BAB III

METODE PENELITIAN

3.1 Subjek dan Objek Penelitian

Subjek pada penelitian ini adalah mereka yang memiliki ketertarikan dan minat dalam bidang musik, khususnya bagi mereka yang menyukai *genre* musik pop barat di era 90-an hingga masa kini. Sedangkan objek pada penelitian ini adalah suara instrumen musik gitar dan suara vokal manusia yang didapat dari sebuah *cover* lagu akustik. Data lagu dikumpulkan melalui *platform YouTube* dengan format *file* MP3. Penelitian ini dilakukan secara fleksibel, karena data yang dibutuhkan untuk penelitian dapat dengan mudah diperoleh melalui *platform* media seperti *YouTube*. Selain itu, tempat dilakukannya penelitian ini yaitu di lab data kampus (Institut Teknologi Telkom Purwokerto), yang dilengkapi dengan perangkat keras dan perangkat lunak yang memadai untuk memproses data dengan jumlah dan ukuran yang besar. Hal ini menjadikan penelitian ini dapat dilakukan dengan efisien dan efektif, terlebih lagi dengan dukungan teknologi canggih dalam bidang *deep learning* seperti *autoencoder* dan RNN. Dengan demikian, penelitian ini memiliki potensi besar untuk memberikan hasil yang signifikan dan berdampak luas pada perkembangan dunia musik, terutama pada bidang produksi musik digital.

3.2 Alat dan Bahan Penelitian

Penelitian ini membutuhkan alat dan bahan yang dapat menunjang jalannya proses dan kelancaran penelitian. Beberapa alat yang dibutuhkan diantaranya adalah perangkat keras dengan spesifikasi minimum untuk *processing* data, perangkat lunak untuk pengumpulan data, pengolahan data, pemrosesan data, dan analisis data. Sedangkan bahan yang dibutuhkan adalah *dataset* sebagai objek yang diteliti.

3.2.1 Spesifikasi Perangkat Keras

Perangkat keras yang digunakan pada penelitian ini adalah *Personal Computer* (PC) dan laptop dengan spesifikasi yang terdapat pada tabel di bawah ini.

Tabel 3.1 Tabel Spesifikasi Perangkat Keras (PC)

Komponen Device	Spesifikasi Device
Processor	Intel(R) Core(TM) i7-10700K CPU @3.80GHz 3.79 GHz
RAM	32.0 GB
Storage	345 GB (Local Disk), 585 GB (New Volume)
System type	64-bit operating system, x64-based processor

Tabel 3.2 Tabel Spesifikasi Perangkat Keras (Laptop)

Komponen Device	Spesifikasi Device
Processor	Intel(R) Celeron(R) N4120 CPU @ 1.10GHz 1.10 GHz
RAM	4,00 GB DDR4
Storage	256GB SSD
System type	64-bit operating system, x64-based processor

3.2.2 Spesifikasi Perangkat Lunak

Perangkat lunak yang digunakan pada penelitian ini mencakup sistem operasi dan beberapa aplikasi dengan kegunaan yang terdapat pada tabel di bawah ini.

Tabel 3.3 Tabel Spesifikasi Perangkat Lunak

Jenis	Keterangan	Kegunaan
Sistem Operasi	Windows 10 Pro	Sistem operasi yang digunakan untuk menjalankan perangkat lunak pada PC
	Windows 11 Home Single Language	Sistem operasi yang digunakan untuk menjalankan perangkat lunak pada Laptop
Aplikasi	Google Chrome	Untuk browsing segala kebutuhan terkait penelitian

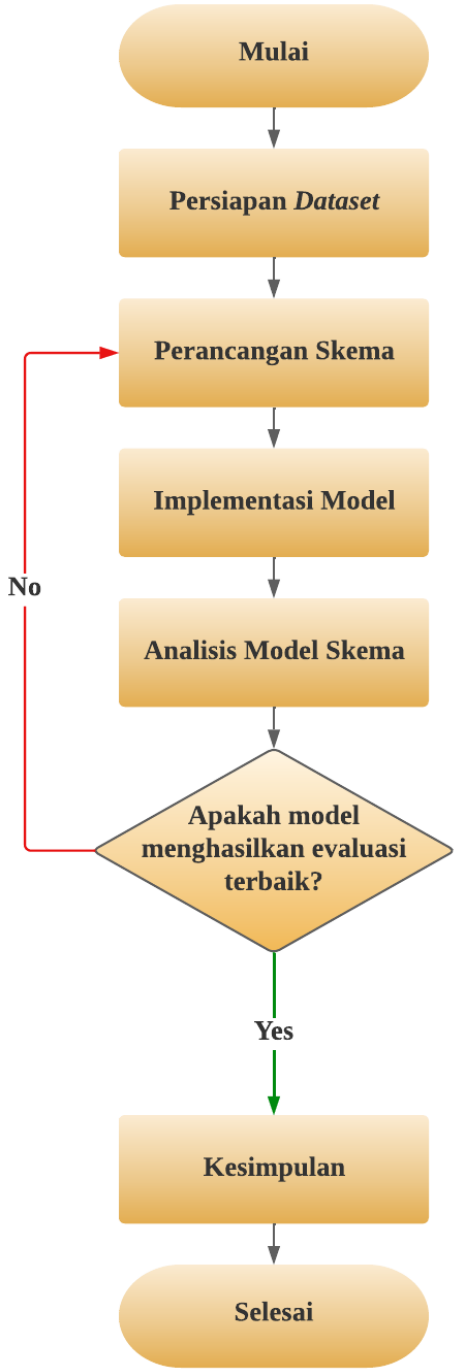
Jenis	Keterangan	Kegunaan
	Google Drive	Menyimpan segala bahan penelitian pada cloude storage
	Google Colaboratory	Platform lembar kerja secara online untuk menjalankan prorgam dan kode pyhton
	Jupyter Notebook	Platform lembar kerja secara offline pada lokal untuk menjalankan prorgam dan kode pyhton
	Google Spreadsheet	Lembar kerja untuk analisis dan evaluasi data
	Visual Studio Code	Editor yang digunakan untuk membangun web interface.
	YouTube	Referensi pengumpulan dataset cover lagu akustik
	SoundCloud	Menyimpan output suara hasil penelitian pada cloud

3.2.3 Bahan Penelitian

Pada penelitian ini, bahan penelitian yang digunakan adalah *dataset* yang berisi sejumlah lagu *cover* akustik yang diperoleh dari *platform YouTube* dalam format MP3. Untuk memisahkan suara vokal dan instrumen pada setiap lagu, digunakan model *spleeter* 5 *stems* sehingga menghasilkan dua *set* data terpisah, yaitu suara vokal dan instrumen. Setelah itu, kedua *set* data tersebut dimasukkan ke dalam variabel *x* dan *y*, dimana *x* berisi data suara vokal dan *y* berisi data suara instrumen.

3.3 Diagram Alir Penelitian

Dalam melakukan penelitian, perlu adanya rangkaian proses atau tahapan penelitian. Hal ini dilakukan agar penelitian dapat berjalan dengan lancar dan mendapatkan hasil yang diharapkan. Berikut adalah proses penelitian yang digambarkan pada diagram alir dibawah ini.



Gambar 3.1 Diagram Alir Penelitian

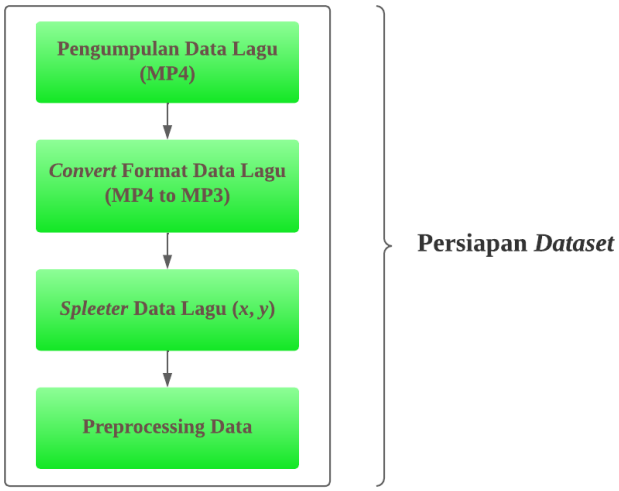
Tahap awal dalam penelitian ini adalah membuat rumusan masalah. Rumusan masalah dilakukan untuk menentukan permasalahan yang akan

diselesaikan dalam penelitian ini. Setelah rumusan masalah ditetapkan, tahap selanjutnya adalah studi literatur *review*. Studi literatur *review* dilakukan untuk mengumpulkan informasi dari penelitian sebelumnya yang terkait dengan topik yang akan diteliti. Studi literatur review juga dapat dilakukan untuk mendapatkan landasan teori yang relevan dengan penelitian yang akan dilakukan.

Dalam penelitian ini, penulis telah melakukan studi literatur *review* berupa kajian pustaka dari penelitian sebelumnya dan landasan teori yang relevan. Hal ini menunjukkan bahwa penelitian ini telah dipersiapkan secara matang sebelum melakukan penelitian. Dengan melakukan studi literatur *review*, penulis dapat memperoleh gambaran tentang topik penelitian dan dapat mengetahui penelitian apa yang sudah dilakukan sebelumnya dalam topik tersebut. Selain itu, studi literatur review juga dapat membantu penulis dalam membangun hipotesis atau konsep penelitian yang lebih terarah dan terfokus.

3.3.1 Persiapan *Dataset*

Pada penelitian ini *dataset* utama yang harus dikumpulkan adalah suara vokal manusia dan suara instrumen musik gitar yang berasal dari sebuah lagu akustik gitar atau rekaman dari *cover* lagu akustik. Jumlah *dataset* yang akan dikumpulkan adalah 250 *dataset* yang dibagi menjadi dua bagian, yaitu 250 suara vokal manusia dan 250 suara instrumen musik gitar yang didapatkan melalui *youtube format* file MP3.



Gambar 3.2 Diagram Alir Persiapan *Dataset*

Proses persiapan *dataset* pada penelitian ini dimulai dengan pengumpulan data lagu melalui sumber *youtube* dalam format mp4. Setelah semua data terkumpul, tahap selanjutnya adalah melakukan konversi format *file* mp4 ke mp3 menggunakan aplikasi *open-source* bernama *ffmpeg*. Data lagu kemudian dipisahkan antara suara vokal dan instrumen musik gitar menggunakan pustaka model U-Nets *machine learning* bernama *Spleeter* dengan variasi 5 *stems*. Setelah dilakukan pemisahan, data lagu siap untuk diproses lebih lanjut pada tahap *preprocessing* data sebelum diimplementasikan pada masing-masing skema. Dalam hal ini, proses persiapan dataset telah dilakukan dengan teliti untuk memastikan kualitas data yang diperoleh memadai dan siap digunakan pada tahap selanjutnya.

3.3.2 Perancangan Skema

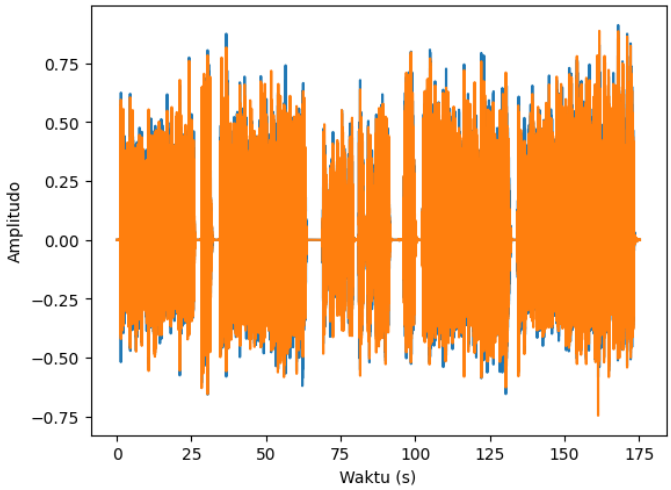
Perancangan skema merupakan teknik penyelesaian yang akan digunakan untuk menyelesaikan permasalahan pada penelitian. Pada penelitian ini, peneliti merancang tiga skema yang berbeda yaitu Skema *Autoencoder* - RNN, Skema *Autoencoder* - Sisir *(Combing)* - RNN, dan Skema DCT – RNN. Setiap skema kemudian dievaluasi dan dianalisis untuk mengetahui kelemahan dan kekurangan

masing-masing skema. Hal ini dilakukan agar pada pembaruan selanjutnya, peneliti dapat menentukan teknik yang dapat menyelesaikan permasalahan yang tidak teratasi pada skema sebelumnya.

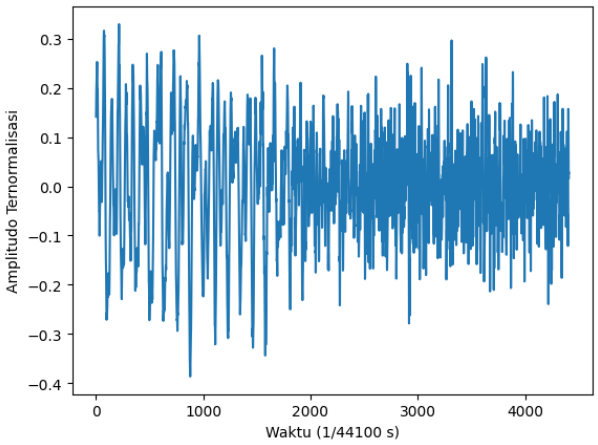
Skema *autoencoder* - RNN merupakan skema awal yang digunakan pada penelitian ini. Kemudian, dilakukan pengembangan skema dengan melakukan pembaruan yang menghasilkan skema *Autoencoder* - Sisir *(Combing)* - RNN. Skema *Autoencoder* - Sisir *(Combing)* - RNN kemudian menjadi dasar bagi pengembangan skema selanjutnya yaitu skema DCT - RNN. Dalam tahap evaluasi dan analisis skema, peneliti akan mengevaluasi kinerja masing-masing skema dan menganalisis hasilnya. Dari hasil evaluasi dan analisis tersebut, peneliti akan mengetahui kelebihan, kelemahan, dan kekurangan masing-masing skema sehingga dapat memperbaiki dan mengembangkan skema selanjutnya dengan lebih baik. Penjelasan masing-masing skema secara rinci akan dijabarkan pada poin-poin di bawah ini.

3.3.2.1 Skema *Autoencoder* - RNN

Skema *autoencoder* - RNN merupakan skema yang mengkombinasikan teknik *deep learning autoencoder* dan RNN. Pada skema ini dilakukan *preprocessing* data dengan cara memotong setiap data (x, y) dari jumlah data 44100*n menjadi 4410*n atau 1/10 sekon dan 8820*n atau 2/10 sekon dari total durasi data lagu. Data *input* dari hasil *preprocessing* data pada skema ini disebut *input* pendek.



Gambar 3.3 Data Suara Asli (Lagu ke-0)

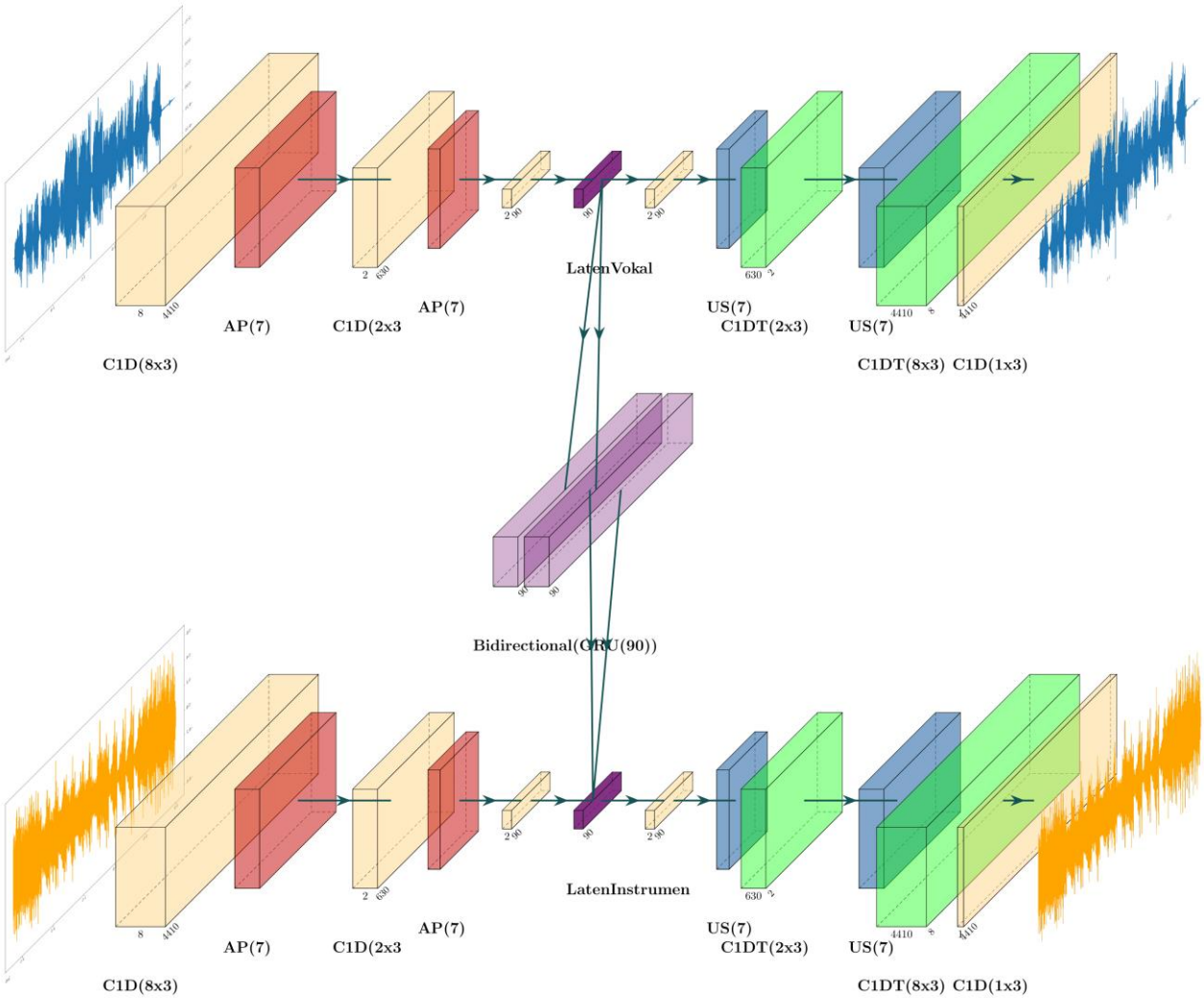


Gambar 3.4 Hasil *Preprocessing* Data Suara (Lagu ke-0)

Pada Gambar 3.3, merupakan contoh data dalam bentuk asli dan sebelum melalu proses *preprocessing* data, sedangkan pada Gambar 3.4 menunjukkan hasil dari salah satu potongan data 4410 atau 0.1s setelah melalui proses *preprocessing* data.

Skema ini akan menyiapkan dua model *autoencoder* dengan *input* masing-masing 4410 atau 8820. Model *autoencoder* pertama akan di*training* dengan data suara vokal (*x*) sebagai *input* dan *output*, sedangkan autoencoder kedua akan di*training* dengan data suara instrumen (*y*) sebagai

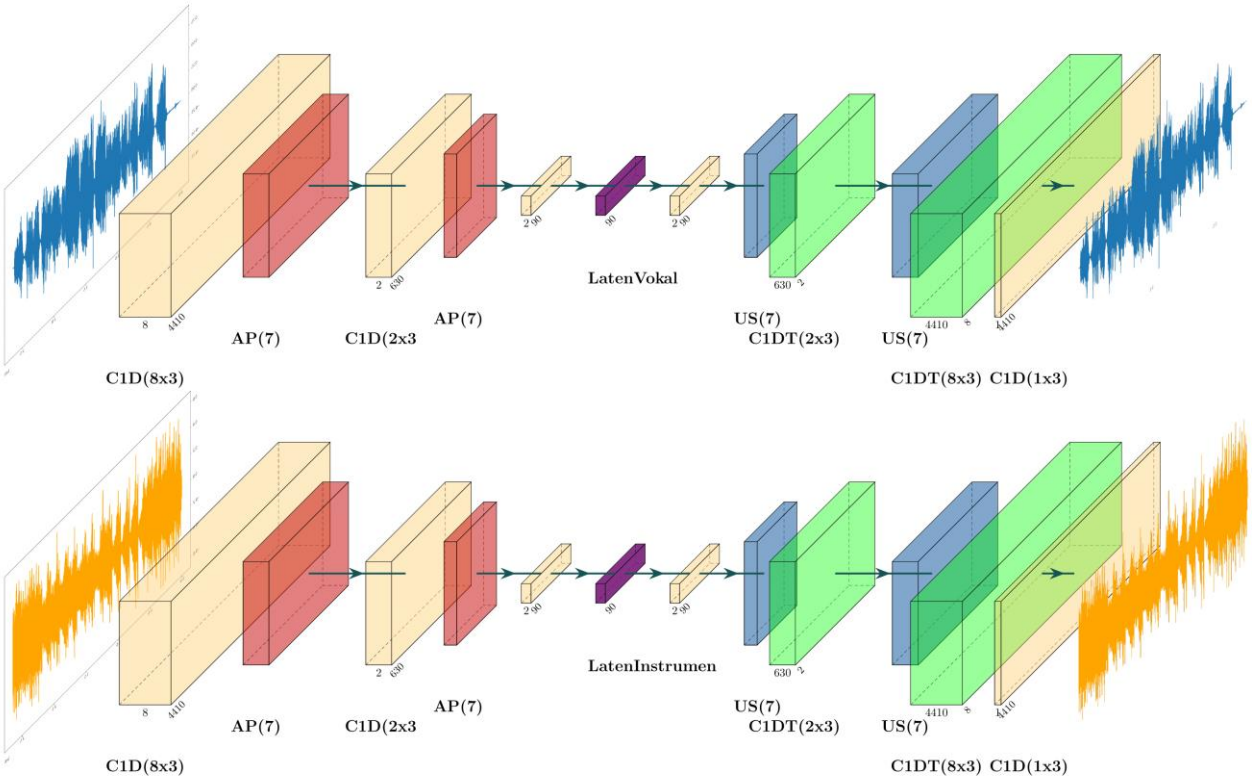
input dan *output*. Berikut adalah gambaran lengkap pemodelan diagram skema *autoencoder* – RNN.



Gambar 3.5 Ilustrasi Pemodelan Skema *Autoencoder* - RNN

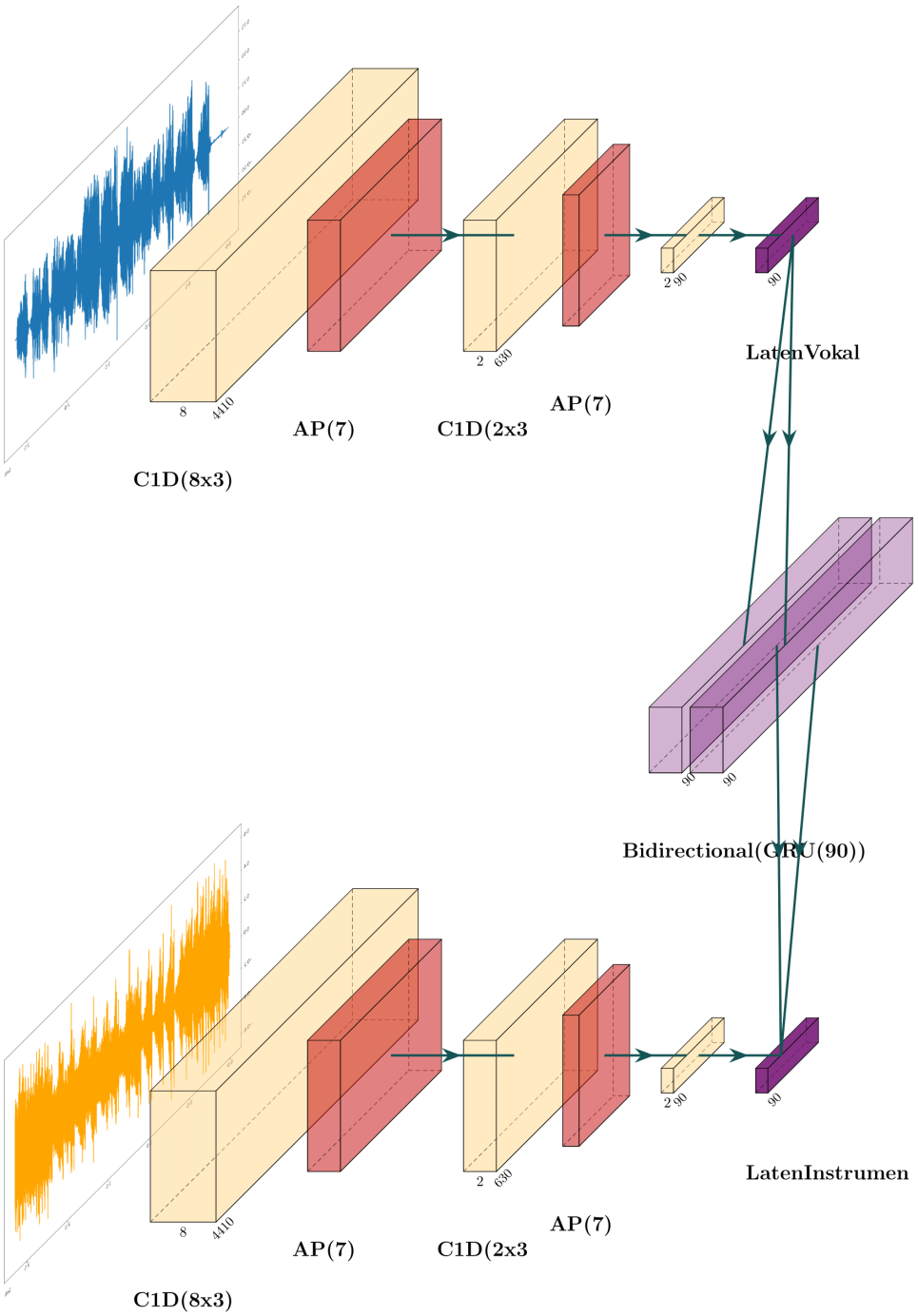
Pada contoh ilustrasi pemodelan skema di atas, diketahui arsitektur kedua model *autoencoder* menggunakan spefikasi 2-*layers* Conv1D, 2-*layers pooling*, dan 1-*fully connected layer*. Model *encoder* memiliki susunan arsitektur Conv1D(8x3) – AveragePooling(7) – Conv1D(2x3) – AveragePooling(7) , sedangkan model *decoder* memiliki susunan arsitektur cermin dari model *encoder*, yaitu UpSampling(7) – Conv1D(2x3) – UpSampling(7) – Conv1D(8x3) dengan *fully connected layer* Conv1D(1x3).

Untuk membuat model *autoencoder* yang dapat menyederhanakan *input* data suara vokal (*x*) dan input data instrumen (*y*) menjadi data *latent*, maka akan dilakukan *training* kedua model dengan *autoencoder*. Berikut adalah contoh ilustrasi *training* model yang ditunjukkan pada gambar di bawah.



Gambar 3.6 Ilustrasi Pelatihan *Autoencoder Input* Pendek

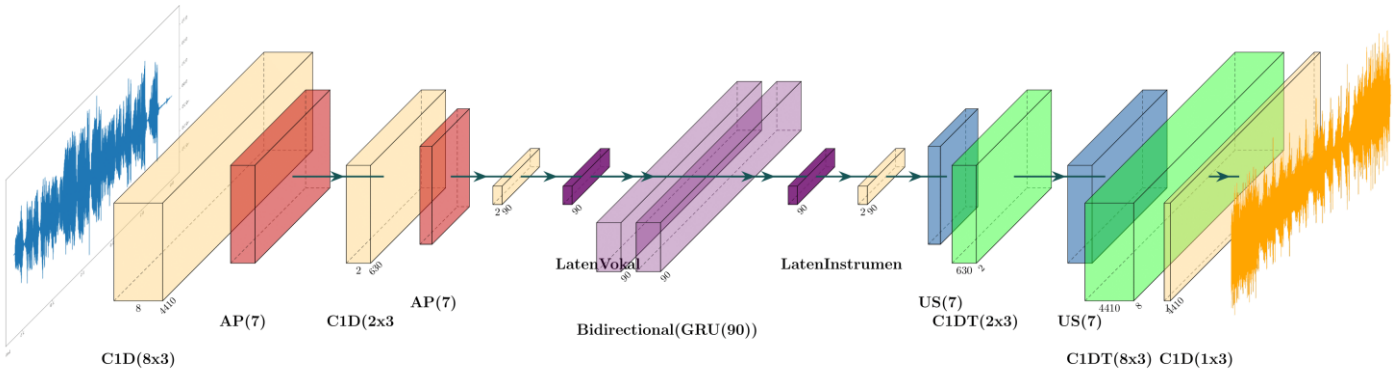
Pada Gambar 3.6, terdapat dua diagram, diagram atas adalah representasi dari model *autoencoder* pertama yang di*training* menggunakan data suara vokal (*x*) dan diagram bawah adalah representasi dari model *autoencoder* kedua yang di*training* menggunakan data instrumen (*y*). Fase ini merupakan proses untuk mendapatkan data *latent* dengan ukuran sekecil mungkin. Hal ini dilakukan untuk memungkinkan *training* model RNN-GRU yang lebih cepat dan meminimalkan pemakaian memori yang lebih banyak.



Gambar 3.7 Ilustrasi Pelatihan *Latent* Data Suara Vokal terhadap Data Suara Instrumen (*Input* Pendek)

Pada tahap selanjutnya, akan dimanfaatkan model *encoder* dari setiap *autoencoder* untuk melatih fitur data *latent* dari masing-masing data

suara vokal dan suara instrumen. Harapannya agar didapatkan model dengan evaluasi *loss* yang sekecil mungkin.

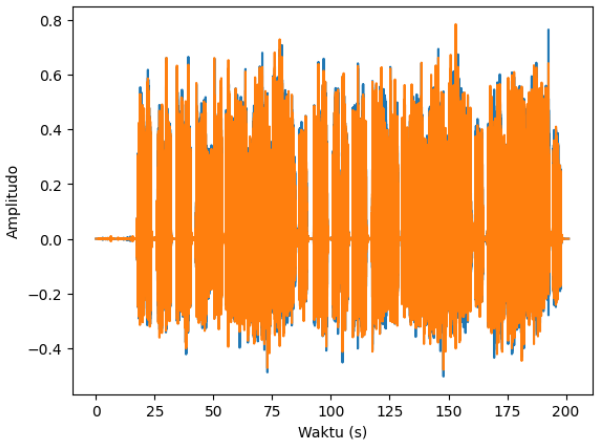


Gambar 3.8 Model Utama Skema *Autoencoder* – RNN Dengan *Input* Pendek

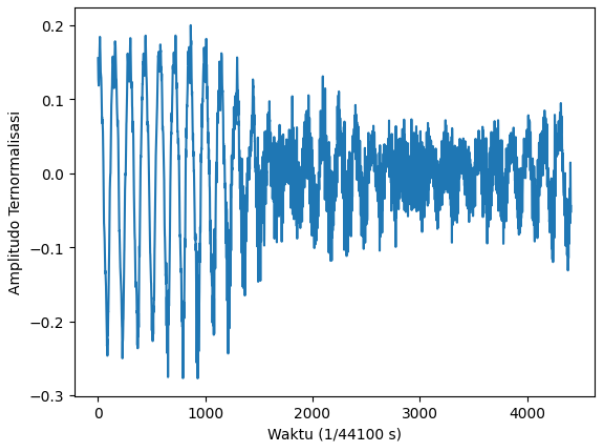
Proses terakhir pada skema ini [adalah dengan menggabungkan semua model *autoencoder* yang telah di*training* menjadi satu model utama dengan *input* pendek. Bagian model *encoder* dari *autoencoder* pertama akan dihubungkan ke model GRU, dan *output* dari model GRU akan dihubungkan ke bagian model *decoder* dari *autoencoder kedua*. Langkah ini memungkinkan model utama untuk mengekstrak fitur dari *input* pendek dengan menggunakan model GRU dan kemudian menghasilkan *output* dengan memanfaatkan model *decoder* dari *autoencoder* kedua.

3.3.2.2 Skema *Autoencoder* - Sisir *(Combing)* - RNN

Skema *autoencoder* – Sisir (*Combing*) – RNN merupakan *scalling* dari skema *autoencoder* – RNN. Pada skema ini dilakukan *preprocessing* data dengan cara memotong setiap data (*x*, *y*) dari jumlah data 44100**n* menjadi 4410*64 atau 64/10 sekon dan 8820*64 atau 128/10 sekon dari total durasi data lagu. Data *input* dari hasil *preprocessing* data pada skema ini disebut *input* sisir.



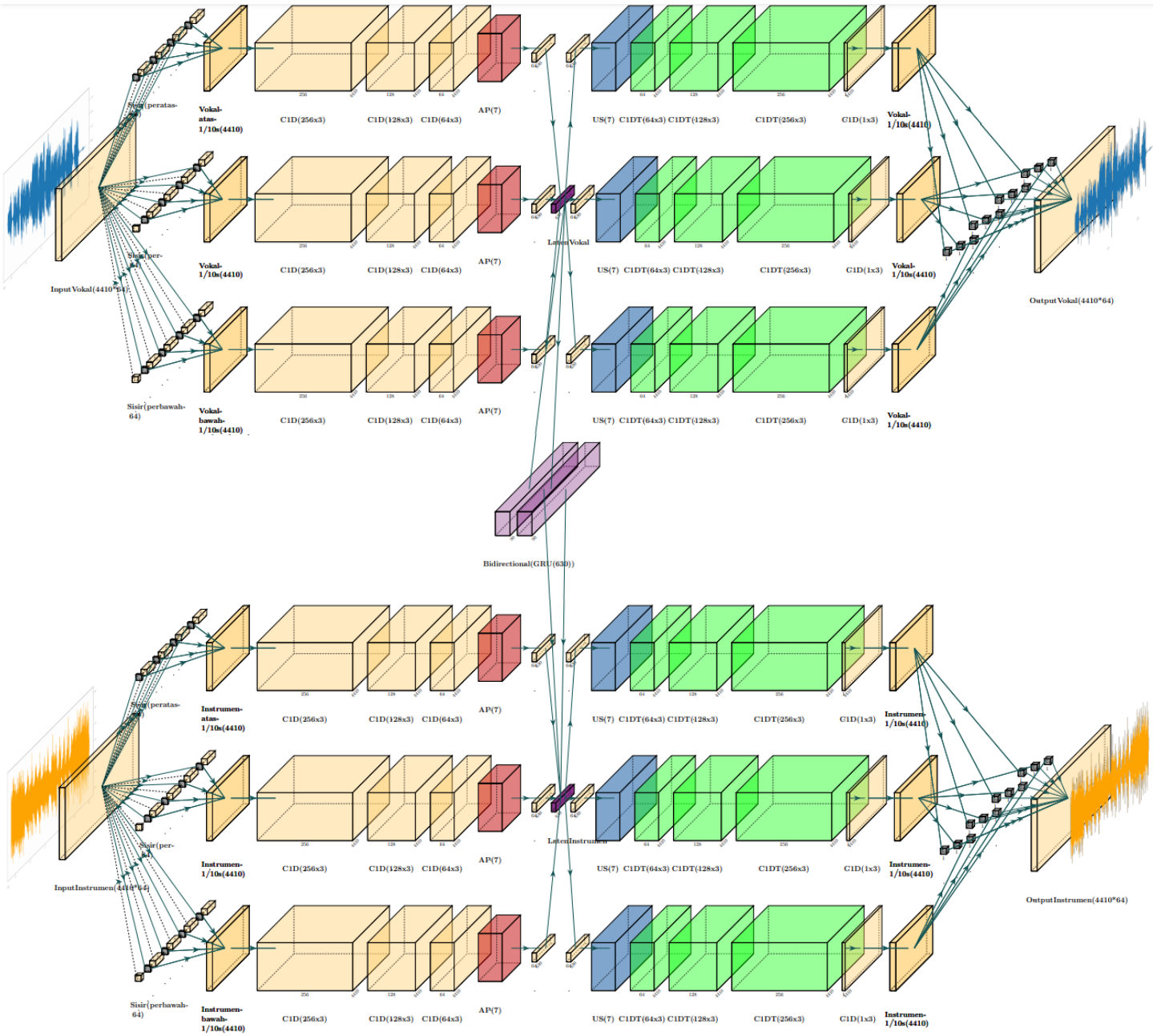
Gambar 3.9 Data Suara Asli (Lagu ke-1)



Gambar 3.10 Hasil *Preprocessing* Data Suara (Lagu ke-1)

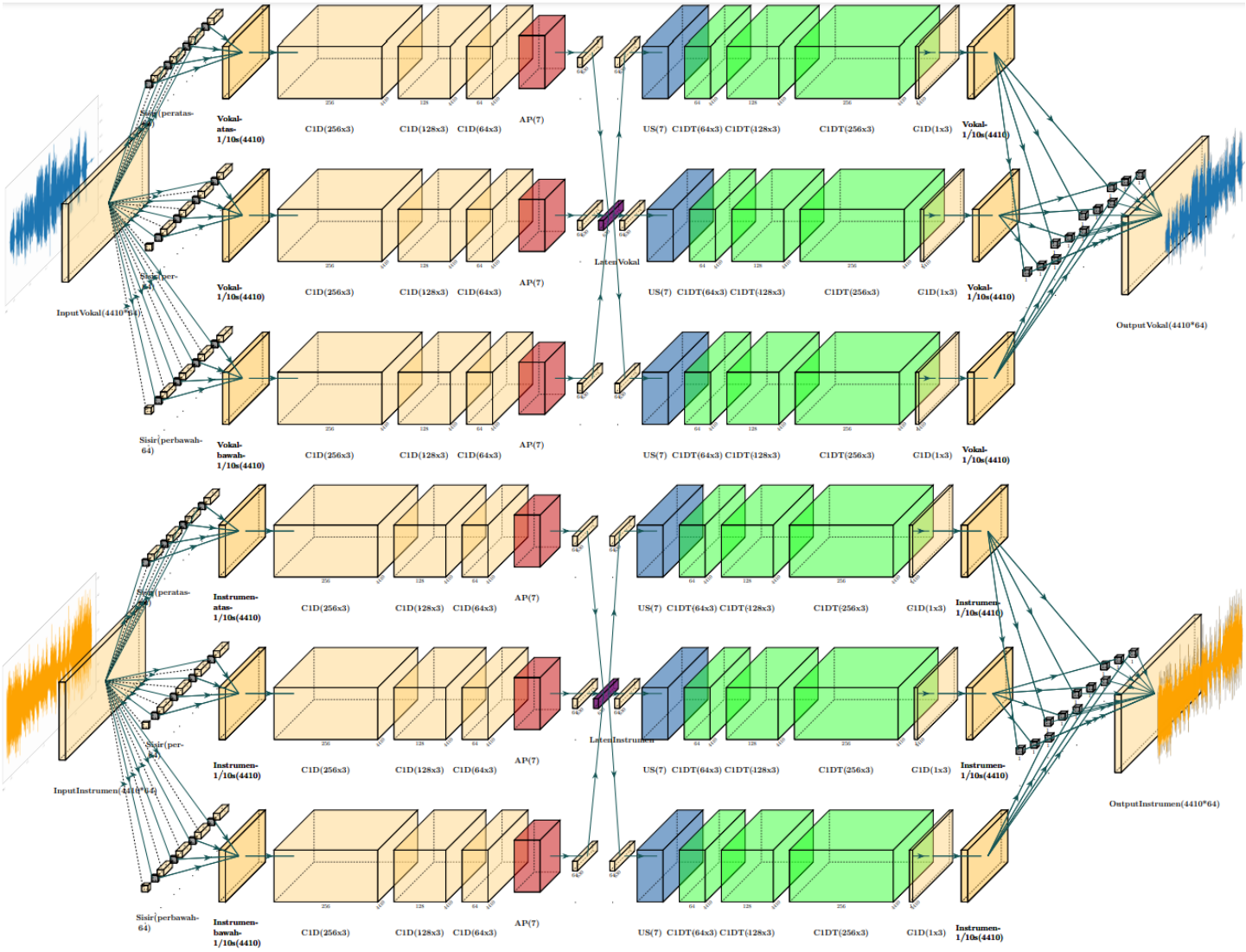
Pada Gambar 3.9, merupakan contoh data dalam bentuk asli dan sebelum melalu proses *preprocessing* data, sedangkan pada Gambat 3.10 menunjukkan hasil dari salah satu potongan data 4410 atau 0.1 setelah melalui proses *preprocessing* data menggunakan teknik sisir.

Pada skema ini, dilakukan modifikasi pada konfigurasi model *input* yang mana berdasarkan pada masukan yang disusun secara berurutan menyisir, atau dengan kata lain menggunakan *input* sisir. Berikut adalah gambaran lengkap pemodelan diagram skema *autoencoder* – sisir (*combing*) – RNN.



Gambar 3.11 Ilustrasi Pemodelan Skema *Autoencoder* - Sisir (*Combing*) – RNN

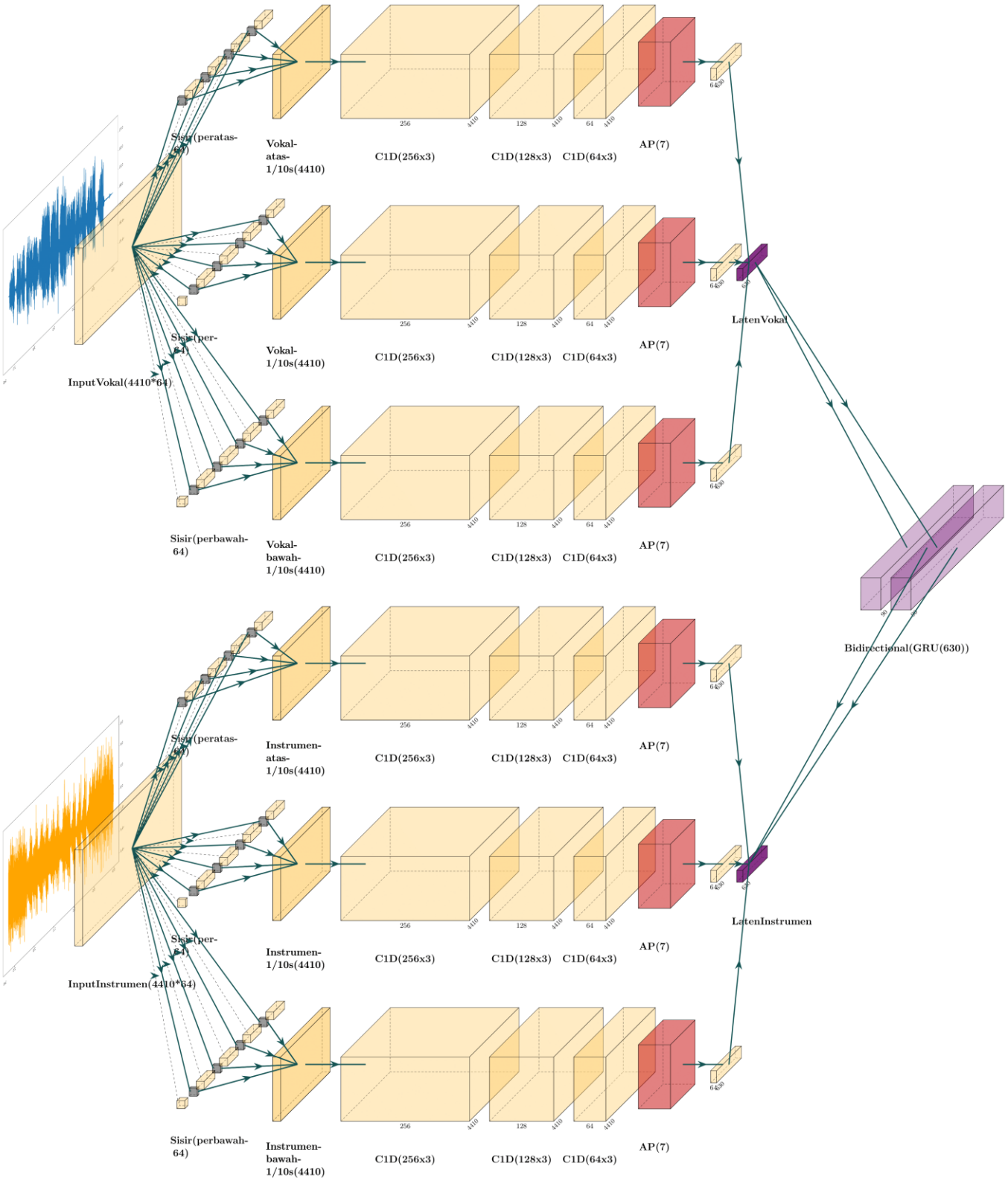
Pada contoh ilustrasi pemodelan skema di atas, diketahui arsitektur kedua model *autoencoder* menggunakan spefikasi 3-*layers* Conv1D, 1-*layers pooling*, dan 1-*fully connected layer*. Model *encoder* memiliki susunan arsitektur Conv1D(256x3) – Conv1D(128x3) – Conv1D(64x3) – AveragePooling(7) , sedangkan model *decoder* memiliki susunan arsitektur cermin dari model *encoder*, yaitu UpSampling(7) – Conv1D(64x3) – Conv1D(128x3) – Conv1D(256) dengan *fully connected layer* Conv1D(1x3).



Gambar 3.12 Ilustrasi Pelatihan *Autoencoder Input* Sisir

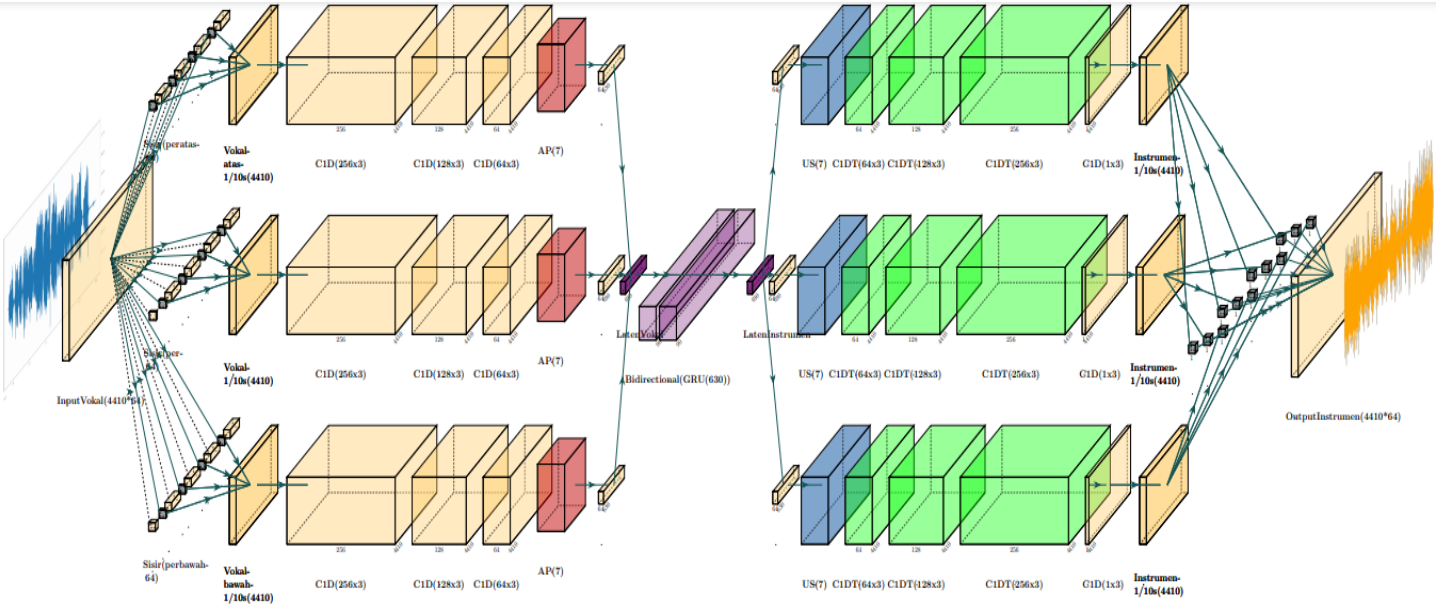
Pada Gambar 3.12, diketahui bahwa input pada proses ini dibagi menjadi beberapa bagian berdasarkan pengaturan sisir. Akan tetapi, pada ilustrasi di atas hanya menunjukkan tiga bagian awal dari area yang disisir, tetapi pada *training* sebenarnya, digunakan 64 bagian dari area yang disisir yang mencakup data selama 6,4 sekon. Tujuannya tetap sama, yaitu untuk mendapatkan data *latent* yang sekecil mungkin.

Langkah selanjutnya sama dengan teknik sebelumnya, di mana pada fase ini, bagian model *encoder* dari setiap *autoencoder* digunakan karena tujuan saat ini adalah melatih fitur *latent* dari masing-masing data suara vokal (x) dan data instrumen (y).



Gambar 3.13 Ilustrasi Pelatihan *Latent* Data Suara Vokal terhadap Data Suara Instrumen (*Input* Sisir)

Tahap terakhir, untuk menggabungkan semua model yang telah di*training* menjadi model utama, model utama untuk skema *autoencoder* – sisir (*combing*) – RNN dibuat seperti yang ditunjukkan pada Gambar 3.14.

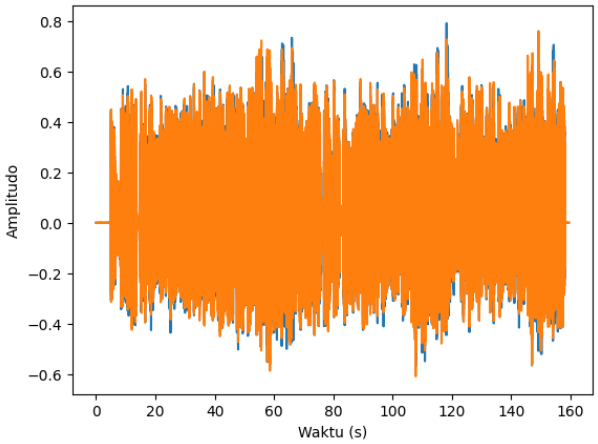


Gambar 3.14 Model Utama Skema *Autoencoder* - Sisir (*Combing*) - RNN Dengan *Input* Sisir

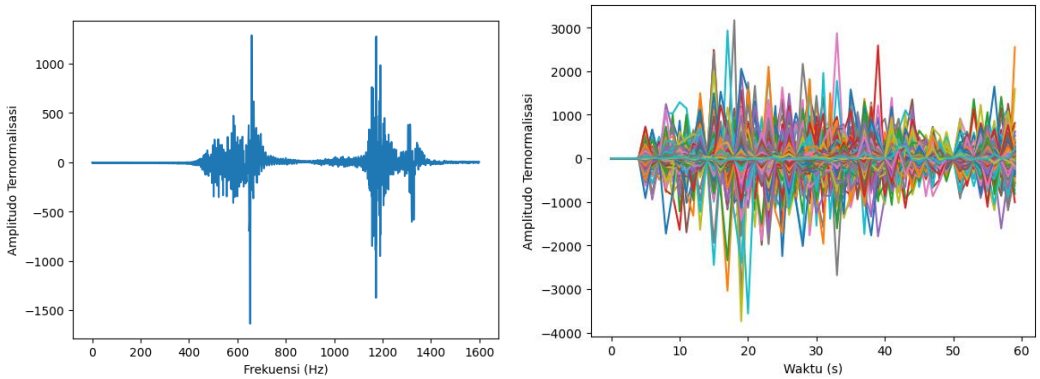
Berdasarkan perancangan kedua skema, peneliti akan mencoba membuat beberapa arsitektur model yang berbeda untuk mengimplementasikan kedua jenis skema yang telah dijabarkan sebelumnya. Hal ini penting untuk dilakukan agar dapat membandingkan dan mengevaluasi efektivitas dari masing-masing arsitektur dalam memproduksi instrumen musik. Selama proses pelatihan, model-model tersebut akan dinilai berdasarkan evaluasi metrik umum seperti *loss*, *bias*, dan standar deviasi yang dihasilkan. Setelah model optimal teridentifikasi, kami akan melakukan *scalling* dengan melakuan *training* ulang model tersebut dengan *dataset* yang lebih besar dan epoch yang lebih lama untuk meningkatkan performa model.

3.3.2.3 Skema *Discrete Cosine Transform* (DCT) - RNN

Skema ini merupakan pengembangan dari kedua skema sebelumnya. Pada skema ini dilakukan *preprocessing* data dengan cara memotong setiap data (*x*, *y*) per 30 atau 60 sekon dari total durasi data lagu, dan kemudian diubah ke dalam bentuk DCT. Setelah itu, akan dilakukan proses *filtering* frekuensi pada rentang 0 – 800 *Hertz* atau 0 – 1600 *Hertz*. Data *input* dari hasil *preprocessing* data pada skema ini disebut *input* DCT.



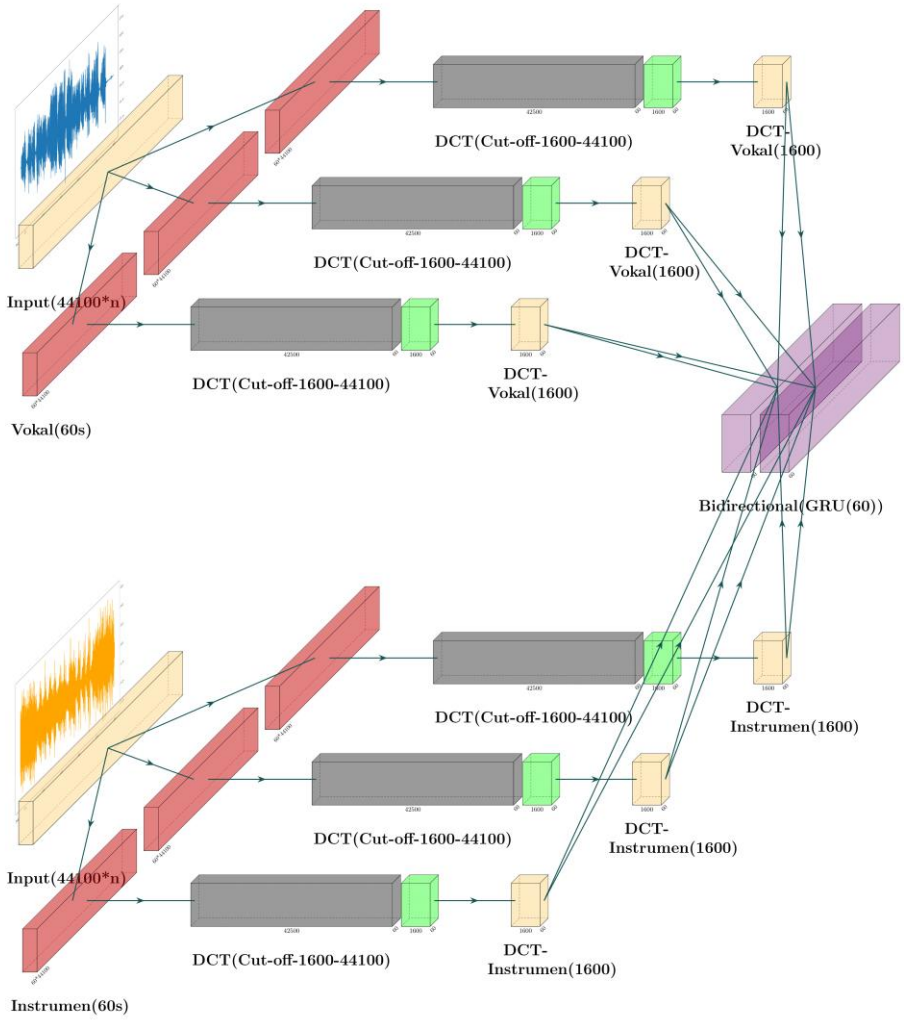
Gambar 3.15 Data Suara Asli (Lagu ke-2)



Gambar 3.16 Hasil *Preprocessing* Data Suara (Lagu ke-2)

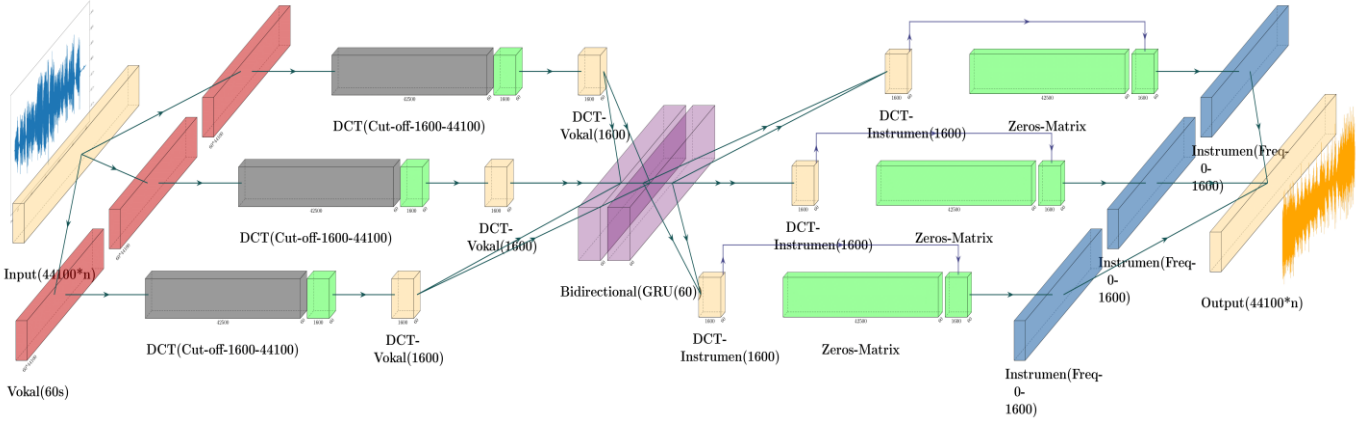
Pada Gambar 3.15, merupakan contoh data dalam bentuk asli dan sebelum melalui proses *preprocessing* data, sedangkan pada Gambar 3.16 menunjukkan hasil dari salah satu potongan 1600 data frekuensi DCT setelah melalui proses *preprocessing* data.

Pada skema ini, teknik *autoencoder* tidak lagi digunakan untuk mengekstrak data lagu. Namun, akan digunakan secara langsung potongan data lagu berdurasi 30 atau 60 sekon. Setiap potongan data akan diubah menjadi bentuk DCT per 30 atau 60 sekon dan hanya pada rentang frekuensi 0 – 800 *Hertz* atau 0 – 1600 *Hertz* yang akan diambil. Data potongan yang telah disiapkan akan memiliki dimensi 30 × 800 atau 60 × 1600. Ukuran data yang lebih kecil ini akan diproses lebih cepat oleh RNN dengan lapisan yang lebih sedikit dibandingkan dengan data berdimensi 800 × 30 atau 1600 × 60. Berikut adalah gambaran lengkap pemodelan diagram skema RNN – DCT .



Gambar 3.17 Ilustrasi Pelatihan Data Suara Vokal terhadap Data Suara Instrumen (*Input* DCT)

Pada tahap awal, dilakukan *training* model RNN menggunakan data *latent* dalam bentuk DCT dari data suara vokal (*x*) terhadap data *latent* dalam bentuk DCT dari data instrumen (*y*).

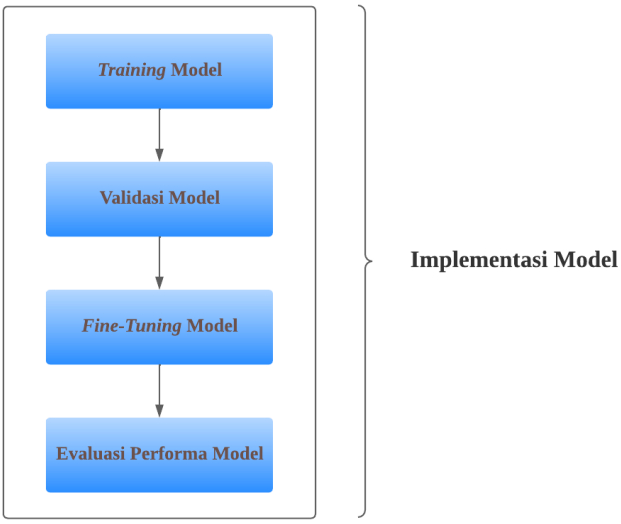


Gambar 3.18 Model Utama Skema RNN-DCT

Setelah *training* model RNN, maka tahap selanjutnya adalah menghasilkan model utama pada skema RNN – DCT dengan struktur arsitektur model utama yang dapat dilihat pada Gambar 3.18. Hasil akhir dari model ini adalah suara instrumen musik pada rentang frekuensi 0 – 800 *Hertz* atau 0 – 1600 *Hertz*.

Kemudian, untuk hasil *output* yang lebih kompleks dan bervariasi, maka dibuat model lain yang spesifik untuk menerima *input* data lagu dengan frekuensi 1600 – 3200 *Hertz*, 3200 – 4800 *Hertz*, dan seterusnya. Melalui cara ini, akan dihasilkan beberapa model *output* yang berbeda dan kemudian menggabungkannya menjadi satu, sehingga memperoleh hasil suata instrumen yang lengkap dengan seluruh frekuensi yang ada.

3.3.3 Implementasi Model



Gambar 3.19 Diagram Alir Implementasi Model

Pada tahap selanjutnya, akan dilakukan implementasi model-model pada setiap skema. Implemenasi model akan meliputi beberapa tahapan, yaitu *training* model, validasi model, *fine-tuning* model, dan evaluasi performa model menggunakan metrik *loss*, *bias*, dan standar deviasi. Tahapan *training* model akan dilakukan dengan memasukkan data ke dalam model, dan mengoptimalkan model untuk meminimalkan nilai *loss*, sehingga model dapat mempelajari pola dan fitur-fitur dari data yang dimasukkan.

Selanjutnya, tahapan validasi model akan dilakukan untuk mengevaluasi performa model selama proses *training*. Dalam tahap ini, model akan diuji menggunakan data validasi yang tidak pernah dilihat oleh model sebelumnya, dan dihitung nilai akurasi atau performa model terhadap data validasi tersebut.

Setelah tahap validasi, tahapan *fine-tuning* model akan dilakukan untuk mengoptimalkan model lebih lanjut. Tahap ini dilakukan dengan mengubah beberapa hyperparameter model, seperti *learning rate* dan jumlah *epoch*, untuk mendapatkan model yang lebih baik dan akurat.

Tahapan terakhir dalam proses pelatihan model adalah evaluasi performa model menggunakan metrik *loss*, *bias*, dan standar deviasi. Metrik-metrik ini akan digunakan untuk menilai seberapa baik performa model dalam mempelajari pola

dan fitur-fitur dari data yang dimasukkan. Dalam evaluasi performa model, model-model yang dihasilkan akan dibandingkan satu sama lain, sehingga dapat dipilih model terbaik yang sesuai dengan kebutuhan. Dengan implementasi model-model pada setiap skema diharapkan dapat menghasilkan model yang akurat dan dapat memprediksi dengan baik.

3.3.4 Analisis Model Skema

Tahap ini merupakan tahap di mana implementasi model pada tiap skema telah selesai dilakukan. Analisis model skema ini dilakukan untuk mengetahui kelemahan dan kekuatan masing-masing model dalam mengatasi permasalahan yang dihadapi. Analisis model skema yang dilakukan meliputi analisis evaluasi model, interpretasi performa model, serta optimasi hyper-parameter. Dalam analisis evaluasi model, peneliti akan melakukan perhitungan terhadap nilai *loss*, *bias*, dan standar deviasi untuk diperoleh nilai rerata-nya. Dari nilai rerata tiap komponen yang dihasilkan, maka akan dibandingkan satu sama lain model mana yang memiliki hasil rerata yang terbaik.

Selain itu, interpretasi performa model dapat memberikan pemahaman lebih dalam tentang karakteristik data yang digunakan untuk melatih dan menguji model. Dalam hal ini, peneliti dapat melakukan analisis terhadap distribusi data, pola-pola data, serta karakteristik data yang mempengaruhi performa model.

Terakhir, optimasi hyper-parameter dapat membantu praktisi ilmu data dalam menentukan strategi untuk meningkatkan performa model. Analisis ini meliputi faktor-faktor seperti karakteristik *dataset*, arsitektur model, serta teknik-teknik *preprocessing* data yang digunakan. Dengan melakukan analisis model skema, peneliti dapat memperoleh wawasan yang lebih dalam tentang performa model dan faktor-faktor yang mempengaruhinya. Hal ini dapat membantu dalam pengambilan keputusan terkait pemilihan model yang tepat, serta strategi untuk meningkatkan performa model.