BAB II

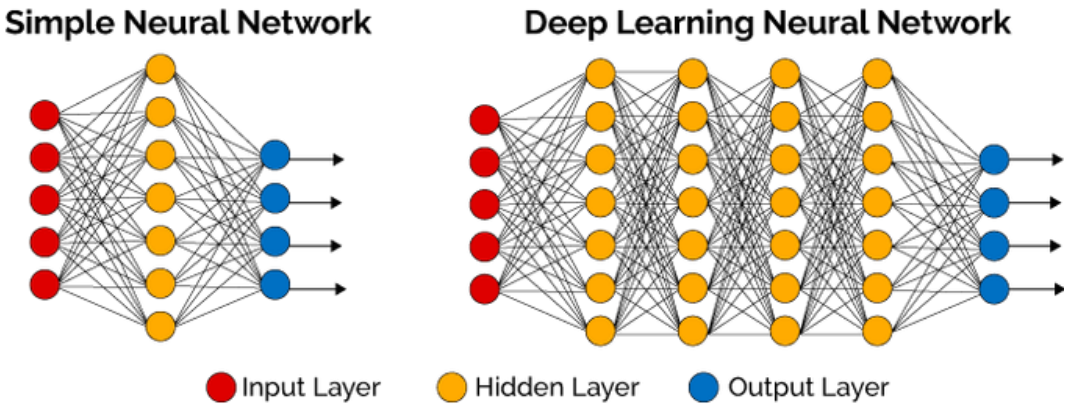
LANDASAN TEORI

Seperti yang telah dipaparkan dalam sistematika penulisan pada bab sebelumnya, bab ini akan membahas tentang dasar-dasar teori penunjang yang berkaitan dengan topik pada penelitian ini. Adapun teori-teori yang akan dipaparkan adalah sebagai berikut.

2.1 *Deep Learning*

Deep learning merupakan sebuah algoritma *neural network* yang menggunakan metadata sebagai *input* dan mengolah *input* tersebut menggunakan sekumpulan fungsi transformasi *non-linier* yang ditata berlapis-lapis dan mendalam [8]. Algoritma *deep learning* dirancang untuk menganalisis data dengan struktur logika yang mirip dengan bagaimana manusia mengambil keputusan. Algoritma ini memiliki kemampuan menangkap fitur atau ciri yang relevan dari suatu data untuk keperluan pemecahan suatu masalah.

Pada *deep learning* terdapat *hidden layer* (lapisan tersembunyi) yang bertugas untuk melatih serangkaian fitur unik berdasarkan *output* dari jaringan sebelumnya [9]. Algoritma ini menjadi semakin kompleks dan bersifat abstrak ketika jumlah *hidden layer* semakin bertambah banyak. *Neural network* yang dimiliki *deep learning* terbentuk dari hirarki sederhana dengan beberapa lapisan tingkat tinggi atau banyak lapisan (*multi layer*). Berikut pada **Gambar 2.1** adalah ilustrasi perbedaan antara *neural network* sederhana yang hanya menggunakan satu atau dua *hidden layer* dengan *deep learning* yang menggunakan banyak *hidden layer*.



Gambar 2.1 Perbandingan *neural network* sederhana dengan *deep learning neural network* [8]

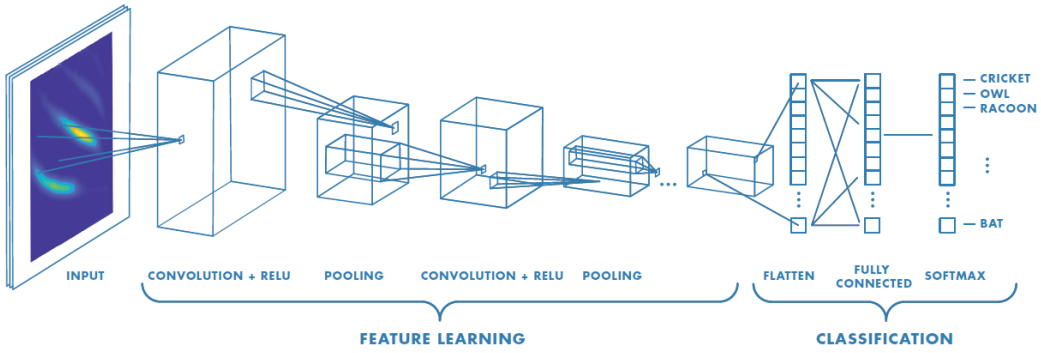
Seperti yang tampak pada pada **Gambar 2.1,** *neural network* sederhana yang menggunakan jaringan *feed forward* hanya mempunyai satu atau dua *hidden layer.* Sedangkan pada *deep learning neural network* terdapat *hidden layer* yang jumlahnya diatas dua *layer* keatas yang bahkan bisa sampai ratusan *layer*. Penggunaan *hidden layer* yang banyak inilah yang memungkinkan *deep learning* mampu menganalisis data dari berbagai dimensi, sehingga dapat menarik kesimpulan dari suatu permasalahan dengan lebih akurat dan detail.

Sama seperti *neural network* pada umumnya, *layer* pada *deep learning neural network* terbagi atas tiga *layer* utama yakni *input layer, hidden layer,* dan *output layer.* Ketiga *layer* tersebut memiliki perannya masing-masing. *Input layer* berfungsi untuk menampung data mentah seperti suara, teks, atau gambar. *Hidden layer* bertugas untuk meneliti dan mengklasifikasikan data tersebut berdasarkan referensi yang sudah ada. *Output layer* berfungsi untuk menyajikan hasil penelitian dan klasifikasi dari suatu data. Algoritma *deep learning* terbagi lagi kedalam beberapa jenis, namun algoritma yang paling sering digunakan yakni C*onvolutional Neural Network (CNN), Recurrent Neural Network (RNN),* dan *Long Short-Term Memory Network (LSTM).* Pada penelitian ini, algoritma *deep learning* yang

digunakan adalah *Convolutional Neural Network (CNN).*

2.2 *Convolutional Neural Network (CNN)*

Convolutional neural network (CNN) adalah salah satu algoritma *deep learning* yang dirancang untuk mengolah data dua dimensi. CNN biasanya digunakan untuk mempelajari dan mendeteksi *feature* pada sebuah gambar [10]. Jika *neural network* pada umumnya menggunakan data *array* 1 dimensi sebagai *input*, pada algoritma CNN data yang digunakan sebagai *input* adalah data dua dimensi [5]. Sama seperti *neural network* pada umumnya, CNN juga terdiri dari banyak *neuron* yang memiliki *weight, bias*, dan *activation function.* Berikut pada **Gambar 2.2** menunjukkan tahapan pada algoritma CNN dalam memproses suatu *input* gambar.



Gambar 2.2 Tahapan proses pada CNN [10]

Seperti yang dapat kita lihat pada **Gambar 2.2**, tahapan proses pada CNN terdiri dari dua bagian utama, yakni *feature learning* dan *classification.* Berikut ini adalah penjelasan dari kedua bagian tersebut.

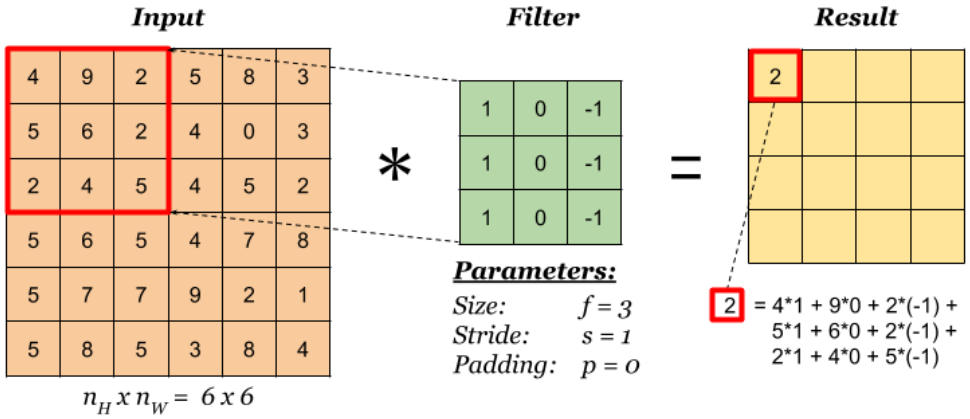
2.2.1 *Feature Learning*

Feature learning adalah proses *encoding* dari sebuah gambar menjadi *feature* yang berupa nilai-nilai yang merepresentasikan gambar tersebut. Proses ini terdiri dari beberapa *layer* yang saling bekerjasama untuk mengambil ciri dari

sebuah gambar. Berikut ini adalah penjelasan dari setiap *layer* yang terdapat pada *feature learning*.

1. *Convolution Layer*

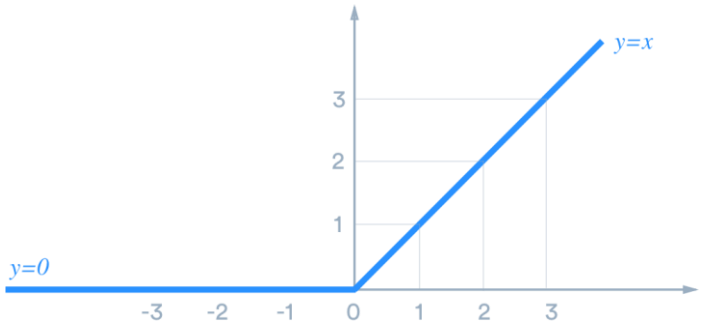
   Convolution layer adalah *layer* pertama dalam algoritma CNN. *Convolution layer* bekerja dengan cara melakukan *sliding window* terhadap *input* gambar. *Convolution layer* terdiri dari neuron yang membentuk sebuah filter dengan panjang dan tinggi (*pixels*) tertentu. Filter akan digeser pada keseluruhan bagian dari gambar. Setiap pergeseran akan dilakukan operasi '*dot'* antara *input* dan filter, sehingga menghasilkan sebuah nilai *output* tertentu [11]. Berikut pada **Gambar 2.3** adalah gambaran operasi '*dot*' antara *input* dengan filter pada *convolution layer* untuk menghasilkan *output* tertentu.



Gambar 2.3 Operasi '*dot*' antara *input* dengan filter [11]

2. *Rectified Linear Unit (ReLU) activation*

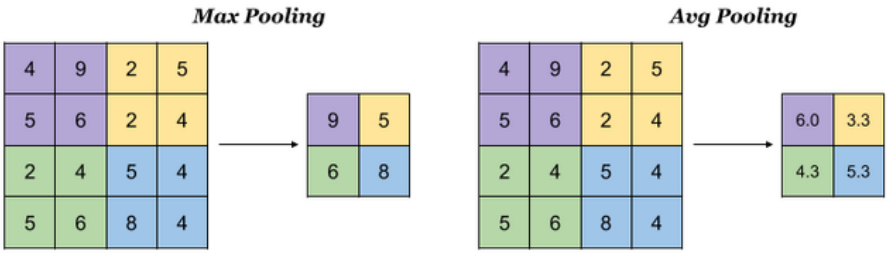
   ReLU layer berfungsi untuk menerapkan fungsi aktivasi terhadap nilai *output* hasil konvolusi pada *convolution layer*. Keluaran dari fungsi aktivasi dinyatakan sebagai 0 (nol) apabila *input*-nya negatif. Namun jika *input*-nya positif, maka *output*-nya akan sama dengan nilai *input* fungsi aktivasi itu sendiri [12]. Berikut pada **Gambar 2.4** adalah grafik fungsi aktivasi pada *ReLU layer*.



Gambar 2.4 Grafik fungsi aktivasi pada *ReLU layer* [12]

3. *Pooling Layer*

Pooling layer merupakan bagian dari *feature learning* pada CNN yang berperan untuk mengurangi dimensi dari matriks hasil konvolusi (*convolved feature*). Tujuan dari penggunaan *pooling layer* adalah untuk mempercepat proses komputasi. Hal ini dapat terjadi karena setelah melewati *pooling layer*, parameter yang harus di-*update* semakin sedikit, sehingga resiko terjadinya *overfitting* semakin minim. Sama seperti pada *convolution layer*, *pooling layer* juga memiliki filter dengan ukuran tertentu yang akan melakukan proses *sliding window* terhadap matriks *input*. Terdapat dua jenis *pooling* yang sering digunakan, yakni *max pooling* dan *average pooling* [11]. **Gambar 2.5** memperlihatkan perbedaan antara kedua jenis *pooling* tersebut.



Gambar 2.5 Perbedaan *max pooling* dengan *average pooling* [11]

Seperti yang dapat kita amati pada **Gambar 2.5**, *max pooling* akan mengambil nilai terbesar dari matriks *input* berdasarkan cakupan filter dari *max*

pooling layer. Sedangkan *average pooling layer* akan mengambil nilai rata rata dari matriks *input* berdasarkan cakupan filternya.

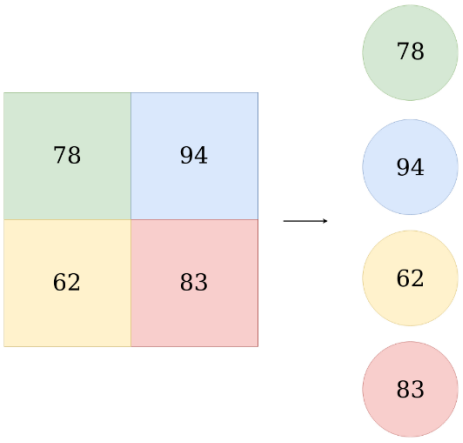
2.2.2 *Classification*

Proses *classification* berfungsi untuk melakukan klasifikasi terhadap setiap *neuron* yang telah di ekstraksi pada proses *feature learning*. Bagian ini terdiri dari beberapa *layer* yang saling berkaitan satu sama lain. Berikut ini adalah penjelasan setiap fungsi pada bagian *classification*.

1. *Flatten*

   Feature map yang dihasilkan pada proses *feature learning* adalah dalam bentuk *multidimensional array*. Sedangkan *input* untuk *fully connected layer* haruslah data dalam bentuk vektor. Oleh karena itu, dibutuhkan sebuah fungsi yang dapat mengubah data *multidimensional array* kedalam bentuk vektor.

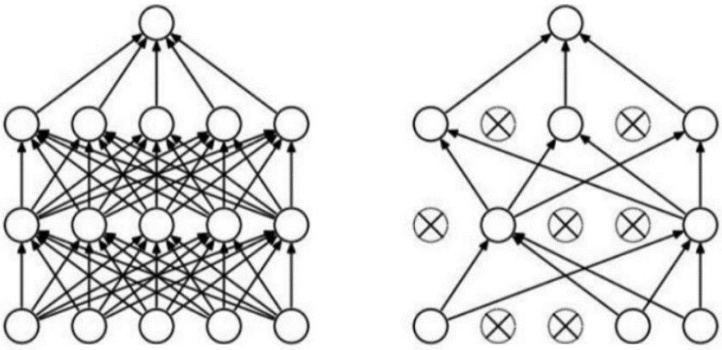
   Flatten berfungsi untuk membentuk ulang (*reshape*) *feature map* dari *multidimensional array* menjadi vektor [13]. Hal ini diperlukan agar nilai-nilai tersebut dapat digunakan sebagai *input* pada *fully connected layer*. Berikut pada **Gambar 2.6** adalah ilustrasi proses *reshape feature map* pada *flatten*.



Gambar 2.6 Proses *reshape feature map* pada *flatten* [13]

2. *Dropout Regularization*

Dropout adalah teknik regularisasi *neural network* dimana beberapa neuron akan dipilih secara acak dan tidak dipakai dalam proses *training.* Dengan menghilangkan suatu neuron, berarti menghilangkannya sementara dari jaringan yang ada [14]. **Gambar 2.7** memperlihatkan perbandingan antara *neural network* yang tidak menggunakan *teknik dropout*, dengan *neural network* yang menggunakan teknik *dropout*.



Gambar 2.7 Perbandingan *neural network* tanpa *dropout* dan setelah menggunakan teknik *dropout* [14]

Tujuan dari penggunaan teknik ini adalah untuk mencegah terjadinya *overfitting.* Selain itu, teknik ini juga dapat mempercepat proses *learning* pada arsitektur CNN secara keseluruhan.

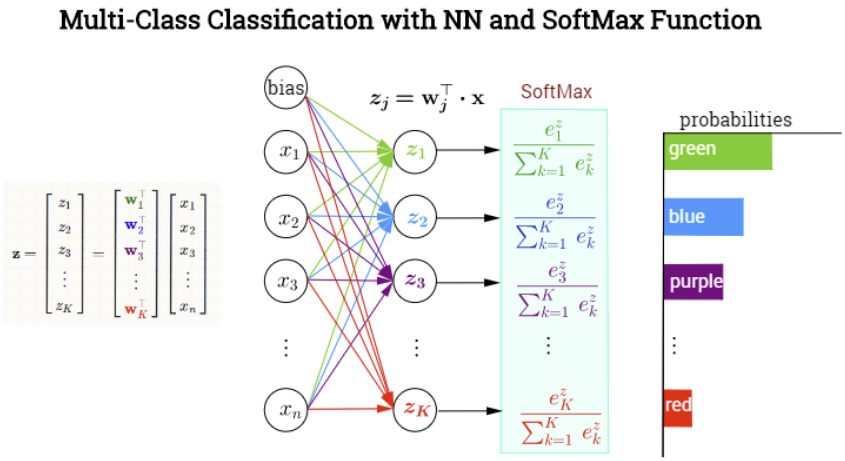
3. *Fully connected layer*

Fully connected layer merupakan *feed forward neural networks* yang terdiri dari *hidden layer, activation function, ouput layer,* dan *loss function* [15]. *Fully connected layer* sering digunakan pada *multi layer perceptron* dengan tujuan untuk melakukan transformasi pada dimensi data agar data-data tersebut dapat diklasifikasikan secara *linear*. *Input* pada *fully connected layer* adalah data dari *feature learning.* Data dari proses *feature learning* yang merupakan *input* pada

layer ini adalah data dalam bentuk vektor, yang sebelumnya telah diproses oleh *flatten*.

4. *Softmax*

Peran utama dari *softmax* dalam proses klasifikasi adalah untuk menghitung probabilitas dari setiap kelas target terhadap semua kelas target yang ada. Rentang probabilitas *output* pada *softmax* adalah nilai dari 0 hingga 1, dan apabila semua nilai probabilitas dari kelas target tersebut dijumlahkan, maka nilainya akan sama dengan satu. *Softmax* menggunakan eksponensial dari nilai *input* yang diberikan dan jumlah nilai eksponensial dari semua nilai dalam *output*. Rasio eksponensial dari nilai *input* dan jumlah nilai eksponensial adalah *output* dari fungsi *softmax*. Berikut pada **Gambar 2.8** adalah gambaran umum penggunaan *softmax* dalam proses klasifikasi.



Gambar 2.8 Penggunaan *softmax* dalam proses klasifikasi [16]

2.3 *Software*

Terdapat beberapa macam *software* yang dapat digunakan dalam merancang sistem dengan metode seperti yang terdapat pada penelitian ini. Pemilihan *software* yang tepat dan sesuai kebutuhan akan sangat berpengaruh

dalam proses perancangan pada sistem ini. Pada penelitian ini, MATLAB merupakan *software* utama yang digunakan. Selain itu, terdapat beberapa *library/toolbox* MATLAB yang digunakan secara bersamaan. Berikut ini adalah penjelasan dari *software* MATLAB dan *library* yang digunakan pada penelitian ini.

2.3.1 MATLAB

MATLAB (*Matrix Laboratory*) merupakan sebuah program untuk menganalisis dan mengkomputasi data *numerik*. MATLAB juga sekaligus merupakan sebuah bahasa pemrograman yang dikembangkan oleh "The Mathwork Inc." yang hadir dengan fungsi dan karakteristik yang berbeda dengan bahasa pemrograman lainnya. MATLAB biasanya digunakan untuk penelitian, pengembangan, dan desain suatu sistem. MATLAB dapat digunakan untuk membantu proses komputasi dan perhitungan di berbagai bidang ilmu seperti matematika, finansial, statistika, teknik dan komputasi, biologi, jaringan, dan lain-lain. Salah satu kelebihan utama dari MATLAB adalah hadir dengan dukungan yang lengkap untuk berbagai jenis kebutuhan. Hal ini dibuktikan dengan banyaknya *library/toolbox* yang tersedia untuk MATLAB. Hal inilah yang memudahkan ketika hendak membuat suatu sistem tertentu yang membutuhkan algoritma yang sangat kompleks seperti *deep learning*.

Pada penelitian ini, MATLAB merupakan *software* utama yang digunakan untuk perancangan sistem. Dimulai dari proses pembacaan sinyal audio, proses klasifikasi menggunakan CNN, sampai kepada proses mengendalikan perangkat elektronik semuanya dilakukan menggunakan MATLAB.

2.3.2 *Library*

Dalam proses perancangan sistem pada penelitian ini, terdapat beberapa *library* MATLAB yang digunakan. *Library* yang digunakan dalam perancangan sistem pada penelitian ini adalah sebagai berikut.

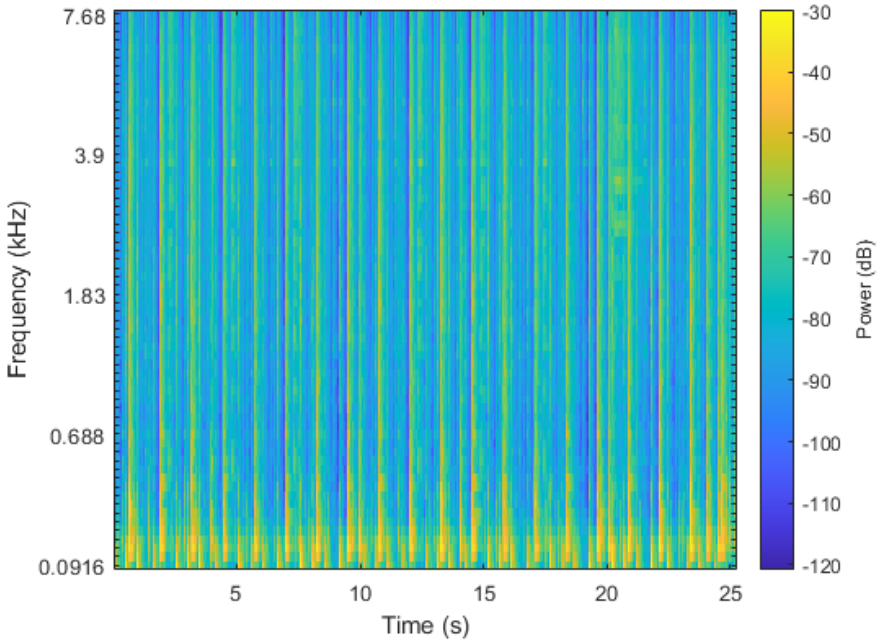
1. *Audio Toolbox*

   Audio toolbox menyediakan berbagai macam *tools* untuk pemrosesan sinyal audio, analisis suara, dan pengukuran akustik. *Audio toolbox* memungkinkan kita untuk meng-*import data set* audio, serta mengkstrak fitur dan mentransformasi sinyal audio untuk nantinya digunakan pada algoritma *deep learning* [17]. Terdapat pula fitur pada *library* ini yang memungkinkan kita untuk membaca sinyal audio secara *real-time*, dengan syarat bahwa komputer yang digunakan harus memiliki *sound card* dan mikrofon.

   Audio toolbox juga menyediakan algoritma untuk *feature extraction* seperti *Mel Frequency Cepstral Coefficient (MFCC).* Algoritma MFCC berfungsi untuk mengekstraksi data sinyal audio sehingga bisa dibedakan ciri yang terdapat pada data sinyal suara tersebut. Metode MFCC pada umumnya digunakan untuk merepresentasikan sinyal suara dalam domain frekuensi. Representasi sinyal audio dalam domain frekuensi ini disebut *spectrogram.*

   Spectrogram adalah representasi spektral atau warna suara yang bervariasi terhadap waktu, yang mana hal ini menunjukkan intensitas energi spektral dari suatu sinyal audio. *Spectrogram* dapat memuat hal-hal yang bersifat detail dari suatu sinyal audio. Dengan demikian kita dapat membedakan suatu sinyal audio dengan sinyal audio lainnya dengan mengamati bentuk *spectrogram*-nya.

Berikut pada **Gambar 2.9** adalah contoh bentuk *spectrogram* dari suatu sinyal audio yang diproses menggunakan *audio toolbox*.



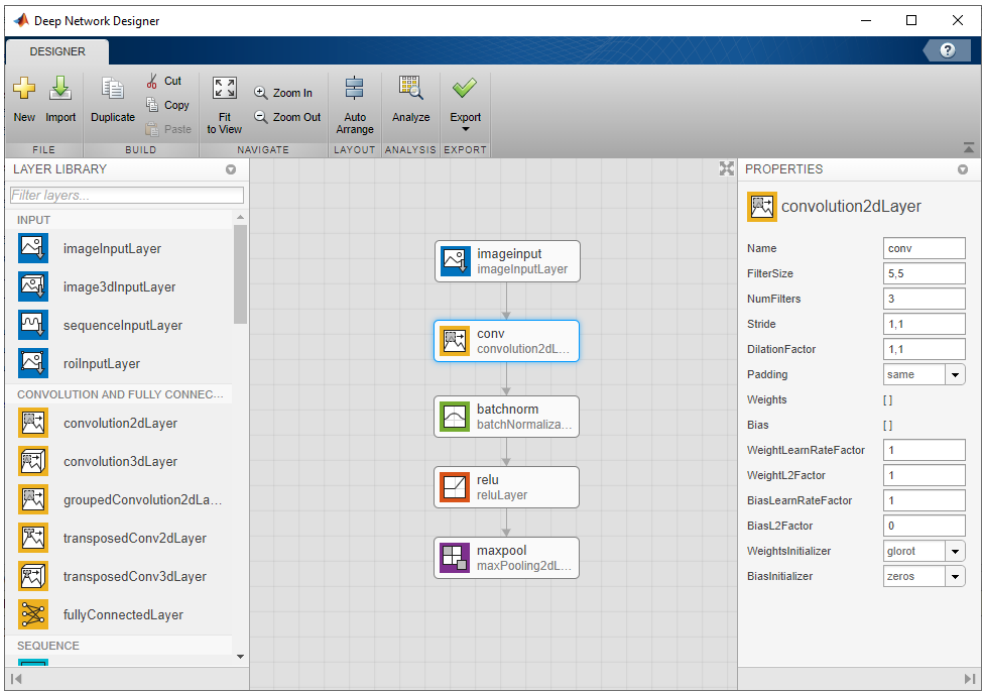
Gambar 2.9 Contoh bentuk *spectrogram* [17]

Seperti yang dapat dilihat pada **Gambar 2.9**, *spectrogram* merepresentasikan data sinyal audio/suara dalam grafik frekuensi terhadap waktu. Contoh diatas merupakan salah satu bentuk *spectrogram* dengan frekuensi antara 62.5 Hz sampai 8 KHz. Warna pada *spectrogram* diatas mewakili intensitas energi spektral dari suatu sinyal audio.

2. *Deep Learning Toolbox*

   Deep learning toolbox menyediakan *framework* untuk merancang dan mengimplementasikan *neural network* dengan algoritma, *pretrained models*, dan aplikasi. *Toolbox* ini memungkinkan kita untuk menggunakan *Convolutional Neural Network (CNN)* dan *Long Short-Term Memory (LSTM)* untuk melakukan klasifikasi dan regresi pada data gambar, data dalam deret waktu, dan data dalam bentuk teks [18].

Deep learning toolbox menyediakan sebuah lembar kerja khusus, yang sangat memudahkan dalam merancang algoritma *deep learning*. Lembar kerja tersebut bernama *deep network designer*. Berikut pada **Gambar 2.10** adalah tampilan dari *deep network designer*.



Gambar 2.10 Tampilan *deep network designer*

Seperti yang terdapat pada **Gambar 2.10**, *deep network designer* memungkinkan kita untuk merancang arsitektur *neural network* berbasis grafis/ diagram blok. Diagram blok dari algoritma *neural network* yang dibuat pada *deep network designer*, kemudian dapat diubah kedalam bahasa pemrograman MATLAB yang berbasis *text*.

3. *Arduino Support Package*

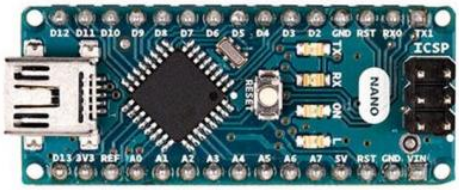
Arduino Support Package merupakan dukungan yang memungkinkan kita menggunakan MATLAB untuk berkomunikasi secara interaktif dengan *board* mikrokontroler arduino. *Arduino support package* mampu mengerjakan tugas tugas seperti berikut [19].

- Membaca data sensor analog dan digital dari *board* arduino
- Mengendalikan perangkat lain dengan *ouput* digital atau PWM
- Menggerakkan *motor DC*, *servo*, dan *stepper*
- Mengakses perangkat *peripheral* dan sensor yang terhubung melalui koneksi I2C atau SPI
- Berkomunikasi dengan *board* arduino melalui komunikasi *serial* (USB) atau secara *nikrabel* melalui jaringan WI-FI
- Membangun *add-on custom* untuk *interface* dengan *hardware* atau *library* tambahan.

MATLAB adalah bahasa pemrograman interpretasi tingkat tinggi, hal inilah memungkinkan data hasil dari instruksi pada *pin input/output* arduino dapat segera dilihat tanpa kompilasi [19]. MATLAB mencakup ribuan fungsi matematika, teknik, dan fungsi *plotting* yang dapat dengan dengan cepat menganalisa dan mengumpulkan data dari *board* arduino.

2.4 Arduino Nano

Arduino nano merupakan sebuah *board development microcontroller* berukuran kecil yang berbasis mirokontroler ATmega328P [20]. Arduino nano adalah salah satu dari sekian banyak *board* mikrokontroler bersifat *open-source* yang dikembangkan oleh arduino. Arduino nano kurang lebih memiliki fungsi yang sama seperti arduino uno, arduino mega, dan berbagai jenis *board* arduino lainnya namun hadir dengan dimensi yang jauh lebih kecil. Berikut pada **Gambar 2.11** adalah bentuk dari arduino nano.



Gambar 2.11 Arduino Nano [20]

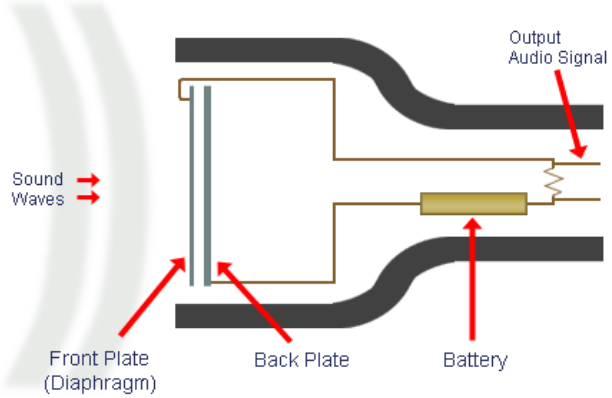
Dikarenakan hadir dengan dimensi yang jauh lebih kecil dibandingkan jenis arduino lainnya, terdapat beberapa fitur dan kelengkapan pada arduino lainnya yang tidak dapat ditemui pada arduino nano. Namun fitur-fitur umum dan penting tetap dipertahankan pada arduino nano. Berikut ini adalah spesifikasi lengkap dari arduino nano [20].

- *Microcontroller* : ATmega328P
- *Architecture* : AVR
- *Operating Voltage* : 5 V
- *Flash Memory* : 32 KB
- *SRAM* : 2 KB
- *Clock Speed* : 16 MHz
- *Analog In Pins* : 8
- *EEPROM* : 1 KB
- *DC Current per I/O Pins* : 40mA (I/O Pins)
- *Input Voltage* : 7-12 V
- *Digital I/O Pins* : 22
- *PWM Output* : 6
- *Power Consumption* : 19mA

2.5 Mikrofon

Mikrofon adalah sebuah transduser yang berfungsi sebagai komponen pengubah satu bentuk energi ke bentuk energi lainnya. Mikrofon mengkonversi energi akustik (Gelombang suara) menjadi energi listrik (Sinyal audio). Berdasarkan teknik konversinya (Konversi energi akustik menjadi energi listrik), mikrofon dibagi kedalam beberapa jenis yakni *dynamic microphone, condenser microphone, electret microphone, ribbon microphone,* dan *crystal microphone.*

Pada penelitian ini, jenis mikrofon yang akan digunakan adalah *condenser microphone* (Mikrofon kondensor). Kondensor adalah sebuah kapasitor, yang merupakan sebuah komponen yang berfungsi untuk menyimpan energi listrik dalam bentuk medan elektrostatik. Mikrofon kondensor membutuhkan *supply* daya *eksternal* agar dapat bekerja. Kelebihan dari mikrofon kondensor adalah mampu menghasilkan sinyal audio yang lebih kuat dibandingkan mikrofon jenis lain. Hal ini dikarenakan, mikrofon jenis ini cenderung lebih sensitif dan responsif dibandingkan mikrofon dinamis. Oleh karena itu, mikrofon kondensor akan jauh lebih baik untuk digunakan untuk menangkap detail-detail kecil pada suara [21]. Berikut pada **Gambar 2.12** adalah komponen utama pada sebuah mikrofon kondensor.



Gambar 2.12 Mikrofon kondensor [21]

Seperti yang dapat kita lihat pada **Gambar 2.12**, mikrofon kondensor terdiri dari dua buah plat dengan tegangan listrik diantara keduanya. Pada mikrofon kondensor, salah satu plat terbuat dari material yang sangat ringan dan berfungsi sebagai diafragma. Ketika terkena gelombang suara, plat diafragma ini akan bergetar menyebabkan terjadinya perubahan jarak antar kedua plat sehingga menyebabkan terjadinya perubahan kapasitansi. Lebih jelasnya, ketika kedua plat saling merapat, kapasitansi akan meningkat dan terjadi penambahan arus. Ketika kedua plat saling menjauh, kapasitansi akan berkurang dan terjadi pelepasan arus.

Berdasarkan *directional properties*, mikrofon dapat dikelompokkan ke dalam tiga kategori utama, yaitu *Omni-directional, Unidirectional, dan Bidirectional.* Pada penelitian ini mikrofon yang akan digunakan adalah mikrofon kondensor dengan *directional propeties ominidirectional* seperti yang dapat kita lihat pada **Gambar 2.13.**

Gambar 2.13 Mikrofon kondensor dengan *polar pattern omni-directional* [22]

Berikut adalah spesifikasi lengkap dari mikrofon kondensor seperti yang tampak pada **Gambar 2.13** [22]**.**

- *Transducer* : *Electret condenser*
- *Polar Pattern* : *Omni-directional*

- *Frequency range* : 65 Hz – 18 KHz
- *Signal/Noise* : 74 dB SPL
- *Sensitivity* : -30dB +/- 3dB/0 dB=1V/Pa, 1KHz
- *Output Impedance* : 1000 Ohm *or less*
- *Battery type* : LR44
- *Connector* : 3.5mm(1/8”) 4-*pole gold plug*