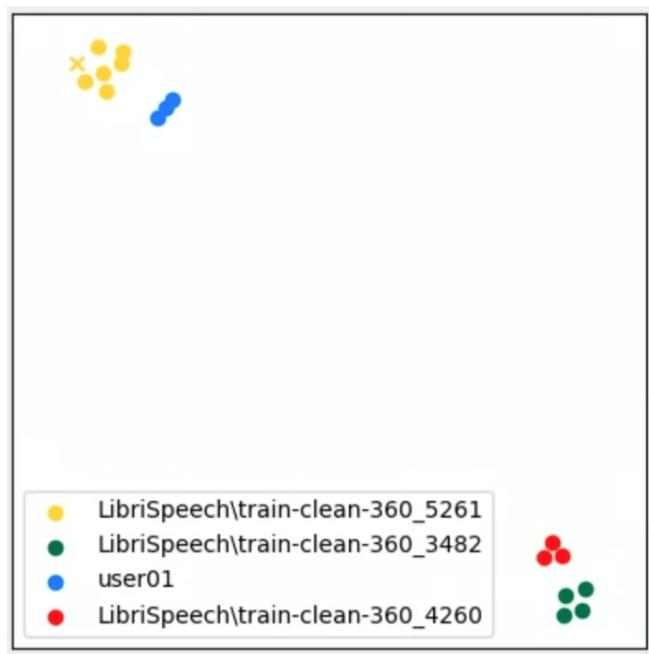
Model Speech Vector to TTS (SV2TTS) terdiri dari tiga bagian, masing-masing dilatih secara individual. Hal ini memungkinkan setiap bagian dilatih pada data independen, sehingga mengurangi kebutuhan untuk mendapatkan data multispeaker berkualitas tinggi.



Gambar 8.1 Arsitektur SV2TTS umum Sumber: Jia, Zhang, dan Weiss et al.

8.1.1 Speaker Encoder

Bagian pertama dari model SV2TTS adalah encoder speaker. Tugas speaker encoder adalah mengambil beberapa input audio (dikodekan sebagai bingkai spektogram mel), dari speaker tertentu, dan mengeluarkan embedding yang menangkap "bagaimana suara speaker". Encoder pembicara tidak peduli dengan kata-kata yang diucapkan pembicara, atau tentang kebisingan di latar belakang, yang dia pedulikan hanyalah suara pembicara, misalnya, suara bernada tinggi/rendah, aksen, nada, dll. Semua fitur ini digabungkan menjadi vektor berdimensi rendah, yang secara formal dikenal sebagai vektor-d, atau secara informal sebagai penyematan speaker. Akibatnya, ujaran yang diucapkan oleh penutur yang sama akan saling berdekatan pada penyematan penutur, sedangkan tuturan yang diucapkan oleh penutur yang berbeda akan berjauhan pada penyematan penutur.



Gambar 8.2 Representasi visual dari embeddings, setiap warna adalah speaker yang berbeda
Sumber: Jia, Zhang, dan Weiss et al.

Untuk belajar menghasilkan embeddings ini, penulis menjelaskan proses berikut:

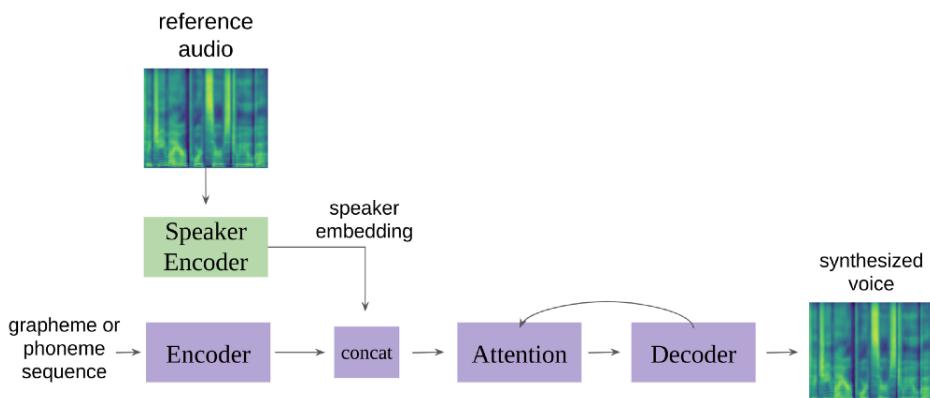
1. Pertama, contoh audio ucapan disegmentasi menjadi klip 1,6 detik tanpa transkrip dan diubah menjadi spektrogram mel.
2. Kemudian speaker encoder dilatih untuk mengambil dua sampel audio dan memutuskan apakah speaker yang sama memproduksinya atau tidak. Sebagai produk

sampingan, ini memaksa encoder speaker untuk membuat embeddings yang mewakili suara speaker.

Proses pelatihan mirip dengan jaringan saraf siam jika Anda sudah familiar dengan mereka.

8.1.2 Synthesizer

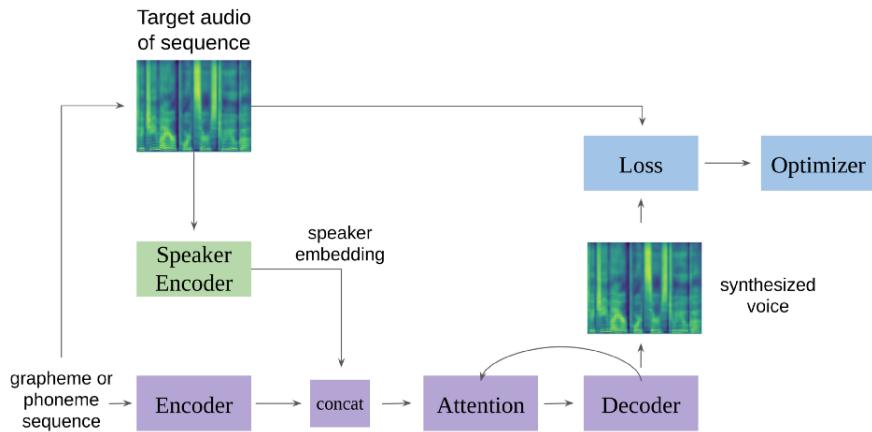
Synthesizer adalah bagian dari SV2TTS yang menganalisis input teks untuk membuat spektrogram mel, yang kemudian diubah oleh vocoder menjadi suara. Synthesizer mengambil urutan teks — dipetakan ke fonem (unit terkecil dari suara manusia, misalnya, suara yang Anda buat saat mengucapkan 'a'), bersama dengan embeddings yang dihasilkan oleh encoder speaker, dan menggunakan arsitektur Tacotron 2 untuk menghasilkan bingkai dari spektrogram mel secara berulang



Gambar 8.3 Model Arsitektur Multi-Speaker Voice Cloning Sumber: Jia, Zhang, dan Weiss et al.

Berikut Proses Training Synthesizer:

1. Pertama, kami mengumpulkan urutan fonem dan spektrogram mel dari pembicara yang mengucapkan kalimat itu.
2. Kemudian, spektrogram mel diteruskan ke encoder speaker untuk menghasilkan embedding speaker.
3. Selanjutnya, pembuat enkoder penyintesis menggabungkan penyandian urutan fonemnya dengan penyematan speaker.
4. Spektrogram mel dihasilkan secara berulang oleh decoder dan bagian perhatian dari synthesizer
5. Terakhir, spektrogram mel dibandingkan dengan target awal untuk menghasilkan kerugian, yang kemudian dioptimalkan



Gambar 8.4 Training Synthesizer Sumber: Jia, Zhang, dan Weiss et al.

8.1.3 Vocoder

Pada titik ini, synthesizer telah membuat spektrogram mel, tetapi kami masih belum dapat mendengarkan apa pun. Untuk mengubah spektrogram mel menjadi gelombang audio mentah, penulis menggunakan vocoder. Vocoder khusus yang digunakan di sini didasarkan pada model WaveNet DeepMind , yang menghasilkan bentuk gelombang audio mentah dari teks, dan pada satu titik canggih untuk sistem TTS.



Gambar 8.5 Vocoder Sumber: Jia, Zhang, dan Weiss et al.

8.1.4 Alur Model SV2TTS

berikut adalah alur modelnya:

1. Encoder speaker mendengarkan sampel audio yang diberikan dan menghasilkan embedding
2. Synthesizer mengambil daftar fonem dan penyematan speaker, kemudian menghasilkan spektrogram mel
3. Vocoder saraf menguraikan spektrogram mel menjadi bentuk gelombang audio yang dapat kita dengarkan

8.1.5 Mengukur Kualitas Hasil

Penulis menggunakan penilai manusia untuk mengukur kealamian dan kesamaan ucapan yang dihasilkan model. Kealamian mengukur seberapa "manusia" ucapan itu terdengar, sementara kesamaan mengukur seberapa mirip suara pidato yang disintesis dengan pembicara aslinya. Penilai berinteraksi melalui GUI yang terlihat seperti:

Instructions

In this task, your job is to evaluate if the two speech audio samples are from the same speaker. Please release this task if any of the following are true:

- You think you do not have good listening ability.
- There is considerable background noise (street noise, loud fan/air-conditioner, open TV/radio, people talking, etc).
- For any reason, you can't hear the audio samples.

Task

How are you listening to the speech samples?

- Headphones, with no noise in the background.** I am listening to the speech sample using headphones and there is no noise around me (people talking, music playing, air-conditioners, fans, etc.).
 Headphones, with some low-level noise in the background. I am listening to the speech sample using headphones and there is some low-level noise around me (people talking, music playing, air-conditioners, fans, etc.).
 Audio speakers or other.

Speaker Similarity Between Two Speech Samples

Please listen to the two speech samples below (Sample A and Sample B) and rate how similar they are. Your rating should reflect your evaluation of how close the voices of the two speakers sound. You **should not judge the content, grammar, or audio quality** of the sentences; instead, just focus on the similarity of the speakers to one another.

Speech samples (please listen at least **two times each**)

Sample A	▶ 0:00 / 0:02	<input type="range"/>		<input type="button" value="▶"/>
Sample B	▶ 0:00 / 0:03	<input type="range"/>		<input type="button" value="▶"/>

Please rate the similarity of the two speech samples



Comment (optional)

Gambar 8.6 Pengukuran Kualitas Model Sumber: Jia, Zhang, dan Weiss et al.

Teknik penggunaan rating crowdsourced seperti ini sering disebut dengan Mean Opinion Score atau MOS. Penulis menemukan bahwa hasil akhir untuk kealamian akhirnya terlihat seperti:

System	VCTK Seen	VCTK Unseen	LibriSpeech Seen	LibriSpeech Unseen
Ground truth	4.43 ± 0.05	4.49 ± 0.05	4.49 ± 0.05	4.42 ± 0.07
Embedding table	4.12 ± 0.06	N/A	3.90 ± 0.06	N/A
Proposed model	4.07 ± 0.06	4.20 ± 0.06	3.89 ± 0.06	4.12 ± 0.05

Gambar 8.7 MOS Sumber: Jia, Zhang, dan Weiss et al.

Jika Anda ingin analisis lengkap ini, saya akan merekomendasikan membaca bagian 3.1 dari makalah asli. Namun, secara ringkas:

1. Model SV2TTS yang diusulkan mencapai sekitar 4,0 MOS di semua kumpulan data.
2. LibriSpeech (salah satu set data pelatihan) diklasifikasikan sebagai kurang alami, karena tidak ada tanda baca dalam transkrip, sehingga menyulitkan model untuk mempelajari jeda.
3. Sistem tabel penyematan menggunakan tabel pencarian penyematan speaker tetapi sebaliknya memiliki arsitektur yang sama.
4. Karena memiliki tabel pencarian, tidak dapat digeneralisasi, dan oleh karena itu tidak dapat dievaluasi pada kumpulan data yang tidak terlihat.
5. Kealamian dari contoh yang tidak terlihat dan yang terlihat sangat mirip, yang sangat mengesankan.

Dan untuk kesamaan ucapan, hasilnya terlihat seperti:

System	Speaker Set	VCTK	LibriSpeech
Ground truth	Same speaker	4.67 ± 0.04	4.33 ± 0.08
Ground truth	Same gender	2.25 ± 0.07	1.83 ± 0.07
Ground truth	Different gender	1.15 ± 0.04	1.04 ± 0.03
Embedding table	Seen	4.17 ± 0.06	3.70 ± 0.08
Proposed model	Seen	4.22 ± 0.06	3.28 ± 0.08
Proposed model	Unseen	3.28 ± 0.07	3.03 ± 0.09

Gambar 8.8 Speech Similarity Sumber: Jia, Zhang, dan Weiss et al.

Sekali lagi, Anda mungkin ingin membaca bagian 3.2 dari makalah asli untuk analisis lengkapnya. Tapi secara ringkas:

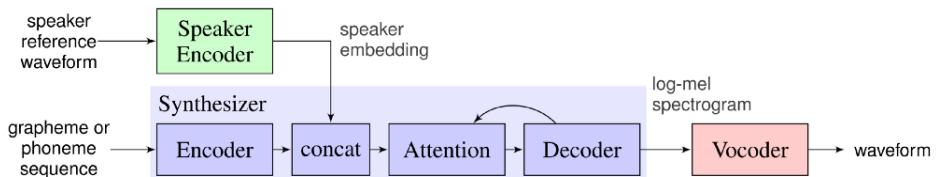
1. Skor lebih tinggi untuk VCTK (set data pelatihan lainnya), menunjukkan bentuk dataset yang lebih terstruktur.
2. Skor model SV2TTS antara "cukup mirip" dan "sangat mirip" pada skala evaluasi untuk pembicara yang tidak terlihat.
3. Meskipun ini sulit diukur secara matematis, model secara keseluruhan menangkap karakteristik pembicara secara akurat.

BAB 9

MULTI-SPEAKER VOICE CLONING

9.1 Model SV2TTS

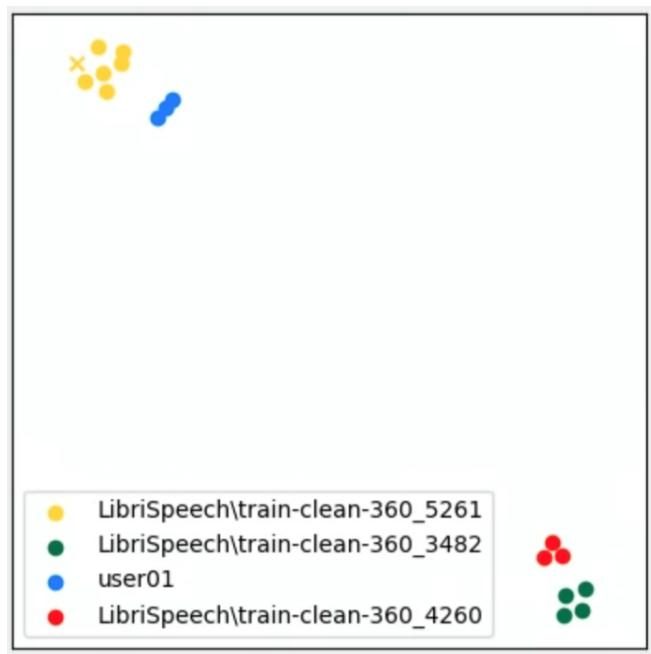
Model Speech Vector to TTS (SV2TTS) terdiri dari tiga bagian, masing-masing dilatih secara individual. Hal ini memungkinkan setiap bagian dilatih pada data independen, sehingga mengurangi kebutuhan untuk mendapatkan data multispeaker berkualitas tinggi.



Gambar 9.1 Arsitektur SV2TTS umum Sumber: Jia, Zhang, dan Weiss et al.

9.1.1 Speaker Encoder

Bagian pertama dari model SV2TTS adalah encoder speaker. Tugas speaker encoder adalah mengambil beberapa input audio (dikodekan sebagai bingkai spektrogram mel), dari speaker tertentu, dan mengeluarkan embedding yang menangkap "bagaimana suara speaker". Encoder pembicara tidak peduli dengan kata-kata yang diucapkan pembicara, atau tentang kebisingan di latar belakang, yang dia pedulikan hanyalah suara pembicara, misalnya, suara bernada tinggi/rendah, aksen, nada, dll. Semua fitur ini digabungkan menjadi vektor berdimensi rendah, yang secara formal dikenal sebagai vektor-d, atau secara informal sebagai penyematan speaker. Akibatnya, ujaran yang diucapkan oleh penutur yang sama akan saling berdekatan pada penyematan penutur, sedangkan tuturan yang diucapkan oleh penutur yang berbeda akan berjauhan pada penyematan penutur.



Gambar 9.2 Representasi visual dari embeddings, setiap warna adalah speaker yang berbeda
Sumber: Jia, Zhang, dan Weiss et al.

Untuk belajar menghasilkan embeddings ini, penulis menjelaskan proses berikut:

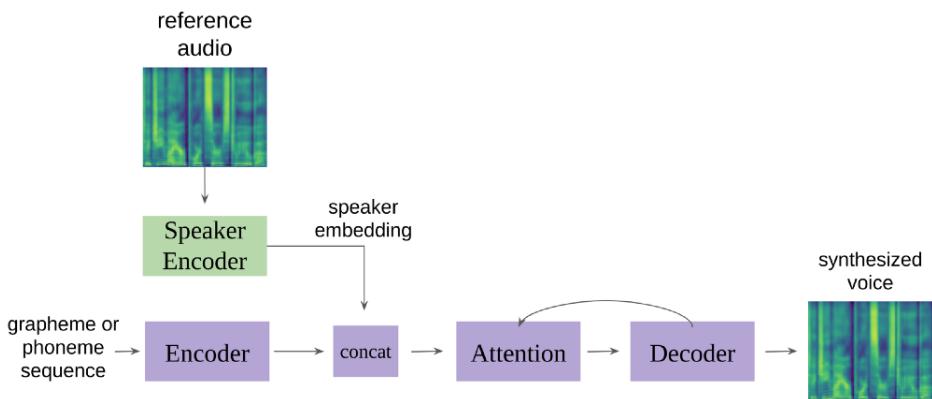
1. Pertama, contoh audio ucapan disegmentasi menjadi klip 1,6 detik tanpa transkrip dan diubah menjadi spektrogram mel.
2. Kemudian speaker encoder dilatih untuk mengambil dua sampel audio dan memutuskan apakah speaker yang sama memproduksinya atau tidak. Sebagai produk

sampingan, ini memaksa encoder speaker untuk membuat embeddings yang mewakili suara speaker.

Proses pelatihan mirip dengan jaringan saraf siam jika Anda sudah familiar dengan mereka.

9.1.2 Synthesizer

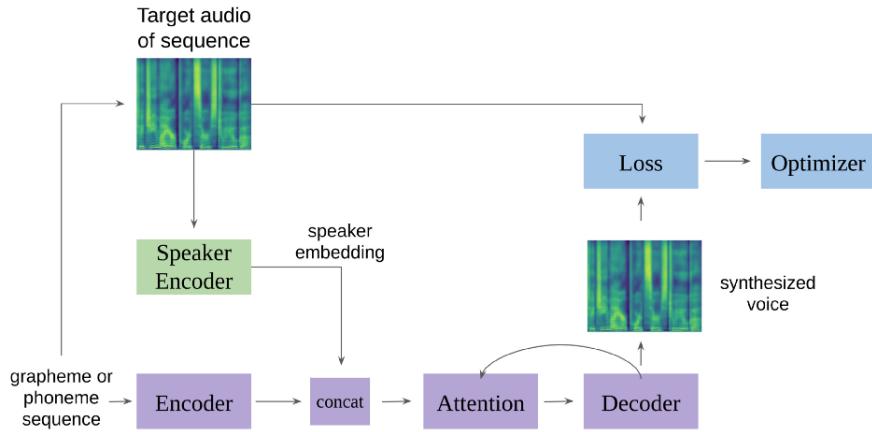
Synthesizer adalah bagian dari SV2TTS yang menganalisis input teks untuk membuat spektrogram mel, yang kemudian diubah oleh vocoder menjadi suara. Synthesizer mengambil urutan teks — dipetakan ke fonem (unit terkecil dari suara manusia, misalnya, suara yang Anda buat saat mengucapkan 'a'), bersama dengan embeddings yang dihasilkan oleh encoder speaker, dan menggunakan arsitektur Tacotron 2 untuk menghasilkan bingkai dari spektrogram mel secara berulang



Gambar 9.3 Model Arsitektur Multi-Speaker Voice Cloning Sumber: Jia, Zhang, dan Weiss et al.

Berikut Proses Training Synthesizer:

1. Pertama, kami mengumpulkan urutan fonem dan spektrogram mel dari pembicara yang mengucapkan kalimat itu.
2. Kemudian, spektrogram mel diteruskan ke encoder speaker untuk menghasilkan embedding speaker.
3. Selanjutnya, pembuat enkoder penyintesis menggabungkan penyandian urutan fonemnya dengan penyematan speaker.
4. Spektrogram mel dihasilkan secara berulang oleh decoder dan bagian perhatian dari synthesizer
5. Terakhir, spektrogram mel dibandingkan dengan target awal untuk menghasilkan kerugian, yang kemudian dioptimalkan



Gambar 9.4 Training Synthesizer Sumber: Jia, Zhang, dan Weiss et al.

9.1.3 Vocoder

Pada titik ini, synthesizer telah membuat spektrogram mel, tetapi kami masih belum dapat mendengarkan apa pun. Untuk mengubah spektrogram mel menjadi gelombang audio mentah, penulis menggunakan vocoder. Vocoder khusus yang digunakan di sini didasarkan pada model WaveNet DeepMind , yang menghasilkan bentuk gelombang audio mentah dari teks, dan pada satu titik canggih untuk sistem TTS.



Gambar 9.5 Vocoder Sumber: Jia, Zhang, dan Weiss et al.

9.1.4 Alur Model SV2TTS

berikut adalah alur modelnya:

1. Encoder speaker mendengarkan sampel audio yang diberikan dan menghasilkan embedding
2. Synthesizer mengambil daftar fonem dan penyematan speaker, kemudian menghasilkan spektrogram mel
3. Vocoder saraf menguraikan spektrogram mel menjadi bentuk gelombang audio yang dapat kita dengarkan

9.1.5 Mengukur Kualitas Hasil

Penulis menggunakan penilai manusia untuk mengukur kealamian dan kesamaan ucapan yang dihasilkan model. Kealamian mengukur seberapa "manusia" ucapan itu terdengar, sementara kesamaan mengukur seberapa mirip suara pidato yang disintesis dengan pembicara aslinya. Penilai berinteraksi melalui GUI yang terlihat seperti:

Instructions

In this task, your job is to evaluate if the two speech audio samples are from the same speaker. Please release this task if any of the following are true:

- You think you do not have good listening ability.
- There is considerable background noise (street noise, loud fan/air-conditioner, open TV/radio, people talking, etc).
- For any reason, you can't hear the audio samples.

Task

How are you listening to the speech samples?

- Headphones, with no noise in the background. I am listening to the speech sample using headphones and there is no noise around me (people talking, music playing, air-conditioners, fans, etc.).
 Headphones, with some low-level noise in the background. I am listening to the speech sample using headphones and there is some low-level noise around me (people talking, music playing, air-conditioners, fans, etc.).
 Audio speakers or other.

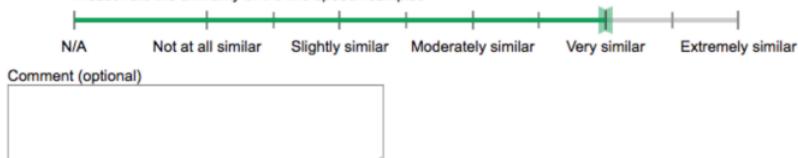
Speaker Similarity Between Two Speech Samples

Please listen to the two speech samples below (Sample A and Sample B) and rate how similar they are. Your rating should reflect your evaluation of how close the voices of the two speakers sound. You **should not judge the content, grammar, or audio quality** of the sentences; instead, just focus on the similarity of the speakers to one another.

Speech samples (please listen at least **two times each**)

Sample A	▶ 0:00 / 0:02	<input type="range"/>	🔊	<input type="range"/>
Sample B	▶ 0:00 / 0:03	<input type="range"/>	🔊	<input type="range"/>

Please rate the similarity of the two speech samples



Gambar 9.6 Pengukuran Kualitas Model Sumber: Jia, Zhang, dan Weiss et al.

Teknik penggunaan rating crowdsourced seperti ini sering disebut dengan Mean Opinion Score atau MOS. Penulis menemukan bahwa hasil akhir untuk kealamian akhirnya terlihat seperti:

System	VCTK Seen	VCTK Unseen	LibriSpeech Seen	LibriSpeech Unseen
Ground truth	4.43 ± 0.05	4.49 ± 0.05	4.49 ± 0.05	4.42 ± 0.07
Embedding table	4.12 ± 0.06	N/A	3.90 ± 0.06	N/A
Proposed model	4.07 ± 0.06	4.20 ± 0.06	3.89 ± 0.06	4.12 ± 0.05

Gambar 9.7 MOS Sumber: Jia, Zhang, dan Weiss et al.

Jika Anda ingin analisis lengkap ini, saya akan merekomendasikan membaca bagian 3.1 dari makalah asli. Namun, secara ringkas:

1. Model SV2TTS yang diusulkan mencapai sekitar 4,0 MOS di semua kumpulan data.
2. LibriSpeech (salah satu set data pelatihan) diklasifikasikan sebagai kurang alami, karena tidak ada tanda baca dalam transkrip, sehingga menyulitkan model untuk mempelajari jeda.
3. Sistem tabel penyematan menggunakan tabel pencarian penyematan speaker tetapi sebaliknya memiliki arsitektur yang sama.
4. Karena memiliki tabel pencarian, tidak dapat digeneralisasi, dan oleh karena itu tidak dapat dievaluasi pada kumpulan data yang tidak terlihat.
5. Kealamian dari contoh yang tidak terlihat dan yang terlihat sangat mirip, yang sangat mengesankan.

Dan untuk kesamaan ucapan, hasilnya terlihat seperti:

System	Speaker Set	VCTK	LibriSpeech
Ground truth	Same speaker	4.67 ± 0.04	4.33 ± 0.08
Ground truth	Same gender	2.25 ± 0.07	1.83 ± 0.07
Ground truth	Different gender	1.15 ± 0.04	1.04 ± 0.03
Embedding table	Seen	4.17 ± 0.06	3.70 ± 0.08
Proposed model	Seen	4.22 ± 0.06	3.28 ± 0.08
Proposed model	Unseen	3.28 ± 0.07	3.03 ± 0.09

Gambar 9.8 Speech Similarity Sumber: Jia, Zhang, dan Weiss et al.

Sekali lagi, Anda mungkin ingin membaca bagian 3.2 dari makalah asli untuk analisis lengkapnya. Tapi secara ringkas:

1. Skor lebih tinggi untuk VCTK (set data pelatihan lainnya), menunjukkan bentuk dataset yang lebih terstruktur.
2. Skor model SV2TTS antara "cukup mirip" dan "sangat mirip" pada skala evaluasi untuk pembicara yang tidak terlihat.
3. Meskipun ini sulit diukur secara matematis, model secara keseluruhan menangkap karakteristik pembicara secara akurat.

BAB 10

TUTORIAL PEMBUATAN MULTI-SPEAKER VOICE CLONING

10.1 Pengenalan Anaconda

Anaconda merupakan sebuah software yang mendistribusikan bahasa pemrograman python dan R untuk keperluan komputasi ilmiah seperti data science, machine learning, data processing skala-luas, analisis prediksi, dan lain sebagainya. Anaconda memiliki anaconda navigator yang didalamnya terdapat software-software yang dapat digunakan untuk membuat program python dan R. Salah satu software yang akan kita gunakan yaitu Spyder. Sebelum kita membuat program kita harus menginstall anaconda terlebih dahulu. Instalasi pada kali ini menggunakan 2 sistem operasi yaitu Windows 10 x64 dan Ubuntu 19.04. Hal yang dibutuhkan adalah laptop dengan os windows atau ubuntu dan internet.

10.1.1 Instalasi Anaconda 3 Windows 10 x64

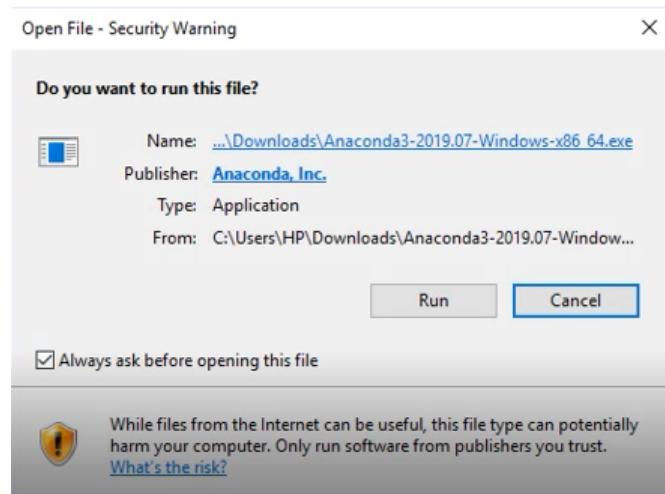
Hal yang harus diperhatikan sebelum melakukan instalasi *Anaconda Python*

1. *Download Anaconda Python* <https://www.anaconda.com/distribution/>
2. Perhatikan versi dari sistem operasi yang digunakan (versi 32bit atau 64bit)

3. Download file anaconda yang sesuai dengan versi sistem operasi (32bit atau 64bit)

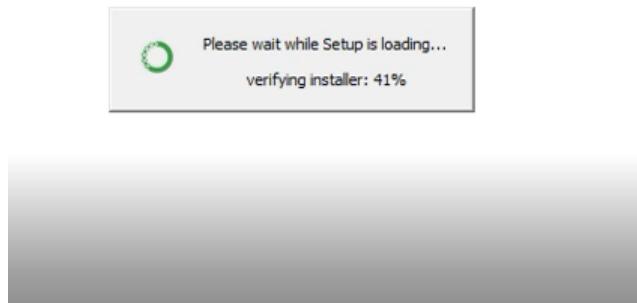
Berikut langkah-langkah instalasi anaconda.

1. Buka aplikasi *installer Anaconda* yang telah didownload lalu akan muncul gambar 10.1, lalu pilih run untuk menjalankan proses instalasi.



Gambar 10.1 Run Setup Anaconda

2. Tunggu hingga *setup loading* selesai seperti pada gambar 10.2.



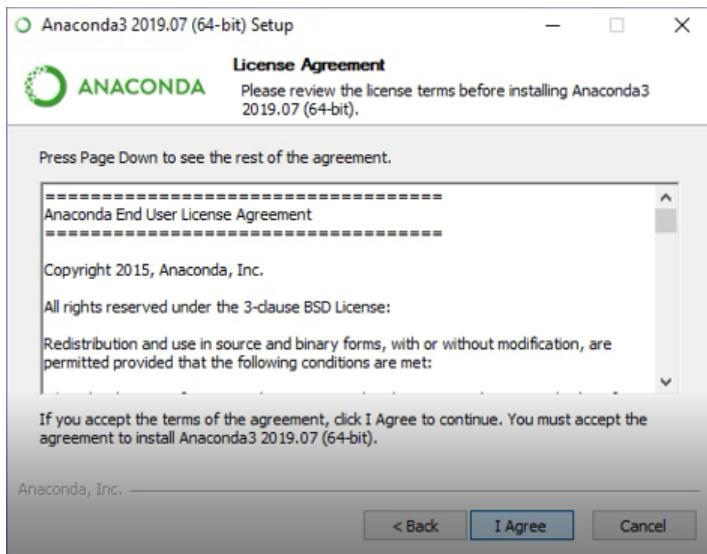
Gambar 10.2 Setup Loading

3. Jika *setup loading* telah selesai, maka klik *next* seperti pada gambar 10.3.



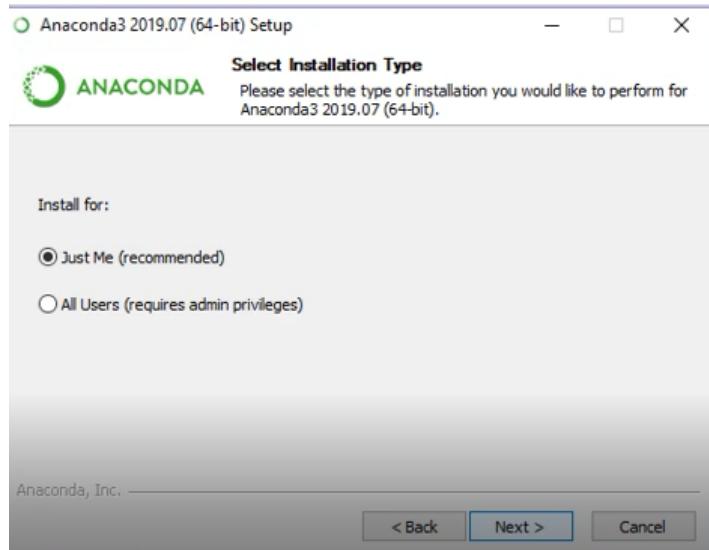
Gambar 10.3 Welcome to Anaconda Setup

4. Pada *License Agreement* klik *I Agree* karena jika teman-teman tidak menyetujui lisensi anaconda maka teman-teman tidak akan bisa melanjutkan proses instalasi. lakukan langkah ini seperti pada gambar 10.4



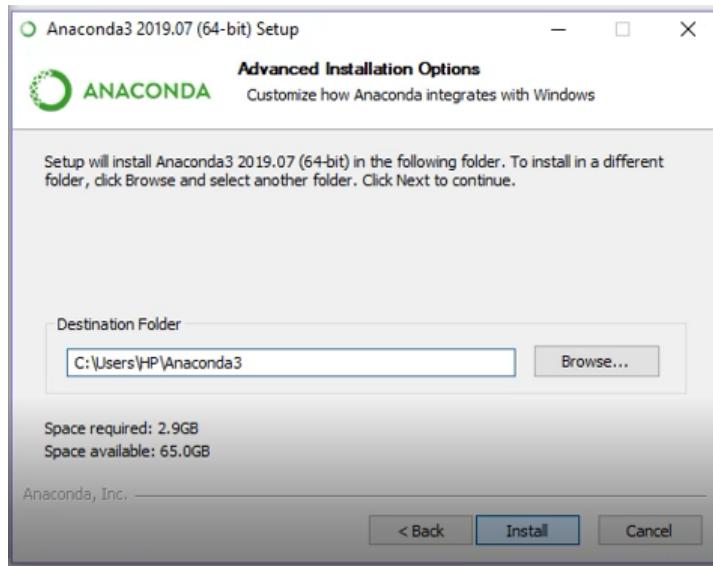
Gambar 10.4 License Agreement

5. Kemudian pilih *Just Me(Recomended)* agar sesuai dengan komputer yang digunakan, kemudian klik *next* seperti pada gambar 10.5.



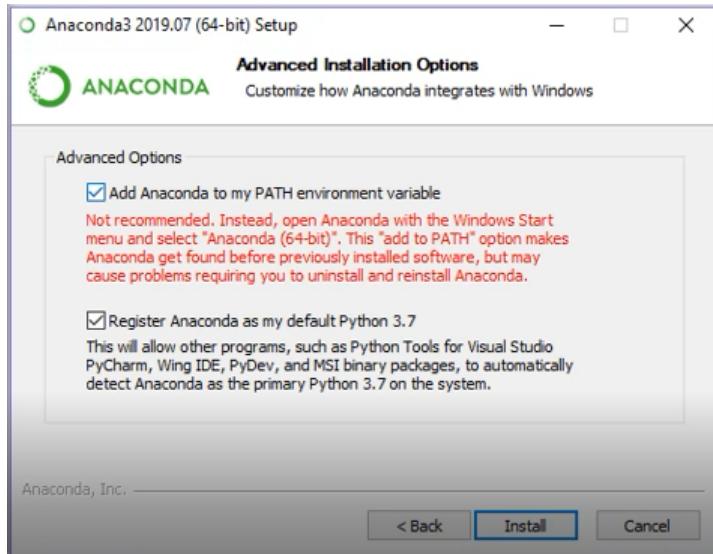
Gambar 10.5 *Just Me(recomended)*

6. Kemudian pilih direktori tempat kita akan *menginstall anaconda* seperti pada gambar 10.6



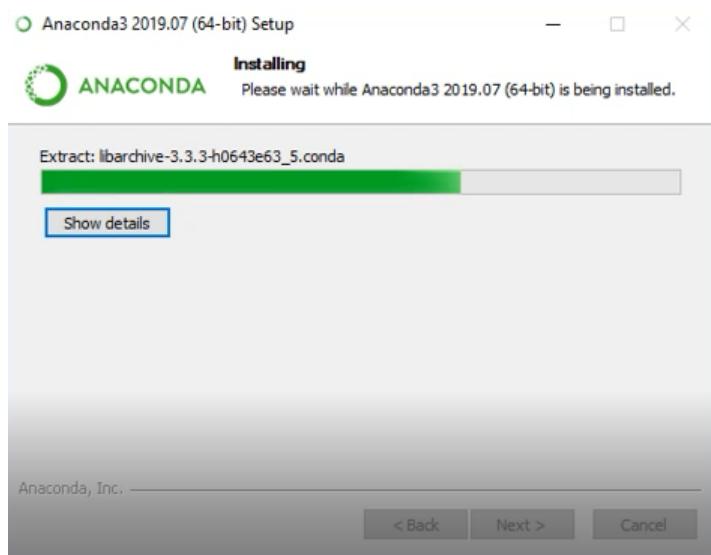
Gambar 10.6 Pilih lokasi

- Kemudian centang *Add Anaconda to my Path environment variable*, agar saat melakukan instalasi *package anaconda*, *package* tersebut akan langsung tertuju ke *path anaconda* tidak ke aplikasi yang lain. kemudian Klik *install* seperti pada gambar 10.7.



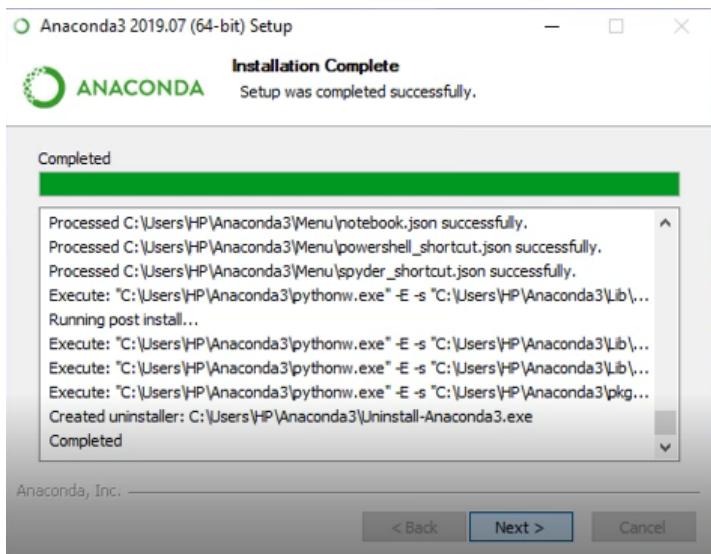
Gambar 10.7 Centang Anaconda to my PATH

8. Tunggu sampai proses *installasi* selesai seperti pada gambar 10.8.



Gambar 10.8 Waiting Installation Complete

9. Apabila instalasi telah selesai maka akan terlihat seperti gambar 10.9, kemudian klik *next*



Gambar 10.9 Installation Complete

10. apabila muncul gambar 10.10, maka klik *next*



Gambar 10.10 *Anaconda+JetBrains*

11. Jika instalasi telah selesai maka akan ada ucapan terima kasih telah menginstall anaconda 3 seperti pada gambar 10.11, hal ini menandakan bahwa teman-teman telah selesai dan berhasil melakukan instalasi anaconda. Kemudian klik *finish* untuk mengakhiri instalasi.



Gambar 10.11 *Thanks for install Anaconda*

10.1.2 Update Anaconda dan Spyder

Kenapa kita harus melakukan update anaconda dan spyder? melakukan update diperlukan agar software yang kita gunakan merupakan software yang terbaru, karena versi lama dan versi baru akan memiliki banyak perbedaan dan akan menjadi masalah nantinya ketika kita membuat program atau mengimportkan modul-modul python yang digunakan. Berikut cara mengupdate Spyder:

1. Buka anaconda prompt, lalu ketikkan perintah conda update spyder
2. Konfirmasi update dengan mengetikkan y, lalu tekan enter
3. Tunggu hingga installan selesai

Berikut cara mengupdate Anaconda:

1. Buka anaconda prompt, lalu ketikkan perintah conda update anaconda
2. Konfirmasi update anaconda dengan mengetikkan y dan kemudian tekan enter
3. Tunggu hingga installan selesai

10.1.3 Instalasi Anaconda Ubuntu 19.04

Untuk instalasi python pada Ubuntu 19.04 dibutuhkan sebagai berikut:

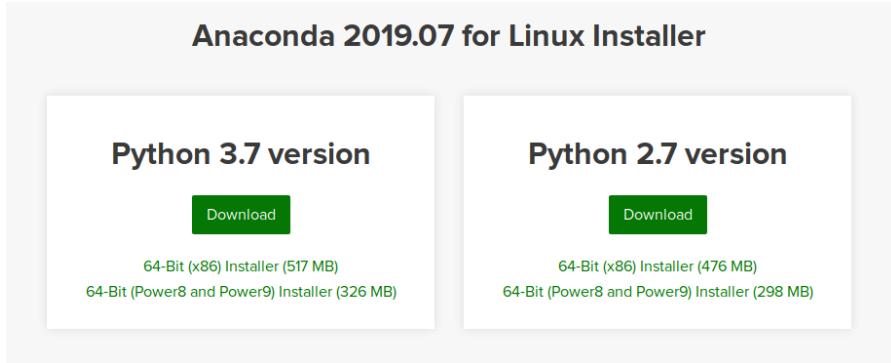
1. Internet

2. Anaconda installer (64bit or 32bit)

3. enter, dan yes atau no

Ikuti langkah berikut:

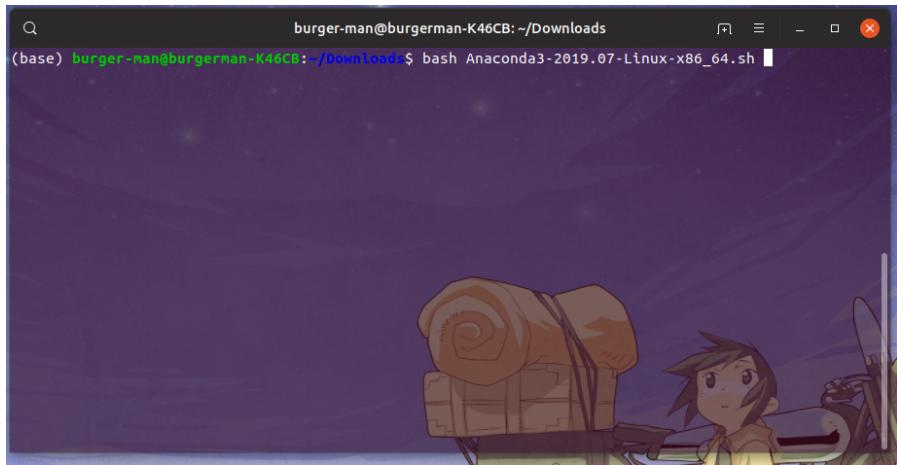
1. Pertama kita kunjungi situs <https://www.anaconda.com/distribution/#download-section> seperti gambar 10.12 dan pilih **64-Bit (x86) Installer (517 MB)**



Gambar 10.12 Gambar halaman download

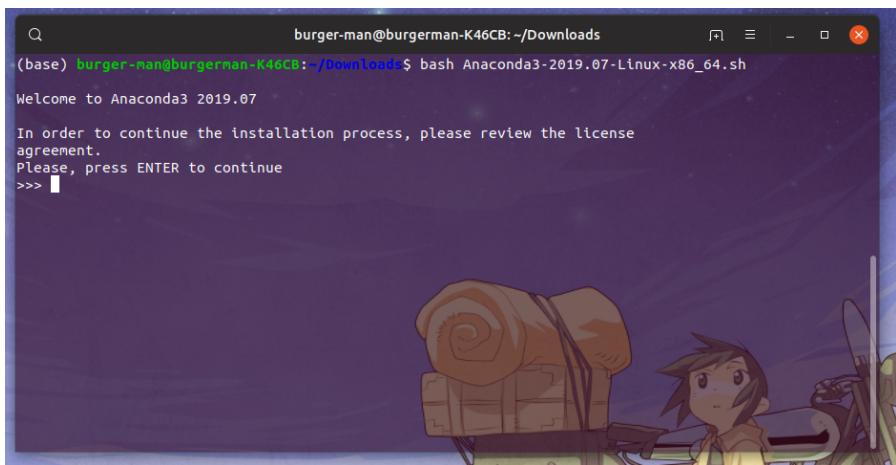
2. Kedua kita buka **terminal** kita lalu arahkan ke direktori kita menyimpan file download anaconda

3. Ketiga kita ketikkan sebagai berikut **bash namafileanaconda.sh** lalu enter, contoh seberti gambar 10.13



Gambar 10.13 Gambar install anaconda

4. Setelah itu, tekan **ENTER** saja seperti gambar 10.14



Gambar 10.14 Gambar eksekusi anaconda

5. Lalu akan muncul sebuah tulisan **End User License Agreement** seperti gambar 10.15, tekan **ENTER** dan tahan hingga seperti gambar



Gambar 10.15 Gambar anaconda license agreement

Gambar 10.16 Gambar perintah yes or no

- Lalu setelah muncul perintah '**yes**' or '**no**' ketik **yes** lalu enter
 - Setelah itu muncul path direktori instalasi anaconda kita seperti gambar 10.17 lalu tekan enter

```
Q burger-man@burgerman-K46CB: ~/Downloads
Please answer 'yes' or 'no':'
>>>
Please answer 'yes' or 'no':'
>>> yes

Anaconda3 will now be installed into this location:
/home/burger-man/anaconda3

- Press ENTER to confirm the location
- Press CTRL-C to abort the installation
- Or specify a different location below

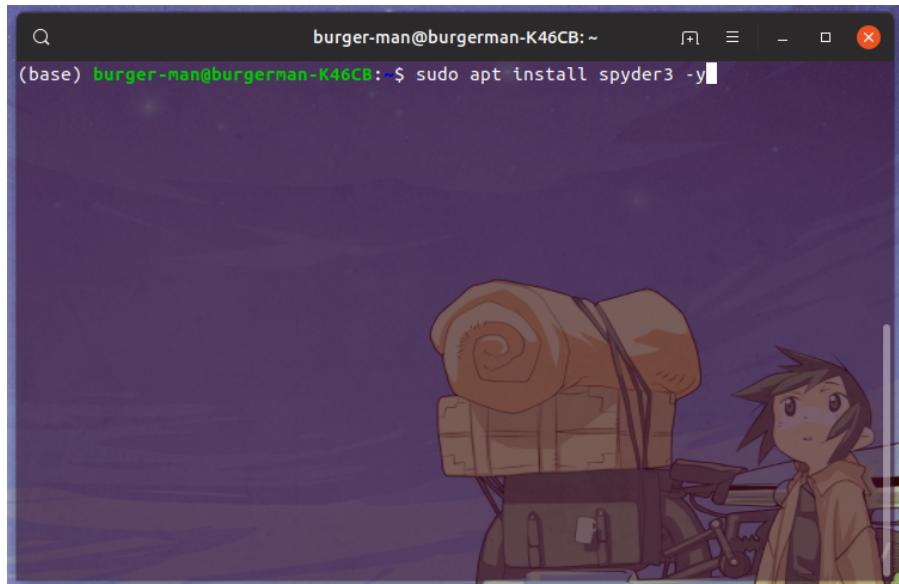
[~/home/burger-man/anaconda3] >>> [REDACTED]
```



Gambar 10.17 Gambar path anaconda

Setelah kita selesai instalasi anaconda jangan lupa juga untuk menginstal spyder ide, caranya seperti berikut:

- (a) ketikkan perintah `sudo apt install spyder3 -y` seperti gambar 10.18



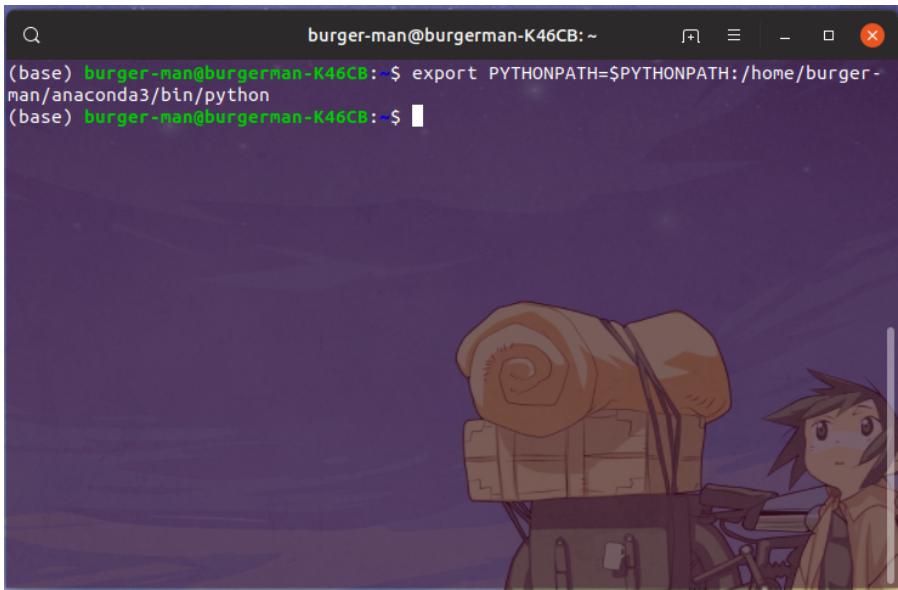
Gambar 10.18 Gambar perintah install spyder3

(b) lalu jalankan dengan perintah **spyder** atau **spyder3**

10.1.4 Konfigurasi Python

Setelah kita selesai instal Anaconda dan Spyder, selanjutnya kita akan mempelajari bagaimana cara setting environments python kita? caranya sebagai berikut

1. pertama kita buka terminal kita lalu ketikkan perintah **export PYTHONPATH=\$PYTHONPATH:\$PWD** contoh seperti gambar 10.19, lalu enter

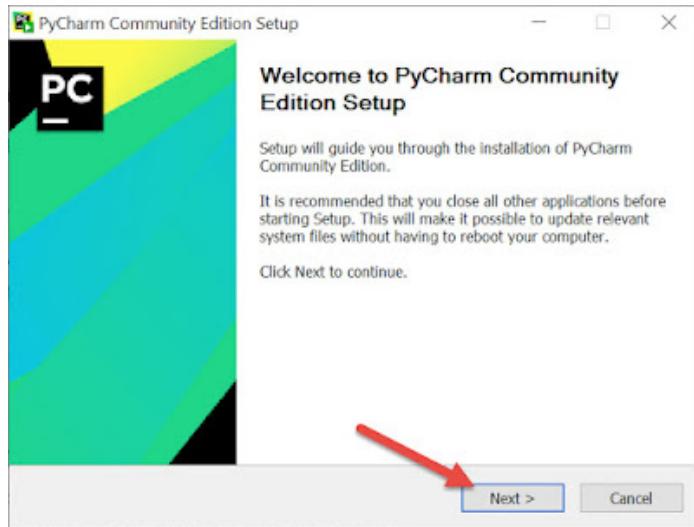


Gambar 10.19 Gambar setpath

10.2 Instalasi Pycharm

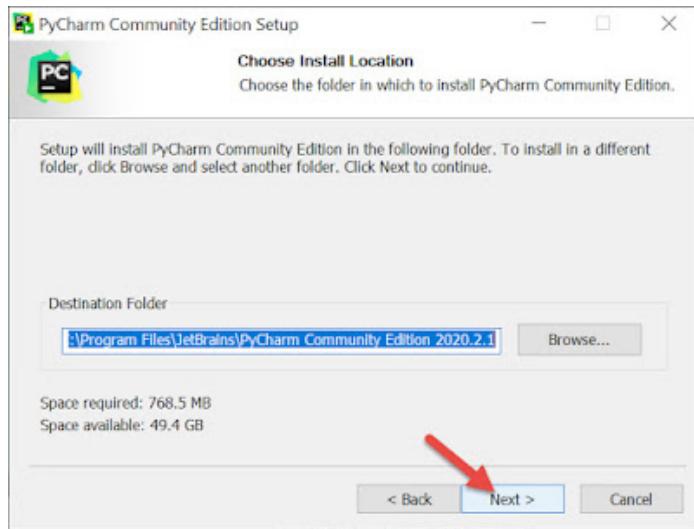
Ikuti langkah-langkah berikut untuk install pycharm di sistem operasi windows:

1. Download Installer Pycharm dari situs resminya yaitu <https://www.jetbrains.com/pycharm/>
2. Jalankan Installer PyCharm yang telah selesai di download, kemudian klik Next untuk melanjutkan instalasi seperti pada contoh gambar 10.20.



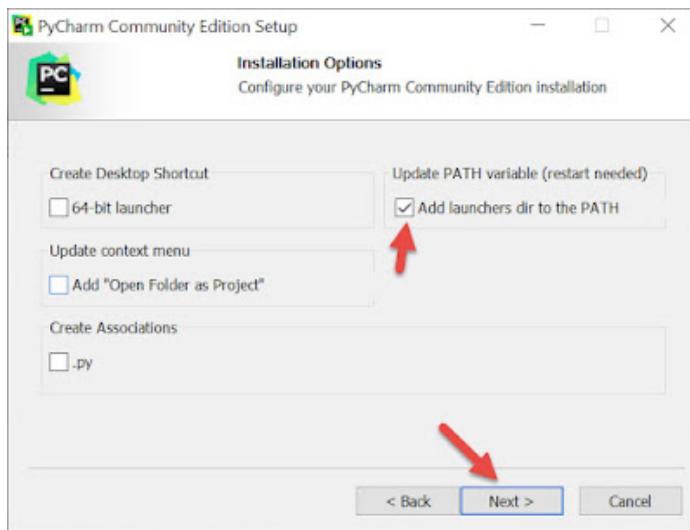
Gambar 10.20 Klik Next untuk Install Pycharm

3. Tentukan destination folder jika diperlukan, atau dibiarkan default kemudian klik Next.



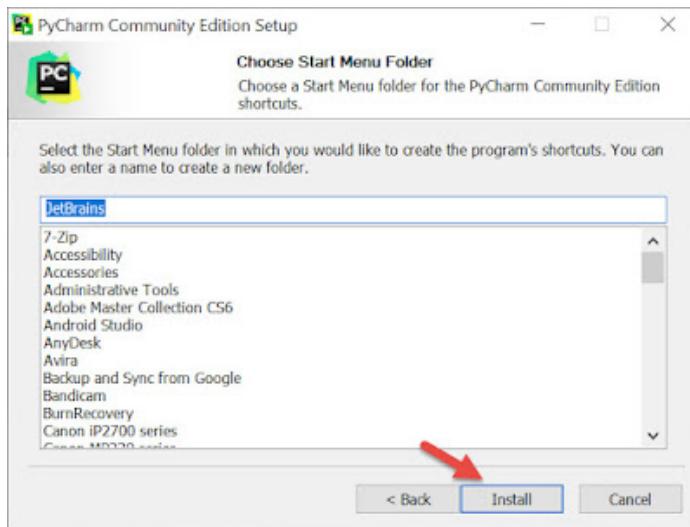
Gambar 10.21 Menentukan lokasi folder dan klik next

4. Checklist pada "Add launchers dir to PATH" kemudian klik Next.



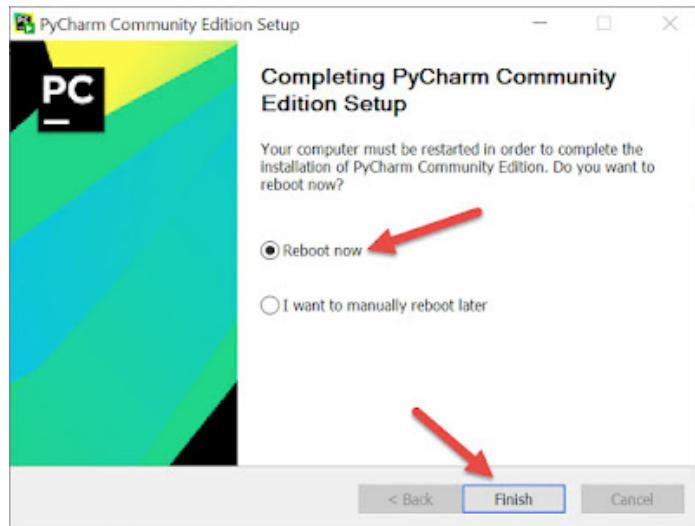
Gambar 10.22 Add launchers dir to PATH

5. Klik Install untuk melanjutkan



Gambar 10.23 Klik Install

6. Tunggu proses installasi PyCharm sampai selesai.



Gambar 10.24 Reboot Now

- Setelah proses installasi PyCharm selesai, klik Reboot now, lalu klik Finish.

10.3 Membuat Akun Github

Github merupakan software developer untuk menyimpan dan sharing project-project yang bersifat opensource, karena bersifat opensource maka project tersebut dapat dikembangkan oleh programmer lain yang ingin berkontribusi pada project, sangat bagus untuk project yang dikembangkan oleh team maupun individu.

Github memiliki banyak keunggulan diantaranya sebagai tempat menyimpan backup project, adanya free hosting, memudahkan developer, mendukung semua bahasa pemrograman dan github juga berguna sebagai tempat penyimpanan project secara gratis.

Berikut tata cara membuat akun github:

- buka browser kemudian kunjungi situs github.
- klik tombol sign up



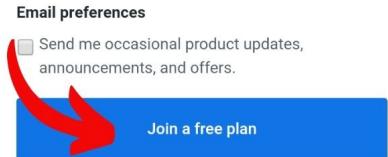
Gambar 10.25 Sign Up | sumber: <https://omcyber.com/cara-membuat-akun-github/>

3. isi formulir data diri seperti username, email, password, dan verify your account. Pastikan data yang diisikan benar.

The image shows the GitHub sign-up form. It consists of several input fields and descriptive text. The first field is labeled 'Username *' with a red asterisk indicating it is required. The second field is labeled 'Email address *' with a red asterisk. The third field is labeled 'Password *' with a red asterisk. Below these fields is a note: 'Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more](#)'. The fourth section is titled 'Email preferences' and contains a checkbox followed by the text: 'Send me occasional product updates, announcements, and offers.' The fifth section is titled 'Verify your account' and contains a text input field with a yellow 'Omcyber.com' watermark. The entire form is set against a light gray background.

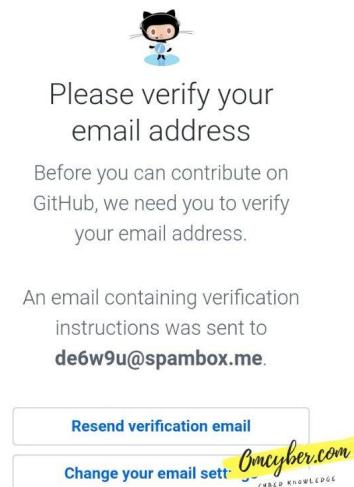
Gambar 10.26 Form Daftar

4. klik tombol create an account untuk membuat akun baru github.
5. silahkan pilih free plan dengan klik tombol join a free plan



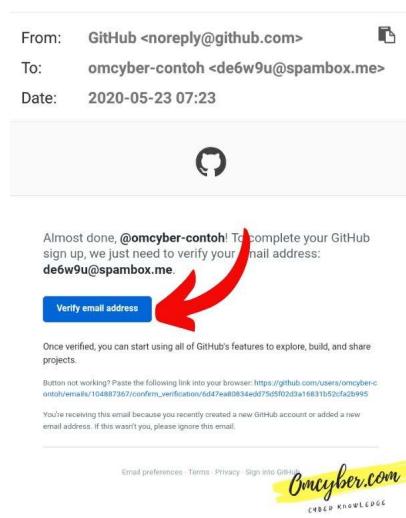
Gambar 10.27 Free Plan

6. verifikasi email agar akun github terverifikasi



Gambar 10.28 Verifikasi Email

7. buka inbox email yang didaftarkan, buka email dari github dan klik verifikasi email address.



Gambar 10.29 Verifikasi Email Address

- setelah berhasil verifikasi email maka akun github telah berhasil dibuat dan ter-verifikasi.



What do you want to do first?

Every developer needs to configure their environment, so let's get your GitHub experience optimized for you.

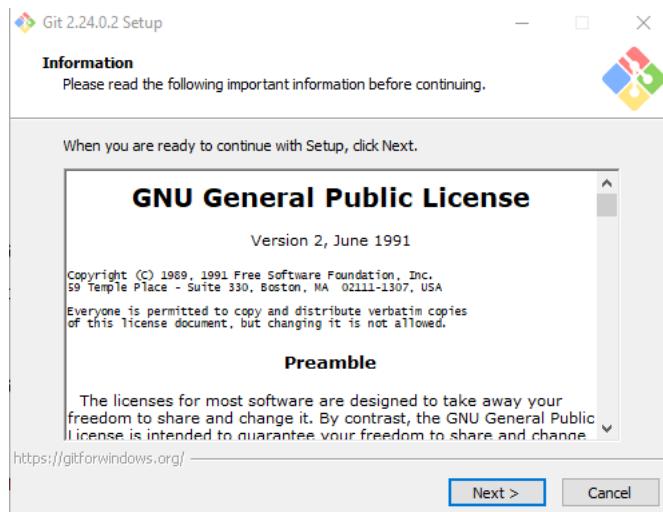


Gambar 10.30 Verifikasi Email Berhasil

10.4 Download dan Install Git

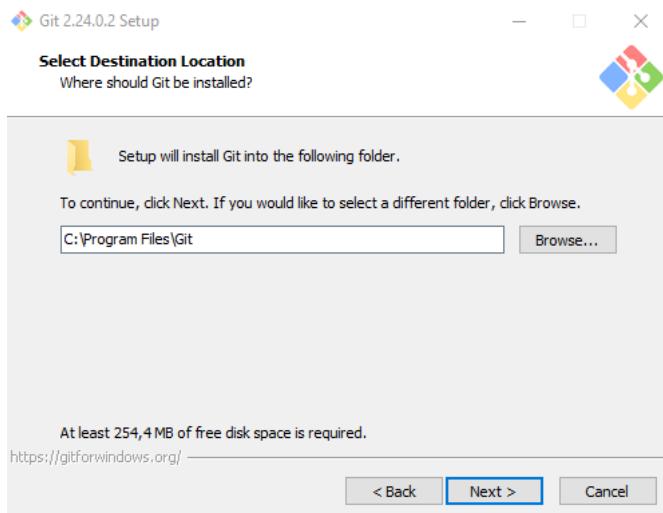
Sebelum melakukan instalasi git, kita memerlukan file instalasi git terlebih dahulu. File ini bisa teman-teman dapatkan dengan mengunduh file dari situs resmi git, berikut link yang dapat teman-teman kunjungi <https://git-scm.com/downloads>. Download file yang sesuai dengan sistem operasi pada komputer teman-teman. Jika komputer teman-teman memiliki sistem operasi Windows 64bit maka teman-teman harus mengunduh file instalasi git untuk windows 64bit. Ikuti tahapan berikut untuk melakukan instalasi git:

1. Buka file instalasi git yang telah di download, kemudian klik next.



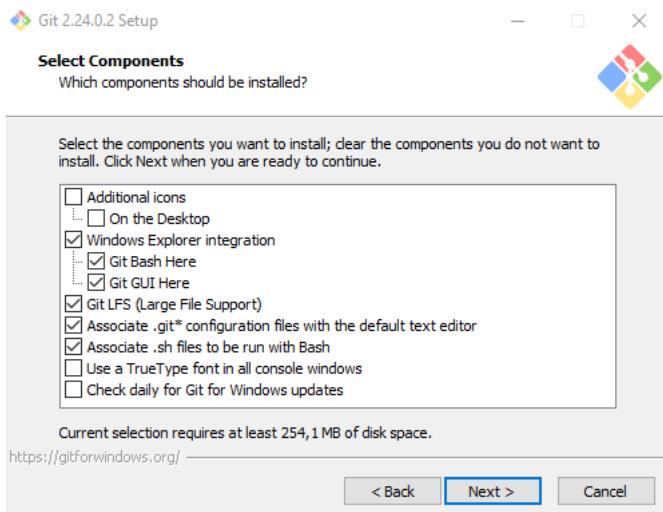
Gambar 10.31 Install Git

2. Pilih lokasi instalasi git, disarankan install pada lokasi seperti gambar



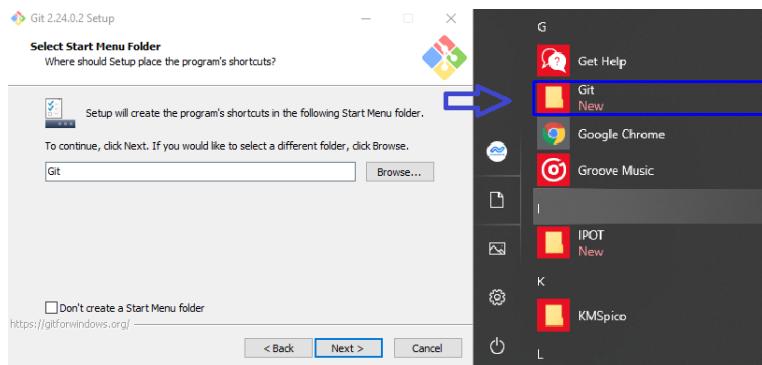
Gambar 10.32 Lokasi Instalasi Git

3. Selanjutnya pilih komponen tambahan untuk install git, sesuaikan dengan kebutuhan teman-teman. Jika sudah maka klik next untuk melanjutkan instalasi.



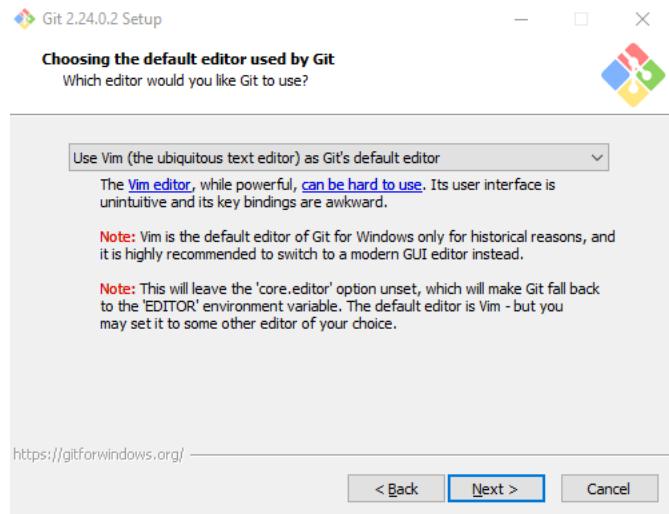
Gambar 10.33 Komponen Tambahan

4. Tentukan nama aplikasi git, untuk memudahkan pencarian aplikasi maka disarankan menggunakan nama Git saja.



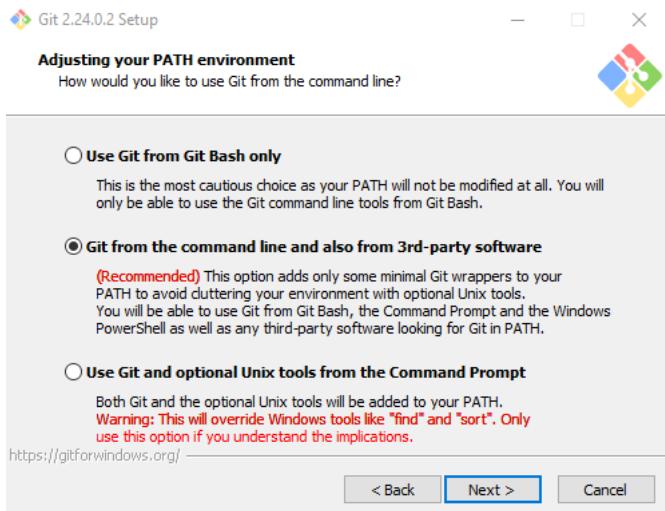
Gambar 10.34 Select Start Menu Folder

5. pilih file editor untuk dikombinasikan dengan git, pada tutorial ini saya menggunakan vim editor lalu pilih next.



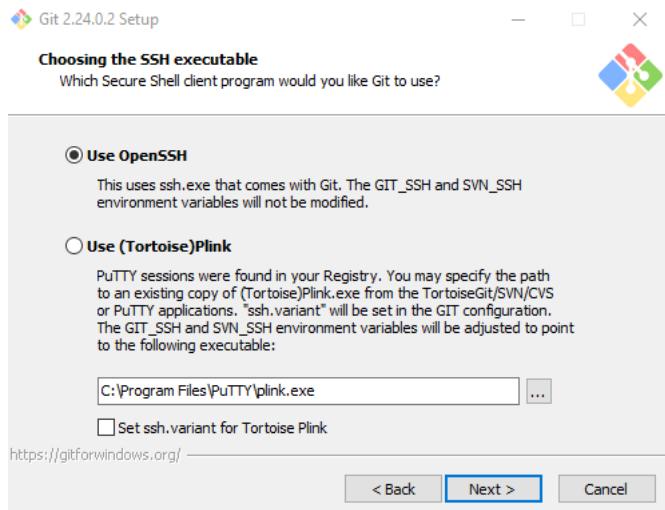
Gambar 10.35 Choose Default Editor

6. selanjutnya kita perlu mengatur path environment, Pilih Git from the command line and also from 3rd-party software agar saat menjalankan perintah Git dapat dikenali di Command Prompt (CMD) pada Windows.



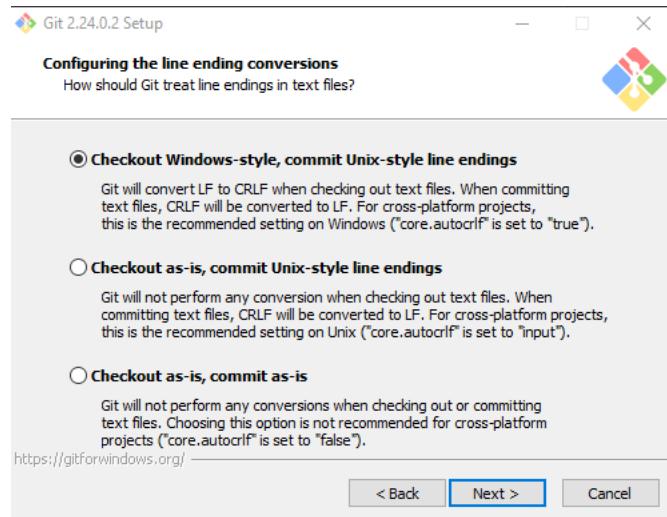
Gambar 10.36 PATH Environment

7. kemudian pilih aplikasi SSH, pada tutorial ini saya memilih Use OpenSSH yang merupakan aplikasi default SSH dari Git lalu klik next.



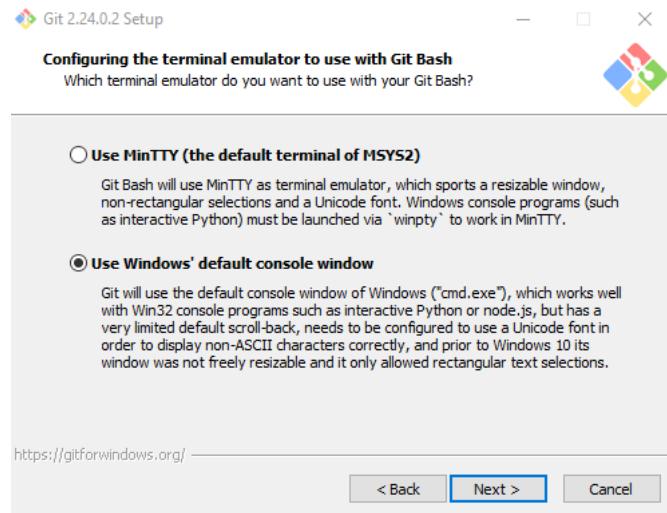
Gambar 10.37 Choose SSH

8. selanjutnya pilih Checkout Windows-style, commit Unix-style line endings dan klik next



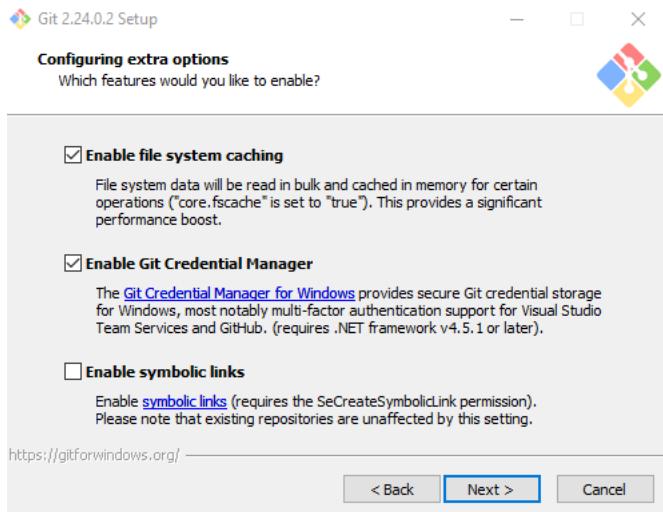
Gambar 10.38 Configuring the line ending

9. pilih Use Windows' default console windows kemudian klik next



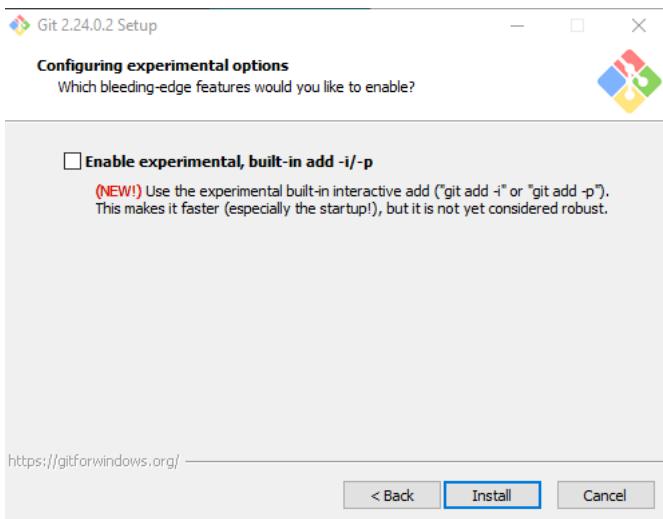
Gambar 10.39 Configuring the terminal emulator

10. pilih opsi ekstra yaitu Enable File System Caching agar Git memiliki fungsi system caching dan Enable Git Credential Manager agar Git bisa dikombinasikan dengan aplikasi lain seperti Visual Studio, Android Studio, dan GitHub. Selanjutnya klik next.



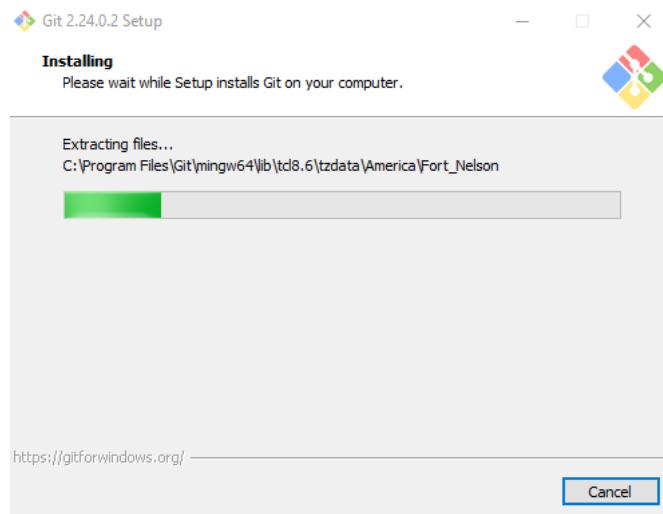
Gambar 10.40 Configuring extra options

11. setelah menambahkan konfigurasi ekstra maka teman-teman bisa melakukan instalasi git dengan klik install.



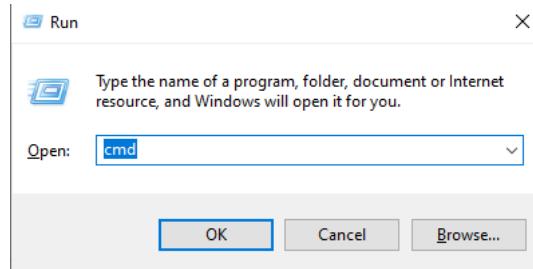
Gambar 10.41 Install

12. Tunggu hingga proses instalasi selesai.



Gambar 10.42 Proses Install

13. Jika instalasi telah selesai maka teman-teman bisa cek versi git untuk memastikan apakah git telah berhasil terinstal di komputer teman-teman dengan membuka command prompt atau cmd.



Gambar 10.43 Command Prompt (CMD)

14. ketikkan perintah git –version lalu tekan enter untuk melihat versi git yang telah teman-teman install.

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.18362.418]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\dafit>git --version
git version 2.24.0.windows.2

C:\Users\dafit>
```

Gambar 10.44 Versi Git

10.5 Konfigurasi Git

Berikut hal yang perlu teman-teman lakukan apabila telah install git di komputer, teman-teman perlu melakukan konfigurasi email dan username akun git teman-teman. Ikuti tahapan berikut untuk melakukan konfigurasi git:

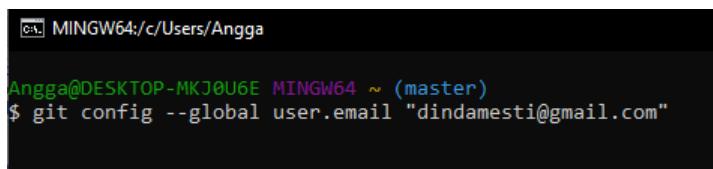
1. Konfigurasikan email dan username git yang telah teman-teman daftarkan pada github dengan mengetikkan perintah

```
git config --global user.name "username anda"  
git config --global user.email "email_anda@gmail.com"
```



The screenshot shows a terminal window titled 'MINGW64:/c/Users/Angga'. The command \$ git config --global user.name "dindamajesty13" is entered and executed successfully.

Gambar 10.45 Konfigurasi Username Git



The screenshot shows a terminal window titled 'MINGW64:/c/Users/Angga'. The command \$ git config --global user.email "dindamesti@gmail.com" is entered and executed successfully.

Gambar 10.46 Konfigurasi Email Git

2. membuat ssh key dengan mengetikkan perintah

```
ssh-keygen -t rsa -b 4096 -C "email_anda@gmail.com"
```

lalu tekan enter sebanyak tiga kali, pertama untuk membuat direktori penyimpanan ssh, kedua dan ketiga untuk menambahkan passphrase.

```
saras@DESKTOP-DCDKHUR MINGW64 ~
$ ssh-keygen -t rsa -b 4096 -C "sarascayya@gmail.com" 1
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/saras/.ssh/id_rsa): 2
Created directory '/c/Users/saras/.ssh'.
Enter passphrase (empty for no passphrase): 3
Enter same passphrase again: 4
Your identification has been saved in /c/Users/saras/.ssh/id_rsa.
Your public key has been saved in /c/Users/saras/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:C68xLrRsWaVX681mt0bVE7tXUzjN0g8Ad9MtM1QCVcc sarascayya@gmail.com
The key's randomart image is:
+---[RSA 4096]---+
|       .++oooo |
|       . +.+E |
|       o + * |
|       . . . o + |
|       .oS. . . + |
|       . oo... . +o+ |
|       o +o.o. + ..%+ |
|       * . + . B. = |
|       . .o     =. . . |
+---[SHA256]---+
```

Gambar 10.47 Generate SSH key

3. jika ssh key telah berhasil digenerate, ketikkan perintah

```
cd .ssh/
ls
```

perintah ini digunakan untuk berpindah direktori ke direktori ssh dan menampilkan list file yang ada didalam direktori tersebut.

```
[c:\] MINGW64:/c/Users/Angga/.ssh
Angga@DESKTOP-MKJ0U6E MINGW64 ~ (master)
$ cd .ssh/
Angga@DESKTOP-MKJ0U6E MINGW64 ~/ssh (master)
$ ls
config  id_rsa  id_rsa.pub  id_rsa_angga  id_rsa_angga.pub  known_hosts  known_hosts.old
```

Gambar 10.48 Direktori SSH

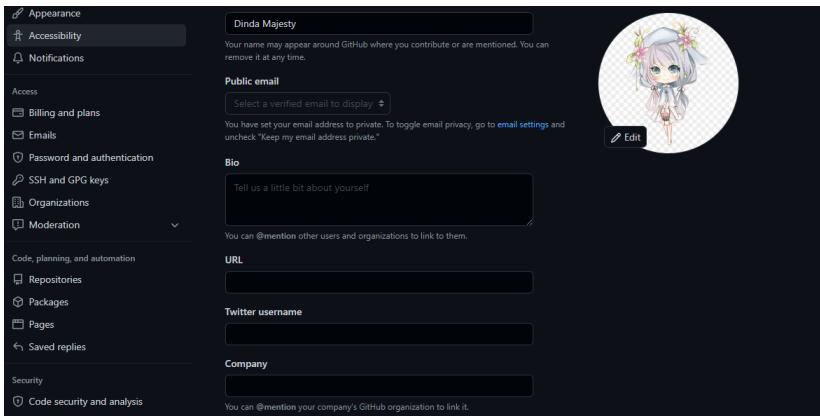
4. ketikkan perintah berikut untuk menampilkan ssh yang telah digenerate, kemudian copy ssh key yang tampil dilayar teman-teman.

```
cat id_rsa.pub
```

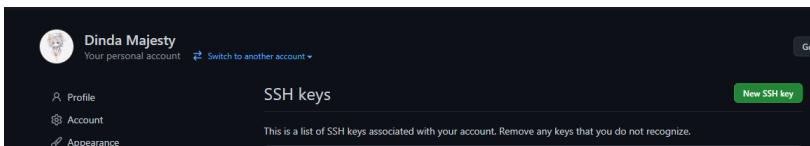
```
Angga@DESKTOP-MKJ0U6E MINGW64 ~/ .ssh (master)
$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQgQCndl/CJd50MCWkWxN4Tz0K2oBY+kNwG9bchXe91wrPQnb6tFZMLrsyRS
f16hpOKzq239ldQd8R2RsWtp1SoNKz+91r+rtDjXl+5ACKz99vZ2T5rjnR4t+j44aq09Q3kQDq06tTrr+GsOpbhly6amkrHb
u0Qkq8pasIG2ow2d3onX14za/3cxI90mXX7upd3iF785CVV1bJX1THXRgT352UdN651F0q0Lc1CD80MEEM4w4KKw7rS8ax1Ue
KtcvCVQeYSDSyDgqywCr31bvCMJa55WC10bk+n=4JZSMd49ngZ00CeSoXfffRLdpFYK4x1+i1x8ASn4Eaq5VMaNo3W21TKzs
DuqAztLe6WPNBwrf5CxYggxz3bj+yhakLF/7H9IxVdgu1XPX2P7t5FOw1e6ziJgJMB5EGk= dindamesti@gmail.com
```

Gambar 10.49 SSH Key

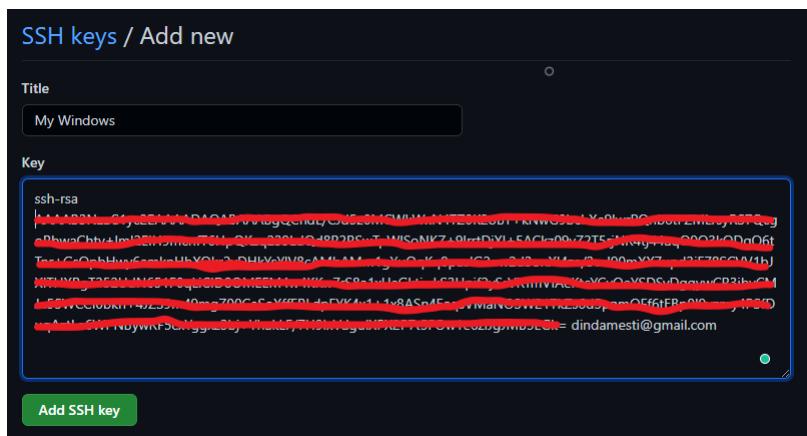
5. pilih menu SSH dan GPG keys

**Gambar 10.50** Menu SSH dan GPG keys

6. klik tombol New SSH key

**Gambar 10.51** New SSH key

7. isikan title, pada bagian key paste ssh key yang telah di copy sebelumnya dan klik add SSH key



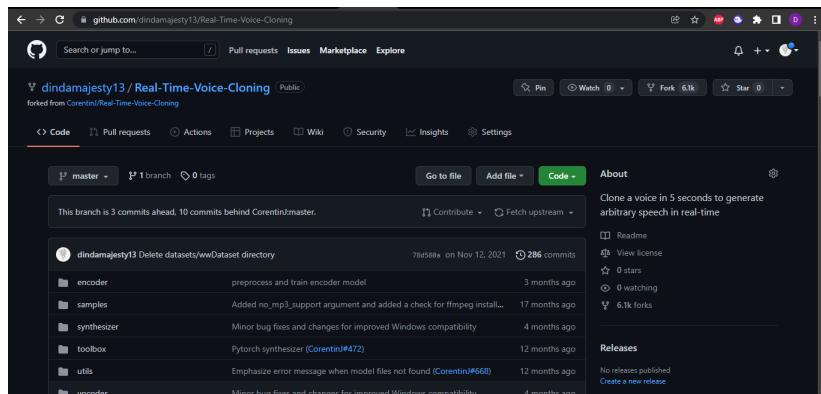
Gambar 10.52 Add SSH key

- sekarang teman-teman telah berhasil menambahkan ssh key pada akun github teman-teman. Selanjutnya teman-teman dapat mengklon Real-Time-Voice-Cloning project.

10.6 Fork dan Clone Voice Cloning Repository

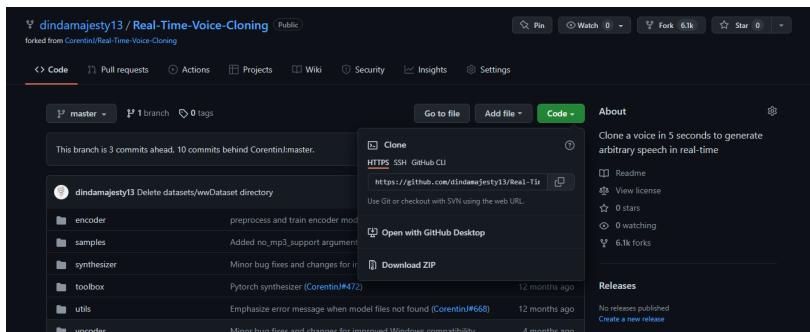
berikut tutorial clone repo voice cloning menggunakan github:

- Kunjungi link berikut <https://github.com/dindamajesty13/Real-Time-Voice-Cloning>
- Klik fork, maka repo Real-Time-Voice-Cloning akan ada di github anda.



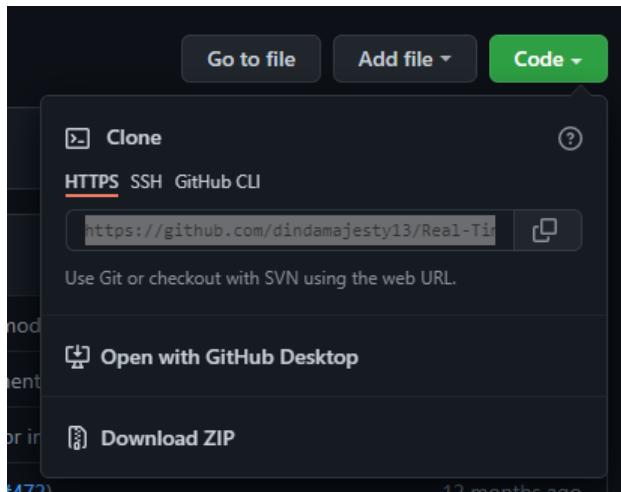
Gambar 10.53 Fork Repository

3. buka project Real-Time-Voice-Cloning yang ada direpo anda lalu klik pada bagian code.



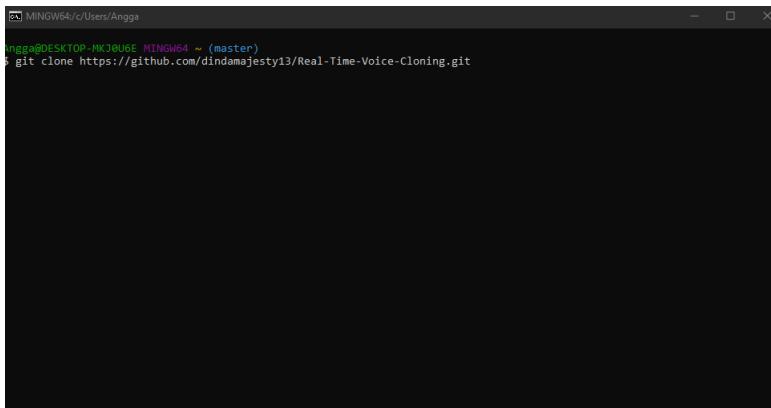
Gambar 10.54 Klik Tombol Code

4. pilih HTTPS dan salin url yang ada dibawah tulisan HTTPS



Gambar 10.55 Copy Url

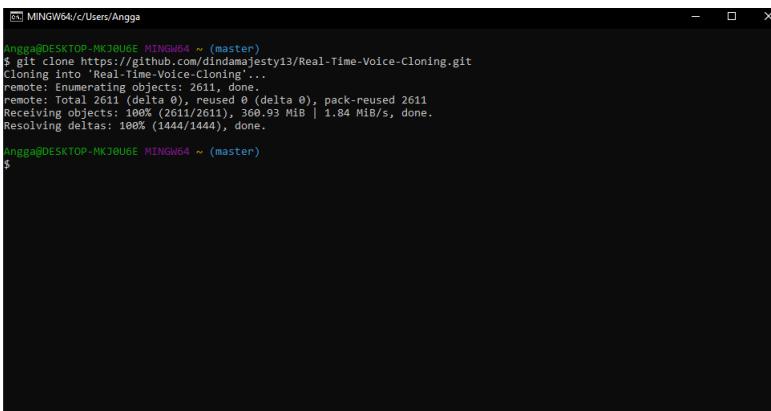
5. buka gitbash kemudian ketikkan git clone lalu paste url yang telah di copy sebelumnya.



```
MINGW64 / c/Users/Angga  
angga@DESKTOP-MKJ0U6E MINGW64 ~ (master)  
$ git clone https://github.com/dindamajesty13/Real-Time-Voice-Cloning.git
```

Gambar 10.56 Git Clone Repository

6. Lalu tekan enter pada keyboard, tunggu hingga proses clone selesai.

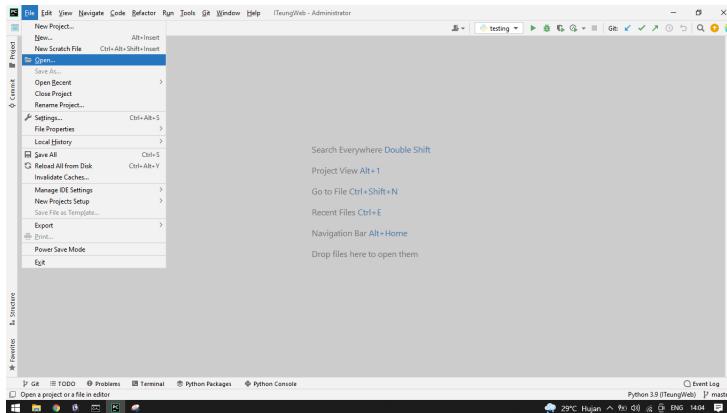


```
MINGW64 / c/Users/Angga  
angga@DESKTOP-MKJ0U6E MINGW64 ~ (master)  
$ git clone https://github.com/dindamajesty13/Real-Time-Voice-Cloning.git  
Cloning into 'Real-Time-Voice-Cloning'...  
remote: Enumerating objects: 2611, done.  
remote: Total 2611 (delta 0), reused 0 (delta 0), pack-reused 2611  
Receiving objects: 100% (2611/2611), 360.93 MiB | 1.84 MiB/s, done.  
Resolving deltas: 100% (1444/1444), done.  
$
```

Gambar 10.57 Proses Clone Repository

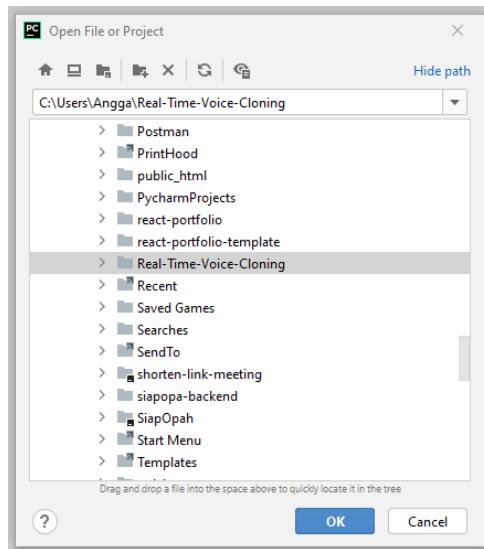
7. jika proses clone telah selesai, buka pycharm.

8. klik menu file, pilih open.



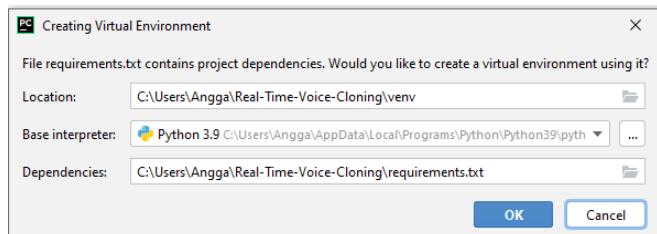
Gambar 10.58 Pycharm

9. cari folder Real-Time-Voice-Cloning, kemudian klik ok



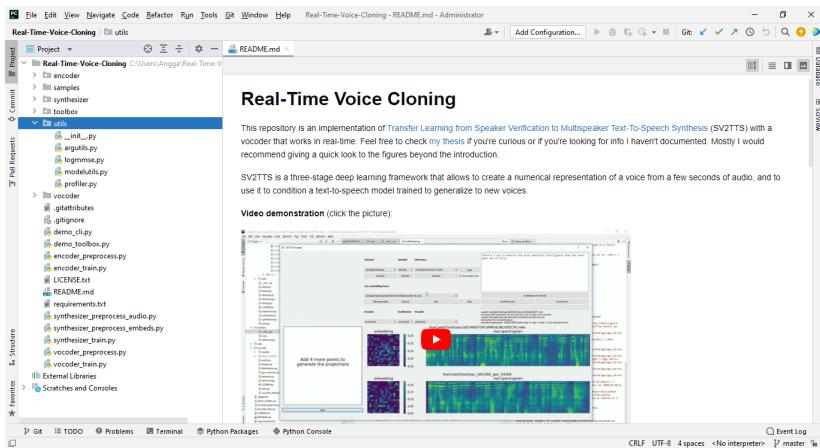
Gambar 10.59 Lokasi Folder Repotori

10. pycharm akan membuka file project Real-Time-Voice-Cloning, jika terdapat pop up create virtual environment klik ok jika versi python sama dengan 3.7, jika versi python 3.9 seperti gambar 10.60 maka klik cancel dan ikuti tahapan Menambahkan Interpreter ke dalam Voice Cloning Project dibawah.



Gambar 10.60 Pop Up Create Virtual Environment

11. jika berhasil maka anda akan melihat tampilan seperti pada gambar 10.61



Gambar 10.61 Real-Time-Voice-Cloning project

10.7 Menambahkan Interpreter ke dalam Voice Cloning Project

Sebelum menjalankan project menggunakan Pycharm kita membutuhkan interpreter, saya menyarankan untuk menggunakan virtual environment atau conda environment jika anda pengguna anaconda. Dalam pembuatan interpreter penggunaan python 3.7 sangat direkomendasikan karena pada pembuatan voice cloning ini menggunakan tensorflow versi 1.x sehingga teman-teman harus mendownload python versi 3.7, ffmpeg, pytorch, dan requirements. Berikut tata pembuatan virtual environment untuk project voice cloning ini:

10.7.1 Download dan Install Python 3.7

1. download python versi 3.7 pada website resmi python atau kunjungi link berikut <https://www.python.org/downloads/release/python-371/>

2. Pilih file yang sesuai dengan sistem operasi dan processor pada komputer teman-teman.

Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		99f78ecbf766ea449c4d9e7eda19e83	22802018	SIG
XZ compressed source tarball	Source release		0a57e9022c07fad3adb2ee5f8568edb	16960060	SIG
macOS 64-bit/32-bit installer	macOS	for Mac OS X 10.6 and later	ac6630338b53b9e5b9dbb1bc2390a21e	34360623	SIG
macOS 64-bit installer	macOS	for OS X 10.9 and later	b69d52f22e73e1fe37322337eb199a53	27725111	SIG
Windows help file	Windows		b5ca69aa44aa46cd8cf2b527d699740	8534435	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	74f919be8add2749e73d2d91eb6d1da5	6879900	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	4c9fd65b437ad39352e57f15ce832bc	26260496	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	6d866305db7e3d523ae0eb252ebd9407	1333960	SIG
Windows x86 embeddable zip file	Windows		aa4188ea480a64a3ea87e72e09f4c097	6377805	SIG
Windows x86 executable installer	Windows		da2451f28e4cc133c5f0638459993c	25537464	SIG
Windows x86 web-based installer	Windows		20b163041935862876433708819c97db	1297224	SIG

Gambar 10.62 Download Python 3.7

3. Centang add python 3.7 to PATH agar python yang diinstal ditambahkan ke environment variabel komputer teman-teman, kemudian klik install now.



Gambar 10.63 Add Python to PATH

4. Jika instalasi telah selesai maka teman-teman akan melihat tampilan seperti gambar 10.64



Gambar 10.64 Success Install Python 3.7

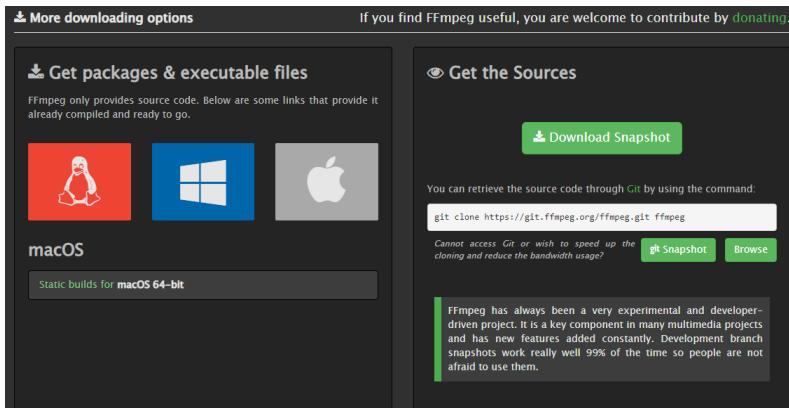
- Untuk memastikan apakah python 3.7 telah benar-benar berhasil terinstal teman-teman dapat melakukan pengecekan melalui command prompt (CMD) dengan mengetikkan perintah python kemudian tekan enter.

A screenshot of a command prompt window. The text inside reads:
C:\Users\ACER>python
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:05:16) [MSC v.1915 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>

Gambar 10.65 Check Python Version

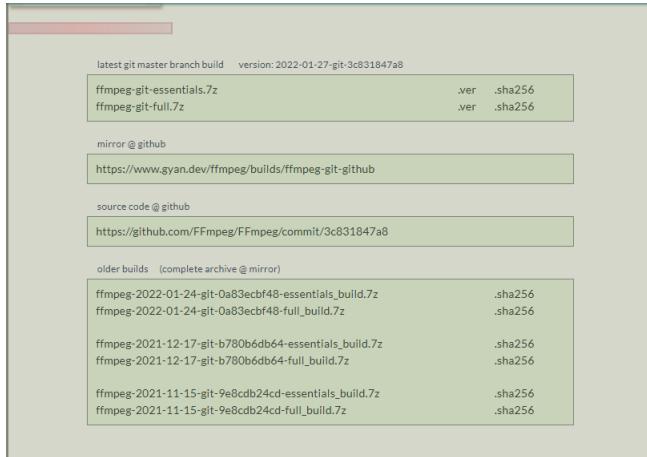
10.7.2 Download dan Install FFmpeg

- Setelah menginstal python 3.7, kita harus menginstal ffmpeg, kunjungi link berikut untuk mendapatkan file instalasi ffmpeg <https://ffmpeg.org/download.html>. Sesuaikan file download dengan sistem operasi yang teman-teman gunakan. Pada tutorial ini saya menggunakan system operasi windows.



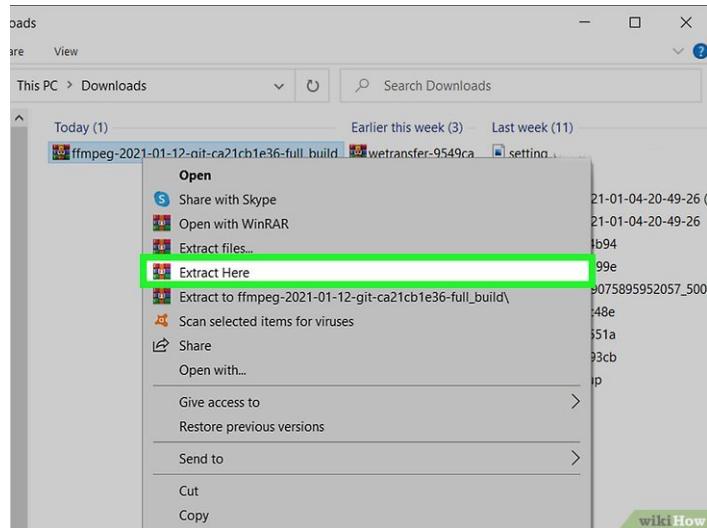
Gambar 10.66 Download FFmpeg

- Pilih Windows builds from gyan.dev. Kemudian teman-teman akan diarahkan ke halaman seperti pada gambar 10.67



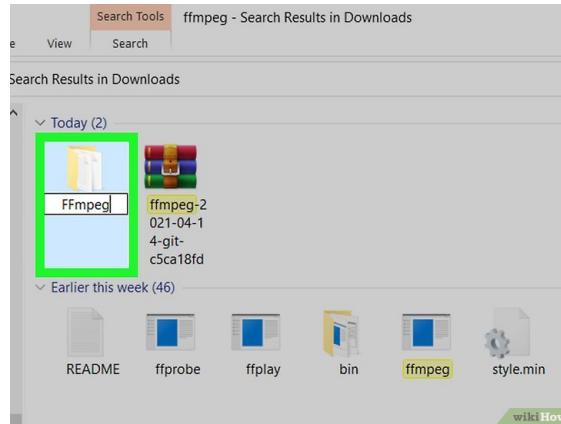
Gambar 10.67 Download FFmpeg for Windows

- Download file `ffmpeg-git-full.7z` kemudian extract file instalasi FFmpeg yang telah didownload.



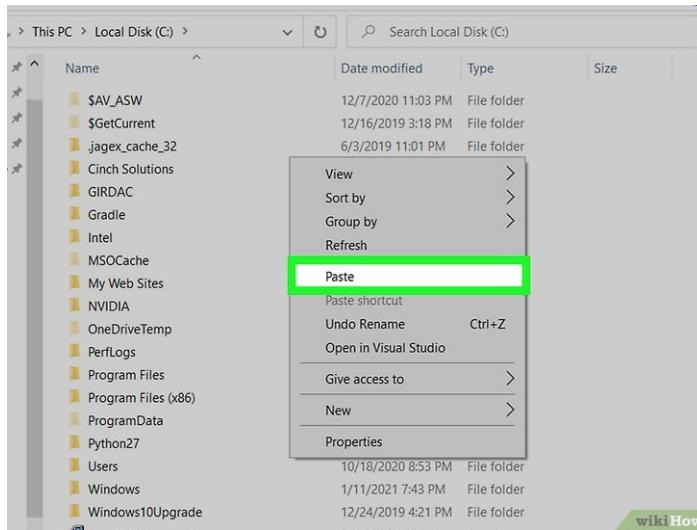
Gambar 10.68 Extract File Instalasi

4. Ganti nama direktori menjadi FFmpeg

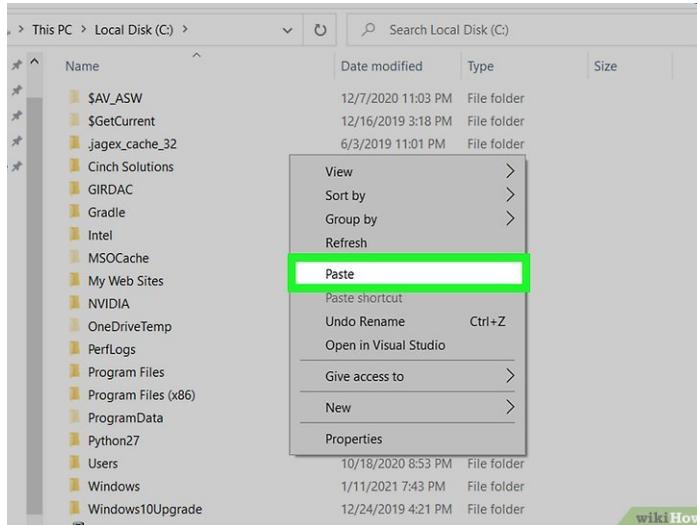


Gambar 10.69 Ganti Nama Folder

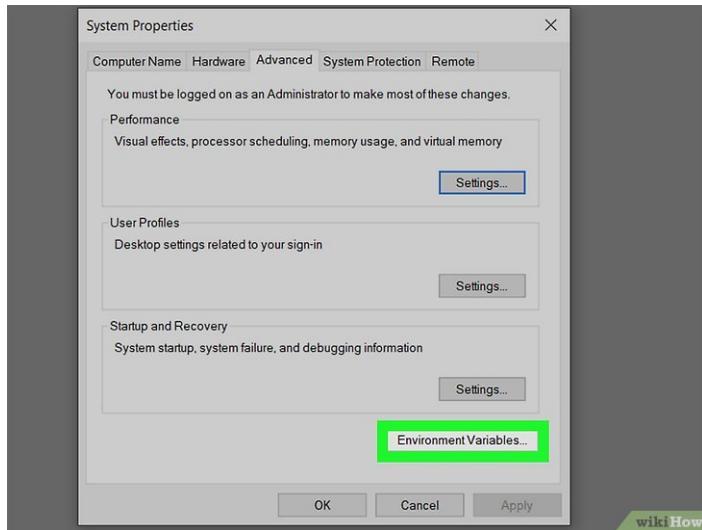
5. Copy folder dan Paste folder ke Local Disk C

**Gambar 10.70** Move Folder

6. Klik start menu pada komputer anda lalu ketikkan environment variable. Klik edit environment variables for your account

**Gambar 10.71** Move Folder

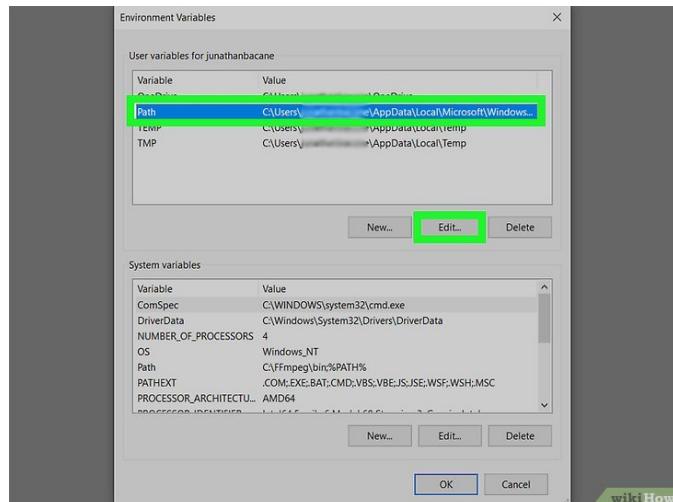
7. Klik menu environment variables



Gambar 10.72 Environment Variables

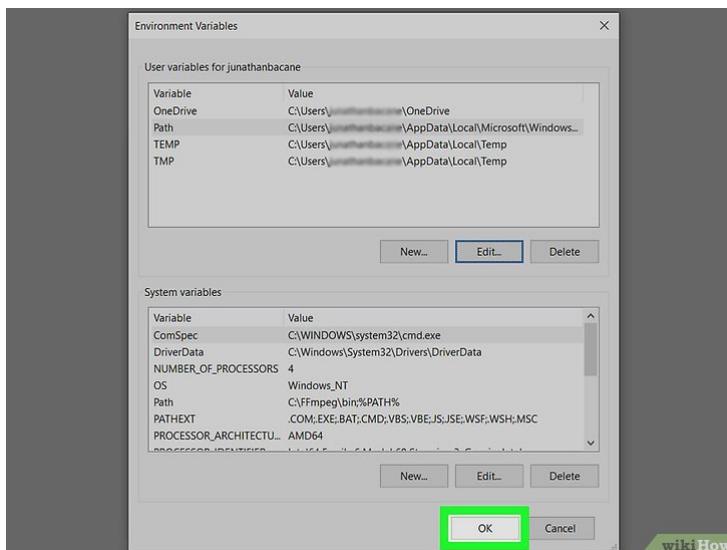
- Pada bagian PATH klik edit maka pop up window edit environment variable seperti pada gambar 10.73 akan muncul. Ketikkan lokasi folder bin FFmpeg lalu klik OK.

C:\FFmpeg\bin



Gambar 10.73 Add FFmpeg to PATH

9. Klik OK untuk menyimpan perubahan pada environment variables



Gambar 10.74 Save Changes

10. Untuk melakukan pengecekan apakah FFmpeg telah berhasil di install pada komputer, teman-teman bisa melakukan pengecekan dengan mengetikkan ffmpeg -version pada command prompt.

```
C:\Windows\system32>ffmpeg -version
ffmpeg version 2021-09-08-git-5e7e2e5031-full_build-www.gyan.dev Copyright (c) 2000-2021
built with gcc 10.3.0 (Rev5, Built by MSYS2 project)
configuration: --enable-gpl --enable-version3 --enable-static --disable-w32threads --dis
fig --enable-iconv --enable-gnuTLS --enable-libxml2 --enable-gmp --enable-lzma --enable-
librist --enable-libsrt --enable-libssh --enable-libzmq --enable-avisynth --enable-lib
sdl2 --enable-libdav1d --enable-libzvbi --enable-libravie --enable-libsvtav1 --enable-li
libx265 --enable-libxvid --enable-libaom --enable-libopenjpeg --enable-libvpx --enable-l
ibfreetype --enable-libfribidi --enable-libvidstab --enable-libvmaf --enable-libzimg --e
nable-cuvid --enable-ffnvcodec --enable-nvdec --enable-nvenc --enable-d3d11va --enable-d
bg1slang --enable-vulkan --enable-opencl --enable-libcdio --enable-libgme --enable-libmo
le-libopencore-amrwb --enable-libmp3lame --enable-libshine --enable-libtheora --enable-l
c --enable-libilbc --enable-libgsm --enable-libopencore-amrnb --enable-libopus --enable-
able-libdspa --enable-libbs2b --enable-libflite --enable-libmysofa --enable-librubberband
aprint
libavutil      57. 4.101 / 57. 4.101
libavcodec     59. 7.102 / 59. 7.102
libavformat    59. 5.100 / 59. 5.100
libavdevice    59. 0.101 / 59. 0.101
libavfilter     8. 7.101 /  8. 7.101
libswscale      6. 1.100 /  6. 1.100
libswresample   4. 0.100 /  4. 0.100
libpostproc    56. 0.100 / 56. 0.100

C:\Windows\system32>
```

Gambar 10.75 Check FFmpeg Version

10.7.3 Tutorial Membuat Virtual Environment

10.7.4 Tutorial Membuat Conda Environment

10.7.5 Install Requirements

Perhatikan struktur folder project voice cloning teman-teman maka akan terlihat file dengan nama requirements.txt, file tersebut berisikan library-library yang digunakan dalam project ini. Library tersebut harus diinstall terlebih dahulu agar project dapat dijalankan dengan baik. Berikut tata cara instalasi requirements:

1. Jika teman-teman menggunakan conda environment maka buka anaconda prompt dan ketikkan perintah berikut untuk pindah directory dan mengaktifkan conda environment:

```
cd conda/envs
conda activate your_environment
```



Gambar 10.76 Select Pytorch with Conda

2. Ketikkan perintah berikut untuk menginstal requirements, tunggu hingga proses instalasi selesai.

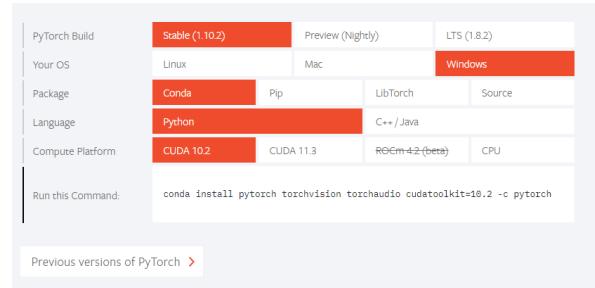
```
pip install -r requirements.txt
```



Gambar 10.77 Select Pytorch with Conda

3. Jika teman-teman menggunakan virtual environment maka buka command prompt dan ketikkan perintah berikut untuk pindah directory dan mengaktifkan virtual environment:

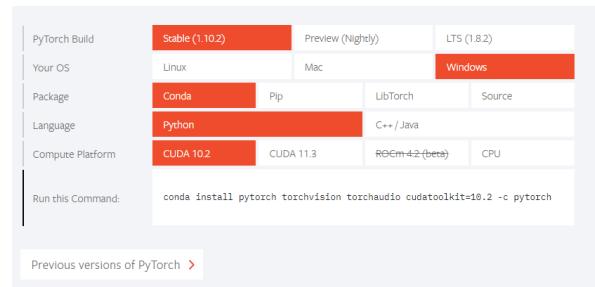
```
cd your-project
source my-env/bin/activate
```



Gambar 10.78 Select Pytorch with Conda

4. Ketikkan perintah berikut untuk menginstal requirements, tunggu hingga proses instalasi selesai.

```
conda install --file requirements.txt
```



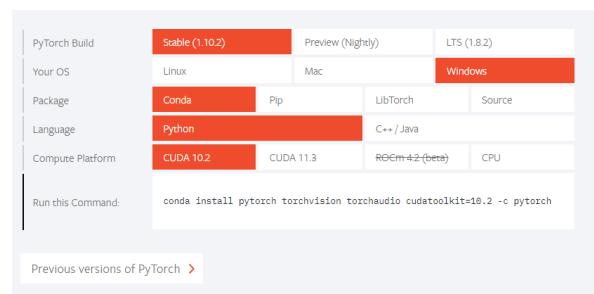
Gambar 10.79 Select Pytorch with Conda

10.7.6 Download and install Pytorch dengan Anaconda

Pytorch merupakan library open source yang digunakan dalam pengembangan dan pelatihan neural network. Dalam project ini, apabila komputer teman-teman memiliki GPU yang mendukung maka saya sarankan untuk menggunakan GPU dalam proses training. Tidak masalah jika tidak memiliki GPU, teman-teman dapat melakukan proses training menggunakan CPU. Instalasi Pytorch dapat dilakukan dengan cuda ataupun Pytorch dan cpu saja.

Berikut cara instalasi Pytorch:

1. Kunjungi situs web pytorch, berikut link website <https://pytorch.org/>.
2. Perhatikan gambar 10.80. Pastikan teman-teman memilih stable build pytorch dan menyesuaikan sistem operasi dengan komputer teman-teman. Apabila teman-teman menggunakan conda environment maka pilih conda dan bahasa pemrograman python. Jika teman-teman menggunakan GPU maka select Cuda dengan versi yang teman-teman inginkan, pilih CPU apabila menggunakan CPU. Pada run this command teman-teman akan melihat perinta instalasi menggunakan conda, lalu copy perintah tersebut.



Gambar 10.80 Select Pytorch with Conda

3. Jika teman-teman menggunakan conda environment, buka anaconda prompt, lalu ketikkan perintah berikut untuk pindah direktori.

```
cd conda/envs
```

perintah ini untuk mengaktifkan conda environment teman-teman. Ganti your-environment dengan nama conda environment yang sebelumnya telah teman-teman buat.

```
conda activate your-environment
```

4. Jika conda environment telah aktif maka teman-teman paste kan perintah yang telah di copy sebelumnya lalu tekan enter.

**Gambar 10.81** Select Pytorch with Pip

5. Tunggu hingga proses instalasi Pytorch selesai.

**Gambar 10.82** Select Pytorch with Pip

6. untuk memastikan apakah pytorch sudah berhasil di install maka teman-teman dapat melakukan pengecekan dengan cara mengetikkan perintah python lalu tekan enter.

**Gambar 10.83** Select Pytorch with Pip

7. Selanjutnya teman-teman ketikkan perintah import torch lalu tekan enter.



Gambar 10.84 Select Pytorch with Pip

8. Ketikkan perintah berikut untuk melihat versi pytorch yang telah berhasil di install.

```
print(torch.__version__)
```

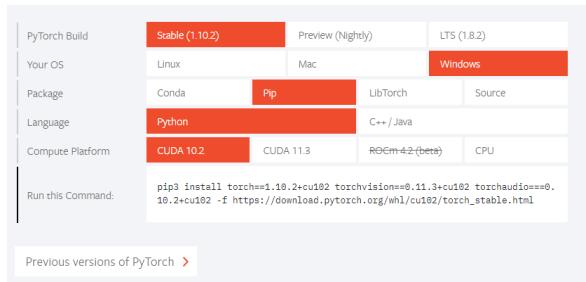


Gambar 10.85 Select Pytorch with Pip

10.7.7 Download and install Pytorch dengan Pip

Berikut cara instalasi Pytorch menggunakan pip

1. Kunjungi situs web pytorch, berikut link website <https://pytorch.org/>.
2. Perhatikan gambar 10.91. Pastikan teman-teman memilih stable build pytorch dan menyesuaikan sistem operasi dengan komputer teman-teman. Apabila menggunakan virtual environment maka pilih Pip lalu copy perintah pada run this command.



Gambar 10.86 Select Pytorch with Pip

3. Jika teman-teman menggunakan virtual environment, buka terminal pada Pycharm, lalu ketikkan perintah berikut untuk mengaktifkan virtual environment teman-teman. Ganti my-env dengan nama environment yang telah teman-teman buat sebelumnya.

```
source my-env/bin/activate
```

4. Jika virtual environment telah aktif maka teman-teman paste kan perintah yang telah di copy sebelumnya lalu tekan enter.



Gambar 10.87 Select Pytorch with Pip

5. Tunggu hingga proses instalasi Pytorch selesai.



Gambar 10.88 Select Pytorch with Pip

6. untuk memastikan apakah pytorch sudah berhasil di install maka teman-teman dapat melakukan pengecekan dengan cara mengetikkan perintah python lalu tekan enter.



Gambar 10.89 Select Pytorch with Pip

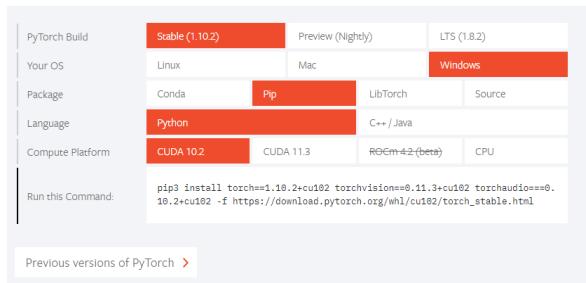
7. Selanjutnya teman-teman ketikkan perintah import torch lalu tekan enter.



Gambar 10.90 Select Pytorch with Pip

8. Ketikkan perintah berikut untuk melihat versi pytorch yang telah berhasil di install.

```
print(torch.__version__)
```



Gambar 10.91 Select Pytorch with Pip

BAB 11

TUTORIAL PEMBUATAN PROJECT MENGGUNAKAN GOOGLE COLAB

Google colab atau Google Colaboratory merupakan solusi yang sangat baik apabila teman-teman memiliki masalah pada tutorial sebelumnya atau mungkin komputer teman-teman tidak memiliki aspek yang cukup memadai dalam pembuatan project ini. Google colab adalah sebuah software buatan Google yang digunakan untuk keperluan research di bidang data science dan machine learning. Hal yang lebih menakjubkannya lagi, google colab menyediakan free GPU yang akan sangat kita butuhkan dalam project ini. Selain GPU masih banyak keunggulan lainnya dari google colab yang akan membantu kita seperti fitur yang memungkinkan google colab terhubung dengan google drive sebagai media penyimpanan dataset, kodingan, hasil training dan model. Karena dapat di simpan di google drive tentu membuat project ini mudah untuk dibagikan kepada tim riset agar dapat mengetahui progres dan hasilnya.

Apabila teman-teman merupakan orang yang menyukai sesuatu hal yang gratis tetapi tidak praktis maka saya sarankan untuk menggunakan google colab untuk mengerjakan project ini. Tetapi jika teman-teman memiliki komputer dengan spesifikasi GPU yang sangat baik dan melebihi GPU gratis yang diberikan google colab maka akan lebih baik menggunakan komputer pribadi saja. Masing-masing cara memiliki kelebihan dan kekurangan. Menurut saya memanfaatkan google colab me-

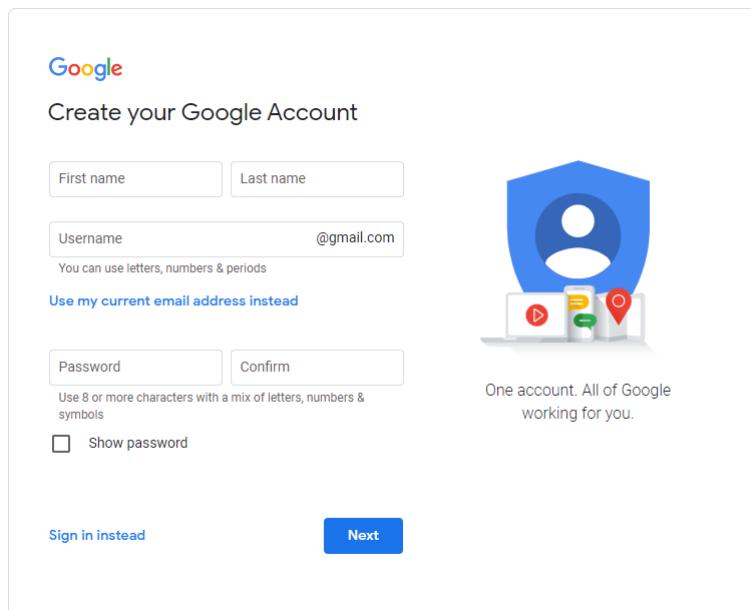
mang gratis namun sangat tidak praktis karena teman-teman akan membutuhkan akun google yang sangat banyak. Sementara itu menggunakan komputer pribadi mungkin membutuhkan lebih banyak biaya tetapi akan sangat praktis, tidak butuh mendaftar banyak akun google dan tidak terbatas waktu. Perlu teman-teman ketahui bahwa free GPU google colab tidak berlaku selamanya tetapi hanya 12 jam per akun google setelah itu teman-teman harus mengganti akun google yang telah mencapai limit penggunaan GPU dengan akun google yang baru.

Berikut tata cara pembuatan Voice Cloning Project menggunakan google colab:

11.1 Membuat Akun Google

Sebelum kita memasuki tata cara penggunaan google colab, kita harus mengetahui cara untuk membuat akun google terlebih dahulu. Ikuti ya tahap-tahap berikut ini:

1. Buat sebuah akun google baru, kunjungi link berikut <https://accounts.google.com/signup/v2/webcreateaccount?hl=en&flowName=GlifWebSignIn&flowEntry=SignUp>.



Gambar 11.1 Create Google Account

2. Isikan data diri teman-teman pada form seperti nama, username, dan password. Kemudian klik next

The screenshot shows the 'Create your Google Account' page. At the top, there are input fields for 'First name' (akun) and 'Last name' (training). Below these are fields for 'Username' (akuntraining38) and 'Email address' (@gmail.com). A note says 'Available: abuat0854 fortraining28 buata7332'. There's also a link 'Use my current email address instead'. Further down are password fields ('Password' and 'Confirm') with a note 'Use 8 or more characters with a mix of letters, numbers & symbols'. A 'Show password' checkbox is present. At the bottom are 'Sign in instead' and 'Next' buttons.

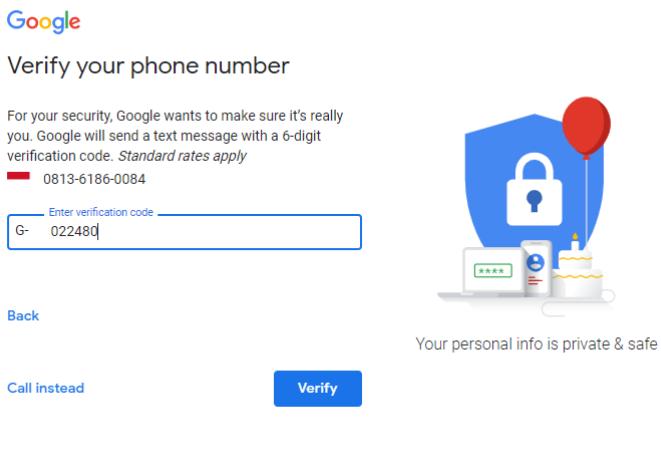
Gambar 11.2 Isi form pendaftaran

- Isikan nomor handphone teman-teman yang aktif untuk verifikasi pembuatan akun google lalu klik next.

The screenshot shows the 'Verify your phone number' page. It explains that Google wants to make sure it's really you and will send a text message with a 6-digit verification code. A note states 'Standard rates apply'. A dropdown menu shows a selected phone number (628120000000) which is highlighted in blue. Below the input field are 'Back' and 'Next' buttons. To the right, there's an illustration of a blue hexagon containing a white padlock and a red balloon, with icons for a smartphone, laptop, and other devices below it. A note at the bottom right says 'Your personal info is private & safe'.

Gambar 11.3 Isikan Nomor Handphon

4. Cek handphone teman-teman apakah ada sms dari google yang berisikan kode verifikasi, jika ada masukkan 6 digit kode verifikasi tersebut lalu klik verify.



Gambar 11.4 Verify Phone Number

5. Isikan data-data tambahan seperti email pemulihan, tanggal lahir, dan jenis kelamin pada form. Klik Next.



akun, welcome to Google

akuntraining38@gmail.com

Phone number (optional)

Google will use this number only for account security. Your number won't be visible to others. You can choose later whether to use it for other purposes.

Recovery email address (optional)

We'll use it to keep your account secure

Month

Day

Year

Your birthday

Gender



Your personal info is private & safe

[Why we ask for this information](#)

[Back](#)

[Next](#)

Gambar 11.5 Form Personal Data

6. Pada tahap Get more from your number, teman-teman bisa klik skip atau klik yes, tergantung pada pilihan masing-masing. Jika mengklik Yes, I'm in maka teman-teman telah setuju bahwa nomor handphone teman-teman dapat digunakan diseluruh layanan google.



Get more from your number

If you like, you can add your phone number to your account for use across Google services. [Learn more](#)

For example, your number will be used to

Receive video calls & messages

G Make Google services, including ads, more relevant to you

[More options](#)



Your personal info is private & safe

[Back](#)

[Skip](#)

[Yes, I'm in](#)

Gambar 11.6 Get more from your number

- Pada tahap Privacy and Terms klik I agree untuk menyelesaikan pendaftaran akun google.

Combining data

We also combine this data among our services and across your devices for these purposes. For example, depending on your account settings, we show you ads based on information about your interests, which we can derive from your use of Search and YouTube, and we use data from trillions of search queries to build spell-correction models that we use across all of our services.

You're in control

Depending on your account settings, some of this data may be associated with your Google Account and we treat this data as personal information. You can control how we collect and use this data now by clicking "More Options" below. You can always adjust your controls later or withdraw your consent for the future by visiting My Account ([myaccount.google.com](#)).

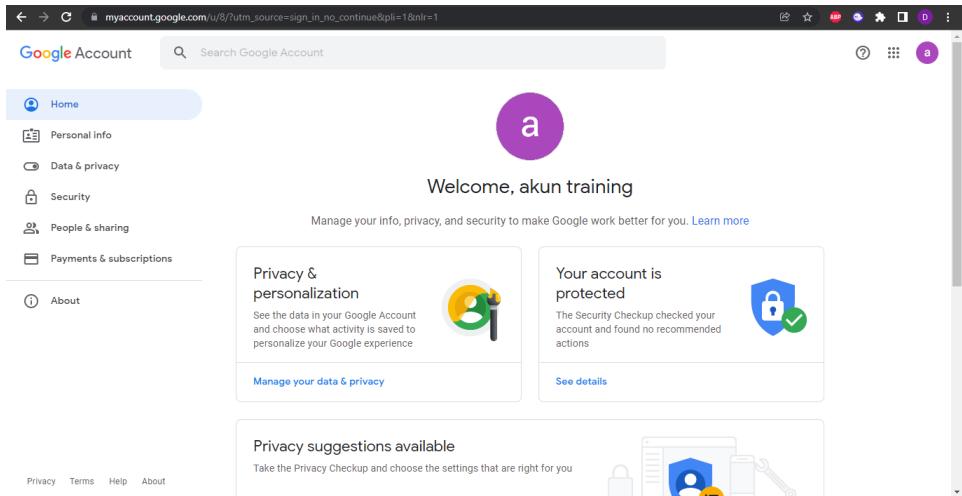
[More options ▾](#)

[Cancel](#)

[I agree](#)

Gambar 11.7 Create Google Account

8. Jika tampilan website seperti gambar 11.8 maka selamat, teman-teman telah berhasil membuat akun google teman-teman.

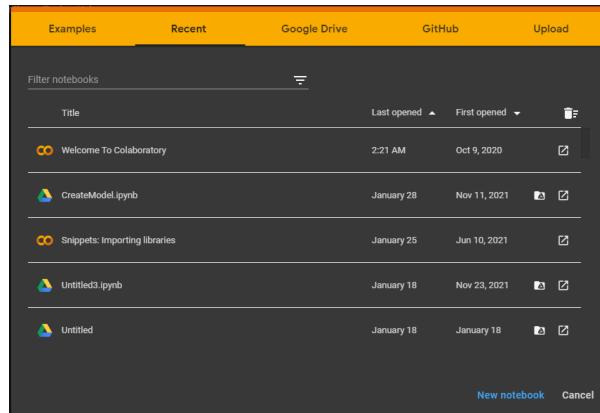


Gambar 11.8 Create Google Account

11.2 Membuat New Project Pada Google Colab

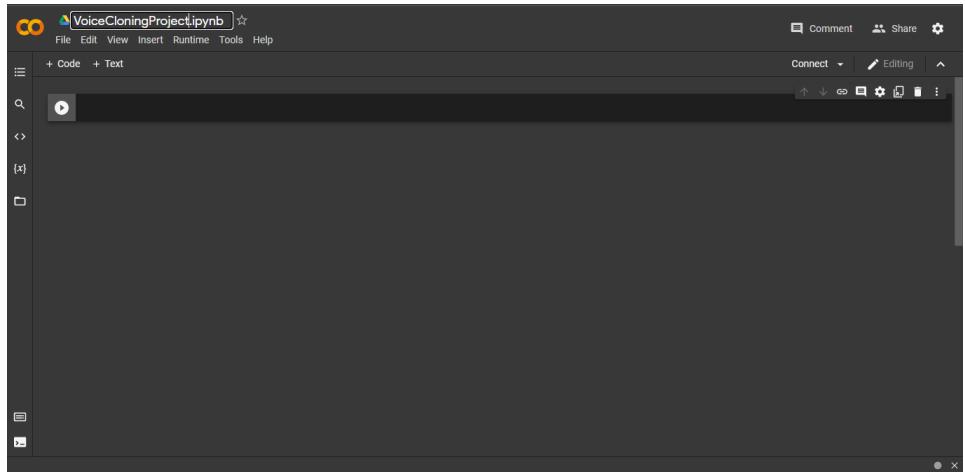
Saya akan mengajarkan kepada teman-teman cara membuat project baru di google colab dan menuliskan script code program yang akan kita gunakan untuk preprocess-ing dataset dan training tiga model yang akan kita butuhkan pada project ini. Berikut langkah-langkah pembuatannya:

1. Kunjungi website google colab atau akses link berikut <https://colab.research.google.com/>. Klik new project untuk membuat project google colab.



Gambar 11.9 Create New Project

2. Rename nama project menjadi VoiceCloningProject untuk memudahkan teman-teman melakukan pencarian file project ini pada google drive.



Gambar 11.10 Rename File

3. Ketikkan script berikut untuk melakukan mounting google colab ke google drive agar teman-teman dapat mengakses penyimpanan pada google drive.

```
1 from google.colab import drive
2 drive.mount('/content/drive')
```

Listing 11.1 Mounting Google Drive

```
From google.colab import drive
drive.mount('/content/drive')
```

Gambar 11.11 Script Mounting Google Drive

4. Klik text untuk mempercantik tampilan project teman-teman lalu ketikkan #Mounting.

```
#Mounting
[ ] from google.colab import drive
drive.mount('/content/drive')
```

Gambar 11.12 Add Text to Project

5. Tambahkan kode untuk berpindah direktori ke folder Colab Notebooks yang ada didalam penyimpanan google drive teman-teman.

```
[language=Python, caption=Change Directory]
%cd /content/drive/MyDrive/ColabNotebooks
```

The screenshot shows a Google Colab notebook titled "VoiceCloningProject.ipynb". The code cell contains the following Python code:

```
[x] [ ] from google.colab import drive  
drive.mount('/content/drive')  
  
[x] %cd /content/drive/MyDrive/ColabNotebooks
```

Gambar 11.13 Change Directory

- 6.Tambahkan lagi text dan ketikkan #Clone Repository lalu klik code untuk menambahkan kode berikut ini untuk clone repository dari github dan menginstall semua library yang ada di file requirements.txt.

```
1 %tensorflow_version 1.x  
2 import os  
3 from os.path import exists, join, basename, splitext  
4  
5 git_repo_url = 'https://github.com/dindamajesty13/Real-Time-Voice  
-Cloning.git'  
6 project_name = splitext(basename(git_repo_url))[0]  
7 if not exists(project_name):  
8     # clone and install
```

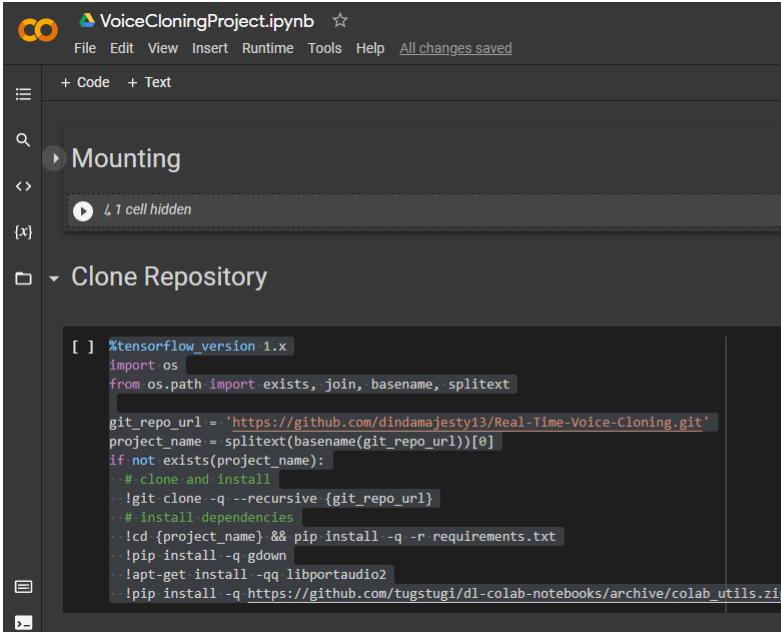
```

9   !git clone -q --recursive {git_repo_url}
10  # install dependencies
11  !cd {project_name} && pip install -q -r requirements.txt
12  !pip install -q gdown
13  !apt-get install -qq libportaudio2
14  !pip install -q https://github.com/tugstugi/dl-colab-notebooks/
      archive/colab_utils.zip

```

Listing 11.2 Clone Repository

Silahkan ganti git_repo_url dengan link repository teman-teman.



The screenshot shows a Google Colab notebook titled "VoiceCloningProject.ipynb". The interface includes a top navigation bar with File, Edit, View, Insert, Runtime, Tools, Help, and a "All changes saved" indicator. Below the navigation is a toolbar with "+ Code" and "+ Text" buttons. The main workspace is divided into sections: "Mounting" and "Clone Repository". The "Clone Repository" section contains the following Python code:

```

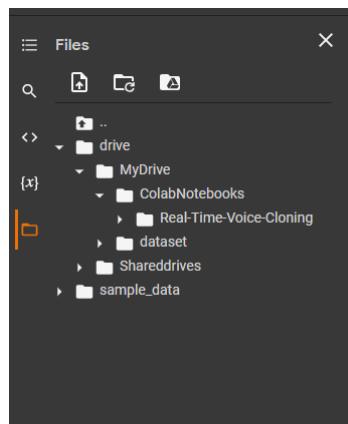
[ ] %tensorflow_version 1.x
import os
from os.path import exists, join, basename, splitext

git_repo_url = 'https://github.com/dindamajesty13/Real-Time-Voice-Cloning.git'
project_name = splitext(basename(git_repo_url))[0]
if not exists(project_name):
    # clone and install
    !git clone -q --recursive {git_repo_url}
    # install dependencies
    !cd {project_name} && pip install -q -r requirements.txt
    !pip install -q gdown
    !apt-get install -qq libportaudio2
    !pip install -q https://github.com/tugstugi/dl-colab-notebooks/archive/colab_utils.zip

```

Gambar 11.14 Script Clone Repository

- Apabila Clone Project telah selesai maka klik icon folder di kiri tampilan google colab teman-teman lalu cari folder Real-Time-Voice-Cloning untuk memastikan bahwa repositori telah berhasil di clone.



Gambar 11.15 Check Folder

8. Buka file config.py yang terdapat didalam folder encoder, edit kode berikut sesuai dengan dataset yang teman-teman gunakan. Ganti value dari key clean sesuai dengan nama folder tempat teman-teman menyimpan dataset. Jika teman-teman hanya menggunakan satu dataset maka cukup abaikan salah satunya. Saya sarankan teman-teman mengganti nama variabel sesuai dengan dataset teman-teman.

```

1 #dataset 1
2 common_voice = {
3     "train": {
4         "clean": ["common_voice"]
5     },
6     "test": {
7         "clean": ["test_clean_data"]
8     },
9 }
10
11 #dataset 2
12 titml_datasets = {
13     "train": {
14         "clean": ["titml"]
15     },
16     "test": {
17         "clean": ["wwDataset"]
18     },
19 }
20
21 #contoh
22 nama_dataset = {
23     "train": {
24         "clean": ["nama_folder_dataset"]
25     },
26     "test": {
27         "clean": ["nama_folder_dataset"]
28     },

```

29 }

Listing 11.3 Config Dataset

9. Buka file preprocess.py yang ada didalam folder encoder lalu edit kode berikut, sesuaikan dengan dataset yang teman-teman gunakan.

```
1 #import dataset sesuai dengan nama dataset pada config.py
2 #jika hanya satu dataset saja cukup importkan satu dataset , jika
3     lebih dari dua maka importkan dan pisahkan dengan koma
4
5
6 #ganti common_voice menjadi nama dataset pada config.py dan
7     sesuaikan dengan yang diimportkan
8 #ganti wav sesuai dengan ekstensi file audio dataset teman-teman
9     (mp3, flac, wav, m4a)
10
11
12 def preprocess_clean_dataset(datasets_root: Path, out_dir: Path,
13     skip_existing=False):
14     for dataset_name in common_voice["train"]["clean"]:
15         # Initialize the preprocessing
16         dataset_root, logger = _init_preprocess_dataset(
17             dataset_name, datasets_root, out_dir)
18         if not dataset_root:
19             return
20
21         # Preprocess all speakers
22         speaker_dirs = list(dataset_root.glob("*"))
23         _preprocess_speaker_dirs(speaker_dirs, dataset_name,
24             datasets_root, out_dir, "wav",
25             skip_existing, logger)
26
27 #sama dengan diatas .
28 #catatan: jika teman-teman hanya menggunakan satu dataset maka
29     hapus function ini , abaikan , atau komen.
30 #jika menggunakan tiga dataset maka tambahkan function ini dan
31     sesuaikan dengan dataset teman-teman seperti cara di atas.
32
33 def preprocess_speech_dataset(datasets_root: Path, out_dir: Path,
34     skip_existing=False):
35     for dataset_name in titml_datasets["train"]["clean"]:
36         # Initialize the preprocessing
37         dataset_root, logger = _init_preprocess_dataset(
38             dataset_name, datasets_root, out_dir)
39         if not dataset_root:
40             return
41
42         # Preprocess all speakers
43         speaker_dirs = list(dataset_root.glob("*"))
44         _preprocess_speaker_dirs(speaker_dirs, dataset_name,
45             datasets_root, out_dir, "wav",
46             skip_existing, logger)
```

Listing 11.4 Preprocessing Function

10. Selanjutnya edit file encoder_preprocessing.py seperti pada kode 11.5.

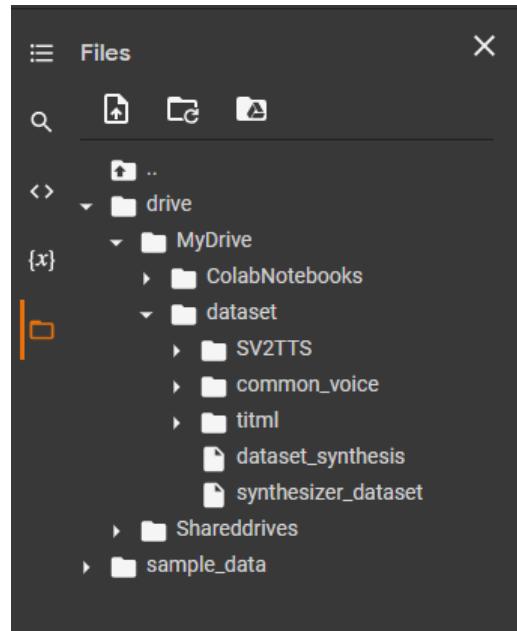
```
1 from encoder.preprocess import preprocess_clean_dataset ,  
2     preprocess_speech_dataset  
3 from utils.argutils import print_args  
4 from pathlib import Path  
5 import argparse  
6  
7 if __name__ == "__main__":  
8     class MyFormatter(argparse.ArgumentDefaultsHelpFormatter ,  
9         argparse.RawDescriptionHelpFormatter):  
10        pass  
11  
12    parser = argparse.ArgumentParser(  
13        description="Preprocesses audio files from datasets ,  
14        encodes them as mel spectrograms and "  
15        " writes them to the disk. This will allow you  
16        to train the encoder. The "  
17        "datasets required are at least one of  
18        VoxCeleb1 , VoxCeleb2 and LibriSpeech ."  
19        " Ideally , you should have all three. You  
20        should extract them as they are "  
21        " after having downloaded them and put them in  
22        a same directory , e.g.:\\n"  
23        " -[datasets_root]\\n"  
24        " -LibriSpeech\\n"  
25        " -train-other-500\\n"  
26        " -VoxCeleb1\\n"  
27        " -wav\\n"  
28        " -vox1.meta.csv\\n"  
29        " -VoxCeleb2\\n"  
30        " -dev",  
31        formatter_class=MyFormatter  
32    )  
33    parser.add_argument("datasets_root" , type=Path , help=\  
34        "Path to the directory containing your LibriSpeech/TTS  
35        and VoxCeleb datasets .")  
36    parser.add_argument("-o" , "--out_dir" , type=Path , default=  
37        argparse.SUPPRESS , help=\  
38        "Path to the output directory that will contain the mel  
39        spectrograms. If left out ,"  
40        " defaults to <datasets_root>/SV2TTS/encoder/")  
41    parser.add_argument("-d" , "--datasets" , type=str ,  
42        default="titml,common_voice" , help=\  
43        "Comma-separated list of the name of the datasets you  
44        want to preprocess. Only the train "  
45        "set of these datasets will be used. Possible names:  
46        librispeech_other , voxceleb1 , "  
47        "voxceleb2 .")  
48    parser.add_argument("-s" , "--skip_existing" , action="  
49        store_true" , help=\  
50        "Whether to skip existing output files with the same name  
51        . Useful if this script was "  
52        "interrupted .")  
53    parser.add_argument("--no_trim" , action="store_true" , help=\  
54        "
```

```
41     "Preprocess audio without trimming silences (not
42     recommended).")
43     args = parser.parse_args()
44
45     # Verify webrtcvad is available
46     if not args.no_trim:
47         try:
48             import webrtcvad
49         except:
50             raise ModuleNotFoundError("Package 'webrtcvad' not
51             found. This package enables "
52             "'noise removal and is recommended. Please install
53             and try again. If installation fails, "
54             "'use --no_trim to disable this error message.'")
55     del args.no_trim
56
57     # Process the arguments
58     args.datasets = args.datasets.split(",")
59     if not hasattr(args, "out_dir"):
60         args.out_dir = args.datasets_root.joinpath("SV2TTS", "encoder")
61     assert args.datasets_root.exists()
62     args.out_dir.mkdir(exist_ok=True, parents=True)
63
64     # Preprocess the datasets
65     print_args(args, parser)
66     preprocess_func = {
67         "titml": preprocess_speech_dataset,
68         "common_voice": preprocess_clean_dataset,
69     }
70     args = vars(args)
71     for dataset in args.pop("datasets"):
72         print("Preprocessing %s" % dataset)
73         preprocess_func[dataset](**args)
```

Listing 11.5 Preprocessing Encoder Model

Pada argument dataset masukkan nama dataset yang teman-teman gunakan, pada kode diatas saya menggunakan dataset titml dan common voice berbahasa indonesia.

11. Upload dataset teman-teman ke google drive dengan struktur direktori seperti gambar 11.16



Gambar 11.16 Struktur Dataset

- 12.Tambahkan text dan ketikkan #Preprocessing encoder lalu tambahkan kode dan ketikkan kode program untuk menjalankan preprocessing encoder model.

```
! python /content/drive/MyDrive/ColabNotebooks/Real-Time-Voice-Cloning/encoder_preprocess.py /content/drive/MyDrive/dataset
```

Listing 11.6 Script to Run Preprocessing Speaker Encoder Model

The screenshot shows a Google Colab notebook titled "VoiceCloningProject.ipynb". The interface includes a top bar with File, Edit, View, Insert, Runtime, Tools, Help, and a "All changes saved" message. On the right, there are buttons for Comment, Share, Connect, Editing, and other settings. The left sidebar has sections for Mounting, Clone Repository, and Preprocessing Encoder, each with a "4 2 cells hidden" or "4 1 cell hidden" indicator. The main workspace contains a single command cell: "[] !python /content/drive/MyDrive/ColabNotebooks/Real-Time-Voice-Cloning/encoder_preprocess.py /content/drive/MyDrive/dataset".

Gambar 11.17 *Preprocessing Speaker Encoder Model*

