BAB 2
LANDASAN TEORI

Landasan teori yang digunakan untuk mendukung pengerjaan penelitian ini, dijabarkan secara mendalam dan spesifik. Mengenai Music Emotion Recognition, ektraksi fitur *mel-spectrogram* dan *chroma*, *Convolutional Neural Network*, dan *Convolutional Recurrent Neural Network* serta metrik yang digunakan selama proses penelitian.

2.1 Music Emotion Recognition (MER)

Music Emotion Recognition merupakan sistem dari proses identifikasi emosi dan memodelkannya menjadi data yang bisa dibaca oleh algoritma dengan menggunakan teknik pemrosesan sinyal audio dan *machine learning* [7]. MER bekerja dengan mengekstraksi emosi dari berbagai fitur seperti *chroma*, MFCC, dan *spectral*. Fitur-fitur yang ada tersebut akan dianalisis dan diproses melalui komputasi untuk mendapatkan label emosi [8].

Model MER terbagi menjadi 2, *Categorical* dan *Dimensional*. *Categorical* mengklasifikasikan emosi yang terdapat pada lagu menjadi label emosi diskrit, seperti*happy, sad, angry, relaxed*. *Categorical* mengklasifikasikan emosi ke dalam ruang kontinu 2 dimensi menjadi regresi *valence* dan *arousal* [7][8].

2.1.1 Categorical

Gambar 2.1. Hevner's emotion model, 1936

Sumber: [18]

Gambar 2.1 merupakan gambar dari *categorical emotion model* yang dikemukan oleh Hevner pada tahun 1936. *Categorical* merepresentasikan emosi ke dalam model diskrit yang berisi dari emosi-emosi yang ada, Hevner mengklasifikan emosi menjadi 8 kelompok, *sad, longing, lyric, jumping, joyful, warm, majestic, chanting*[8][11]. Model *categorical* menggunakan pendeketan terhadap emosi sesuai dengan refleksi dari sifat alaminya dan emosi yang berlawanan, berlawanan pula sifatnya [7][18].

Pada gambar 2.1 terlihat kuadran 2 merupakan *sad* yang berisi emosi seperti

sadness, depression, dan tragic. Di sisi yang berlawanan kuadran 6 merupakan *joyful* yang berisi emosi seperti *happy, cheerful, dan bright*. Berdasarkan penelitian menggunakan Hevner emotion model, musik dengan kunci mayor membuat orang bahagia, kunci minor membuat perasaan menjadi sedih. Tempo juga menjadi kunci dari emosi yang dirasakan manusia dalam mendengarkan lagu, ketika mendengarkan lagu dengan tempo yang cepat, orang akan cenderung memaknai perasaan senang, dan tempo yang rendah orang akan cenderung berada dalam emosi sedih [18][19]. Berikut uraian untuk ke-delapan penjelasan emosi berdasarkan *Hevner's Emotion Model*:

1. Chanting (khusyuk-berirama)
   Emosi yang bersifat repetitif, khidmat, dan menyerupai nyanyian ritual, memberi kesan spiritual atau meditatif. Contoh: "Gregorian Chant" (paduan suara monastik), Ave Maria (Franz Schubert).

2. Sad (sedih)
   Emosi yang mengekspresikan kesedihan, duka, atau melankolis, umumnya ditandai tempo lambat dan tangga nada minor. Contoh: "Someone Like You" (Adele), "Hurt" (Johnny Cash).

3. Longing (kerinduan)
   Emosi yang menyiratkan kerinduan, kehampaan, atau kenangan yang mendalam, seringkali terasa reflektif dan penuh perasaan. Contoh: "Yesterday" (The Beatles), "Right Here Waiting" (Richard Marx).

4. Lyric (liris)
   Emosi yang lembut, mengalir, dan puitis, menekankan keindahan melodi yang tenang dan ekspresif. Contoh: "Clair de Lune" (Claude Debussy), "The Scientist" (Coldplay).

5. Jumping (riang-energik)
   Emosi yang dinamis, ceria, dan penuh gerak, ditandai ritme yang hidup dan tempo cepat. Contoh: "Happy" (Pharrell Williams), "Can't Stop the Feeling!" (Justin Timberlake).

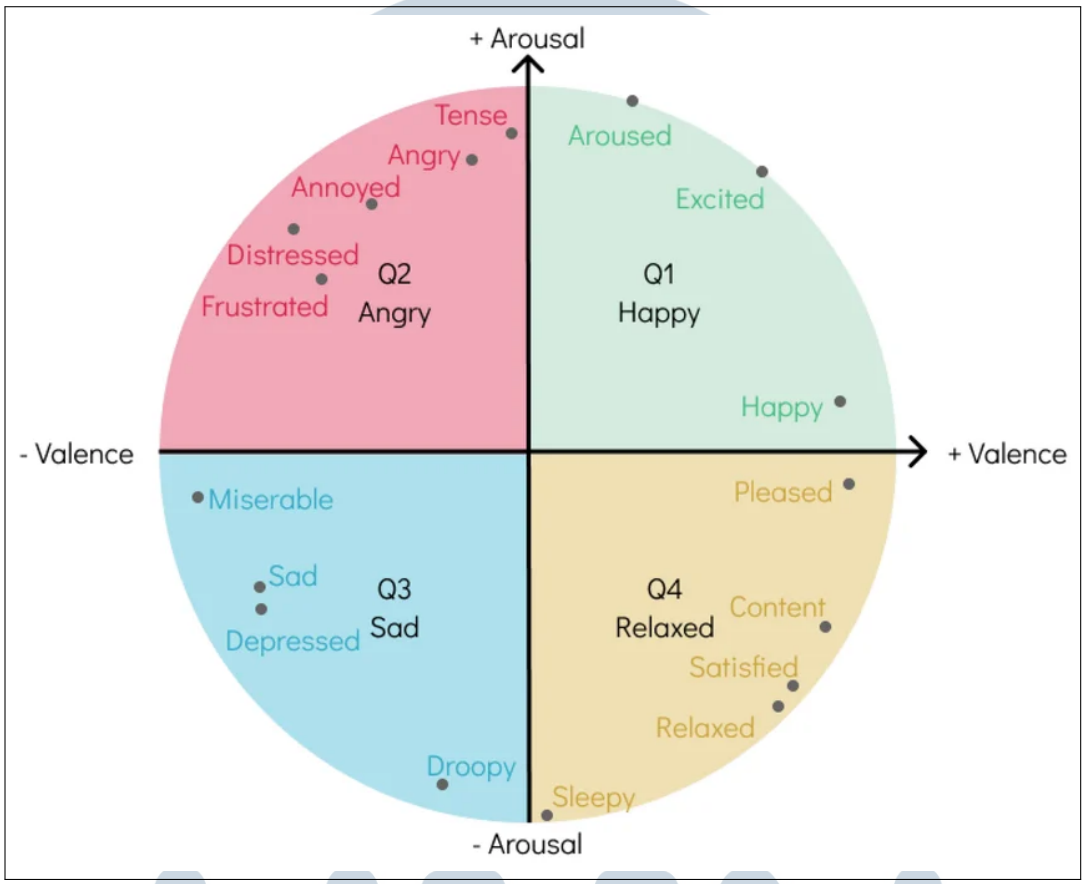
6. Joyful (gembira)
   Emosi kebahagiaan yang cerah dan positif, memancarkan keceriaan dan optimisme. Contoh: "Here Comes the Sun" (The Beatles), "Walking on Sunshine" (Katrina and the Waves).

7. Warm (hangat)
   Emosi yang lembut, penuh kasih, dan menenangkan, memberi kesan nyaman dan intim. Contoh: ”Thinking Out Loud” (Ed Sheeran), ”Can’t Help Falling in Love” (Elvis Presley).

8. Majestic (agung)
   Emosi yang megah, khidmat, dan berwibawa, seringkali terdengar besar dan penuh kekuatan. Contoh: ”Indonesia Raya” (Lagu Kebangsaan Indonesia), ”We Are the Champions” (Queen).

Pendekatan kategorikal seperti model Hevner ini memiliki kelebihan berupa kemudahan interpretasi, namun dianggap kurang fleksibel dalam merepresentasikan emosi musik yang bersifat kontinu dan dapat berubah seiring waktu. Dengan keterbatasan tersebut, digunakan pendekatan dimensional (valence–arousal) pada penelitian ini.

2.1.2 Dimensional



Gambar 2.2. Russel's Emotion Model, 1980

Sumber: [20]

Gambar 2.2 merupakan *dimensional emotion model* yang dikemukakan oleh Russel pada tahun 1980. Berbeda dengan pendekatan kategorikal yang merepresentasikan emosi sebagai label diskrit, pendekatan *dimensional* merepresentasikan emosi ke dalam ruang dua dimensi yang bersifat kontinu, sehingga mampu menangkap perubahan emosi yang halus dan tidak terbatas pada kategori tertentu [5]. Model ini disusun atas dua sumbu utama, yaitu *valence* pada sumbu horizontal dan *arousal* pada sumbu vertikal.

Valence (sumbu x) merepresentasikan sifat emosi pada rentang negatif hingga positif, yakni sejauh mana suatu emosi terasa menyenangkan atau tidak menyenangkan. Sementara itu, *arousal* (sumbu y) merepresentasikan intensitas atau tingkat aktivasi emosi, dari kondisi tenang hingga aktif/energik [8][11]. Rentang nilai kedua sumbu ini umumnya berada dalam skala $[-1, +1]$ atau $[0, 1]$ tergantung

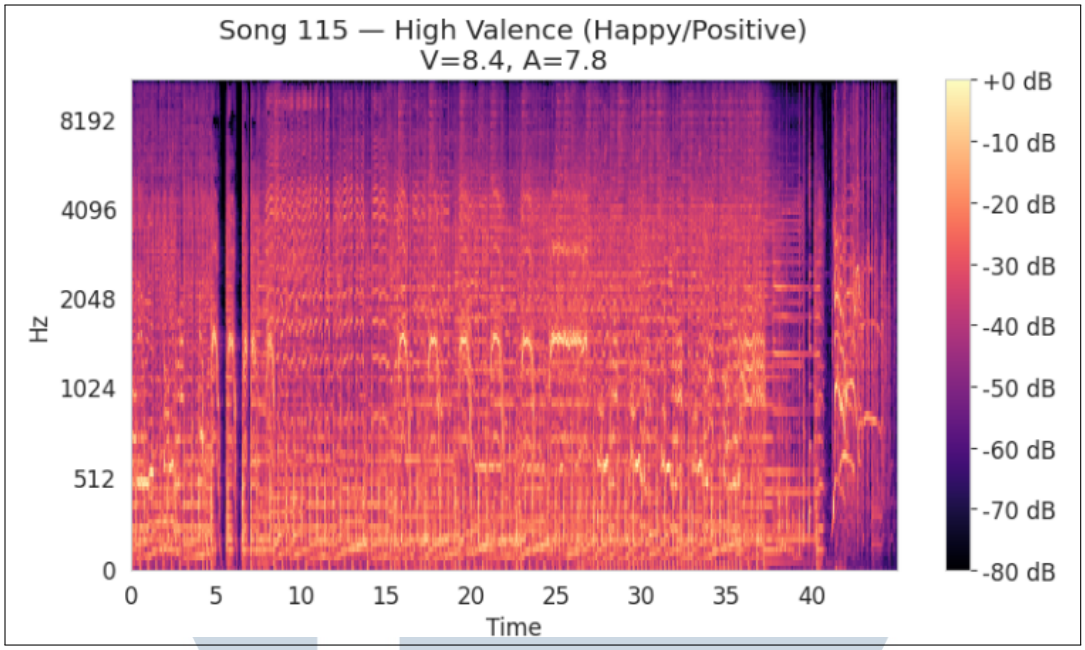
dataset yang digunakan. Pada skala $[-1, +1]$, nilai *valence* $+1$ menunjukkan emosi yang sangat positif seperti *happy*, nilai -1 menunjukkan emosi yang sangat negatif seperti *sad*, dan nilai 0 menunjukkan kondisi netral; demikian pula *arousal* $+1$ mengindikasikan emosi yang sangat aktif dan *arousal* -1 menunjukkan kondisi yang sangat tenang [5][6].

Kombinasi kedua sumbu tersebut membentuk empat kuadran emosi. Kuadran 1 (*valence* positif, *arousal* tinggi) merepresentasikan emosi *happy*, contohnya lagu *"Happy"* (Pharrell Williams) yang ceria dan berenergi. Kuadran 2 (*valence* negatif, *arousal* tinggi) merepresentasikan emosi *angry* atau *tense*, contohnya lagu *"Chop Suey"* (System of A Down) yang agresif dan bertenaga. Kuadran 3 (*valence* negatif, *arousal* rendah) merepresentasikan emosi *sad*, contohnya lagu *"Someone Like You"* (Adele) yang lambat dan melankolis. Kuadran 4 (*valence* positif, *arousal* rendah) merepresentasikan emosi *relaxed*, contohnya lagu *"One Summer's Day"* (Joe Hisaishi) yang tenang namun terasa menyenangkan [6].

Kemampuan model dimensional dalam merepresentasikan emosi secara kontinu inilah yang menjadikannya lebih sesuai untuk menangkap dinamika emosi musik dibandingkan pendekatan kategorikal, sehingga model ini digunakan sebagai dasar dalam penelitian ini.

2.2 Mel-Spectrogram

Sinyal suara dari sebuah audio berbentuk amplitudo terhadap waktu, dan untuk memahami isi dari frekuensi sebuah suara digunakan *Short-time Fourier Transform* (STFT). STFT mengubah sinyal audio menjadi potongan dengan jangka waktu yang pendek (misal 20 - 30 milidetik), pada jendela potongan tersebut dihitung frekuensi dari sinyal audio yang ada [21].



Gambar 2.3. Mel-Spectrogram

Gambar 2.3 merupakan gambar hasil dari proses ekstraksi mel-spectrogram yang dilakukan pada DEAM dataset untuk song ID: 115. Representasi *mel-spectrogram* tidak hanya berfungsi sebagai *input* numerik bagi model, tetapi juga memuat pola visual yang dapat diamati dan diinterpretasikan keterkaitannya dengan emosi yang direpresentasikan sebuah lagu. Pada *mel-spectrogram*, sumbu horizontal merepresentasikan waktu, sumbu vertikal merepresentasikan frekuensi dalam skala mel, dan intensitas warna merepresentasikan besarnya energi (dalam satuan dB) pada suatu frekuensi dan waktu tertentu, warna yang lebih terang menunjukkan energi yang lebih tinggi.

Intensitas dan sebaran dari energi merupakan indikator untuk arousal, aspek ini terlihat pada tingkat kepadatan dan kecerahan energi pada *spectrogram*. Lagu dengan *arousal* tinggi, seperti musik yang energik dan bertempo cepat, cenderung menampilkan warna yang lebih terang, padat, dan menyebar hingga ke frekuensi tinggi, yang mengindikasikan besarnya energi akustik. Sebaliknya, lagu dengan *arousal* rendah, seperti musik yang tenang, cenderung menampilkan warna yang lebih gelap dan energi yang terkonsentrasi pada frekuensi rendah. Karena itu, dimensi *arousal* dapat diamati secara langsung dari tingkat kecerahan dan sebaran energi pada *mel-spectrogram*.

Pola temporal dan ritme dapat terlihat oleh *mel-sepctrogram*. Struktur vertikal yang muncul secara berulang dan teratur pada sumbu waktu

mengindikasikan pola ritmis atau ketukan (*beat*) sebuah lagu. Garis-garis energi vertikal yang rapat menunjukkan tempo yang cepat, sedangkan pola yang lebih renggang menunjukkan tempo yang lambat. Perubahan intensitas energi di sepanjang sumbu waktu juga dapat memperlihatkan dinamika lagu, misalnya transisi dari bagian yang tenang menuju bagian klimaks yang lebih intens.

Sebaran energi pada sumbu frekuensi merepresentasikan karakteristik timbre atau warna suara. Energi yang dominan pada frekuensi tinggi cenderung memberikan kesan suara yang cerah atau tajam, sedangkan energi yang dominan pada frekuensi rendah memberikan kesan suara yang berat atau lembut. Karakteristik timbre ini turut berkontribusi terhadap nuansa emosi yang dipersepsikan pendengar.

Meskipun *mel-spectrogram* sangat informatif dalam merepresentasikan *arousal*, representasi ini memiliki keterbatasan dalam menggambarkan dimensi *valence* secara langsung. Hal ini disebabkan *valence* lebih banyak ditentukan oleh aspek harmoni dan tonalitas, seperti penggunaan tangga nada mayor atau minor, yang tidak terlihat secara eksplisit sebagai pola visual pada *mel-spectrogram*. Dua lagu dengan tingkat energi yang serupa dapat memiliki *valence* yang berbeda, dengan satu lagu terdengar gembira dan lainnya terdengar sedih, namun keduanya dapat menampilkan pola *mel-spectrogram* yang relatif mirip. Keterbatasan ini yang mendasari penggunaan fitur *chroma* sebagai pelengkap untuk menangkap informasi harmoni yang menjadi kunci bagi prediksi *valence*.

$$X(m,k) = \sum_{n=0}^{N-1} x[n+mR]\, w[n]\, e^{-j\frac{2\pi}{N}kn} \tag{2.1}$$

Rumus 3.8 merupakan rumus perhitungan paling dasar untuk mengubah sinyal audio menjadi data yang bisa dibaca oleh komputer. $X(m,k)$ merupakan nilai output berupa angka kompleks pada matriks *spectrogram* yang merepresentasikan distribusi energi sinyal. m mewakili indeks frame waktu, nilai $m = 0,1,2,...$ merepresentasikan urutan potongan sinyal dari awal hingga akhir sebuah lagu. k mewakili indeks frekuensi, merepresentasikan tingkat frekuensi tertentu yang sedang dianalisis pada *frame* ke *-m*. $x[n]$ merupakan data audio digital asli, n mewakili indeks sampel audio pada domain waktu. $x[n+mR]$ berfungsi sebagai pemotong dan menghaluskan sinyal audio. $n+mR$ berfungsi sebagai melakukan *sliding* atau pergesaran pembacaan sinyal dari *frame* 1 ke *frame* ke dengan perhitungan pergeseran sebesar R. w: Jenis jendela yang digunakan.

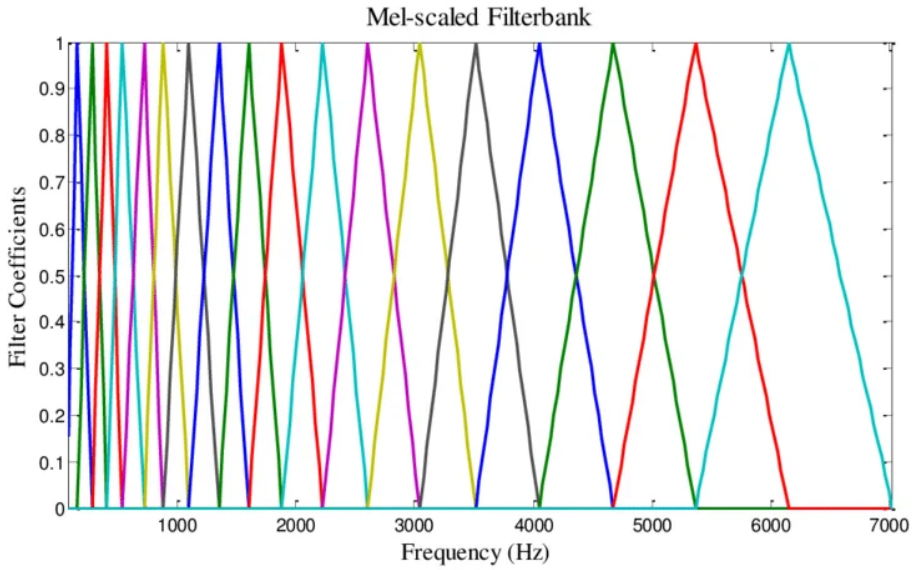
Fungsinya membuat ujung potongan sinyal melandai ke angka nol untuk mencegah munculnya *noise* frekuensi palsu akibat pemotongan kasar. R: Hop length, ini menentukan seberapa jauh jendela bergeser untuk mengambil bingkai berikutnya. mR: Menentukan titik pusat di mana jendela tersebut saat ini diletakkan pada keseluruhan sinyal. $e^{-j\frac{2\pi}{N}kn}$ berdasarkan Rumus Euler, hasil bilangan eksponensial kompleks ini sebenarnya adalah representasi dari gelombang Sinus dan Cosinus murni pada frekuensi tertentu. j bilangan imajiner ($\sqrt{-1}$). N jumlah titik sampel yang dianalisis dalam satu bingkai (N_{FFT}). Mengalikan potongan audio dengan gelombang sinus/cosinus murni berfrekuensi k. Jika audio tersebut mengandung frekuensi yang sama dengan gelombang pelacak, hasil perkaliannya akan bernilai besar (terjadi resonansi). $\sum_{n=0}^{N-1}$ menjumlahkan hasil perkalian dari perhitungan pada sampel demi sampel. Penjumlahan inilah yang menghasilkan satu nilai titik skalar tunggal pada matriks output STFT(m,k) [21].

Hasil perhitungan STFT menghasilkan *spectrogram* yang divisualisasi menjadi frekuensi terhadap waktu, serta warna yang merepresentasikan amplitudo (dB) yang semakin terang warnanya semakin tinggi desibel yang dihasilkan[14]. *Spectrogram* menggunakan skala frekuensi linear (Hertz), pendengaran manusia menggunakan frekuensi non-linear, manusia tidak bisa membedakan suara antara 10.000 Hz dan 10.100 Hz oleh karena itu digunakan *mel scale* sebagai representasi dari pendengaran manusia terhadap audio [12][14]. *Mel scales* merupakan skala perseptual nada yang dirancang agar jarak yang sama pada skala tersebut dirasakan memiliki jarak nada yang sama oleh pendengaran manusia.

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \tag{2.2}$$

Rumus 2.2 merupakan rumus yang digunakan untuk merubah spectrogram menjadi *mel scale* melalui *mel filter bank*, hal ini digunakan untuk merepresentasikan pendengaran manusia secara asli kepada sistem komputasi. $M(f)$ merupakan nilai frekuensi akhir dalam *mel scale* (output). Ini adalah representasi dari ”nada” seperti yang dirasakan oleh telinga manusia. f frekuensi akustik asli dari sinyal audio dalam satuan Hertz (Hz) (input). 2595 merupakan angka konstanta penskalaan (scaling constant). Angka ini dipilih secara empiris melalui eksperimen psikoakustik agar referensi standarnya terpenuhi, yaitu 1000 Hz harus persis sama dengan 1000 Mel. Pendengaran manusia tidak mempersepsikan kenaikan suara secara linear, $\log_{10}$ digunakan untuk menempatkan frekuensi yang

tinggi menjadi segitiga yang kecil dan rapat. 700 merupakan titik tengah atas pendengaran manusia, dibawah 700 manusia dapat mendengar secara *linear* dan diatas 700 manusia mendengar secara logaritmik. 1 sebagai pencegah error.



Gambar 2.4. Mel Filter Bank

Sumber: [22]

Gambar 2.4 adalah representasi dari *Mel Filter Bank*. *Mel filter bank* merupakan set filter berbentuk segitiga (segitiga ini lebih rapat di frekuensi rendah dan semakin melebar di frekuensi tinggi) dikalikan dengan spektrogram linear. Filter ini berfungsi meringkas dan mengelompokkan frekuensi-frekuensi tinggi menjadi satu kesatuan[22].

2.2.1 Chroma dan Chromagram

Hasil ekstraksi dari *mel-spectrogram* menghasilkan matriks 2D yang merepresentasikan distribusi energi suara berdasarkan persepsi pendengaran manusia yang bisa dibaca oleh model. Meskipun demikian, pelatihan model yang hanya bergantung pada representasi *mel-spectrogram* sering kali menemui kendala. Model sering kali kesulitan menangkap perbedaan nilai *valence* pada lagu-lagu yang memiliki tingkat *arousal* yang mirip jika hanya menggunakan *mel-spectrogram*, sehingga rentan terjadi kesalahan prediksi [23]. Untuk mengatasi kelemahan tersebut dan meminimalisir galat prediksi, ekstraksi fitur *chroma* diterapkan untuk memperkaya representasi audio. Fitur *chroma* memproyeksikan

seluruh spektrum energi suara ke dalam 12 *bins* yang mewakili 12 kelas nada (*pitch classes*) semiton dalam satu oktaf standar pada sistem tala musik Barat, yaitu C, C#, D, D#, E, F, F#, G, G#, A, A#, dan B [24].

Penggunaan fitur *chroma* mampu untuk meningkatkan model dalam melakukan prediksi terhadap emosi musik dikarenakan fitur *chroma* memperjelas dari struktur melodi, progresi akord, dan harmoni sebuah lagu. Literatur pada bidang *Music Emotion Recognition* (MER) menunjukkan bahwa informasi harmoni berkorelasi sangat tinggi dengan dimensi *valence*, yakni tingkat kebahagiaan atau kesedihan sebuah lagu [25]. Secara umum, progresi akord mayor merepresentasikan nilai *valence* yang tinggi sehingga memberikan persepsi emosi yang positif atau bahagia. Sebaliknya, akord minor cenderung merepresentasikan nilai *valence* yang rendah, yang mendasari persepsi emosi sedih atau negatif [23].

Dalam literatur pemrosesan sinyal audio, penting untuk membedakan terminologi antara *chroma* dan *chromagram*. Fitur *chroma* merujuk pada vektor satu dimensi yang merepresentasikan distribusi energi pada 12 kelas nada di satu titik waktu (*frame*) tertentu. Sementara itu, *chromagram* merupakan matriks dua dimensi berukuran $12 \times T$ yang terbentuk dari gabungan vektor-vektor *chroma* secara berurutan sepanjang sumbu waktu (T) dari awal hingga akhir lagu.

Proses komputasi ekstraksi ini terjadi beriringan dengan pemrosesan *mel-spectrogram*. Nilai frekuensi yang dihasilkan oleh perhitungan *Short-Time Fourier Transform* (STFT) diambil dan dimasukkan ke dalam algoritma pembentukan *chroma*. Jika pada frekuensi 440 Hz (nada A) memiliki intensitas energi yang tinggi pada suatu *frame*, matriks *chromagram* akan menampilkan warna yang terang pada titik waktu dan kelas nada tersebut. Pola kombinasi dari intensitas nada-nada dominan inilah yang secara kolektif merepresentasikan progresi akord yang dimainkan sepanjang lagu, memberikan panduan sinyal sekuensial yang krusial bagi lapisan konvolusi pada jaringan saraf tiruan untuk memprediksi emosi lagu secara presisi.

Dalam komputasinya, pemetaan spektrum frekuensi menjadi kelas nada dilakukan dengan terlebih dahulu mengonversi nilai frekuensi ke dalam skala *pitch* MIDI [24] melalui persamaan berikut:

$$p(f) = 69 + 12\log_2\left(\frac{f}{f_{\text{ref}}}\right) \tag{2.3}$$

Persamaan 2.3 menjelaskan perhitungan untuk mencari nomor indeks nada pada skala MIDI ($p(f)$). Konstanta 69 merupakan nomor indeks MIDI yang secara baku

ditetapkan untuk nada A4, sedangkan 12 melambangkan jumlah pembagian nada (semiton) dalam satu oktaf. Variabel f adalah nilai frekuensi aktual yang dihasilkan oleh STFT, dan f_{ref} merupakan frekuensi referensi global yang umumnya dipatok pada 440 Hz. Setelah nomor indeks skala MIDI didapatkan, algoritma melakukan pelipatan oktaf (*octave folding*) untuk memasukkan nada tersebut ke dalam 12 kelas nada dasar menggunakan operasi modulo:

$$c = p \pmod{12} \tag{2.4}$$

Persamaan 2.4 menunjukkan proses perhitungan indeks kelas nada (c). Operasi $\pmod{12}$ memastikan bahwa seluruh tinggi-rendahnya oktaf diabaikan, sehingga rentang nilai yang dihasilkan selalu berada secara melingkar di antara 0 hingga 11 (di mana C=0, C#=1, ..., B=11). Hal ini menjadikan matriks *chromagram* yang dihasilkan sangat stabil dan kebal terhadap perubahan timbre dari berbagai instrumen musik, namun tetap sangat sensitif terhadap perubahan harmoni [24].

Tabel 2.1. 12 Kelas Nada (Pitch Class) dan Frekuensi Referensi

Indeks Chroma (c)	**Kelas Nada (Pitch Class)**	**Frekuensi Referensi (Hz)**
0	C	261,63
1	C# / Db	277,18
2	D	293,66
3	D# / Eb	311,13
4	E	329,63
5	F	349,23
6	F# / Gb	369,99
7	G	392,00
8	G# / Ab	415,30
9	A	440,00 (f_{ref})
10	A# / Bb	466,16
11	B	493,88

Sumber: ENTEL's Inspired Acoustics [26]

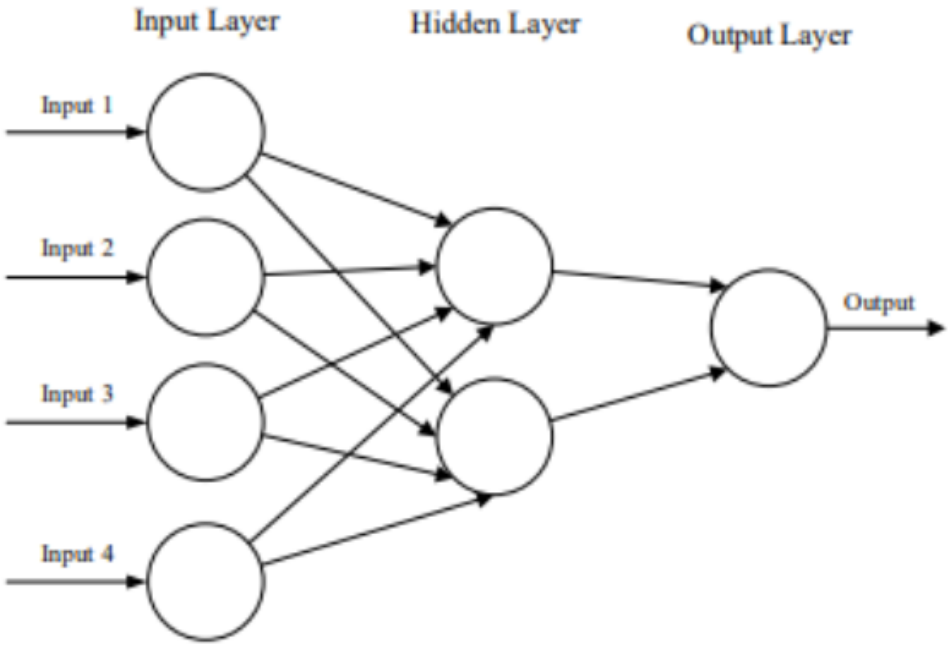
2.3 *Convolutional Neural Network*

Convolutional Neural Networks (CNNs) merupakan varian ANN yang dirancang khusus untuk pengenalan pola pada gambar visual [27]. Perkembangan CNN pertama kali dilakukan di Fukushima, Jepang dengan nama *Neocognitron* pada tahun 1980, yang tidak terpengaruh pada perubahan posisi gambar dalam pengenalan pola gambar [28].

CNN lebih efisien untuk tugas pemrosesan gambar karena mampu mengenkripsi fitur-fitur khusus gambar dalam arsitekturnya, sehingga mengurangi jumlah parameter yang diperlukan. Kelebihan ini membuat CNN lebih cocok untuk memproses data gambar yang kompleks dengan melibatkan lapisan khusus seperti *convolutional layer* untuk mengekstraksi fitur lokal dari gambar, *pooling layer* untuk mereduksi dimensi data (downsampling), konektivitas lokal, dan lapisan mendalam lainnya [29].

Jenis arsitektur *Deep Learning* sangat efektif dalam mengolah data yang memiliki struktur spasial seperti gambar dan video [30]. CNN memiliki kemampuan untuk memahami dan mengekstraksi pola serta fitur-fitur penting secara hierarkis dari data gambar yang diukur menjadi output yang diinginkan [31]. Akan tetapi struktur dasarnya masih serupa dengan FNN, karena aliran data bergerak maju dari lapisan input menuju lapisan output tanpa adanya loop atau aliran balik [32].

CNN dirancang untuk meneliti citra visual pada gambar, namun arsitektur ini mampu mengadaptasi terhadap sinyal audio. Dalam memproses sinyal audio agar dapat terlihat oleh arsitektur, dilakukan pemrosesan pengubahan sinyal audio 1D menjadi representasi spasial 2D berupa *mel-spectrogram*. CNN memperlakukan hasil *mel-spectrogram* seperti citra visual gambar dengan representasi waktu (sumbu horizontal) dan frekuensi (sumbu vertikal).



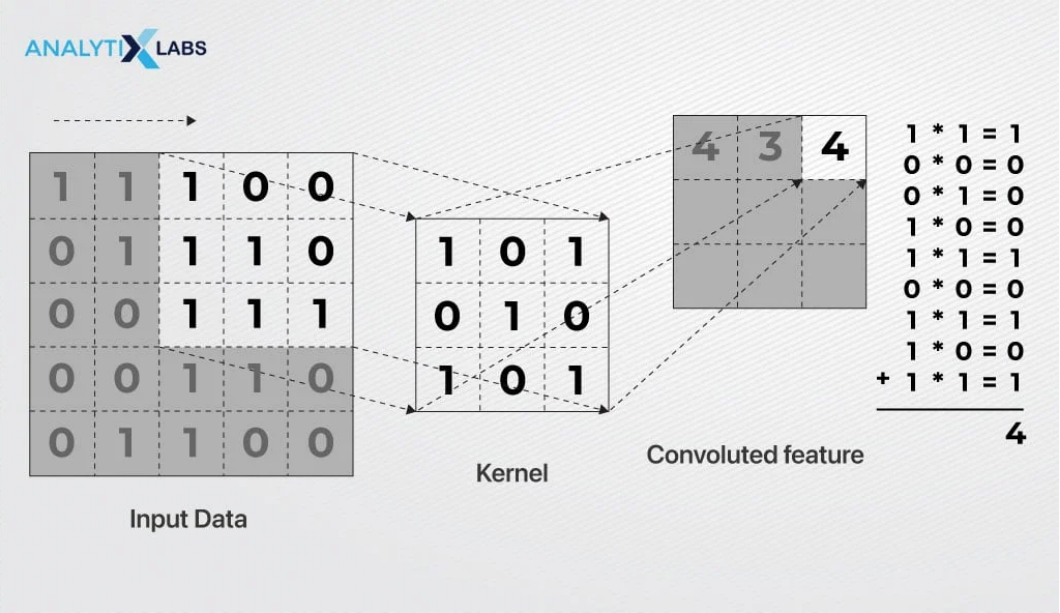
Gambar 2.5. Struktur Umum ANN

Sumber: [32]

Gambar 2.5 merupakan struktur umum dari ANN. ANN berisi dari lapisan input dan output yang diproses didalam beberapa lapisan dalam hidden layer [32]. Model umum CNN terdiri dari empat lapisan; *convolutional layer, pooling layer, ReLU activation, fully connected layer*

2.3.1 *Convolutional Layer*

Convolutional layer merupakan tahap pertama dalam CNN, pada lapisan ini convolutional *layer* akan menggunakan filter untuk memproses gambar yang akan digunakan untuk *layer* berikutnya. Filter tersebut akan bergeser bertahap pada seluruh gambar dan mengekstrak fitur-fitur dari *region-region* kecil yang tumpang tindih. Hasil dari pemfilteran ini adalah feature map [33].



Gambar 2.6. Convolutional Layer

Sumber: [34]

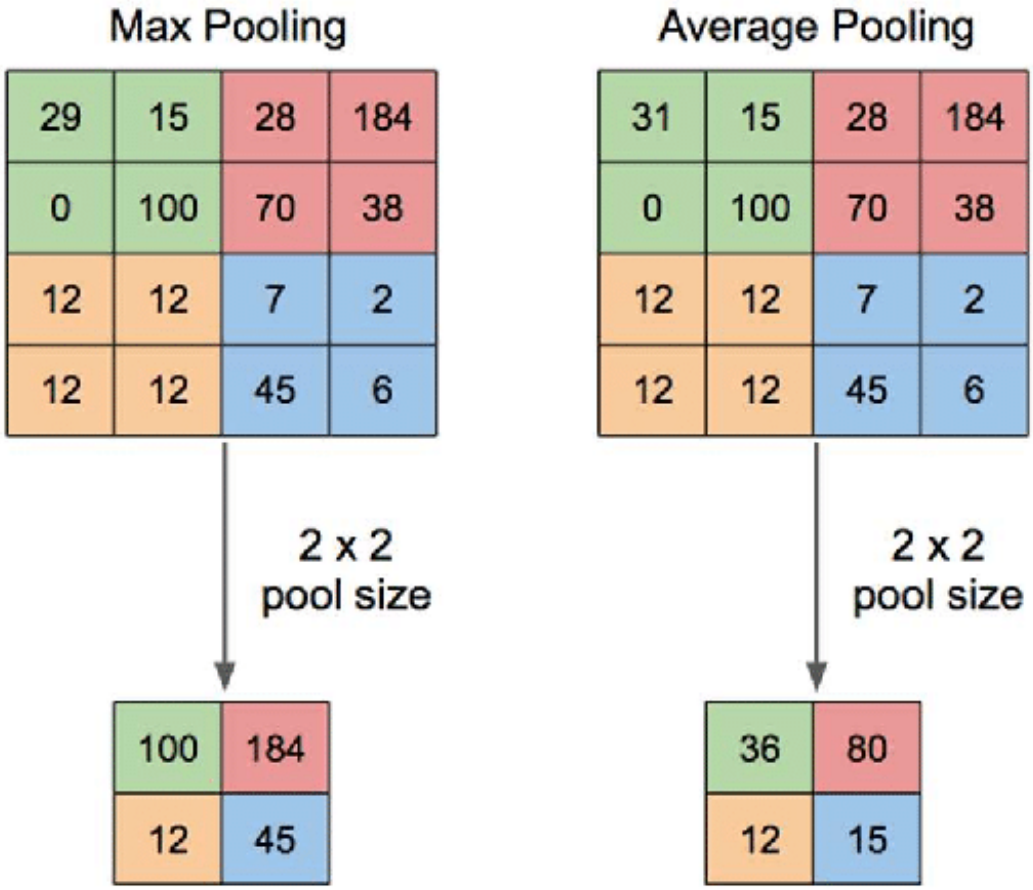
Gambar 2.6 memperlihatkan perhitungan yang dilakukan oleh setiap filter untuk setiap piksel pada gambar. Setelah perhitungan selesai, filter akan bergeser ke nilai input sebelah kanannya dan melakukan perhitungan yang sama untuk menentukan nilai output matrix dari gambar yang telah di proses oleh *convolutional layer*. Proses ini serupa dengan perhitungan perkalian matriks.

$$Y(i,j) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} X(i+m, j+n) W(m,n) + b \tag{2.5}$$

Rumus 2.5 merupakan rumus untuk mengubah gambar menjadi *feature map* oleh *convolutional layer* yang bisa di proses oleh *pooling layer*. $Y_{i,j}$ merupakan nilai *output* (*feature map*) dengan i, j adalah baris dan kolom. $\sum_{m=0}^{M-1} \sum_{n=0}^{N-1}$ melambangkan operasi akumulasi pada grid 2 dimensi sebesar filter yang digunakan. $X(i+m, j+n)$ merupakan matriks input, indeks $i+m$ dan $j+n$ mendefinisikan area spesifik (seukuran filter) pada matriks input yang saat ini sedang ditimpa oleh filter ($M \times N$). Saat indeks i dan j berubah, area ini seolah-olah bergeser (*sliding*). $W(m,n)$ merupakan matriks hasil dari filter yang dilakukan. b adalah bias yang berfungsi untuk menggeser kurva aktivasi, memberikan fleksibilitas ekstra bagi jaringan saraf untuk menyesuaikan hasil *output*.

2.3.2 *Pooling Layer*

Setelah *convolutional layer*, dilakukan *pooling layer*, pada lapisan ini gambar yang sebelumnya telah diproses pada tahap konvolusi, dikurangi lagi dari dimensi spasialnya dan melakukan *down-sampling* pada *feature map*. Pada tahap *pooling layer*, tidak hanya menambah jangkauan penglihatan dari kernel konvolusi sebelumnya, tahap ini juga memudahkan kompleksitas komputasi dalam pemprosesan gambar yang detail dikarenakan *pooling layer* mengurangi resolusi *feature map* dengan memastikan fitur - fitur gambar yang penting tetap ada untuk diproses di layer berikutnya [33]. Terdapat beberapa tipe dalam *Pooling Layer*, *Max Pooling*, *Min Pooling*, *Average Pooling*, dan *Global Pooling*. Jenis yang paling umum digunakan adalah *Max Pooling*, jenis ini menggunakan nilai maksimum dari setiap *sub-region* pada *feature map* [35]. *Max pooling* berguna untuk menonjolkan fitur akustik yang paling dominan, seperti frekuensi nada utama atau energi ketukan perkusi yang kuat, sekaligus mempertahankan invarian terhadap pergeseran temporal (waktu) yang minor. *Average pooling* menggunakan *mean* dari hasil *feature map* untuk memperhitukangkan hasil matriks layer. *Average pooling* dapat mengurangi noise yang ada pada gambar, sehingga dapat menghasilkan gambar yang lebih baik [36].



Gambar 2.7. Pooling Layer

Sumber: [37]

Gambar 2.7 memperlihatkan perhitungang *max pooling* dan *average pooling*, *max pooling* mengklasifikan dari matriks 4 x 4 menjadi 2 x 2 dengan menggunakan nilai input terbesar sebagai nilai output. *Average pooling* menggunakan nilai rata-rata (*mean*) dalam menghasilkan nilai output. Pergerakan pada proses perhitungan matriks disebut *stride*, dalam gambar 2.7 dilakukan *stride* = 2, bergeser ke 2 kolom kanan.

2.3.3 *Flatten Layer*

Flatten layer berguna untuk *reshaping*. Lapisan ini berada pada *convolutional layer* dan *pooling layer* dan menghubungkan hasil dengan *fully connected layer*. Hasil dari *convolutional layer* adalah berupa *feature map* yang berindikasi matriks, sedangkan *fully connected layer* hanya dapat menerima garis vektor 1D. Oleh karena hal tersebut, *flatten layer* penting dalam model CNN agar

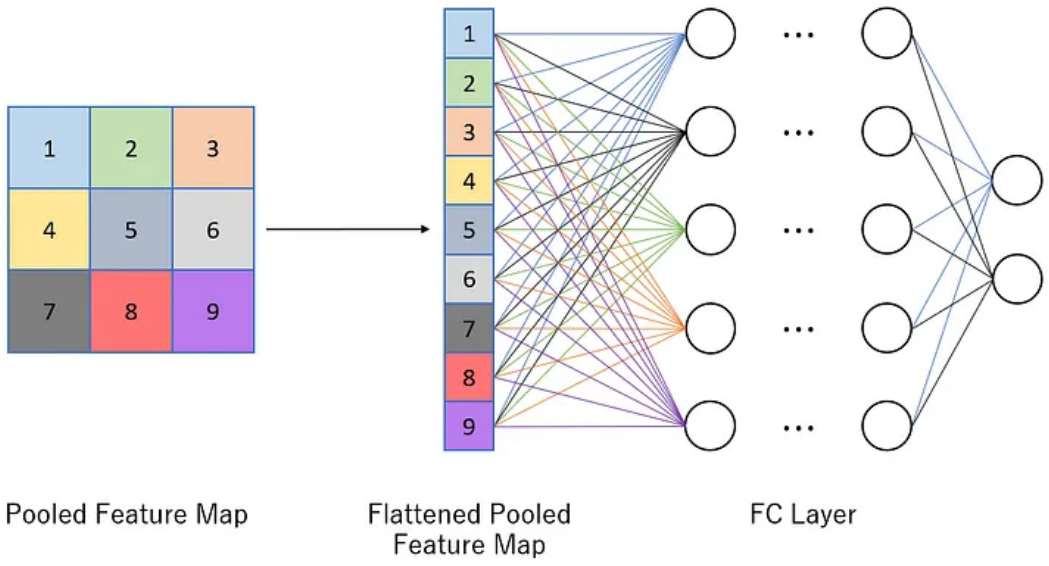
hasil dapat diperlihatkan sesuai dengan kriteria.

2.3.4 *Dropout Layer*

Dropout layer berguna untuk regularisasi, hal ini dilakukan untuk mencegah adanya *overfitting* terhadap data selama proses training. *Dropout layer* ditempatkan setelah proses *convolutional layer* dan *fully connected layer* diterapkan pada iterasi training berikutnya. *Dropout layer* berfungsi dengan mematikan neuron secara acak, hal ini memaksa jaringan lain untuk tidak bergantung dan belajar dari data yang ada.

2.3.5 *Fully Connected Layer*

Hasil akhir dari *convolutional layer* dan *pooling layer*, lapisan ini juga dikenal sebagai *dense layer*. *Fully connected layer* berguna untuk menerima seluruh representasi fitur tingkat tinggi (*high-level features*) yang telah diekstraksi dan direduksi dimensinya oleh lapisan konvolusi dan *pooling* sebelumnya [38]. Lapisan ini bertugas memetakan (*mapping*) fitur-fitur kompleks tersebut ke dalam ruang prediksi akhir. *Output* dari lapisan yang terhubung sepenuhnya dihitung dengan menerapkan bobot pada setiap input, diikuti oleh fungsi aktivasi. Matriks bobot yang menghubungkan neuron di lapisan sebelumnya ke lapisan saat ini dipelajari selama pelatihan [36][39].



Gambar 2.8. Fully Connected Layer

Sumber: [40]

Gambar 2.8 menampilkan hasil matriks dari *pooling layer* lalu di-*flatten* menjadi vektor 1D agar dapat diproses oleh *fully connected layer*.

2.3.6 Fungsi Aktivasi ReLU

Fungsi aktivasi adalah fungsi non-linier yang memungkinkan jaringan menyelesaikan masalah kompleks. Setiap fungsi aktivasi menerima nilai input dan menerapkan operasi matematika. Dalam arsitektur CNN, fungsi aktivasi digunakan setelah proses konvolusi atau *pooling* pada feature map untuk menghasilkan pola fitur. ReLU telah terbukti efektif dalam melakukan operasi non-linear yang bekerja pada setiap elemen piksel dalam *feature map* [41], nilainya diatur kembali menjadi nol jika negatif.

$$f(x) = \max(0, x) \tag{2.6}$$

Rumus 2.6 memperlihatkan perhitungan untuk setiap proses aktivasi pada layer, jika nilai output (x) yang dikeluarkan negatif, maka akan dikembalikan menjadi 0, dan jika nilai output (x) yang dihasilkan positif, hasil tersebut akan tetap dipertahankan [42]. Penerapan ReLU memungkinkan jaringan saraf tiruan untuk memodelkan representasi dan hubungan non-linear yang kompleks dari *feature map*, sekaligus memitigasi masalah gradien yang menghilang (*vanishing gradient problem*) selama proses komputasi propagasi balik (*backpropagation*).

- Fungsi Aktivasi Linear (Identity)
  Berbeda dengan masalah klasifikasi yang membutuhkan keluaran berupa probabilitas diskrit, penelitian ini merupakan masalah regresi yang bertujuan untuk memprediksi nilai intensitas kontinu dari dimensi emosi *valence* dan *arousal*. Oleh karena itu, pada lapisan keluaran (*output layer*) arsitektur model, digunakan fungsi aktivasi *Linear* atau identitas. Fungsi ini beroperasi dengan meneruskan nilai yang dihasilkan oleh neuron sebelumnya tanpa menerapkan transformasi non-linear apa pun.

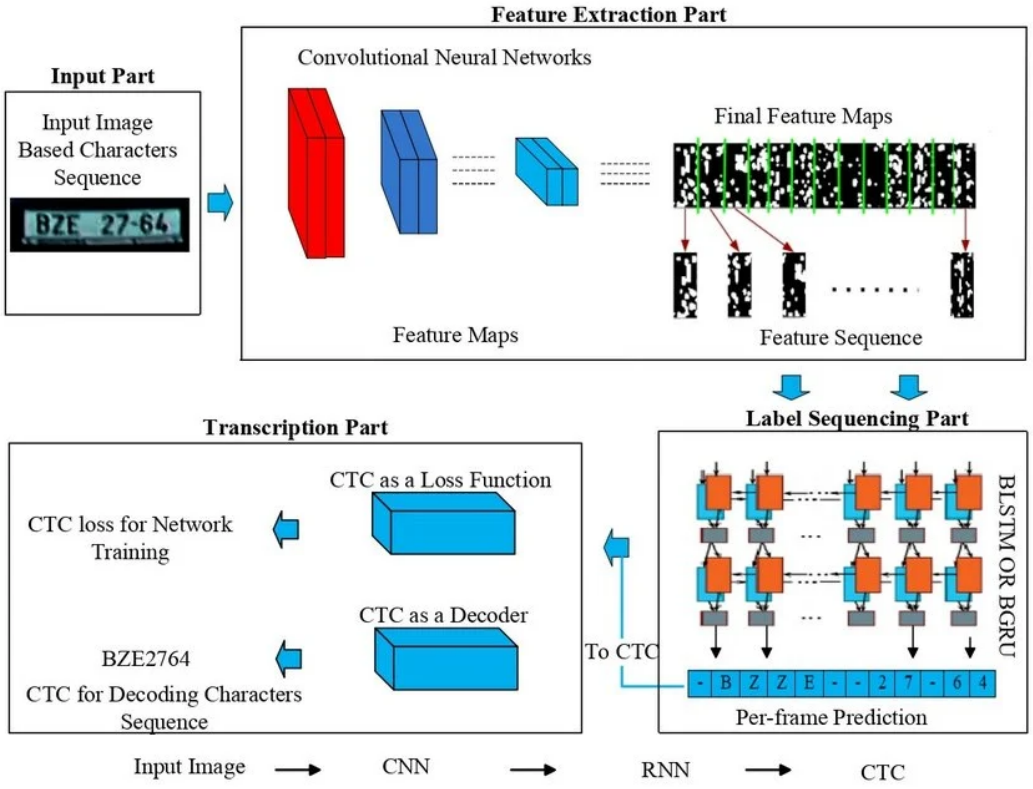
  $$f(x) = x \tag{2.7}$$

  Penggunaan fungsi linear pada simpul (*node*) terakhir memungkinkan jaringan saraf tiruan untuk memetakan hasil ekstraksi fitur secara langsung ke dalam bentuk nilai prediksi emosi yang kontinu. Nilai prediksi kontinu ini

yang selanjutnya akan dievaluasi selisih atau tingkat kesalahannya terhadap anotasi target asli menggunakan fungsi kerugian (*loss function*) berbasis regresi.

2.4 CRNN

Convolutional Recurrent Neural Network (CRNN) merupakan arsitektur jaringan saraf dalam yang menggabungkan keunggulan *Convolutional Neural Network* (CNN) dan *Recurrent Neural Network* (RNN) untuk memproses data sekuensial seperti audio, teks, maupun video [43]. CNN digunakan untuk mempelajari fitur spektral audio, mengekstrak representasi temporal dengan wilayah reseptif terbatas untuk meningkatkan diskriminasi antar bingkai, tetapi informasi globalnya tidak mencukupi [44]. CRNN digunakan untuk memproses ekstraksi pola spasial pada *mel-spectrogram* dengan representasi *time-frequency* dan memodelkan depedensi temporal antar *frame* sehingga mampu untuk menangkap sinyal dinamika [45].



Gambar 2.9. Struktur Umum CRNN

Sumber: [46]

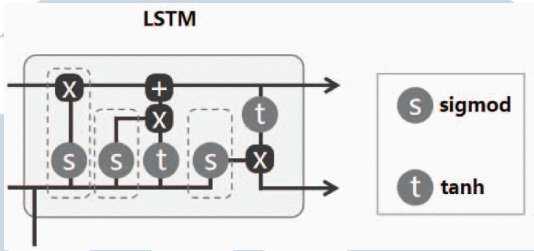
Gambar 2.9 merupakan gambar dari struktur umum pada CRNN. CRNN tersusun dari 3 komponen utama, lapisan pertama *Convolutional Neural Network*, lapisan kedua *Recurrent Neural Network*, dan lapisan ketiga merupakan lapisan nilai output [47]. CRNN menerima input hasil ekstraksi dari *mel-spectrogram*, lapisan CNN mempelajari pola frekuensi pada ekstraksi *mel-spectrogram* tersebut, dimensi waktu dapat diproses sebagai urutan oleh RNN, kemudian satu atau beberapa lapisan LSTM/GRU untuk merangkum informasi temporal, dan akhirnya lapisan penuh-terhubung (fully connected) untuk menghasilkan *output* akhir seperti nilai *valence* dan *arousal* atau probabilitas kelas emosi. Pada fase ini RNN fokus untuk memodelkan informasi konteks yang mengkompensasi hilangnya informasi global dari CNN [44]. Pendekatan ini dirancang untuk mengatasi keterbatasan CNN murni yang cenderung memperlakukan berbagai *frame* waktu secara independen dan kurang mampu merepresentasikan perubahan emosi yang gradual dalam satu lagu.

Studi sebelumnya menunjukkan bahwa kombinasi CNN dan RNN dalam arsitektur CRNN berkinerja lebih baik dibandingkan CNN murni dalam berbagai tugas audio, seperti deteksi peristiwa suara, lokalisasi peristiwa suara, dan klasifikasi suara [48]. CRNN telah terbukti secara efektif mengintegrasikan fitur spasial (pola frekuensi, ritme, dan transisi antar peristiwa) dan temporal (ritme, transisi antar peristiwa) dengan jumlah parameter yang relatif kecil. Regresi emosi dalam ruang *valence-arousal* juga telah dilakukan dengan metode berbasis CNN–RNN dalam bidang *Music Emotion Recognition* (MER). Dataset seperti *MediaEval Emotion in Music* dilaporkan memiliki hasil yang kompetitif [44][45][49].

2.4.1 GRU

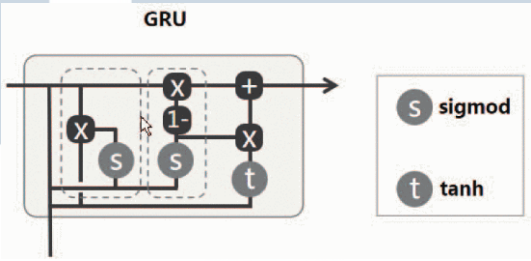
Gated Recurrent Unit pertama kali diperkenalkan oleh Cho, et.al pada tahun 2014. GRU merupakan varian dari RNN yang menggunakan mekanisme gerbang (*gating*) untuk mengatasi masalah *vanishing gradient* dan kesulitan mempelajari dependensi jangka panjang pada data sekuensial sambil mempertahankan kemampuan memodelkan dependensi temporal jangka panjang dengan jumlah parameter yang lebih sedikit. [50]. LSTM menggunakan 3 gerbang utama, *input*, *forget*, dan *output* serta vektor state terpisah. GRU hanya mempunyai dua gerbang untuk mengontrol memori *update gate* dan *reset gate*. Hal ini membuat struktur unit menjadi lebih sederhana dan jumlah parameter menjadi lebih kecil.

GRU sering dipilih dalam dataset yang tidak terlalu banyak dikarenakan GRU mampu melakukan kompromi antara kemampuan pemodelan temporal yang baik dan efisiensi komputasi [51].



Gambar 2.10. Struktur Long Short Term Memory

Sumber: [52]



Gambar 2.11. Struktur Gated Recurrent Unit

Sumber: [52]

Gambar 2.10 merupakan gambar dari struktur *Long Short Term Memory* (LSTM) dan gambar 2.11 merupakan gambar dari struktur *Gated Recurrent Unit* (GRU). *Update gate* pada GRU berfungsi sebagai kontrol terhadap informasi masa lalu yang akan dipertahankan dan dibawa ke langkah berikutnya. *Reset gate* berfungsi sebagai kontrol terhadap informasi masa lalu dalam melakukan perhitungan pada state yang terbaru [53]. Mekanisme ini memungkinkan GRU untuk "mengingat" pola-pola jangka panjang yang penting sekaligus "melupakan" informasi kurang relevan. Karena itu, GRU cukup efektif untuk menangani perubahan emosi yang berlangsung secara bertahap dalam sebuah lagu. Beberapa studi empiris, khususnya pada pemodelan musik polifonik dan pemrosesan sinyal ucapan, menunjukkan bahwa kinerja GRU hampir setara dengan LSTM.

- *Reset Gate*

  Gerbang ini berfungsi untuk menentukan seberapa banyak informasi dari memori masa lalu (langkah waktu sebelumnya, h_{t-1}) yang harus dilupakan

atau diabaikan oleh jaringan [53].

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{2.8}$$

Rumus 2.8 merupakan rumus untuk reset gate pada arsitektur GRU. r_t adalah nilai *reset gate*. Nilai ini didapatkan dari perkalian input saat ini (x_t) dengan matriks parameter bobot W_r, dijumlahkan dengan perkalian memori sebelumnya (h_{t-1}) dengan matriks parameter bobot U_r. Hasilnya dilewatkan pada fungsi aktivasi non-linear sigmoid (σ), yang menekan nilainya berada di antara 0 dan 1. Semakin mendekati 0 nilainya, semakin banyak memori lama yang diabaikan.

- *Update Gate*

  Gerbang ini mengontrol seberapa banyak informasi dari masa lalu (h_{t-1}) yang akan dipertahankan dan dibawa menuju masa depan [53].

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{2.9}$$

Rumus 2.9 merupakan rumus untuk *update gate* pada arsitektur GRU. z_t dikalkulasi menggunakan fungsi sigmoid (σ) beserta matriks bobotnya sendiri (W_z dan U_z). Nilai ini nantinya bertindak sebagai penyeimbang linear antara memori masa lalu dan memori baru.

- *Candidate Hidden State*

  Tahap ini merumuskan kandidat informasi baru yang potensial untuk disimpan [53].

$$\tilde{h}_t = \tanh(W x_t + U(r_t \odot h_{t-1})) \tag{2.10}$$

Rumus 2.10 merupakan rumus untuk menentukan kandidat baru yang berpotensial untuk diteruskan kedalam perhitungan state selanjutnya. Kandidat memori baru ($\tilde{h}_t$) dibentuk dengan menggabungkan input saat ini (x_t) dengan memori lama (h_{t-1}) yang telah dikalikan secara elemen-demi-elemen ($\odot$) dengan *reset gate* (r_t). Fungsi aktivasi *hyperbolic tangent* (tanh) digunakan untuk menjaga stabilitas sebaran nilai agar tetap berada dalam rentang -1 hingga 1.

- *Final Hidden State*

  Langkah terakhir adalah menghitung memori aktual yang akan menjadi *output* jaringan pada waktu ke-t sekaligus dikirimkan sebagai input untuk langkah waktu berikutnya [53].

$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t \tag{2.11}$$

  Rumus 2.11 merupakan rumus untuk menentukan kandidat baru untuk diteruskan kedalam perhitungan *state* selanjutnya. Pembaruan dilakukan melalui interpolasi linear. Jaringan akan mempertahankan sebagian memori lama (dikontrol oleh $1 - z_t$) dan menambahkan sebagian dari kandidat memori baru (dikontrol oleh z_t).

2.5 Metrik Evaluasi Regressi

Emosi musik direpresentasikan dalam bentuk nilai numerik kontinu pada dimensi *valence* dan *arousal*. *Output* yang dihasilkan oleh *neural network* berupa deret nilai yang berkesinambungan (bukan label kelas diskrit), maka pemodelan ini diklasifikasikan sebagai masalah regresi. Untuk mengukur seberapa akurat model (CNN maupun CRNN) dalam memetakan fitur audio ke dalam ruang emosi. Metrik-metrik ini berfungsi untuk menghitung besaran tingkat kesalahan (simpangan) serta mengukur tingkat keselarasan korelasi antara hasil prediksi model dengan nilai aktual (*ground truth*) dari dataset. Metrik yang digunakan meliputi lima metrik evaluasi utama, yaitu *Mean Absolute Error* (MAE), *Mean Squared Error* (MSE), *Root Mean Squared Error* (RMSE), *Coefficient of Determination* (R^2), dan *Concordance Correlation Coefficient* (CCC) sebagai metrik validasi utamanya.

2.5.1 MAE (*Mean Absolute Error*)

Mean Absolute Error (MAE) merupakan nilai rata-rata dari nilai error pada nilai prediksi dengan nilai aktual. MAE mengukur besaran rata-rata nilai error tanpa mempertimbangkan arahnya (positif atau negatif). Dalam konteks pemodelan dinamika musik, MAE memberikan nilai yang meleset daripada prediksi pada dimensi *valence* dan *arousal* dengan nilai aktualnya (*ground truth*). Karakteristik utama MAE adalah penggunaan pembobotan linier, yang berarti setiap nilai kesalahan diperlakukan sama. Ini berarti MAE sangat kuat dalam mengatasi *outlier*

dalam anotasi, sehingga menjadikannya cara yang baik untuk mengukur kesalahan model rata-rata secara umum [54].

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.12}$$

Rumus 2.12 merupakan rumus dari perhitungan MAE dengan $|y_i - \hat{y}_i|$ menghitung nilai selisih antara asli dan prediksi, kemudian diambil nilai absolutnya (membuang tanda negatif jika ada). $\sum_{i=1}^{n}$ menjumlahkan seluruh nilai absolut tersebut dari *frame* pertama hingga terakhir. $\frac{1}{n}$ membagi total penjumlahan dengan jumlah total *frame* untuk mendapatkan nilai rata-rata (*mean*).

2.5.2 MSE (*Mean Squared Error*)

Mean Squared Error (MSE) menghitung rata-rata perbedaan antara nilai prediksi dan nilai aktual. Proses pengkuadratan dalam MSE memberikan penalti yang jauh lebih besar untuk error yang besar dibandingkan proses MAE. Metrik ini sangat membantu ketika sebuah model perlu menghindari prediksi yang sangat berbeda dari apa yang sebenarnya terjadi [54].

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.13}$$

Rumus 2.13 merupakan rumus dari perhitungan MSE dengan $(y_i - \hat{y}_i)^2$ menghitung selisih antara nilai asli dan prediksi, kemudian dikuadratkan. $\sum_{i=1}^{n} ... \frac{1}{n}$ digunakan untuk menjumlahkan seluruh nilai dan $\frac{1}{n}$ untuk mencari nilai rata-rata sama seperti MAE. Ketika mengkuadratkan angka kurang dari 1, hasilnya akan semakin kecil, dan mengkuadratkan angka lebih besar dari 1, hasilnya akan semakin besar. Dalam rentang emosional [-1, 1], pengkuadratan membuat semua kesalahan menjadi positif dan memperburuk tebakan yang sangat salah.

2.5.3 RMSE (*Root Mean Squared Error*)

RMSE adalah turunan langsung dari MSE, dan dikembalikan ke skala satuan aslinya dengan mengambil akar kuadrat. Perubahan ini membuat RMSE lebih mudah dipahami karena satuan kesalahannya sama dengan rentang dimensi *valence* dan *arousal* (misalnya, [-1,0, 1,0]). RMSE mempertahankan sensitivitas MSE terhadap penyimpangan kesalahan (error) yang besar dengan menggunakan skala

yang lebih mirip dengan satuan data asli [54].

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.14}$$

Rumus 2.14memperlihatkan rumus RMSE yang rumus asli MSE diakarkan untuk mengembalikan nilai error ke dalam satuan unit spasial yang sama persis dengan unit data aslinya (satuan *valence* atau *arousal*), sehingga angkanya lebih intuitif untuk diinterpretasikan.

2.5.4 R^2 (*Coefficient of Determination*)

R-squared (R^2) atau koefisien determinasi adalah metrik statistik yang mengukur proporsi variansi pada variabel terikat yang dapat dijelaskan oleh model. Nilai R^2 berkisar antara 0 hingga 1 (atau direpresentasikan dalam persentase). Nilai yang mendekati 1 mengindikasikan bahwa prediksi model sangat sesuai dengan variasi data aktual [55].

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.15}$$

Rumus 2.15 merupakan rumus untuk menghitung R-Squared dengan $\bar{y}$ adalah nilai rata-rata (*mean*) dari seluruh deret data aktual. $\sum(y_i - \hat{y}_i)^2$ disebut sebagai *Residual Sum of Squares* (RSS), yaitu total kuadrat nilai error. $\sum(y_i - \bar{y})^2$ disebut sebagai *Total Sum of Squares* (TSS), yaitu total variansi data aktual jika dibandingkan dengan rata-rata kasarnya saja. Rumus ini pada dasarnya membandingkan error model (RSS) dengan error model yang paling dasar (TSS). Jika rasio $\frac{RSS}{TSS}$ mendekati 0, maka nilai R^2 akan mendekati 1, yang memberikan penjelasan kurva model sangat identik dengan variansi data asli.

2.5.5 CCC (*Concordance Correlation Coefficient*)

Concordance Correlation Coefficient (CCC) digunakan sebagai metrik utama dalam penelitian ini untuk mengevaluasi kualitas prediksi *valence* dan *arousal*. CCC mengukur tingkat kesesuaian antara nilai prediksi dan nilai sebenarnya (*ground truth*) dengan mempertimbangkan dua aspek sekaligus, yaitu korelasi (apakah pola prediksi bergerak searah dengan label) dan bias (apakah terdapat pergeseran sistematis antara prediksi dan label). CCC dihitung melalui persamaan berikut:

$$\mathrm{CCC} = \frac{2\,\rho\,\sigma_{\hat{y}}\,\sigma_y}{\sigma_{\hat{y}}^2 + \sigma_y^2 + (\mu_{\hat{y}} - \mu_y)^2} \tag{2.16}$$

Pada Persamaan 2.16, $\mu_{\hat{y}}$ dan μ_y berturut-turut adalah rata-rata nilai prediksi dan label; $\sigma_{\hat{y}}^2$ dan σ_y^2 adalah variansi keduanya; sedangkan ρ merupakan koefisien korelasi antara prediksi dan label. Nilai CCC berkisar antara -1 hingga $+1$, di mana nilai 1 menunjukkan kesesuaian sempurna, nilai 0 menunjukkan ketidaksesuaian, dan nilai negatif menunjukkan hubungan yang berlawanan.

Pemilihan CCC sebagai metrik utama didasarkan pada dua keunggulannya. Pertama, CCC bersifat *scale-invariant*, yaitu nilainya tidak berubah meskipun skala data diubah. Sifat ini penting karena model CNN dievaluasi pada anotasi statis berskala $[1, 9]$, sedangkan model CRNN dievaluasi pada anotasi dinamis yang telah di-*z-score*. Metrik kesalahan absolut seperti RMSE dan MAE tidak dapat dibandingkan secara langsung antar-arsitektur karena sensitif terhadap perbedaan skala tersebut, sementara CCC memungkinkan perbandingan yang adil. Kedua, berbeda dengan koefisien korelasi Pearson yang hanya mengukur keselarasan arah, CCC turut memberikan penalti terhadap bias sistematis melalui suku $(\mu_{\hat{y}} - \mu_y)^2$ pada penyebutnya. Dengan demikian, CCC memberikan penilaian yang lebih ketat dan menghindari model yang cenderung memprediksi nilai rata-rata (*regression toward the mean*).

Sebagai acuan interpretasi, penelitian ini menetapkan nilai CCC $\geq$ 0,7 sebagai indikator kesesuaian prediksi yang baik. Ambang ini mengadopsi kategori tingkat kesepakatan (*agreement*) yang digunakan dalam analisis statistik, rentang 0,61–0,80 dikategorikan sebagai kesepakatan substansial (*substantial agreement*) [56]. Nilai 0,7 dipilih sebagai titik tengah representatif dari kategori tersebut, sehingga model yang mencapai atau melampaui ambang ini pada suatu dimensi emosi dianggap telah memiliki keselarasan yang memadai antara prediksi dan persepsi emosi manusia pada dataset.