SKRIPSI

TEXT TO SPEECH BAHASA BUGIS MENGGUNAKAN TACOTRON 2 DENGAN METODE *TRANSFER LEARNING*

Disusun dan diajukan oleh:

NUR HIKMAH
D121 18 1015

PROGRAM STUDI SARJANA TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS HASANUDDIN
GOWA
2024

LEMBAR PENGESAHAN SKRIPSI

TEXT TO SPEECH BAHASA BUGIS MENGGUNAKAN TACOTRON 2 DENGAN METODE *TRANSFER LEARNING*

Disusun dan diajukan oleh

Nur Hikmah
D121 18 1015

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Teknik Informatika
Fakultas Teknik Universitas Hasanuddin
Pada tanggal 8 Mei 2024
dan dinyatakan telah memenuhi syarat kelulusan

Menyetujui,

Pembimbing Utama,	Pembimbing Pendamping,
Dr. Ir. Ingrid Nurtanio, M.T.	Anugrayani Bustamin, S.T., M.T.
NIP 19610813 198811 2 001	NIP 19901201 201807 4 001

Ketua Program Studi,

Prof. Dr. Ir. Indrabayu, S.T., M.T., M.Bus.Sys., IPM, ASEAN. Eng.
NIP 19750716 200212 1 004

PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini;

Nama	: Nur Hikmah
NIM	: D121 18 1015
Program Studi	: Teknik Informatika
Jenjang	: S1

Menyatakan dengan ini bahwa karya tulisan saya berjudul

{*Text To Speech* Bahasa Bugis Menggunakan Tacotron 2 Dengan Metode *Transfer Learning*}

Adalah karya tulisan saya sendiri dan bukan merupakan pengambilan alihan tulisan orang lain dan bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Semua informasi yang ditulis dalam skripsi yang berasal dari penulis lain telah diberi penghargaan, yakni dengan mengutip sumber dan tahun penerbitannya. Oleh karena itu semua tulisan dalam skripsi ini sepenuhnya menjadi tanggung jawab penulis. Apabila ada pihak manapun yang merasa ada kesamaan judul dan atau hasil temuan dalam skripsi ini, maka penulis siap untuk diklarifikasi dan mempertanggungjawabkan segala resiko.

Segala data dan informasi yang diperoleh selama proses pembuatan skripsi, yang akan dipublikasi oleh Penulis di masa depan harus mendapat persetujuan dari Dosen Pembimbing.

Apabila dikemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan isi skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Gowa, Mei 2024

Yang Menyatakan

Nur Hikmah

ABSTRAK

NUR HIKMAH. *Text To Speech Bahasa Bugis Menggunakan Tacotron 2 Dengan Metode Transfer Learning* (dibimbing oleh Ingrid Nurtanio dan Anugrayani Bustamin)

Bahasa daerah seperti Bahasa Bugis, adalah bagian tak terpisahkan dari kekayaan budaya Indonesia yang perlu dilestarikan. Teknologi *Text-to-Speech* (TTS), selain menjadi alat yang bermanfaat dibidang pendidikan, juga dapat digunakan untuk mempromosikan bahasa Bugis dan membuatnya lebih mudah diakses oleh masyarakat luas. Namun, penggunaan teknologi TTS untuk bahasa daerah masih terbatas karena keterbatasan data dan sumber daya komputasi untuk membangun model TTS itu sendiri. Untuk mengatasi hambatan tersebut, penelitian ini bertujuan untuk mengembangkan model TTS bahasa Bugis yang berkualitas tinggi menggunakan arsitektur Tacotron 2 dan metode *transfer learning* meskipun dengan sumber data yang terbatas (*low resource)*.

Penelitian ini menggunakan Tacotron 2, sebuah arsitektur jaringan saraf *end-to-end* yang mampu menghasilkan suara berkualitas tinggi berdasarkan teks *input*. Serta metode *transfer learning* dipilih untuk memanfaatkan pengetahuan yang telah dipelajari dari model Tacotron 2 yang dilatih pada *dataset* bahasa Inggris (*pre-trained* model) dan menyesuaikannya dengan Bahasa Bugis yang memiliki sumber daya data yang terbatas menggunakan teknik *fine-tuning*.

Fine-tuning hyperparameter pada model Bugis TTS yang optimal diperoleh pada *epoch* 183 dari total 200 *epoch*, *learning rate* 1e-5 (0.00001) dan *dropout* 0.1 yang memperoleh *loss training* dan *loss validation* paling rendah yaitu 0.12 dan 0.10. Adapun untuk mengevaluasi kualitas suara yang dihasilkan model Bugis TTS dilakukan dengan metrik *Mean Opinion Score* (MOS) dan metode pendukung lainnya yaitu tes *Turing*, tes *Listening* dan *Confusion Matrix*. Terdapat tiga kriteria penilaian kualitas suara, yaitu dari segi *naturalness* (kealamian ucapan), *fluidity* (kelancaran ucapan) dan *intelligibility* (kejelasan ucapan), yang masing-masing memperoleh nilai MOS 4.30, 4.37 dan 4.40. Dimana skor tertinggi diperoleh oleh tingkat kejelasan ucapan yang juga didukung dengan hasil evaluasi tes *Listening*, dan skor terendah pada tingkat kealamian ucapan yang juga didukung oleh hasil tes *Turing* dan *Confusion Matrix*. Sehingga secara keseluruhan MOS yang diperoleh model Bugis TTS ini adalah 4.36 dari skala 5, dengan skor tersebut sudah menunjukkan bahwa model Bugis TTS yang dibangun dan dikembangkan menggunakan Tacotron 2 dengan pendekatan *transfer learning* ini mampu menghasilkan suara dengan kualitas yang mudah dipahami dan cukup alami untuk bahasa Bugis meskipun hanya dengan *dataset* yang relatif sedikit yaitu 1 jam 5 menit.

ıci: *Text-to-Speech, Tacotron 2 , Transfer Learning*, Bahasa Bugis, *Low*

ABSTRACT

NUR HIKMAH. *Text To Speech Bugis Language Using Tacotron 2 With Transfer Learning Method* (supervised by Ingrid Nurtanio and Anugrayani Bustamin)

Regional languages such as Bugis, are an inseparable part of Indonesia's cultural richness that need to be preserved. Text-to-Speech (TTS) technology, apart from being a useful tool in the field of education, can also be used to promote the Bugis language and make it more accessible to the wider community. However, the use of TTS technology for regional languages is still limited due to limited data and computing resources to build the TTS model itself. To overcome this obstacle, this research aims to develop a high quality Bugis TTS model using the Tacotron 2 architecture and transfer learning method even with limited data sources (low resources).

This research uses Tacotron 2, an end-to-end neural network architecture capable of producing high-quality sound based on input text. And the transfer learning method was chosen to utilize the knowledge that has been learned from the Tacotron 2 model which was trained on an English dataset (pre-trained model) and adapt it to the Bugis language which has limited data resources using fine-tuning techniques.

Optimal fine-tuning hyperparameters of the Bugis TTS model were obtained at epoch 183 out of a total of 200 epochs, learning rates 1e-5 (0.00001) and dropouts 0.1 with the lowest training loss and loss validation of 0.12 and 0.10. As for evaluating the quality of sound produced by the Bugis TTS model, it was carried out using the Mean Opinion Score (MOS) metric and other supporting methods, namely the Turing test the Listening test and Confusion Matrix. There are three criteria for assessing voice quality, namely in terms of naturalness (naturalness of speech), fluidity (speech fluency) and intelligibility (clarity of speech), which respectively obtained MOS scores of 4.30, 4.37 and 4.40. Where the highest score was obtained by the level of clarity of speech which was also supported by the results of the Listening test evaluation, and the lowest score was obtained by the level of naturalness of speech which was also supported by the results of the Turing test and Confusion Matrix. So overall the MOS obtained by the Bugis TTS model is 4.36 on a scale of 5, with this score already showing that the Bugis TTS model which was built and developed using Tacotron 2 with a transfer learning approach is able to produce sounds with a quality that is easy to understand and quite natural for the Bugis language. although only with a relatively small dataset, namely 1 hour 5 minutes.

Keywords: Text-to-Speech, Tacotron 2, Transfer Learning, Buginese Language, Low Resource

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SINGKATAN DAN ARTI SIMBOL

Lambang/Singkatan	Arti dan Keterangan
TTS	*Text To Speech*
seq2seq	*Sequence to Sequence*
STFT	*Short Time Fourier Transform*
FT	*Fourier Transform*
MOS	*Mean Opinion Score*
µs	*Micro Second*
TP	*True Positive*
TN	*True Negative*
FP	*False Positive*
FN	*False Negative*

DAFTAR LAMPIRAN

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Pujian syukur sebesar-besarnya atas kehadirat Allah *Subhanahu Wa Ta'ala* yang setiap tarikan napas penulis adalah milik-Nya, karena atas limpahan kasih sayang, petunjuk, ampunan serta semua rencana indah-Nya, sehingga penulis dapat menyelesaikan skripsi dengan judul "***Text To Speech* Bahasa Bugis Menggunakan Tacotron 2 Dengan Metode *Transfer Learning***". Skripsi ini diajukan untuk memenuhi salah satu syarat guna memperoleh gelar Sarjana (S1) pada Program Studi Teknik Informatika, Fakultas Teknik, Universitas Hasanuddin.

Penulis menyadari bahwa skripsi ini tidak akan terselesaikan tanpa bantuan dan dukungan dari berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih kepada:

1. Allah *Subhanahu Wa Ta'ala* sebagai pemilik nyawa yang tahu segalanya tentang penulis, atas ketenangan hati dan petunjuk yang senantiasa diberikan untuk penulis meskipun sejatinya penulis masih mencintai-Mu dengan cacat atau tepatnya dengan lalai namun dengan kebaikan-Mu tetap membimbing penulis disetiap detik kehidupannya;
2. Rasulullah Muhammad *Shallallahu Alaihi Wasallam* manusia terbaik yang pernah lahir di muka bumi ini yang selalu merindukan umatnya bahkan diakhir hayat dan penulis berharap menjadi bagian dari barisan panjang umatmu yang mendapat syafaat dan berdiri dibelakangmu dengan senyuman terlebar;
3. Kedua orang tua tercinta, atas doa, restu, dan dukungan moril maupun materil yang diberikan dengan tulus tanpa pamrih kepada penulis;
4. Ibu Dr. Ir. Ingrid Nurtanio, M.T., selaku pembimbing utama yang selalu tersenyum dan senantiasa sabar memberikan arahan dan masukan selama proses penyusunan tugas akhir ini;
5. Ibu Anugrayani Bustamin, S.T., M.T., selaku pembimbing pendamping yang selalu meluangkan waktu, tenaga dan pikiran dengan sepenuh hati selama membimbing penulis dalam menyelesaikan tugas akhir ini;
6. Bapak Prof. Dr. Ir. Indrabayu, S.T.,M.T., M.Bus.Sys., IPM, ASEAN. Eng. serta Ibu Prof. Dr. Eng. Intan Sari Areni, S.T., M.T. selaku penguji yang luar biasa memberikan ilmunya melalui revisi-revisi kepada penulis;
7. Bapak dan Ibu Dosen Departemen Teknik Informatika yang sudah memberikan ilmu dan nasihatnya serta membantu selama perkuliahan;
8. Segenap Staf Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah banyak membantu selama proses perkuliahan hingga penyelesaian tugas akhir ini;
9. Teman-teman Synchronous 18 yang telah berbagi ruang, waktu dan memori suka dan duka bersama selama masa perkuliahan;
10. Seluruh keluarga, sahabat dan pihak yang tidak bisa disebutkan satu persatu, atas dukungan dan semangatnya selama penulis menyelesaikan studi.

ıulis berharap semoga Allah *Subhanahu Wa Ta'ala* berkenan membalas baikan dari semua pihak yang telah banyak membantu dalam penyelesaian ıir ini. Penulis menyadari bahwa skripsi ini masih memiliki banyak

kekurangan. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun dari para pembaca demi perbaikan skripsi ini di masa depan.

Akhir kata, penulis berharap skripsi ini dapat bermanfaat bagi pengembangan ilmu pengetahuan dan teknologi, khususnya dalam bidang *Text-to-Speech* untuk pengembangan bahasa daerah di Indonesia.

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Gowa, Mei 2024

Penulis

BAB I
PENDAHULUAN

1.1 Latar Belakang

Indonesia negara yang kaya akan budaya lokal dimana salah satunya adalah kekayaan akan bahasa daerahnya seperti yang dimaksudkan dalam Undang-Undang Dasar Negara Republik Indonesia Tahun 1945 Pasal 32 ayat 2 yang bunyinya "Negara menghormati dan memelihara bahasa daerah sebagai kekayaan budaya nasional". Hampir setiap suku di NKRI memiliki bahasa daerahnya sendiri sebagai alat komunikasi dan ekspresi dalam kelompok dan identitas budayanya. Menurut data dari laman *labbineka kemdikbud* terdapat sekitar 718 bahasa daerah yang ada di Indonesia. Salah satu slogan Badan Pengembangan dan Pembinaan Bahasa Kemendikbud adalah Tri Gatra Bangun Bahasa yaitu "Utamakan bahasa Indonesia, lestarikan bahasa daerah dan kuasai bahasa asing". Melalui ungkapan ini, menggambarkan bahwa bahasa daerah sebagai salah satu kekayaan bangsa Indonesia perlu dilestarikan. Selain itu, bahasa daerah juga berfungsi sebagai pendukung bahasa nasional kita yakni bahasa Indonesia.

Dengan seiring berkembangnya zaman, aksara seolah terlupakan, ditandainya dengan semakin banyaknya generasi muda yang tidak tahu tentang aksara bahasa daerah yang menjadi bukti bahwa aksara semakin tergerus oleh perkembangan zaman. Aksara daerah mengalami dampak dari zaman digital saat ini, yaitu penurunan penggunaan aksara daerah karena tidak tersedianya di sistem digital. Sekarang masih banyak masyarakat yang tidak mengenal aksara daerah, hal ini disebabkan karena bukan lagi aksara ataupun bahasa daerah yang digunakan sehari-hari oleh masyarakat. Penggunaan huruf Latin dalam berkomunikasi sehari-hari ini juga salah satu yang menyebabkan aksara-aksara daerah sulit berkembang karena persoalan pemakaian aksara-aksara daerah juga terkait erat dengan masalah kebiasaan. Aksara daerah merupakan salah satu peninggalan budaya yang tidak ternilai harganya. Jika kita bisa memahami aksara daerah kita juga setidaknya dapat

ɔil pelajaran dari naskah-naskah bahasa daerah yang di dalamnya terdapat
ın pesan moral, adat istiadat dan keagamaan (Winata et al., 2022).

Meskipun dalam budaya membaca dan menulis aksara bahasa daerah sering dikecualikan dari aksara Latin, bukan berarti aksara bahasa daerah tidak perlu untuk dilestarikan apalagi dihilangkan karena itu merupakan salah satu aset kebudayaan dan kekayaan bangsa (Baso & Agussalim, 2022). Dengan aset kekayaan bahasa daerah sebanyak itu tentu saja perlu upaya untuk terus melestarikannya agar tetap hidup di masyarakat. Salah satu upaya untuk melestarikan bahasa daerah adalah dengan pemanfaatan teknologi yang saat ini tengah berkembang pesat salah satunya adalah teknologi kecerdasan buatan. Kecerdasan buatan merupakan sebuah teknologi yang dapat mengadopsi kecerdasan yang dimiliki manusia untuk diimplementasikan ke dalam sebuah komputer. Mengembangkan cara berkomunikasi secara digital membantu melestarikan sejarah dan budaya yang sangat penting bagi pengetahuan manusia. TTS (*Text to speech)* dapat mejadi salah satu cara untuk melestarikan bahasa dan membuatnya dapat diakses oleh generasi sekarang dan mendatang.

Text to speech (TTS) adalah jenis teknologi "*read aloud*" yang mampu membaca simbol dan menghasilkan bentuk gelombang suara secara otomatis (Azizah & Adriani, 2020). Sistem TTS secara umum dibagi menjadi dua bagian proses yaitu pertama proses *Natural Language Processing* (NLP) yang akan mengolah teks dalam sistem ini dan *speech synthesis* yang akan mengubah teks menjadi suara. TTS digunakan dalam berbagai aplikasi, termasuk sistem penerjemah, sistem pembacaan teks untuk tunanetra, alat komunikasi bot, mesin penjawab telepon, pengumuman suara sintetis, aplikasi ucapan untuk penyandang disabilitas berbicara, dan aplikasi pendidikan dan permainan.

Fungsi TTS sebagai media pendukung pelestarian budaya sangatlah mungkin karena TTS juga dapat dimanfaatkan untuk kegiatan pembelajaran suatu bahasa bagi orang asing. Suara yang merupakan keluaran dari TTS akan mempermudah seseorang mempelajari pengucapan suatu kata dalam bahasa tertentu.

Pada penelitian ini akan berfokus membangun teknologi TTS khusus untuk bahasa daerah khususnya bahasa Bugis yang dapat membaca teks berbahasa Bugis ontara) tersebut seperti dialek penutur asli agar lebih memudahkan bagi am yang tidak mengetahui bahasa tersebut juga dapat memudahkan dalam mbelajaran bahasa daerah.

Arsitektur Tacotron 2 digunakan untuk membangun sistem TTS ini. Tacotron 2 (Shen et al., 2017) merupakan sistem TTS *end-to-end* yang dikembangkan oleh tim Google pada tahun 2017 menerapkan arsitektur jaringan saraf untuk sintesis ucapan langsung dari teks. Sistem ini terdiri dari jaringan prediksi fitur *recurrent sequence-to-sequence* yang memetakan karakter ke *spectrogram* skala-mel, diikuti oleh model WaveNet yang bertindak sebagai *vocoder* untuk sintesis bentuk gelombang dari *spectrogram* tersebut. Keunggulan Tacotron 2 adalah suara yang dihasilkan sudah mendekati suara alami dari manusia dibanding metode TTS yang telah dikembangkan sebelumnya seperti Parametric, Concatenative dan versi pendahulunya yaitu Tacotron 1, dengan skor MOS (*Mean Opinion Score*) yang dihasilkan 4.526 dari skor *ground truth* atau suara alami manusia yaitu 4.582.

Oleh karena itu pada penelitian ini mengimplementasikan arsitektur Tacotron 2. Namun untuk membuat model TTS dibutuhkan sumber daya yang besar, mulai dari *dataset* yang dibutuhkan dan perangkat komputasi untuk melatih modelnya. Berdasarkan beberapa penelitian salah satunya penelitian Shen et al., (2017) membutuhkan *dataset* suara dengan durasi 24 jam dan dilatih selama beminggu-minggu bahkan memakan waktu bulanan. Dikarenakan sangat membutuhkan waktu yang lama untuk membangun model dari awal untuk TTS bahasa tertentu, salah satu solusi untuk membuat model TTS dari sumber data yang kecil adalah dengan menggunakan metode *transfer learning* untuk membangun model TTS. Memanfaatkan *pre-trained model* dari model TTS bahasa lain yang sudah dilatih dengan data yang besar, dengan begitu cukup dengan *dataset* kecil kita bisa mengambil manfaat dari *pre-trained model* untuk diadaptasikan dengan *dataset* bahasa target.

1.2 Rumusan Masalah

Rumusan masalah berdasarkan latar belakang dijabarkan sebagai berikut:

1. Bagaimana membangun sistem *text to speech* bahasa Bugis menggunakan arsitektur Tacotron 2 dengan metode *transfer learning*?
2. Bagaimana hasil sintesis suara sistem *text to speech* bahasa Bugis?

1.3 Tujuan Penelitian

Tujuan penelitian dijabarkan sebagai berikut:

1. Membangun sistem *text to speech* bahasa Bugis menggunakan arsitektur Tacotron 2 dengan metode *transfer learning*.
2. Mengetahui kualitas hasil sintesis suara dari sistem *text to speech* bahasa Bugis.

1.4 Manfaat Penelitian

Manfaat penelitian dijabarkan sebagai berikut:

1. Sebagai salah satu upaya untuk tetap melestarikan budaya daerah dibidang bahasa daerah.
2. Dapat menjadi salah satu alat bantu dalam proses pembelajaran bahasa daerah.
3. Sebagai referensi bagi penelitian mengenai *text to speech* untuk bahasa daerah yang bersifat *low resource*.

1.5 Ruang Lingkup

Adapun ruang lingkup dalam penelitian ini adalah:

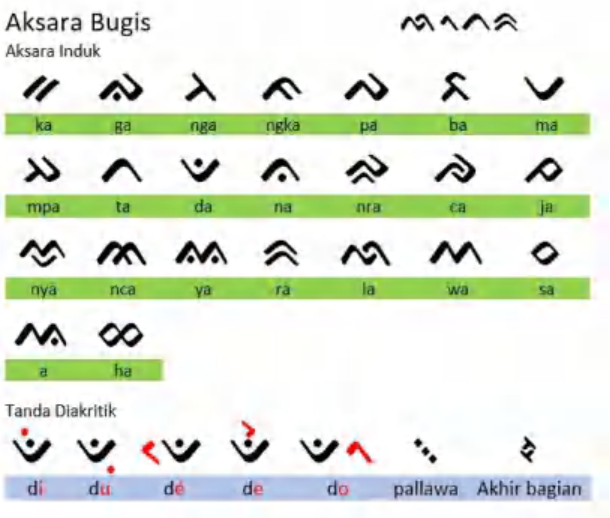
1. Penelitian hanya berfokus pada arsitektur Tacotron 2 dan tidak berfokus pada arsitektur *vocoder.*
2. *Vocoder* yang digunakan adalah WaveGlow.

BAB II
TINJAUAN PUSTAKA

2.1 Bahasa Bugis

Bahasa Bugis merupakan salah satu jenis bahasa daerah yang ada di Sulawesi Selatan dan digunakan oleh suku Bugis yang merupakan salah satu etnis terbesar di Sulawesi Selatan. Bahasa Bugis memiliki huruf tersendiri dalam penulisannya yang dikenal sebagai aksara Lontara.

Aksara Lontara seperti yang ditunjukkan Gambar 1 yang terdiri dari 23 aksara dasar dinamakan *inang sureq* (induk huruf). Setiap konsonan merepresentasikan satu suku kata dengan vokal /a/ yang dapat diubah dengan pemberian diakritik tertentu yang dinamakan *anaq sureq* (anak huruf). *Anaq sureq* ini berjumlah lima buah. Pemberian *anaq sureq* (anak huruf) tertentu dengan posisi tertentu pula maka akan melambangkan suku kata tertentu pula.



Gambar 1. Aksara Lontara
Sumber: https://writingtradition.blogspot.com/

2.2 *Text to Speech*

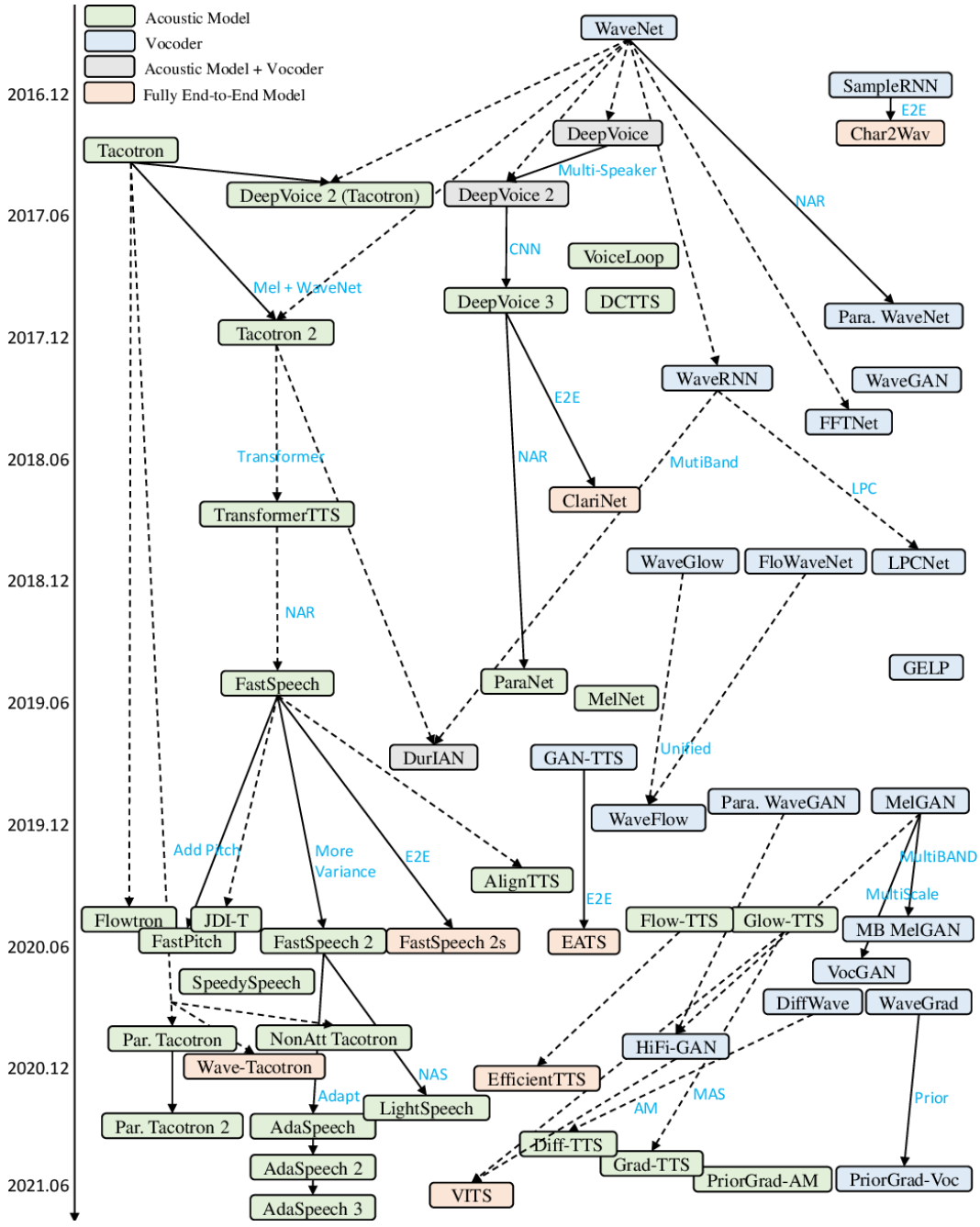
Sistem *text-to-speech* (TTS) sederhananya adalah sistem yang mengubah teks menjadi ucapan dalam bahasa tertentu. TTS adalah teknologi komprehensif yang ın banyak disiplin ilmu seperti akustik, linguistik, pemrosesan sinyal n statistik. Tujuan dibangunnya TTS adalah untuk menghasilkan ucapan

yang mudah dipahami dan tidak dapat dibedakan dengan suara yang dihasilkan antara manusia dan mesin.

Sintesis ucapan memainkan peran penting dalam interaksi manusia dan komputer. Dalam dua puluh tahun terakhir TTS menjadi fokus para peneliti sehingga penerapan TTS semakin meluas seperti pada layanan penjawab telepon komersial, program penerjemah, buku digital, mainan anak hingga perangkat membaca untuk tunanetra dan penyandang cacat lain.

Saat ini metode untuk mengubah teks menjadi ucapan secara umum dibagi menjadi 2 metode, yaitu metode tradisional dan metode sintesis *end-to-end* berbasis *deep learning*. Sistem sintesis ucapan tradisional biasanya mencakup dua modul, *front-end* dan *back-end*. Modul *front-end* menganalisis teks masukan dan mengekstrak informasi linguistik yang dibutuhkan oleh modul *back-end*. Sedangkan sistem sintesis *end-to-end* secara langsung memasukkan teks atau karakter fonetik dan mengeluarkan bentuk gelombang audio. Sehingga tidak membutuhkan pengetahuan linguistik dari ahli, dan dapat menunjukkan gaya pengucapan yang kuat dan kaya serta ekspresi berirama (Li et al., 2021).

WaveNet diusulkan untuk secara langsung menghasilkan bentuk gelombang dari fitur linguistik yang dapat dianggap sebagai model TTS modern berbasis *neural network* pertama (Tan, 2021). Selanjutnya, beberapa model *end-to-end* seperti Tacotron 1/2, Deep Voice 3, dan FastSpeech 1/2 diusulkan untuk menyederhanakan modul analisis teks dan langsung mengambil urutan karakter/fonem sebagai *input*, dan menyederhanakan fitur akustik dengan *spectrogram* dan menggabungkan dengan *vocoder* untuk mengubah *spectrogram* menjadi suara. Kemudian, penelitian terbaru adalah sistem TTS *fully end-to-end* dikembangkan untuk secara langsung menghasilkan bentuk gelombang dari teks tanpa perlu *vocoder* seperti ClariNet, FastSpeech 2s dan EATS. Perkembangan model TTS berbasis *neural network* secara detail dapat dilihat pada Gambar 2 berikut ini:



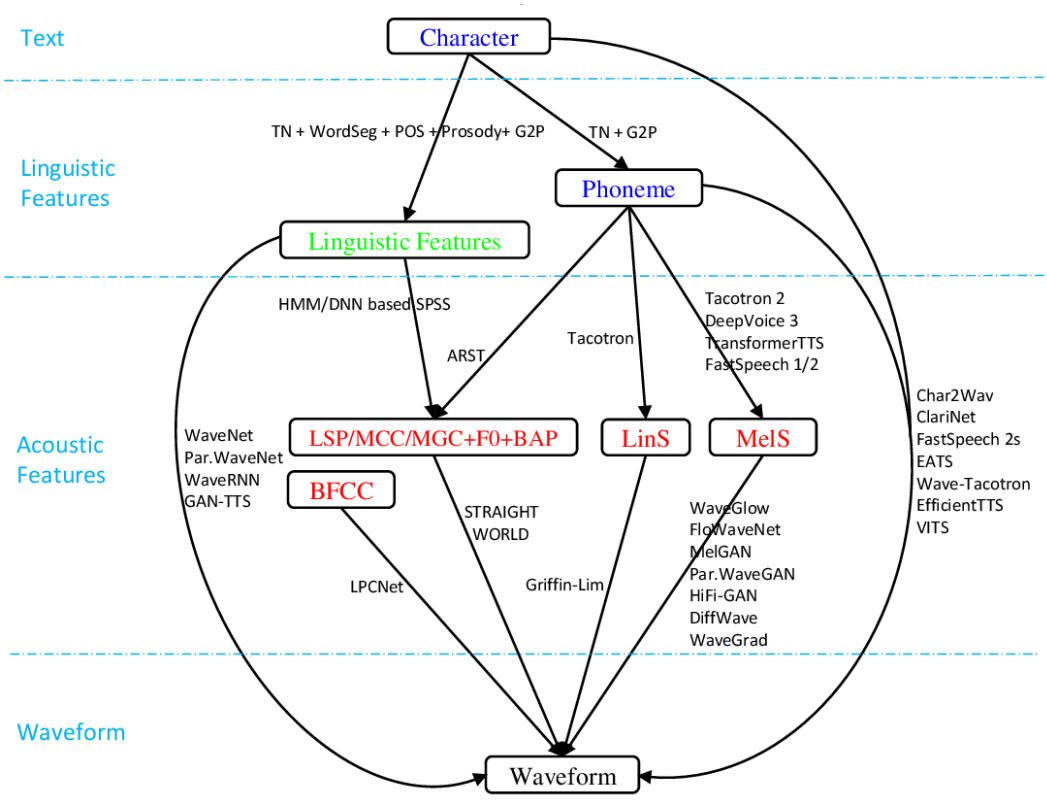
Gambar 2. Evolusi TTS model berbasis *Neural Network*
Sumber: Tan (2021)

Komponen utama dalam sistem TTS modern terdiri dari tiga komponen dasar yaitu modul analisis teks , model akustik dan *vocoder*. Seperti yang ditunjukkan pada Gambar 3, modul analisis teks mengubah urutan teks menjadi fitur linguistik, ustik menghasilkan fitur akustik dari fitur linguistik, kemudian *vocoder* sis bentuk gelombang (*waveform*) dari fitur akustik.

Gambar 3. Komponen dalam *neural* TTS
Sumber: (Tan, 2021)

Gambar 4 menggambarkan detail dari Gambar 3 di atas. Dalam gambar tersebut ada beberapa representasi data dalam proses konversi *text to speech*:

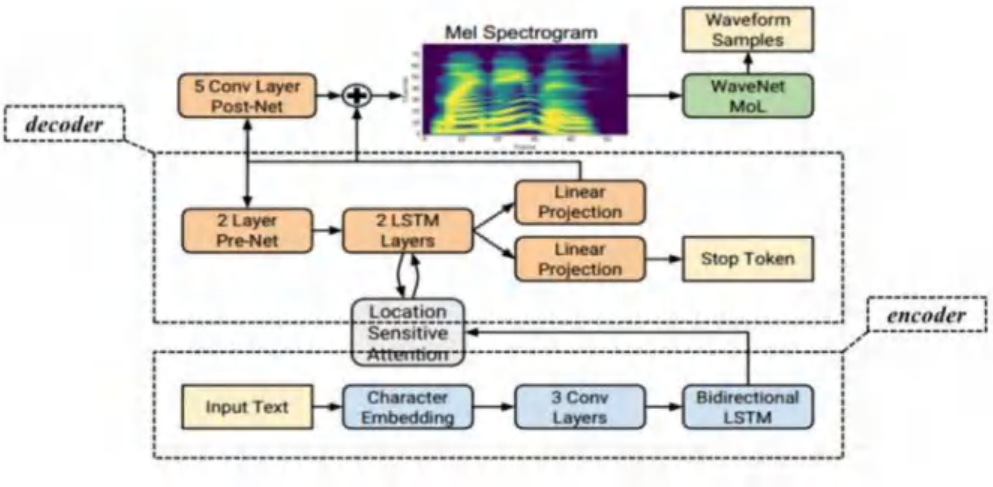
- Teks yang dapat berupa karakter ataupun fonem langsung sebagai *input*.
- Fitur linguistik yang diperoleh melalui analisis teks dan mengandung informasi konteks yang kaya tentang pengucapan dan prosodi.
- Fitur akustik yang merupakan representasi dari bentuk gelombang ucapan. Dalam model TTS *end-to-end*, *mel-spectrogram* biasanya digunakan sebagai fitur akustik, yang diubah menjadi bentuk gelombang menggunakan *vocoder* berbasis *neural network.*
- Bentuk gelombang *speech* (*waveform*) merupakan format akhir yang dihasilkan dari model TTS.



Gambar 4. Taksonomi model TTS berdasarkan *data flows*
Sumber: (Tan, 2021)

2.3 Tacotron 2

Tacotron 2 (Shen et al., 2017) adalah model sintesis ucapan yang diusulkan oleh perusahaan Google pada tahun 2017 dan telah mencapai hasil sintesis yang luar biasa pada kumpulan data ucapan bahasa Inggris. Tacotron 2 memiliki arsitektur model yang lebih ringkas dibanding Tacotron versi sebelumnya, yang pada dasarnya terdiri dari dua komponen. Komponen pertama adalah jaringan prediksi *spectrogram* yang mengambil teks sebagai masukan dan menghasilkan *mel-spectrogram*. Komponen kedua bertindak sebagai *vocoder* dan mengubah *mel-spectrogram* menjadi bentuk gelombang. Dalam publikasi asli Tacotron 2 (Shen et al., 2017), WaveNet digunakan sebagai *vocoder*. Namun saat ini, terdapat banyak pilihan *vocoder* lain yang dapat digunakan. Arsitektur model Tacotron 2 dapat dilihat pada Gambar 5 berikut ini:



Gambar 5. Arsitektur Tacotron 2
Sumber: (Shen et al., 2017)

Fleksibilitas Tacotron 2 yang memproses teks ortografi mentah (tingkat karakter) untuk memprediksi *spectrogram* dari pasangan data membuatnya lebih mudah untuk diterapkan dalam bahasa lain (Azizah & Adriani, 2020). Gambar 6 merupakan skor MOS (*Mean Opinion Score*) dari Tacotron 2 untuk bahasa Inggris yaitu 4.526 yang sangat dekat dengan MOS (4.582) dari rekaman ucapan profesional (*Ground truth*).

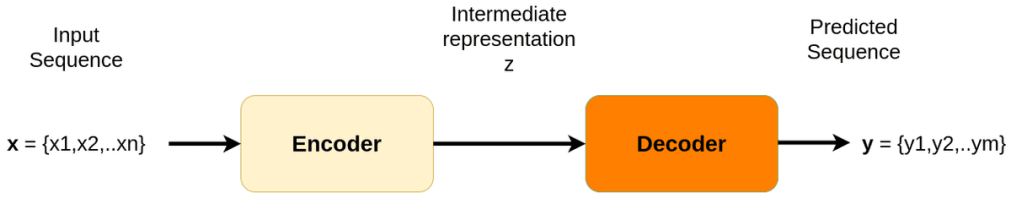
System	MOS
Parametric	3.492 ± 0.096
Tacotron (Griffin-Lim)	4.001 ± 0.087
Concatenative	4.166 ± 0.091
WaveNet (Linguistic)	4.341 ± 0.051
Ground truth	4.582 ± 0.053
Tacotron 2 (this paper)	$\mathbf{4.526 \pm 0.066}$

Gambar 6. Nilai MOS Tacotron 2
Sumber: (Shen et al., 2017)

Tulang punggung Tacotron 2 adalah jaringan *sequence-to-sequence (seq2seq)* yang terdiri dari *encoder* dan *decoder* dengan *attention*. Tujuan *encoder* adalah untuk mengubah teks *input* menjadi *hidden fiture*, yang kemudian digunakan oleh *decoder* berbasis *attention* untuk memprediksi *mel-spectrogram* (Fang et al., 2019). Berikut bagian-bagian penyusun dari model Tacotron 2:

2.3.1 *Sequence to Sequence*

Jaringan *sequence to sequence* dalam *deep learning* adalah arsitektur jaringan saraf yang dirancang untuk memproses data sekuensial, seperti teks, audio dan video, dan menghasilkan keluaran sekuensial lainnya. Model ini biasanya terdiri dari dua komponen utama: *encoder* dan *decoder* seperti pada Gambar 7 berikut ini:



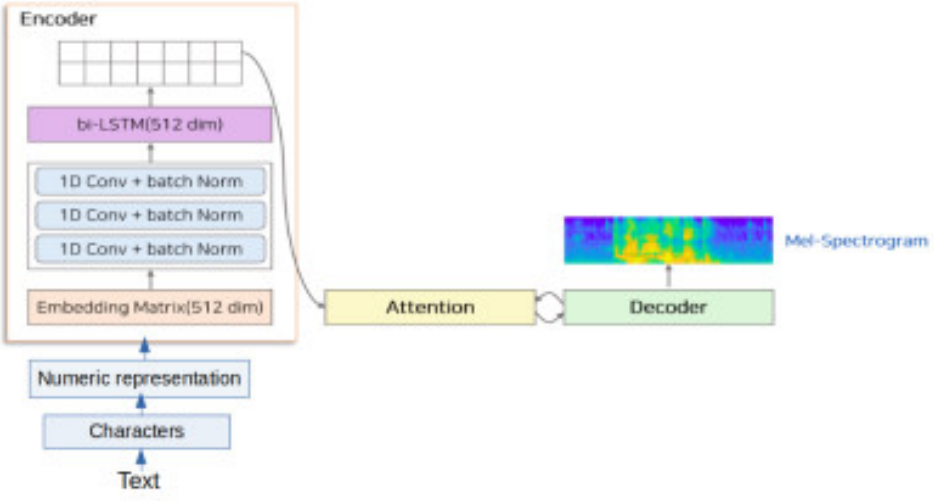
Gambar 7. Komponen dan alur kerja jaringan *seq2seq*
Sumber: https://theaisummer.com/attention/

Encoder memproses data sekuensial dan menghasilkan representasi terkompresi dari data *input*. Biasanya, *encoder* menggunakan jaringan saraf berulang (RNN) seperti *Long Short-Term Memory* (LSTM) atau *Gated Recurrent Unit* (GRU). *Decoder* mengambil representasi yang dihasilkan oleh *encoder* dan menghasilkan *output* sekuensial. *Decoder* juga sering menggunakan RNN, dan *output* dihasilkan langkah demi langkah berdasarkan representasi *encoder* dan ng dihasilkan sebelumnya. Untuk menghubungkan *encoder* dan *decoder*, apa mekanisme yang biasa digunakan, seperti:

- *Attention*: memungkinkan *decoder* untuk "melihat" bagian tertentu dari representasi *encoder* saat menghasilkan setiap elemen *output*.
- *Teacher Forcing*: selama proses *training*, *output* sebenarnya (*ground truth*) pada setiap langkah *decoder* digunakan sebagai *input* untuk langkah berikutnya yang disebut sebagai teknik *teacher forcing*. Ini akan lebih membantu *decoder* belajar menghasilkan *output* yang benar dibandingkan hanya dengan menggunakan data prediksi sebelumnya sebagai *input*.

2.3.2 *Encoder*

Encoder bertanggung jawab untuk mengubah karakter menjadi *hidden vector* yang nantinya akan digunakan oleh *decoder*. *Encoder* Tacotron 2 terdiri dari karakter *embedding matrix* 512 dimensi, 3 *layer* CNN, dan bi-LSTM sebagaimana yang tergambar dalam Gambar 8 berikut ini:

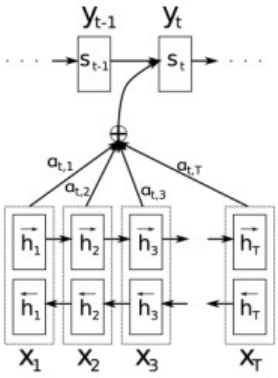


Gambar 8. *Encoder* Tacotron 2
Sumber: (Luel et al., 2022)

Embedding matriks bertugas mengubah *input* yang berupa urutan karakter menjadi *vector embedded* dengan 512 dimensi kemudian melewati tiga lapis *convolution network* (*batch norm* + aktivasi *ReLu*) yang masing-masing berisi 512 kernel konvolusi 5 × 1 dan diteruskan ke satu blok bi-LSTM yang terdiri dari 512 unit (256 setiap arah) dimana ia dikonversi menjadi *encoder feature* atau fitur konteks yang akan diumpan ke *decoder* melalui jaringan *attention.*

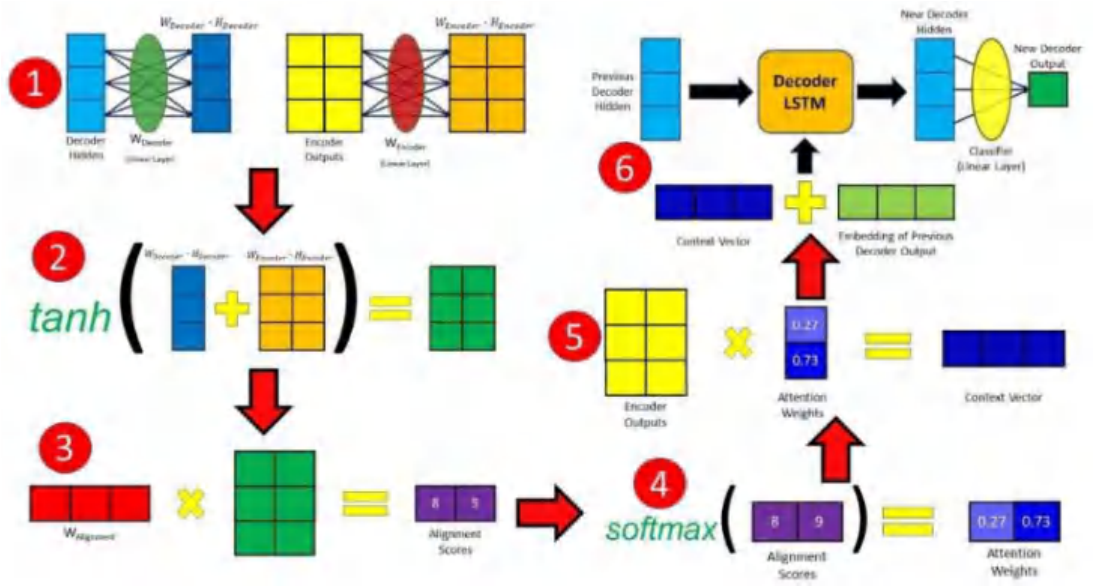
2.3.3 *Attention*

Attention bertanggung jawab untuk mengambil informasi dari *encoder* untuk digunakan oleh *decoder*. Dengan kata lain, mekanisme *attention* dalam *encoder-decoder* adalah mekanisme yang memungkinkan model untuk memfokuskan pada aspek-aspek tertentu dengan memberikan bobot atau "perhatian" kepada bagian-bagian penting dari *input* saat melakukan prediksi atau generasi *output*, hal ini dapat membantu model untuk menghasilkan *output* yang lebih akurat dan koheren. Karena sulitnya arsitektur *encoder-decoder* untuk menghafal rangkaian *input* yang panjang, diperlukan mekanisme *attention* untuk membantu mengatasi kesulitan tersebut. Akibatnya, dalam rangkaian yang panjang, kinerja arsitektur tanpa mekanisme *attention* akan menurun sebab *decoder* hanya mendapat informasi konteks dari hasil akhir seluruh *input encoder* saja, sedangkan dengan adanya *attention* diantara keduanya, maka *decoder* bisa memiliki akses ke informasi konteks setiap *input* teks dari *encoder* melalui *attention* ini. Cara kerja *attention* pada *encoder-decoder* secara umum dapat dilihat pada Gambar 9 yang merupakan mekanisme *attention* diusulkan oleh (Bahdanau et al., 2015) untuk model mesin translasi, biasa disebut *Additive Attention*. Model Tacotron 2 menggunakan varian *Location Sensitive Attention* (Shen et al., 2017) yang merupakan pengembangan dari mekanisme *Additive Attention* (Bahdanau et al., 2015).



Gambar 9. Penerapan *attention* pada jaringan *seq2seq*
Sumber: (Bahdanau et al., 2015)

Untuk mengetahui cara kerja *Location Sensitive Attention* yang digunakan 2, maka tentu saja harus memahami alur *Additive Attention* untuk ıui dimana letak perbedaannya. Detail yang terjadi dalam *Additive* diilustrasikan pada Gambar 10 berikut ini:



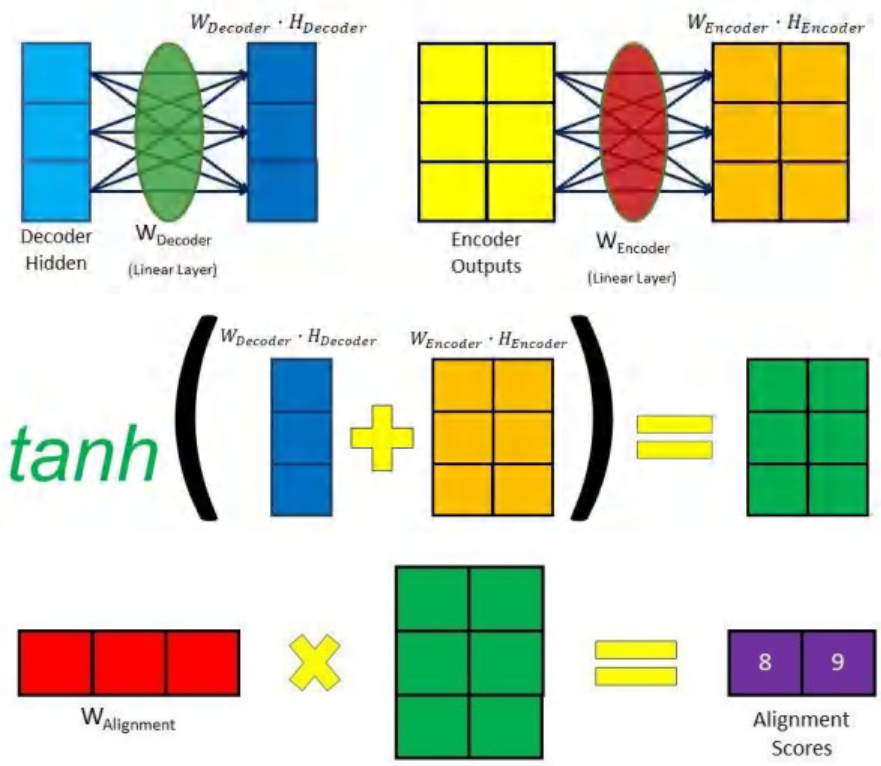
Gambar 10. Ilustrasi cara kerja *attention*
Sumber: (Loye, 2019)

1. Menghitung *alignment score*

Persamaan untuk menghitung *alignment score* sebagai berikut:

$$s_{t,i} = \omega^T tanh(W_1 d_{t-1} + W_2 h_i) \quad (1)$$

Dimana $s_{t,i}$ adalah fungsi untuk menghitung *alignment score*. W_1, W_2, ω adalah parameter yang dipelajari. d_{t-1} adalah *hidden state* yang dihasilkan oleh *decoder* pada langkah waktu sebelumnya, sedangkan h_i adalah *hidden state* dari *encoder*. *Hidden state decoder* dan *output encoder* akan dilewatkan melalui lapisan linier masing-masing dan memiliki bobot tersendiri yang dapat dilatih. Setelah itu, mereka akan dijumlahkan sebelum melewati fungsi aktivasi *tanh*. *Hidden state decoder* akan ditambahkan ke setiap *output encoder*. Terakhir, vektor resultan dari beberapa langkah sebelumnya akan menjalani perkalian matriks dengan vektor yang dapat dilatih, sehingga diperoleh vektor skor penyelarasan akhir yang menyimpan *alignment score* untuk setiap *output encoder*. Jika melihat pada Gambar 10, proses menghitung *alignment score* dilakukan pada langka 1 hingga 3. *Zoom in* untuk langkah 1 hingga 3 diperlihatkan pada Gambar 11 berikut ini:



Gambar 11. Ilustrasi langkah perhitungan *alignment score*
Sumber: (Loye, 2019)

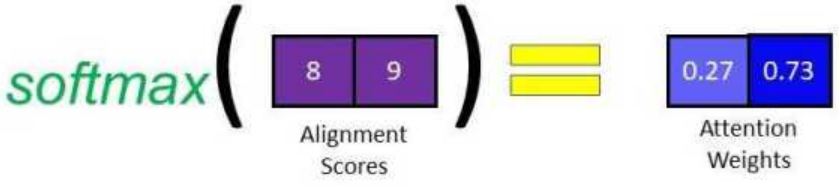
2. Menghitung *attention weight*

Selanjutnya *alignment score* digunakan untuk menghitung *attention weight* $\alpha_{t,i}$ dengan menggunakan fungsi *softmax*:

$$\alpha_{t,i} = \frac{\exp(score_{t,i})}{\sum_{i=1}^{n} \exp(score_{t,i})} \quad (2)$$

$$\alpha_t = [\alpha_{t,1}, \alpha_{t,2}, \alpha_{t,3}, \ldots\ldots \alpha_{t,n}] \quad (3)$$

Fungsi *softmax* akan menyebabkan nilai-nilai dalam vektor berjumlah 1 dan setiap nilai individual akan berada di antara 0 dan 1, oleh karena itu dapat mewakili bobot yang dimiliki setiap *input* pada langkah waktu tersebut seperti contoh hasil *attention weights* pada Gambar 12 di bawah ini yang juga merupakan langkah nomor 4 pada Gambar 10.



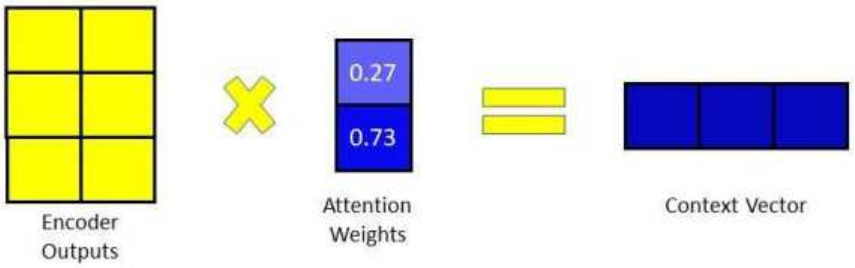
Gambar 12. Ilustrasi perhitungan *attention weight*
Sumber: (Loye, 2019)

3. Menghitung *context vector*

Jika dilihat pada Gambar 10, proses ini merupakan proses kelima. Setelah *attention weight* didapatkan, maka akan digunakan untuk menghitung *context vector* (c_t):

$$c_t = \sum \alpha_{t,i} \, h_i \tag{4}$$

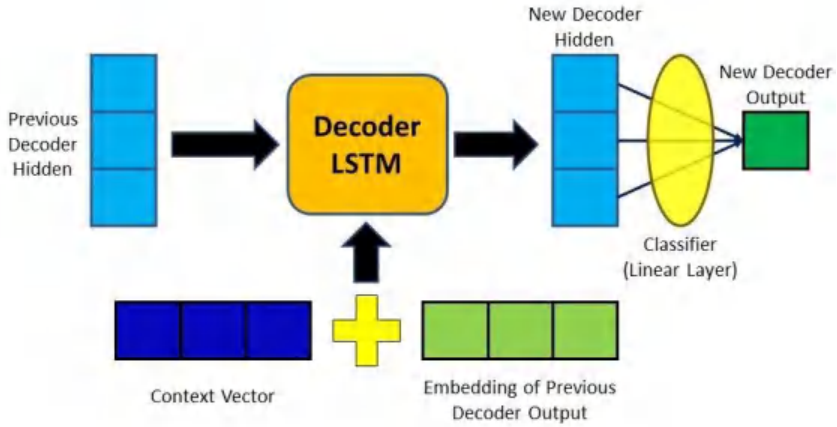
Untuk menghasilkan *context vector* dilakukan perkalian *attention weight* dengan keluaran *encoder* seperti Gambar 13. Karena fungsi *softmax* pada langkah sebelumnya, jika skor elemen *input* tertentu mendekati 1, efeknya dan pengaruhnya terhadap keluaran *decoder* akan diperkuat, sedangkan jika skornya mendekati 0, pengaruhnya akan dilemahkan.



Gambar 13. Ilustrasi perhitungan *context vector*
Sumber: (Loye, 2019)

4. *Context vector* menjadi *input decoder*

Terakhir, *context vector* yang telah hasilkan kemudian akan digabungkan dengan *output decoder* sebelumnya. Kemudian dimasukkan ke dalam sel *decoder* untuk menghasilkan *hidden state* baru yang akan digunakan lagi untuk proses perulangan berikutnya seperti yang diilustrasikan pada Gambar 14 berikut ini:

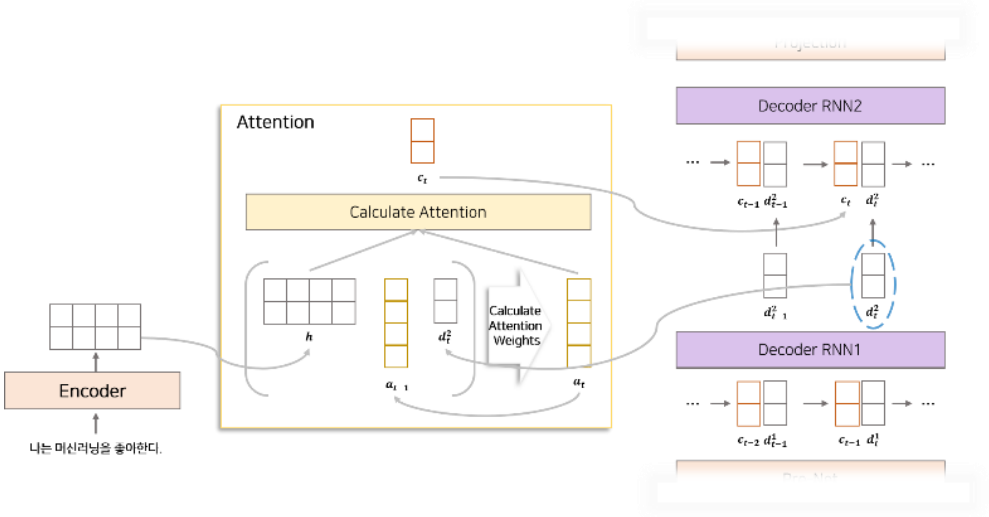


Gambar 14. Penggabungan *context vector* dengan *output decoder*
Sumber: (Loye, 2019)

Sementara itu perbedaan antara *Additive Attention* dan *Location Sensitive Attention* yang digunakan pada Tacotron 2 terletak di perhitungan *attention* score, untuk varian *Location Sensitive Attention* itu sendiri menambahkan informasi *attention weight*, dimana persamaan *Location Sensitive Attention* sebagai berikut:

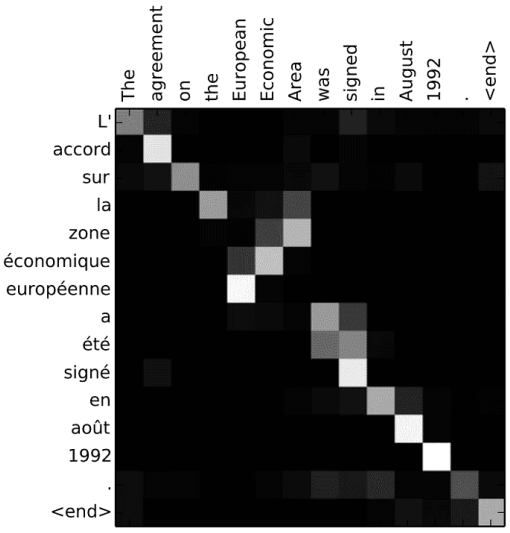
$$s_{t,i} = \omega^T tanh\left(W_1 d_{t-1} + W_2 h_i + W_3 f_{t,i}\right) \quad (5)$$

Jabaran dari $W_3 f_{t,i}$ adalah $f_i = F * \alpha_{i-1}$, dengan W_3 adalah parameter yang dipelajari, F adalah *convolution filter*, α_{i-1} merupakan bobot *attention* sebelumnya, sehingga model dapat fokus pada posisi tertentu dari vektor konteks yang dihasilkan oleh *encoder*. Proses *Location Sensitive Attention* diilustrasikan dalam Gambar 15 berikut ini:



Gambar 15. Alur proses *Location Sensitive Attention*
Sumber: (Jounghee, 2020)

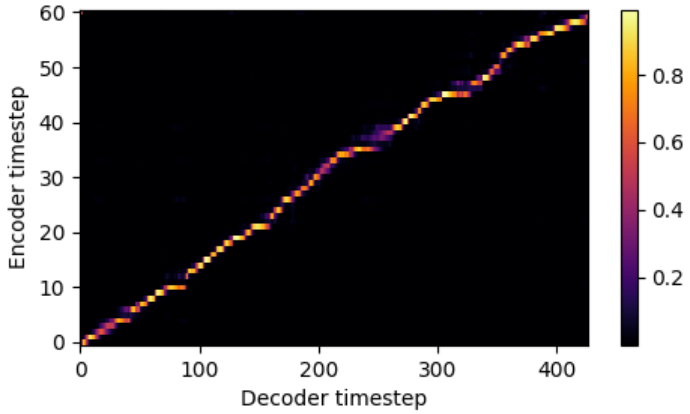
Dalam *paper* aslinya (Bahdanau et al., 2015) mengenai *attention* pada mesin translasi, terdapat visualisasi sederhana dari *attention weight* ($\alpha_{t,i}$). Visualisasi dari *attention weight* ini biasa disebut grafik *alignment*. Grafik *alignment* memberikan wawasan tentang bagaimana model memproses dan memahami hubungan antara token dalam urutan *input* dan *output*. Dalam model TTS berbasis *attention*, seperti Tacotron 2, grafik *alignment* memberikan gambaran visual tentang bagaimana model memperhatikan atau fokus pada representasi tersembunyi (*hidden* ang dihasilkan oleh *encoder* selama menghasilkan *output*. Gambar 16 n grafik *alignment* dari *paper* Bahdanau et al. (2015).



Gambar 16. Grafik *alignment* untuk terjemahan bahasa Inggris ke bahasa Prancis
Sumber: (Bahdanau et al., 2015)

Pada gambar di atas terlihat pola diagonal dari grafik yang menandakan seberapa baik model dalam memperhatikan kata Prancis yang sesuai dari posisi yang sama pada kata bahasa Inggris. Sumbu x dan sumbu y masing-masing plot sesuai dengan kata-kata dalam kalimat sumber (Inggris) dan terjemahan yang dihasilkan (Prancis). Setiap piksel menunjukkan bobot ($\alpha_{t,i}$) dari anotasi kata sumber ke-i untuk kata target ke-t (lihat Persamaan (2)) dalam skala abu-abu (0: hitam, 1: putih).

Tidak jauh berbeda dengan grafik *alignment* pada model TTS Tacotron 2 yang ditunjukkan pada Gambar 17 berikut ini:



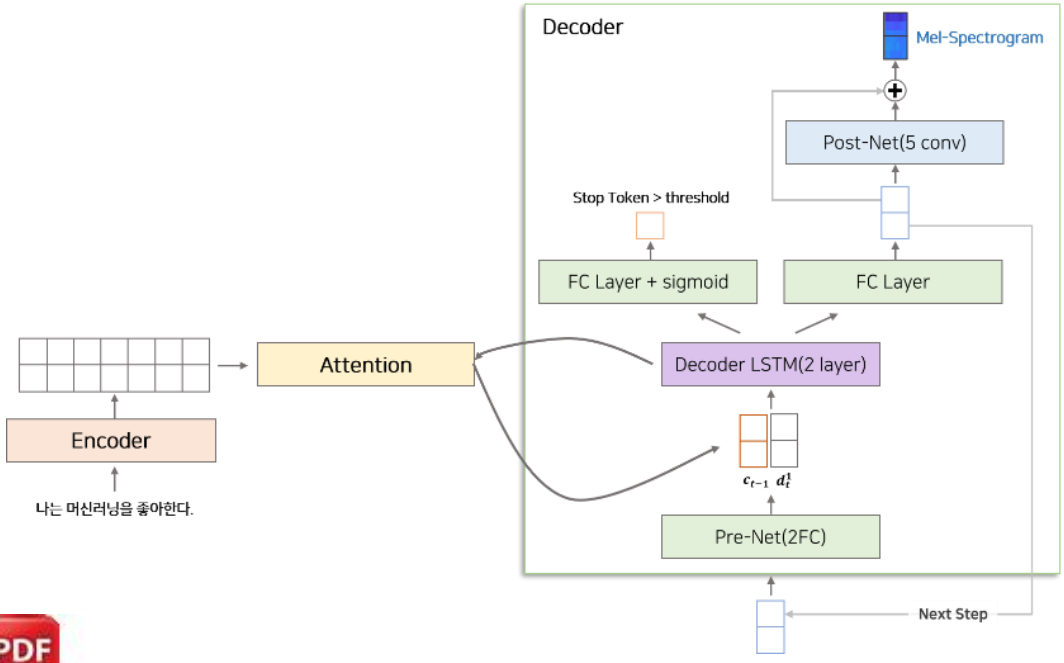
Gambar 17. Contoh grafik *alignment* Tacotron 2

adalah *decoder timestep* (langkah waktu *decoder*) sedangkan sumbu y *encoder timestep* (langkah waktu *encoder*) dan intensitas warna kan bobot *alignment* (0: hitam, 1: putih). Bobot *alignment* menunjukkan

tingkat fokus *decoder* pada *encoder hidden state* pada waktu tertentu. Adapun jumlah *timestep* dalam *encoder* biasanya sama dengan jumlah karakter dalam teks yang diproses. Sedangkan *decoder timestep* tidak langsung berkorelasi dengan jumlah karakter teks *input*, karena setiap *timestep* pada *decoder* menghasilkan representasi audio dari teks *input*. *Decoder* dapat menghasilkan jeda, penundaan, atau memperpanjang suara untuk menyesuaikan dengan intonasi, ritme, atau durasi yang diinginkan. Oleh karena itu, jumlah *decoder timestep* mungkin lebih banyak atau lebih sedikit dari jumlah karakter teks *input*, tergantung pada kompleksitas suara yang ingin dihasilkan dan faktor-faktor lainnya seperti kecepatan bicara, intonasi dan sebagainya.

2.3.4 *Decoder*

Dalam Tacotron 2, *decoder* adalah jaringan saraf berulang yang bersifat *autoregresif* yang bertanggung jawab untuk menghasilkan representasi *mel-spectrogram* secara berurutan menggunakan fitur penyelarasan yang diperoleh melalui *attention* dan informasi *mel-spectrogram* yang dihasilkan pada titik waktu sebelumnya. *Decoder* Tacotron 2 sebagaimana Gambar 18 terdiri dari *Pre-Net*, LSTM, *Projection Layer*, dan *Post-Net*.



Gambar 18. *Decoder* Tacotron 2
Sumber: (Jounghee, 2020)

Pre-Net terdiri dari *2 fully connected layers* (256 dim) + ReLU. Ketika *mel-spectrogram* yang dihasilkan dari titik sebelumnya masuk ke *input decoder* akan melewati *Pre-Net* terlebih dahulu. *Pre-net* adalah bagian yang berperan sebagai *bottle neck* yang menyaring informasi penting.

Decoder LSTM terdiri dari dua lapisan LSTM *uni-directional*. Vektor yang dihasilkan melalui *Pre-Net* dan titik waktu sebelumnya $(t-1)$ digabungkan dengan *context vector* (c_{t-1}) dari *attention* yang kemudian melewati *Decoder* LSTM. *Decoder* LSTM menggunakan informasi dari *attention* dan informasi yang dihasilkan dari *Pre-Net* untuk menghasilkan *output* untuk *timestep* tertentu (t).

Pada setiap titik waktu yang dibuat oleh *decoder* LSTM (t) vektor diteruskan ke dua jalur yaitu jalur untuk menghitung probabilitas kondisi stop dan jalur untuk menghasilkan *mel-spectrogram*. Jalur untuk menghitung probabilitas kondisi stop adalah dengan mengambil vektor yang dihasilkan dari *decoder* LSTM pada setiap titik waktu dan dimasukkan ke *fully connected layer* kemudian menggunakan fungsi *sigmoid* dan mengubahnya menjadi probabilitas antara 0 dan 1. Probabilitas ini sesuai dengan kondisi stop, dan jika ambang batas yang ditetapkan terlampaui, ia bertanggung jawab untuk menghentikan proses generasi *mel-spectrogram*.

Post-Net terdiri dari lima lapisan konvolusi 5D. Lapisan konvolusi memiliki 1 filter dan ukuran kernel 512×5. Vektor mel yang dihasilkan pada langkah sebelumnya melewati *Post-Net* dan struktur *Residual Connection*. *Post-Net* bertanggung jawab untuk memperbaiki atau memperhalus representasi *mel-spectrogram* dan meningkatkan kualitas *mel-spectrogram*, yang merupakan hasil akhir dari Tacotron 2. Terakhir, untuk menghasilkan suara dari *mel-spectrogram* merupakan tugas dari *vocoder*.

2.3.5 *Vocoder*

Dalam sistem TTS, *vocoder* berperan sebagai komponen akhir yang menyintesis suara manusia yang realistis dari representasi akustik yang dihasilkan oleh model pemrosesan sebelumnya. *Vocoder* adalah sejenis konverter yang struksi bentuk gelombang ucapan dari fitur akustik seperti *spectrogram* et al., 2020). Secara sederhana, *vocoder* pada TTS ibarat "alat musik" mainkan partitur musik (representasi akustik yang dalam hal ini adalah

mel-spectrogram yang dihasilkan oleh model Tacotron 2) menjadi alunan nada yang dapat didengar (suara manusia). *Vocoder* berbeda dari prediktor *spectrogram*, dimana dia tidak sepenuhnya bergantung pada bahasa masukan. Peran utamanya adalah mengembalikan representasi *spektral* ke dalam domain waktu. Dengan demikian *vocoder* dianggap bahasa-agnostik yang bisa digunakan untuk bahasa apa saja tanpa *training* ulang dengan bahasa tersebut (Favaro et al., 2009).

Vocoder umumnya terbagi dalam dua kategori yaitu *parametric vocoder* dan *waveform vocoder* atau vocoder berbasis *neural network* (Tan, 2021). Beberapa contoh *parametric vocoder Linear Predictive Coding* (LPC), *Code-Excited Linear Prediction* (CELP), STRAIGHT, WORLD dan masih banyak lagi. Untuk jenis *vocoder* berbasis *neural network* ada WaveNet, WaveGlow, WaveGAN, Hifi-GAN, MelGAN dan masih banyak lagi. Adapun pemilihan *vocoder* tergantung kebutuhan spesifik dari TTS yang dibangun. Banyak faktor yang perlu diperhatikan seperti kualitas suara yang dihasilkan, kompleksitas komputasi, kemudahan penggunaan serta biaya bisa menjadi pertimbangan dalam pemilihan *vocoder*. Tacotron 2 sendiri dalam publikasinya memilih WaveNet sebagai *vocoder* pelengkap sistem TTSnya.

2.3.6 *Loss Function*

Tacotron 2 menggunakan beberapa *loss function* selama *training*. Dua *loss function* utama yang umum digunakan dalam Tacotron 2 adalah:

1. *Mel Spectrogram Loss*

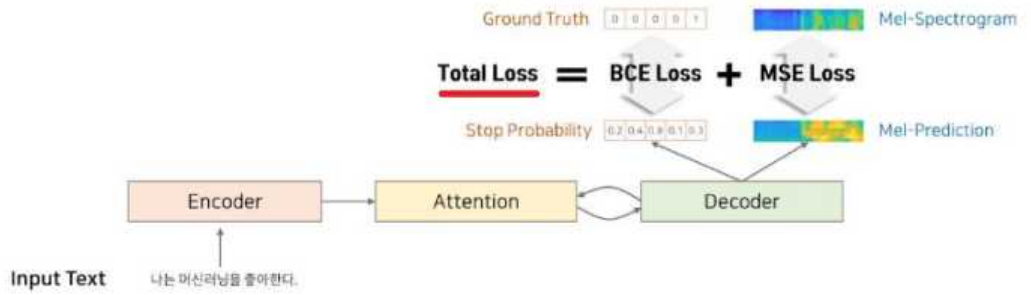
   Loss function ini mengukur perbedaan antara *mel-spectrogram* yang diprediksi oleh model dan *mel-spectrogram* aktual dari suara dalam *dataset training*. Tujuan utamanya adalah untuk memastikan bahwa *mel-spectrogram* yang dihasilkan oleh model mendekati *mel-spectrogram* target sebanyak mungkin. Salah satu metrik yang umum digunakan di sini adalah *Mean Squared Error* (MSE).

2. *Stop Token Loss*

   Loss function ini digunakan untuk melatih model agar dapat memprediksi kapan urutan teks berakhir. Tacotron 2 memproses teks secara urutan, dan prediksi *stop token* membantu model untuk mengetahui kapan

harus berhenti menghasilkan suara. *Stop token loss* biasanya dihitung menggunakan fungsi seperti *Binary Cross Entropy* (BCE).

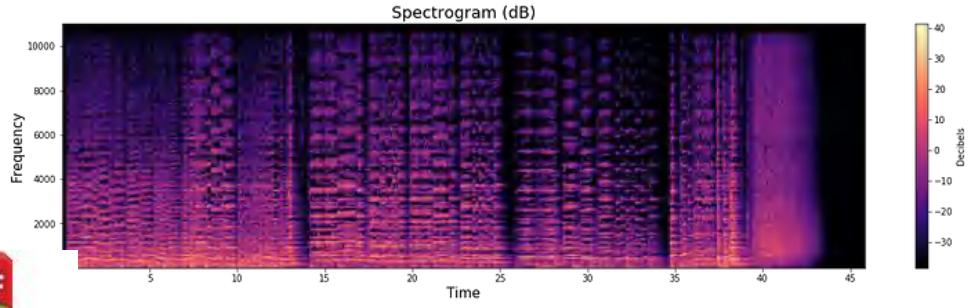
Model secara keseluruhan dilatih dengan mengkombinasikan kedua *loss function* tersebut (Shen et al., 2017) sebagaimana yang diilustrasikan pada Gambar 19. Dengan menggunakan kedua *loss function* ini selama *training*, Tacotron 2 dapat mempelajari representasi yang baik dari teks menjadi suara dengan meminimalkan kesalahan antara *mel-spectrogram* yang dihasilkan dan *mel-spectrogram* target, serta mengontrol penghasilan suara dengan akurat pada setiap langkah waktu.



Gambar 19. *Loss function* Tacotron 2
Sumber: https://kabsung3.tistory.com/

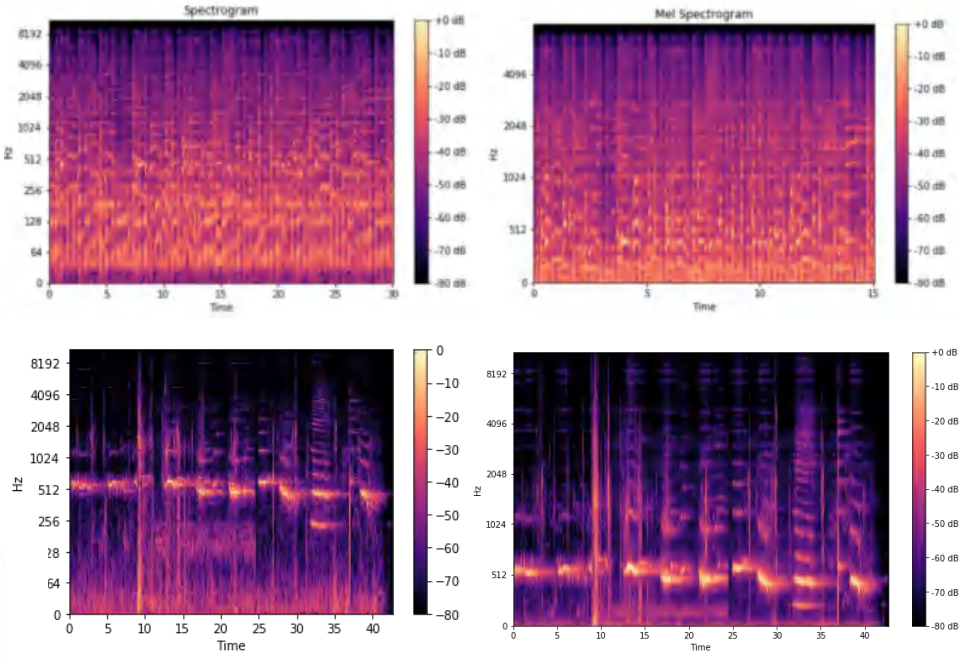
2.4 *Mel-Spectrogram*

Mel-spectrogram merupakan salah satu varian dari *spectrogram*, dimana definisi *spectrogram* itu sendiri adalah sebuah representasi visual dari frekuensi dan intensitas sinyal (amplitudo) suara sepanjang waktu. Ia digambarkan sebagai sebuah grafik seperti Gambar 20 dengan sumbu horizontal (x) mewakili waktu dan sumbu vertikal (y) mewakili frekuensi, intensitas bunyi atau amplitudo digambarkan dengan intensitas warna, dimana warna yang lebih terang menunjukkan intensitas yang lebih tinggi.



Gambar 20. *Spectrogram*
Sumber: (Lewis, 2021)

Perbedaan *spectrogram* biasa dengan *mel-spectrogram* adalah dalam cara merepresentasikan frekuensi. *Spectrogram* biasa menggunakan skala frekuensi linier, yang berarti bahwa setiap unit frekuensi pada sumbu y memiliki jarak yang sama. Hal ini dapat membuat sulit untuk membedakan antara frekuensi yang dekat satu sama lain, terutama pada frekuensi yang tinggi. Sedangkan *mel-spectrogram* menggunakan skala mel untuk sumbu frekuensinya, yaitu skala frekuensi yang tidak linier, di mana jarak antara frekuensi pada skala frekuensi rendah diberi jarak lebih besar, sedangkan frekuensi tinggi dikompresi. Perbedaan keduanya seperti yang terlihat pada Gambar 21. Representasi *mel-spectrogram* ini lebih sesuai dengan cara manusia mendengar, di mana perubahan nada di frekuensi rendah lebih mudah dibedakan dibandingkan dengan frekuensi tinggi. Misalnya jika mendengar pasangan suara berbeda antara 100Hz dan 200Hz, 1000Hz dan 1100Hz, meskipun perbedaan frekuensi sebenarnya antar masing-masing pasangan persis sama 100Hz, namun pasangan 100Hz dan 200Hz akan terdengar lebih jauh perbedaannya dibandingkan pasangan 1000Hz dan 1100Hz yang ternyata lebih sulit menemukan perbedaan antara keduanya. Begitulah cara manusia memandang frekuensi, kita mendengarnya pada skala logaritmik daripada skala linier begitupun dengan persepsi manusia tentang amplitudo suara, kita mendengarnya secara logaritmik daripada linier.



Gambar 21. Perbedaan gambar *spectrogram* dan *mel-spectrogram*
Sumber: (Gartzman, 2019; Roberts, 2020)

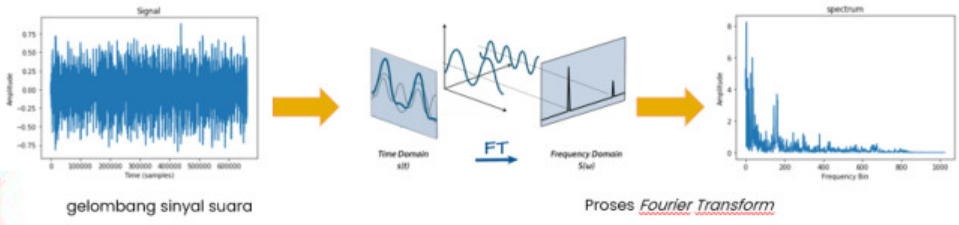
Dengan menggunakan skala mel maka *mel-spectrogram* lebih merepresentasikan cara manusia mendengar dan membedakan suara oleh karena itu lebih cocok digunakan dalam aplikasi TTS dan pengolahan suara lainnya. Untuk mengubah audio menjadi *spectrogram* ataupun *mel-spectrogram* banyak menggunakan algoritma STFT (*Short Time Fourier Transform*). Untuk mengekstrak *mel-spectrogram* dari audio diperlukan total tiga langkah yaitu:

1. *Short-Time Fourier Transform* (STFT)
2. Penskalaan mel dengan *mel filterbanks*
3. Transformasi log

Pertama sinyal dibagi menjadi bagian-bagian kecil yang disebut *windowing* kemudian diterapkan analisis *Fourier Transform* (FT) ke masing-masing *window*, hasilnya akan menjadi *spectrogram*. Langkah kedua adalah menerapkan fungsi *nonlinier* yang disebut *mel-filter bank* ke *spectrogram* untuk memperluas area frekuensi rendah dan mengecilkan area frekuensi tinggi untuk digunakan sebagai fitur. Proses ini akan mengubah fitur menjadi skala yang mudah dipahami oleh manusia untuk menghasilkan ucapan yang lebih jelas. Setelah itu, ketika log diambil dan penskalaan log dilakukan pada area amplitudo, hasil akhirnya disebut *mel-spectrogram*.

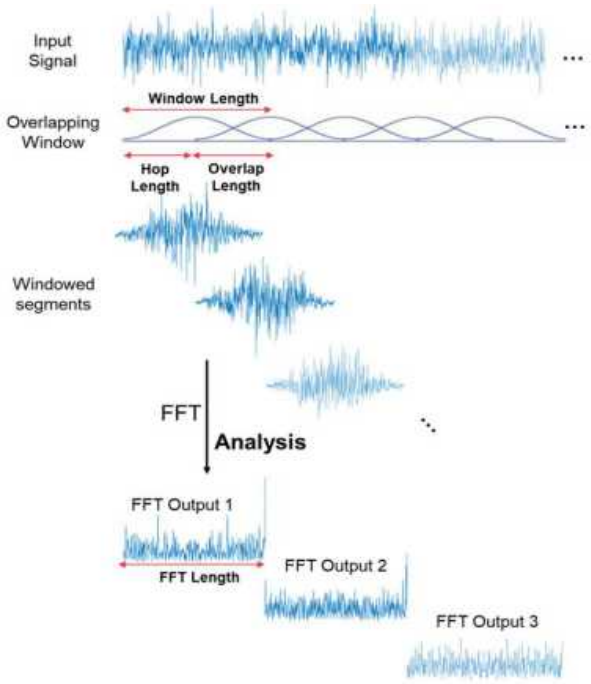
2.4.1 STFT

Short-Time Fourier Transform (STFT) sebenarnya merupakan penggabungan *Fourier Transform* (FT) dan analisis *windowing*. FT adalah teknik matematis yang digunakan untuk mengubah sinyal dari domain waktu (seperti detik) ke domain frekuensi (seperti *Hertz*). Ini mengungkapkan kontribusi frekuensi yang berbeda terhadap sinyal tersebut. Singkatnya digambarkan pada Gambar 22 berikut ini:



Gambar 22. Proses FT

Namun, FT hanya memberikan informasi frekuensi secara keseluruhan, tanpa mempertimbangkan bagaimana frekuensi tersebut berubah seiring waktu. Di situlah *windowing* berperan. STFT membagi sinyal menjadi segmen-segmen pendek yang tumpang tindih, disebut *windows*. FT diterapkan pada masing-masing *window* secara terpisah. Ini memberikan semacam "*snapshot*" frekuensi dalam *window* tersebut. Bekerja diseluruh sinyal dengan cara ini, untuk mendapatkan gambaran tentang bagaimana frekuensi berubah seiring waktu seperti yang digambarkan pada Gambar 23 berikut ini:



Gambar 23. Proses STFT
Sumber: (Doshi, 2021)

Jadi, STFT pada dasarnya adalah FT yang diterapkan pada masing-masing *window segment*. Dengan menggabungkan kekuatan FT dan analisis *windowing*, STFT dapat mengungkapkan:

- Frekuensi yang ada dalam sinyal.
- Amplitudo dari frekuensi tersebut.
- Kapan dan bagaimana frekuensi tersebut berubah seiring waktu.

STFT sangat berguna untuk menganalisis sinyal yang frekuensinya berubah-
erti ucapan, musik, dan banyak sinyal alam lainnya.

2.5 WaveGlow

WaveGlow (NVIDIA, 2018) adalah model sintesis suara yang dikembangkan oleh NVIDIA, mampu menghasilkan audio baru dari *mel-spectrogram*. Ini adalah salah satu dari beberapa model yang digunakan untuk menghasilkan suara manusia yang realistis dari teks dalam konteks sintesis ucapan dengan menggunakan *deep learning* dan teknik terkait. WaveGlow menggunakan arsitektur generatif yang dikenal sebagai WaveNet, yang dikembangkan oleh DeepMind, dan dilatih dengan menggunakan data suara manusia untuk menghasilkan gelombang suara yang berkualitas tinggi. Metode ini memungkinkan model untuk menghasilkan suara yang sangat mirip dengan suara manusia yang sebenarnya.

Keunggulan utama dari WaveGlow adalah kemampuannya untuk menghasilkan suara yang alami dan mudah dipahami oleh manusia. Dibandingkan model lain, WaveGlow adalah tergolong efisien dalam hal waktu dan komputasi, sehingga cocok untuk berbagai aplikasi salah satunya TTS.

2.6 *Transfer learning*

Training yang dimulai dari awal untuk setiap tugas dapat menjadi mahal khususnya pada bidang TTS, karena memerlukan kumpulan data yang besar, memberi label, menemukan arsitektur yang optimal, menyetel parameter, dan mengevaluasi hasil untuk tugas yang diberikan. Oleh karena itu, jauh lebih baik untuk menggunakan pengetahuan sebelumnya yang diperoleh dari tugas lain yang biasa disebut dengan metode *transfer learning.*

Speech syntesis berbasis *deep neural network* membutuhkan data yang sangat besar dalam proses *training* untuk menghasilkan ucapan yang mudah dipahami. TTS berbasis Tacotron 2 untuk domain Indonesia menghasilkan suara yang dapat diterima dan dimengerti oleh pendengaran manusia setidaknya membutuhkan minimal 10 jam data *training* dan data kurang dari tiga jam tidak dapat menghasilkan ucapan yang dapat dipahami (Azizah & Adriani, 2020). Salah satu strategi yang menarik untuk memecahkan masalah sumber daya yang rendah yang ı dalam berbagai penelitian adalah metode *transfer learning*.

ah satu manfaat menggunakan *transfer learning* adalah dapat mengurangi engumpulkan data *training* untuk membangun kembali model dari awal.

Penggunaan *transfer learning* pertama kali yaitu pada bidang visi komputer dengan menggunakan ImageNet (Ezen-can, 2020).

Metode *transfer learning* adalah metode penggunaan kembali bobot lapisan tersembunyi dari jaringan saraf dalam yang telah dilatih sebelumnya untuk tugas yang berbeda. Salah satu teknik dalam *transfer learning* ini ada yang disebut *fine-tuning*, dimana proses *fine-tuning* ini memungkinkan model untuk menyesuaikan bobot dan parameter lainnya untuk menyesuaikan dengan karakteristik *dataset* baru. Proses *transfer learning* dimulai dengan menggunakan model yang telah dilatih sebelumnya pada tugas atau domain yang mirip dengan tugas yang diinginkan, dan kemudian melanjutkan dengan *fine-tuning* model tersebut pada *dataset* spesifik tugas yang baru. Dengan cara ini, *transfer learning* memberikan pemahaman awal yang kuat dari data sumber, sementara *fine-tuning* memungkinkan model untuk menyesuaikan diri dengan data target yang lebih spesifik. Dengan begitu kita bisa mengambil manfaat dari model yang telah dilatih dengan data yang besar untuk digunakan dalam membangun model baru dengan data yang lebih sedikit dan waktu *training* yang lebih singkat. Cara kerja *transfer learning* TTS pada dasarnya suara yang dipelajari dari model yang sudah ada sebelumnya dihilangkan, sementara karakteristik linguistik dari suara tersebut tetap utuh. Oleh karena itu, karena karakteristik linguistik dari model yang sudah ada sebelumnya dapat ditransfer ke penutur lain, sehingga waktu yang dibutuhkan model baru untuk mencapai konvergensi menjadi berkurang (Terblanche et al., 2023).

Terkait pemilihan data sumber bahasa dari rumpun yang sama sebagai acuan untuk *training* data, berdasarkan penelitian terbukti bahwa rasio data sumber dari rumpun bahasa yang sama terhadap total data sumber tidak memengaruhi secara signifikan. Oleh karena itu, rumpun bahasa atau paling tidak metode klasifikasinya yang digunakan di sini, bisa tidak terlalu dipertimbangkan sebagai kriteria untuk memilih bahasa sumber. Klasifikasi rumpun bahasa, meskipun digunakan oleh banyak penelitian, terbukti tidak efektif sebagai kriteria untuk memilih bahasa

Do et al., 2021).

2.7 Metode Evaluasi Kualitas Suara TTS

Pada prinsipnya terdapat metode subjektif dan objektif untuk mengevaluasi hasil sintesis TTS. Namun pada penelitian ini hanya menggunakan metode subjektif. Seperti namanya, evaluasi subjektif bekerja dengan membiarkan beberapa evaluator untuk menilai ucapan langsung melalui telinga manusia. Berikut beberapa metode evaluasi yang digunakan dalam penelitian ini:

1. MOS (*Mean Opinion Score*)

   Diantara banyak metode evaluasi yang diusulkan oleh *International Telecommunication Union* (ITU), MOS yang paling banyak digunakan untuk mengevaluasi hasil TTS. MOS adalah metrik subjektif yang digunakan untuk mengukur kualitas suara secara keseluruhan yang dihasilkan oleh model TTS berdasarkan persepsi manusia. Setiap evaluator menilai hasil sintesis dengan sistem 5 poin. Skor dari 1 sampai 5, mewakili kualitas menjadi sangat buruk, buruk, umum, baik dan sangat baik. Semakin tinggi MOS, semakin baik ucapan yang disintesis (Liu et al., 2021). Rumus untuk menghitung nilai MOS adalah sebagai berikut:

$$MOS = \frac{Total\ Semua\ Skor\ Responden}{Jumlah\ Responden} \quad (6)$$

   Skor MOS dapat diinterpretasikan dalam Tabel 1 berikut ini:

Tabel 1. Kriteria rentang nilai skor MOS

Rentang Skor	Kriteria
5.0	Sangat Baik
4.5 - 4.9	Baik
4.0 - 4.4	Layak
3.5 - 3.9	Cukup
Kurang dari 3.0	Sangat Buruk

2. Tes *Turing*

   Tes *Turing* yang dicetuskan oleh Alan Turing pada tahun 1950 merupakan sebuah uji konsep yang dirancang untuk menilai kemampuan mesin dalam memperlihatkan perilaku cerdas yang setara atau tidak dapat edakan dari manusia. Dalam tes *Turing* klasik, seorang manusia bertindak agai pewawancara dan berkomunikasi dengan dua entitas, satu manusia satu mesin (komputer), tanpa mengetahui identitas keduanya. Jika

pewawancara tidak dapat membedakan antara respon yang diberikan oleh manusia dan mesin, maka mesin tersebut dianggap berhasil "melewati" tes *Turing* tersebut. Dalam konteks pengujian TTS, tes *Turing* dapat digunakan untuk mengevaluasi seberapa baik kualitas suara yang dihasilkan oleh sistem TTS meniru suara manusia secara alami, sehingga responden tidak dapat membedakannya dari suara manusia yang sebenarnya. Responden akan diberikan dua sampel suara yang mengucapkan teks yang sama, dimana salah satunya adalah suara hasil sintesis model TTS dan responden akan diminta untuk menebak sampel suara mana yang merupakan suara manusia asli dan mana yang dihasilkan oleh model TTS. Jika skor tinggi (banyak yang salah menebak) maka model TTS dianggap sudah baik karena suaranya lebih sulit dibedakan dari suara manusia asli. Rumus untuk menghitung nilai presentase setiap opsi untuk setiap pertanyaan dalam survei adalah sebagai berikut:

$$Presentase\ Opsi\ (i) = \left(\frac{Jumlah\ Responden\ yang\ Memilih\ Opsi\ (i)}{Total\ Responden}\right) \times 100\% \qquad (7)$$

Dimana *Opsi (i)* merupakan salah satu opsi yang tersedia dalam pertanyaan survei. Nilai *(i)* adalah nama opsi sampel suara. Kemudian presentase keseluruhan akan dirata-ratakan untuk melihat gambaran umum tentang preferensi responden secara keseluruhan terhadap kedua jenis sampel suara yang diberikan.

Meskipun tes *Turing* secara keseluruhan bukan metode evaluasi standar untuk TTS, tapi ini bisa menjadi salah satu pendekatan untuk menilai kealamian dan kualitas suara yang dihasilkan.

3. Tes *Listening*

Tes *Listening* ada berbagai macam cara, salah satunya dapat diterapkan untuk mengevaluasi kualitas suara hasil dari model TTS. Tes *Listening* untuk penelitian ini adalah metode evaluasi di mana responden diminta untuk mendengarkan sampel suara yang dihasilkan oleh model TTS dan kemudian mengidentifikasi dan diminta memilih teks yang tepat yang sesuai dengan apa yang diucapkan oleh sampel suara tersebut. Tes ini bertujuan untuk ngevaluasi seberapa baik model TTS dapat menghasilkan keluaran suara g dapat dipahami dan sesuai dengan teks yang dimaksud. Evaluasi kukan dengan membandingkan teks yang dipilih oleh responden dengan

teks asli atau yang seharusnya diucapkan oleh sampel suara yang dievaluasi. Jika skor tinggi (banyak yang menjawab benar) maka model TTS dianggap sudah bisa menghasilkan suara dengan tingkat kejelasan yang bagus dan dapat dimengerti oleh manusia. Rumus untuk menghitung nilai presentase setiap opsi untuk setiap pertanyaan dalam survei adalah sebagai berikut:

$$Presentase\ Opsi\ (i) = \left(\frac{Jumlah\ Responden\ yang\ Memilih\ Opsi\ (i)}{Total\ Responden}\right) \times 100\% \qquad (8)$$

Dimana *Opsi (i)* merupakan salah satu opsi yang tersedia dalam pertanyaan survei. Nilai *(i)* adalah nama opsi jawaban dari pertanyaan. Kemudian presentase keseluruhan akan dirata-ratakan untuk menunjukkan proporsi responden yang menjawab pertanyaan dengan benar.

Meskipun metode ini juga bukan metode standar untuk menilai hasil TTS namun dapat membantu dalam menilai kualitas dan keakuratan hasil sintesis suara yang dihasilkan oleh model TTS.

2.8 *Confusion Matrix*

Matriks confusion (*Confusion Matrix*) adalah tabel yang digunakan untuk visualisasi performa model klasifikasi dalam *machine learning*. Ini menunjukkan seberapa baik model tersebut dapat mengidentifikasi kategori yang benar untuk data yang diberikan. *Confusion Matrix* terdiri dari empat sel, yang mewakili jumlah prediksi yang benar dan yang salah dalam klasifikasi. Visualisasi tabel *Confusion Matrix* ditunjukkan pada Gambar 24 berikut ini:

Actual \ Predicted	Positive	Negative
Positive	TRUE POSITIVE	FALSE NEGATIVE
Negative	FALSE POSITIVE	TRUE NEGATIVE

Gambar 24. Visualisasi tabel *Confusion Matrix*

- *True Positive* (TP): Prediksi yang benar positif, yaitu ketika model mprediksi kelas positif dan kelas sebenarnya juga positif.
- *e Negative* (TN): Prediksi yang benar negatif, yaitu ketika model mprediksi kelas negatif dan kelas sebenarnya juga negatif.

- *False Positive* (FP): Prediksi yang salah positif, yaitu ketika model memprediksi kelas positif tetapi kelas sebenarnya negatif.
- *False Negative* (FN): Prediksi yang salah negatif, yaitu ketika model memprediksi kelas negatif tetapi kelas sebenarnya positif.

Dalam konteks model TTS, kita dapat menggunakan *Confusion Matrix* untuk mengevaluasi kinerja model dalam menghasilkan ucapan. *Confusion Matrix* dalam TTS dapat membantu kita memahami seberapa baik model dapat membedakan antara audio yang dihasilkan oleh model (hasil sintesis) dan audio yang berasal dari suara asli manusia. Misalnya, dalam sebuah survei evaluasi model TTS, kita dapat meminta responden untuk mendengarkan dua audio, yaitu hasil sintesis dari model TTS dan yang lainnya adalah suara asli manusia. Kemudian, kita dapat menggunakan *Confusion Matrix* untuk menganalisis berapa banyak responden yang benar mengidentifikasi audio mana yang merupakan suara asli manusia dan audio mana yang dihasilkan oleh model. Dengan demikian, *Confusion Matrix* dalam model TTS akan memiliki empat sel yang mewakili:

- *True Positive* (TP): Jumlah responden yang benar mengidentifikasi suara asli manusia.
- *True Negative* (TN): Jumlah responden yang benar mengidentifikasi hasil sintesis model.
- *False Positive* (FP): Jumlah responden yang salah mengidentifikasi hasil sintesis model sebagai suara asli manusia.
- *False Negative* (FN): Jumlah responden yang salah mengidentifikasi suara asli manusia sebagai hasil sintesis model.

Dengan menggunakan *Confusion Matrix*, kita dapat menghitung berbagai metrik evaluasi seperti *Accuracy* (persentase prediksi yang benar secara keseluruhan), *Precision* (persentase prediksi positif yang benar), *Recall* (persentase data aktual yang diidentifikasi dengan benar), dan *F1-score*, yang membantu dalam mengevaluasi kinerja model (metrik yang menggabungkan *precision* dan *recall*). Keempat metrik tersebut dapat digunakan untuk memberikan gambaran yang lebih tentang performa model. Berikut rumus perhitungan keempat metrik

$$Accuracy: \frac{TN+TN}{TP+TN+FP+FN} \quad (9)$$

$$Precision: \frac{TP}{TP+FP} \quad (10)$$

$$Recall: \frac{TP}{TP+FN} \quad (11)$$

$$F1\ score: 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (12)$$

2.9 Adobe Audition

Adobe Audition yang awalnya dikembangkan sebagai Cool Edit Pro oleh Syntrillium Software sebelum diakuisisi oleh Adobe Systems ini adalah perangkat lunak yang cocok untuk merekam, *mixing*, *mastering* dan mengedit audio dalam proyek film, televisi, musik, maupun *podcast*. Dengan aplikasi Adobe Audition, pengguna bisa melakukan perekaman suara, *editing* kualitas suara, menambah beragam efek suara, menggabungkan berbagai *track* suara menjadi satu *track*, sekaligus menyimpan proyek dengan berbagai format.

2.10 Audacity

Audacity adalah aplikasi editor audio digital yang gratis dan *open source*. Aplikasi ini tersedia untuk Windows, macOS, Linux. Audacity yang mulai dikembangkan pada akhir 1999 dapat digunakan untuk mengolah *file* berbasis audio dengan cara memotong, memperbanyak, menyatukan *track* satu dengan yang lain, merekam suara atau memberikan efek khusus pada suara. Audacity memiliki tampilan audio atau suara yang *user-friendly*, sehingga tampilan tersebut mempermudah suara untuk diolah.