BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam era globalisasi kemajuan teknologi informasi dan komunikasi telah membawa perubahan signifikan dalam berbagai aspek kehidupan manusia, termasuk cara berinteraksi dengan perangkat digital. Salah satu inovasi yang menonjol adalah teknologi pengenalan suara (*Speech Recognition*). Teknologi ini memungkinkan komputer untuk memahami dan memproses bahasa lisan manusia.

Teknologi ASR (*Automatic Speech Recognition*) adalah teknologi yang mampu mengidentifikasi suara pada manusia dan mengubahnya menjadi teks pada komputer. Dalam hal ini, teknologi ASR (*Automatic Speech Recognition*) menjadi peranan penting dalam Aplikasi *Speech To Text Online*. Dimana teknologi ini diimplementasikan sebagai sebuah *library* dalam bahasa pemrograman Python. Dengan memanfaatkan library ini, tentu dapat lebih mudah mengimplementasikan algoritma *deep learning* dalam aplikasi *Speech To Text Online*. Meskipun mengimplementasi algoritma menjadi lebih mudah, tentunya tantangan dalam pengembangannya tidaklah sedikit. Beberapa faktor yang perlu dipertimbangkan termasuk kualitas data suara, hasil teks yang dikeluarkan, keberagaman aksen dan bahasa, serta kemampuan algoritma dalam menangani kebisingan lingkungan. Oleh karena itu, penelitian ini bertujuan untuk mengimplementasikan algoritma *deep learning* dalam aplikasi *speech to text online* dengan bantuan *library Speech Recognition.*

Speech to text merupakan aplikasi *online* yang dirancang agar dapat mengubah suara pada manusia menjadi sebuah teks dalam komputer. Hal tersebut dapat membuat komputer mengerti bahasa manusia melalui perintah suara. *Speech to text* ini dapat diterapkan dalam berbagai aplikasi seperti virtual assistans, penerjemah, sistem pengolahan audio, rekam medis, dan lain sebagainya. Pada umumnya manusia dapat berbicara lebih cepat dari pada mengetik. Oleh karena itu, Aplikasi ini dirancang agar dapat mempermudah segala aktivitas, baik dalam mengambil catatan, dalam membuat sebuah *content creator* maupun membuat teks secara lebih cepat dan efisien, tanpa perlu mengetik secara manual atau menggunakan perangkat keyboard yang umumnya membutuhkan waktu lebih lama.

Implementasi Algoritma *Deep learning* pada aplikasi *Speech To Text Online* merupakan proses menggunakan algoritma *Deep learning* untuk mengubah suara manusia menjadi teks. *Deep learning* digunakan untuk membantu komputer mengenal dan mengerti suara manusia dengan lebih baik, yang dapat dikonversi menjadi teks.

Algoritma *Deep learning* adalah bagian dari *Machine learning* yang memiliki fungsi untuk mempelajari data yang telah tersedia melalui algoritma yang ada (Tilasefana & Putra, 2023). Algoritma *Deep learning* adalah metode dari jaringan syaraf (*Neural Network*) yang menggunakan multi *layer perseptron* sebagai *layer* pembelajarannya. *Layer* tersebut mempunyai neuron yang berjumlah banyak sehingga dapat menghasilkan pembelajaran yang lebih dalam (Laksono, 2018). Salah satu bagian dari Algoritma *Deep Learning* adalah *Recurrent Neural Network* (RNN) yang dapat meningkatkan akurasi pengenalan ucapan dan

menghasilkan transkripsi yang lebih akurat. *Recurrent Neural Network* (RNN) juga memiliki kemampuan dalam menangani data sekuensial (Firmansyah et al., 2020), sistem *Recurrent Neural Network* (RNN) mampu memproses suara secara efisien, menangkap pola-pola ucapan yang kompleks, dan menghasilkan transkripsi teks yang akurat, yang merupakan esensi dari pengenalan suara.

Sesuai dengan latar belakang diatas, penulis akan membuat skripsi tentang pembuatan aplikasi berbasis *website* dengan judul "IMPLEMENTASI ALGORITMA *DEEP LEARNING* PADA APLIKASI *SPEECH TO TEXT ONLINE*".

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang diatas, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana cara mengimplementasikan Algoritma *Deep Learning* pada Aplikasi *Speech To Text Online*?
2. Bagaimana cara proses transkrip suara ke teks secara otomatis pada Aplikasi *Speech To Text Online*?

1.3 Batasan Masalah

Agar pembahasan lebih terarah dan sesuai dengan judul Tugas Skripsi yang telah ditentukan, penulis hanya membahas pokok-pokok bahasan sebagai berikut:

1. Pembuatan Aplikasi *Speech To Text Online* menggunakan bahasa pemrograman Python.
2. Aplikasi *Speech To Text Online* hanya untuk mengubah suara menjadi teks.

3. Dalam Aplikasi *Speech To Text Online* hanya dapat mengubah rekaman file dalam bentuk format mp3 dan wav.

1.4 Tujuan dan Manfaat

1.4.1 Tujuan Penelitian

Tujuan yang ingin dicapai dalam penelitian Tugas Skripsi ini adalah:

1. Mengimplementasikan Algoritma *Deep Learning* pada Aplikasi *Speech To Text Online*, untuk meningkatkan kinerja dan akurasi konversi suara menjadi teks.
2. Mengoptimalkan Aplikasi *Speech To Text Online* menggunakan algoritma *Deep learning*, seperti RNN (*Recurrent Neural Network*), untuk mempermudah proses transkrip otomatis.

1.4.2 Manfaat Penelitian

Adapun manfaat penelitian ini adalah sebagai berikut:

1. Untuk menghasilkan aplikasi yang dapat menjadi transkrip suara ke teks (*speech to* text) dalam Bahasa Indonesia secara otomatis.
2. Untuk mempermudah segala kegiatan yang membutuhkan konversi suara menjadi teks (*speech to text*).

1.5 Metodologi Penelitian

Metodologi penelitian yang digunakan pada penelitian ini adalah:

1. Studi Kepustakaan

Pada tahap ini dilakukan studi kepustakaan yaitu proses mengumpulkan informasi dengan melakukan pengumpulan, mempelajari, dan membaca berbagai bahan referensi yang berkaitan dengan aplikasi, serta Algoritma *Deep Learning.*

2. Analisis dan Perancangan

   Pada tahap ini dilakukan analisis spesifikasi aplikasi dan melakukan perancangan aplikasi, seperti perancangan proses dan antarmuka yang meliputi desain database, sketsa, dan lain sebagainya.

3. Pengkodean

   Pada tahap ini dilakukan pengkodean aplikasi sesuai dengan analisis spesifikasi dan perancangan yang telah ditentukan.

4. Pengujian Aplikasi

   Pada tahap ini dilakukan pengujian terhadap aplikasi konversi suara menjadi teks.

5. Penyusunan Laporan

   Pada tahap ini dilakukan penulisan dokumentasi dan laporan dari aplikasi konversi suara menjadi teks yang dikembangkan.

1.6 Sistematika Penulisan

Sistematika penulisan Tugas Skripsi ini dibagi atas beberapa bab, di mana masing-masing bab dibagi atas beberapa sub agar mempermudah penjelasan mengenai penelitian yang dilakukan dan mempermudah pembaca dalam memahami isi penelitian. Adapun sistematika penulisan Tugas Skripsi ini adalah sebagai berikut:

BAB 1 PENDAHULUAN

Pendahuluan berisi tentang Latar Belakang Masalah, Rumusan Masalah, Tujuan, Manfaat, Batasan Masalah, Metodologi Penelitian dan Sistematika Penulisan dalam pembuatan Tugas Skripsi.

BAB 2 TINJAUAN PUSTAKA

Bab ini berisi teori-teori pengetahuan dasar yang di peroleh dari studi kepustakaan atau literatur dan dokumentasi *internet* yang digunakan untuk memahami permasalahan yang dibahas pada penelitian ini. Teori – teori pengetahuan dasar yang disajikan antara lain tentang aplikasi, serta algoritma *Deep Learning.*

BAB 3 METODE PENELITIAN

Bab ini menguraikan tahapan-tahapan sistematis yang digunakan untuk melakukan kajian penelitian. Tahapan-tahapan tersebut merupakan kerangka yang dijadikan pedoman penelitian untuk mencapai tujuan yang telah ditetapkan. Tahapan tersebut berisi waktu dan tempat penelitian untuk membuat aplikasi konversi suara menjadi teks menggunakan Bahasa Indonesia berbasis *website* dengan Algoritma *Deep Learning.*

BAB 4 HASIL DAN PEMBAHASAN

Bab ini berisi tentang hasil dan pembahasan dari aplikasi konversi suara menjadi teks menggunakan Bahasa Indonesia berbasis *website* dengan Algoritma *Deep Learning.*

BAB 5 KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dari keseluruhan uraian bab – bab penulisan skripsi dan saran yang diajukan untuk pengembangan lebih lanjut.

BAB II

TINJAUAN PUSTAKA

2.1 Aplikasi

2.1.1 Pengertian Aplikasi

Aplikasi dapat diartikan sebagai suatu program berbentuk perangkat lunak yang berjalan pada suatu sistem tertentu yang berguna untuk membantu berbagai kegiatan yang dilakukan oleh manusia. Selain pengertian di atas, ada banyak pengertian dari kata 'Aplikasi' yang dikemukakan oleh para ahli. Berikut ini beberapa definisi aplikasi menurut beberapa ahli yang cukup populer (Huda & Priyatna, 2019):

1. Ali Zaki dan Smitdev Community

   Menurut Ali Zaki dan Smitdev Community, Aplikasi merupakan komponen yang bermanfaat sebagai media untuk menjalankan pengolahan data ataupun berbagai kegiatan lainnya seperti pembuatan ataupun pengolahan dokumen dan file.

2. Sri Widianti

   Menurut Sri Widianti, Aplikasi merupakan sebuah software (perangkat lunak) yang bertugas sebagai front end pada sebuah sistem yang dipakai untuk mengelolah berbagai macam data sehingga menjadi sebuah informasi yang bermanfaat untuk penggunanya dan juga sistem yang berkaitan.

3. Harip Santoso

   Menurut Harip Santoso, Aplikasi merupakan sebuah kelompok file (class, form, report) yang ditujukan sebagai pengeksekusi aktivitas tertentu yang saling berkaitan seperti contohnya aplikasi payroll dan aplikasi fixed asset.

4. Yuhefizar

   Menurut Yuhefizar, Aplikasi adalah program yang sengaja dibuat dan dikembangkan sebagai pemenuh kebutuhan penggunanya dalam menjalankan suatu pekerjaan tertentu.

5. Hengky W. Pramana

   Menurut Hengky W. Pramana, pengertian aplikasi adalah satu unit perangkat lunak yang sengaja dibuat untuk memenuhi kebutuhan akan berbagai aktivitas ataupun pekerjaan, seperti aktivitas perniagaan, periklanan, pelayanan masyarakat, game, dan berbagai aktivitas lainnya yang dilakukan oleh manusia.

2.1.2 Jenis-Jenis Aplikasi

Dalam pengembangannya, aplikasi dapat dikategorikan dalam tiga kelompok, diantaranya(Ayu Rifka Sitoresmi, 2023):

1. Aplikasi *Desktop*, yaitu aplikasi yang hanya dijalankan di perangkat PC komputer atau laptop.
2. Aplikasi *Web*, yaitu aplikasi yang dijalankan menggunakan komputer dan koneksi internet.
3. Aplikasi *Mobile*, yaitu aplikasi yang dijalankan di perangkat *mobile* di mana untuk kategori ini penggunaannya sudah banyak sekali.

2.1.3 Fungsi Aplikasi

Adapun beberapa fungsi dari Aplikasi adalah (Ayu Rifka Sitoresmi, 2023):

1. Untuk mempermudah pekerjaan
2. Sebagai media hiburan
3. Untuk mendapatkan pembaharuan kabar
4. Untuk media pertemanan atau komunikasi
5. Mempermudah kehidupan

2.2 *Website*

2.2.1 Pengertian *Website*

Website dapat diartikan sebagai kumpulan halaman yang berisi informasi data digital baik berupa teks, gambar, animasi, suara dan video atau gabungan dari semuanya yang disediakan melalui jalur koneksi internet sehingga dapat diakses dan dilihat oleh semua orang diseluruh dunia (Rohi Abdulloh, 2022).

2.2.2 Jenis-Jenis *Website*

Menurut Rohi Abdulloh (2022:2) secara umum, *website* dibagi menjadi 3 jenis, yaitu :

1. *Website* Statis

   Website statis yaitu jenis *website* yang isinya tidak diperbaharui secara berkala, sehingga isinya dari waktu ke waktu akan selalu tetap. *Website* jenis ini biasanya hanya digunakan untuk menampilkan profil dari pemilik *website* seperti profil Perusahaan atau organisasi. Untuk saat ini, *website* jenis ini banyak digunakan pada *website* jenis *landing page*.

2. *Website* Dinamis

   Website dinamis yaitu jenis *website* yang isinya terus diperbaharui secara berkala oleh pengelola web atau pemilik *website*. *Website* jenis ini banyak dimiliki oleh perusahaan atau perorangan yang aktivitas bisnisnya memang berkaitan dengan internet. Contoh paling mudah dari *website* jenis ini yaitu *web blog* dan *website* berita.

3. *Website* Interaktif

   Website interaktif pada dasarnya termasuk dalam kategori *website* dinamis, di mana isi informasinya selalu diperbaharui dari waktu ke waktu. Hanya saja, isi informasi tidak hanya diubah oleh pengelola *website* tetapi lebih banyak dilakukan oleh pengguna *website* itu sendiri. Contoh *website* jenis ini yaitu *website* jejaring sosial seperti Facebook dan Twitter atau *website marketplace* seperti Bukalapak, Tokopedia, Shopee, dan sebagainya.

2.3 *Speech To Text*

2.3.1 Pengertian *Speech To Text*

Konversi suara menjadi teks yang biasanya dapat disebut juga dengan *speech to text* adalah perangkat lunak transkripsi yang secara otomatis mengenali ucapan dan mentranskripsikan apa yang diucapkan ke dalam format tertulis yang setara. Secara tradisional, manusia akan mendengarkan *file* audio dan mengetiknya ke dalam *file* teks guna menggunakan kembali konten yang diucapkan untuk media yang berbeda. Namun dengan menggunakan kecerdasan buatan, sekarang komputer dapat dengan mudah mengonversi suara menjadi teks dalam waktu singkat dan

membuat konten tersebut dapat digunakan untuk tujuan yang berbeda, seperti pencarian, *subtitle*, dan wawasan. Pada *Speech to Text* terdapat beberapa pendekatan/metode dalam proses analisis pola suara/bahasa ke dalam bentuk tulisan/teks. *Speech to text* ini dapat diterapkan dalam berbagai hal diantaranya sebagai alat bantu komunikasi bagi tuna rungu (tuli), *smart home*, atau penerjemah. Output dari *speech to text* akan menjadi input dan akan menggantikan input *text* secara manual sehingga mampu mempermudah dalam penerapannya di berbagai bidang. Beberapa pendekan/metode yang sering digunakan diantaranya yaitu: *Deep Neural Network* (DNN), *Recurrent Neural Network* (RNN), *Hidden Markov Model* (HMM), dan lain sebagainya (Laksono, 2018).

2.3.2 Cara Kerja *Speech To Text*

Perangkat lunak transkripsi otomatis mengenali ucapan dengan menggunakan *machine learning* (ML) dan kecerdasan buatan (AI). *Machine learning* adalah teknologi yang melatih komputer dalam pengenalan ucapan dengan menyimpan dan menganalisis volume data ucapan yang sangat tinggi. Konversi suara menjadi teks memberikan hasil yang akurat karena dapat membandingkan pola ucapan yang direkam dengan basis data besar ini. Terdapat dua komponen utama dalam mengkonversi suara menjadi teks (Aws, 2023).

1. Komponen akustik

   Komponen akustik adalah perangkat lunak yang mengonversi *file* audio menjadi urutan unit akustik. Unit akustik adalah sinyal digital yang mewakili gelombang suara atau getaran suara yang dibuat saat berbicara. Teknologi pengenalan ucapan akustik mencocokkan unit akustik dengan suara yang

membentuk bahasa manusia, yang disebut dengan fonem. Misalnya, bahasa Indonesia memiliki 27 fonem, Dimana terdapat 22 fonem konsonan dan 5 fonem vocal, yang digabungkan untuk membentuk semua kata dalam bahasa tersebut. Sehingga dapat digunakan fonem untuk mengonversi suara menjadi teks dalam banyak Bahasa Indonesia secara otomatis.

2. Komponen linguistik

   Komponen akustik mendengar kata, sedangkan komponen linguistik memahami dan mengejanya. Misalnya, banyak kata dalam bahasa Indonesia terdengar sama, tetapi ejaannya berbeda.

   Komponen linguistik menganalisis semua kata sebelumnya dan hubungannya untuk memperkirakan kata apa yang mungkin akan muncul selanjutnya. Komponen tersebut kemudian mengonversi urutan unit akustik menjadi kata, kalimat, dan paragraf yang masuk akal bagi manusia. Teknologi pengenalan ucapan ini mirip dengan fungsi saran otomatis di ponsel cerdas yang secara otomatis menyarankan kata saat ingin mengetik teks.

2.4 *Speech Recognition*

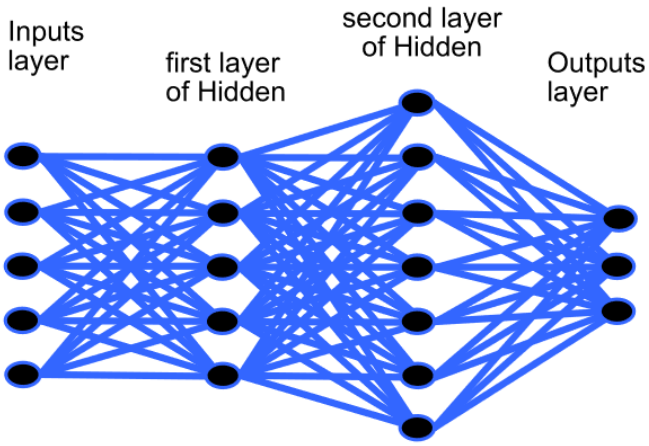
Speech recognition atau yang biasa dikenal dengan *Automatic Speech Recognition* (ASR) adalah suatu pengembangan teknik dan sistem yang memungkinkan komputer dapat menerima masukan berupa kata yang diucapkan. Teknologi ini memungkinkan suatu perangkat dapat mengenali serta memahami kata-kata yang diucapkan dengan cara digitalisasi kata dan mencocokkan sinyal digital tersebut melalui suatu pola tertentu yang tersimpan dalam suatu perangkat. Kata-kata yang diucapkan kemudian diubah menjadi sinyal digital dengan cara

mengubah gelombang suara menjadi suatu kumpulan angka-angka yang kemudian di terjemahkan dengan kode-kode tertentu untuk mengidentifikasikan kata-kata tersebut (Jaman & Fergina, 2021).

2.5 Algoritma *Deep Learning*

Deep Learning adalah cabang dari *Machine Learning* yang terinspirasi oleh struktur dan fungsi otak manusia, yang disebut sebagai jaringan saraf tiruan atau *Artificial Neural Network* (Laksono, 2018).

Deep learning dapat didefinisikan sebagai teknik pembelajaran mesin yang menggunakan jaringan saraf tiruan dengan banyak lapisan neuron untuk mengekstraksi fitur yang kompleks dan mewakili data input yang kompleks dalam hierarki. Jumlah dan kompleksitas lapisan neuron adalah yang membedakan *deep learning* dari pembelajaran mesin konvensional. Kemampuan *deep learning* untuk mempelajari pola dari data yang kompleks dan abstrak menjadikannya sangat sukses dalam pengenalan citra, pemrosesan bahasa alami, pengenalan suara, dan banyak bidang lainnya (Laksono, 2018). Salah satu metode dari algoritma *Deep Learning* ialah *Recurrent Neural Network* (RNN).



Gambar 2.1 Ilustrasi arsitektur pada *Deep Learning*

2.5.1 Prinsip Kerja *Deep Learning*

Menurut (Cholissodin et al., 2020) *Deep Learning* merupakan pembelajaran yang berbasis pada fitur yang berbentuk hirarki, yang mana bentuk fitur hirarki tersebut dapat diskalakan dalam ukuran tertentu yang dapat disesuaikan dengan kasus yang diproses. Sehingga membuat algoritma *Deep Learning* mampu untuk melakukan ekstraksi fitur otomatis dari data mentah secara detail, karena proses ekstraksi yang dilakukan menggunakan struktur eksploitasi, dimana fitur-fitur yang dieksploitasi tersebut tidak mungkin dapat dilihat secara kasat mata. Hal ini dikarenakan distribusi pembeda kelas data biasanya terlalu dalam, sehingga dari fitur level tinggi harus ditransformasi terlebih dahulu ke fitur yang paling rendah yang dapat mudah dipahami oleh mesin pembelajaran. Artinya jika fitur level rendah saja dapat mudah diindentifikasi oleh *Deep Learning*, apalagi yang fitur level tinggi. Perpaduan inilah yang menjadikan *Deep Learning* mampu menghasilkan representasi fitur yang baik dan optimal. Berikut adalah prinsip kerja dari algoritma *Deep Learning* (Cholissodin et al., 2020):

1) Buat arsitektur yang tepat, terutama pada jenis metode *Deep Learning* yang digunakan sesuai dengan kebutuhan dari kompleksitas kasus yang akan diolah dan diselesaikan. Arsitektur tersebut terdiri dari *layer input*, *hidden* (*darknet*/ layer gelap/ tersembunyi), dan *output*.
2) Siapkan data atau item apa saja yang diperlukan ketika proses pelatihan dan pengujian *Deep Learning*.
3) Pertimbangkan untuk tempat implementasinya (misal di local atau menggunakan cloud) dan kode programnya (membuat kode program from

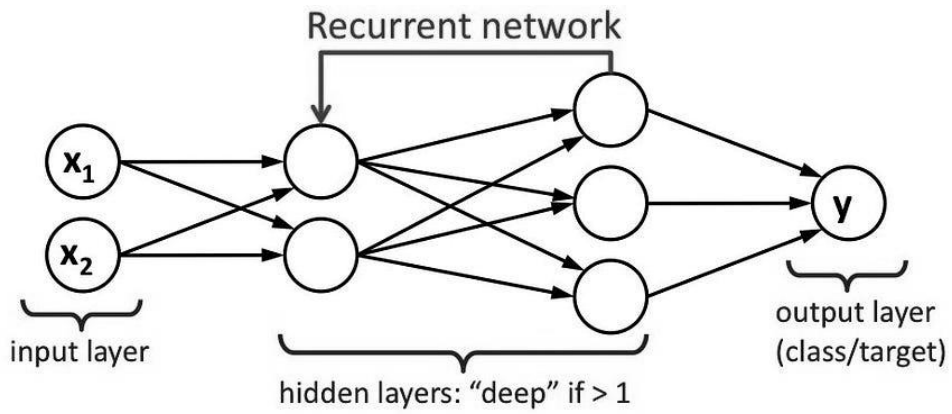
scratch. menggabungkan dengan membuat sebagian secara scratch dan menggunakan Sebagian kode dari library, atau menggunakan full library).

4) Lakukan pengujian yang sesuai dengan strandar dari algoritma *Deep Learning*, mulai dari parameter, sampai ke pengujian bentuk arsitekturnya. Misal pengujiannya secara *waterfall* (sekuensial tanpa *loop*) atau secara *recycle* (sekuensial dengan loop tertentu). Di mana pengujian *recycle* ini mengkondisikan pengujiannya diulang-ulang sampai beberapa kali siklus (perulangan), misal terdapat 3 macam pengujian, pertama menguji parameter A, lalu hasil nilai dari parameter A yang terbaik akan digunakan untuk pengujian parameter B, lalu hasil pegujian nilai dari parameter A dan B yang terbaik akan digunakan untuk pengujian parameter C, lalu hasil pengujian nilai dari parameter B dan C yang terbaik akan digunakan untuk pengujian parameter A (*loop* ke-1 dan seterusnya sampai iterasi tertentu), dan sebaiknya iterasi tersebut dihentikan ketika nilai paramter A, B dan C sudah stabil serta nilai akurasinya sudah stabil pada nilai yang paling tinggi.
5) Pikirkan bagaimana mengoptimasi lagi algoritma *Deep Learning* dari hasil yang didapatkan saat ini.

2.5.2 Metode *Recurrent Neural Network* (RNN)

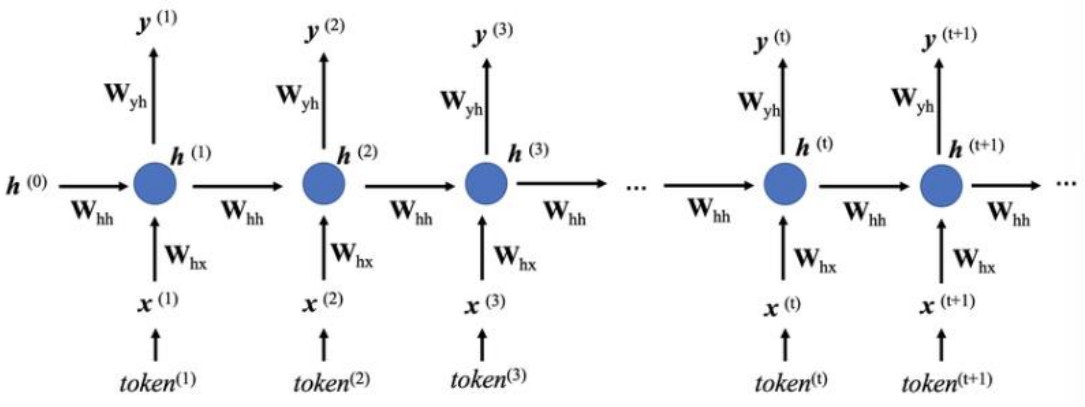
Recurrent Neural Network (RNN) adalah sebuah metode *deep learning* yang dapat melakukan pembelajaran terhadap pola yang terdapat di dalam data sekuensial. *Recurrent Neural Network* (RNN) memiliki kemampuan untuk "mengingat" elemen data yang telah "dipelajari" sebelumya. Dengan kemampuan ini maka RNN mampu memprediksi sebuah elemen (urutan elemen) data yang akan

muncul kemudian berdasarkan data input sebelumnya (Wahyono, 2021). Pengenalan suara, mesin terjemahan, pemodelan bahasa tingkat karakter, klasifikasi gambar, keterangan gambar, prediksi saham, dan rekayasa keuangan merupakan contoh aplikasi RNN (Malau dan Nurjaman, 2019).



Gambar 2.2 Model pada *Recurrent Neural Network*

Recurrent Neural Network (RNN) adalah sebuah kelas *neural network* yang memiliki sejumlah koneksi berupa *loop* yang berfungsi sebagai *input* sebuah *neuron* (selanjutnya disebut sel RNN) (Wahyono, 2021). *Recurrent Neural Network* (RNN) akan memproses data *input* satu per satu secara sekuensial, *hidden layer* pun akan melempar data menuju ke *hidden layer* pada skala waktu selanjutnya. Begitu seterusnya secara sekuensial (Lapian et al., 2018).



Gambar 2.3 Proses RNN ketika melakukan perhitungan

Persamaan Neuron Hidden pada gambar 2.3 ialah (Ramadhika, 2019) :

$$h^{(i)} = \sigma_h (W_{hh} \cdot h^{(i-1)} + W_{hx} x^{(i)} + b_h)$$

Keterangan :

- $h^{(i)}$ adalah nilai neuron hidden pada timestep i
- $x^{(i)}$ adalah nilai input
- σ_h adalah fungsi aktivasi
- W_{hh} adalah parameter weight untuk hubungan antara neuron hidden sebelumnya
- $h^{(i-1)}$ adalah nilai output dari posisi sebelumnya
- W_{hx} adalah parameter weight untuk hubungan antara input x_t dan neuron hidden
- b_h adalah bias neuron hidden

Persamaan Output pada gambar 2.3 ialah (Ramadhika, 2019):

$$y^{(i)} = \sigma_y (W_{yh} h^{(i)} + b_y)$$

Keterangan :

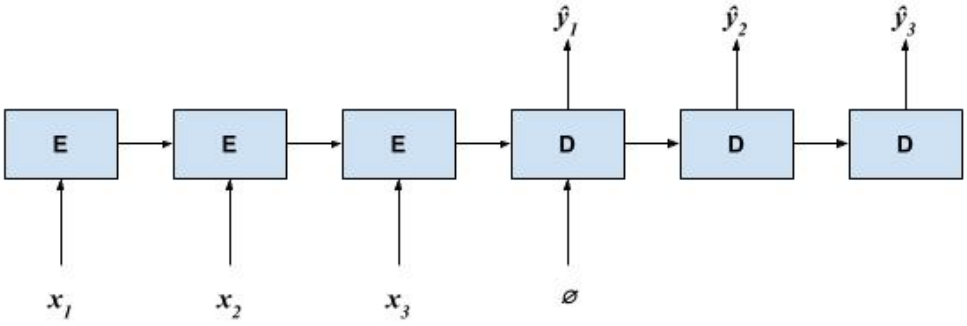
- $y^{(i)}$ adalah output pada timestep i
- σ_y adalah fungsi aktivasi
- W_{yh} adalah parameter weight untuk hubungan antara neuron hidden dan output
- $h^{(i)}$ adalah nilai output
- b_y adalah bias output

2.5.2.1 Jenis – Jenis Pemrosesan *Recurrent Neural Network* (RNN)

Berikut ini ialah jenis-jenis dari pemrosesan pada *Recurrent Neural Network* (RNN) (Prijono, 2018) :

1. *Many To Many* (Banyak Ke Banyak)

   Pada jenis ini, baik input dan output adalah data sekuensial yang jumlahnya banyak.

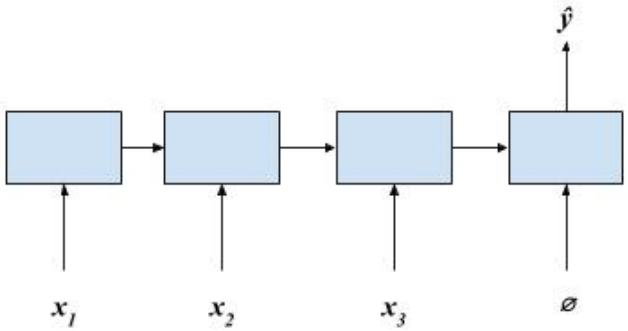


Gambar 2.4 *Many to many* pada RNN

Pada jenis ini jumlah output yang dihasilkan tidak harus sama dengan jumlah input, dan biasanya model akan menghasilkan output hanya setelah semua input selesai diproses, agar model dapat mengetahui sebanyak mungkin konteks untuk menghasilkan output. Contoh penggunaan jenis ini adalah translasi bahasa dan pengenalan suara.

2. *Many to One* (Banyak Ke Satu)

   Pada jenis pemrosesan ini, model memproses banyak input, tapi output yang dihasilkan hanya satu.

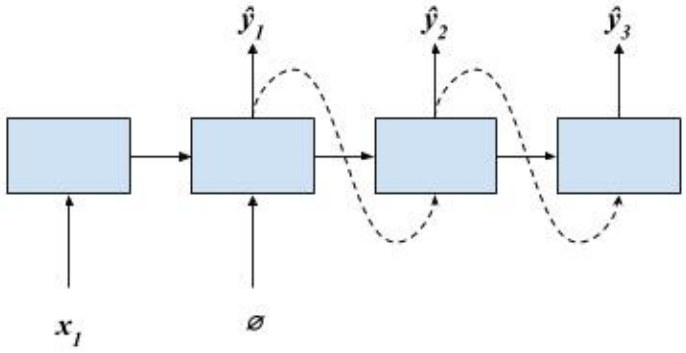


Gambar 2.5 *Many to one* pada RNN

Contoh penggunaan jenis ini adalah deteksi sentimen dan nilai ulasan film otomatis. Pada aplikasi deteksi sentimen misalnya, input berupa kata-kata dan model akan memberikan satu output berupa nilai sentimen yang sesuai.

3. *One To Many* (Satu Ke Banyak)

   Pada jenis pemrosesan ini, model hanya membutuhkan satu input saja, namun bisa menghasilkan banyak output.



Gambar 2.6 *One to many* pada RNN

Penggunaan arsitektur jenis ini terutama pada aplikasi sintesa, misalnya sintesa musik dan sintesa esai. Pada sintesa musik, inputnya hanya satu misalnya jenis music (jazz, dangdut, dan lain sebagainya), lalu model akan mensintesa nada demi nada sehingga menghasilkan musik yang dapat dinikmati.

2.6 Bahasa Pemrograman Web

2.6.1 Python

Menurut Haryodi (2024:10) Python adalah bahasa pemrograman yang mudah dipelajari dan memiliki sintaks yang bersih. Dalam konteks web, Python

digunakan kerangka kerja seperti Django atau Flask untuk membangun aplikasi web dengan fitur yang kompleks. Bahasa pemrograman python dapat digunakan dalam berbagai bidang termasuk *web development*.

Python diciptakan oleh Guido van Rossum di Belanda pada tahun 1990 dan namanya diambil dari acara televisi kesukaan Guido Monty Python's Flying Circus (Muhammad Romzi & Kurniawan, 2020).

2.7 HTML (*Hypertext Markup Language*)

Menurut Rohi Abdulloh (2022:10) HTML merupakan singkatan dari *Hypertext Markup Language*, yaitu bahasa standar web yang dikelola penggunaannya oleh W3C (*World Wide Web Consortium*) berupa tag-tag yang menyusun setiap elemen dari website. html berperan sebagai penyusun struktur halaman website yang menempatkan setiap elemen website sesuai layout yang diinginkan. HTML berperan sebagai pembentuk struktur halaman website yang menempatkan setiap elemen website sesuai layout yang diinginkan.

HTML biasanya disimpan dalam sebuah file berekstensi .html. Untuk mengetikkan skrip html, kita dapat menggunakan teks editor seperti notepad sebagai bentuk paling sederhana atau text editor khusus yang dapat mengenali Setiap unsur skrip html dan menampilkannya dengan warna yang berbeda sehingga mudah dibaca, seperti Notepad++, *Sublime Text*, *Visual Studio Code*, dan masih banyak lagi aplikasi lain yang sejenisnya.

2.8 CSS (*Cascading Style Sheet*)

Menurut Rohi Abdulloh (2022:56) CSS merupakan singkatan dari *Cascading Style Sheet*, yaitu dokumen web yang berfungsi mengatur elemen html dengan berbagai property yang tersedia sehingga dapat tampil dengan berbagai

gaya yang diinginkan. Sebagian orang menganggap CSS bukan termasuk salah satu bahasa pemrograman karena memang strukturnya yang sederhana, hanya berupa kumpulan-kumpulan aturan yang mengatur style elemen HTML. CSS, berperan sebagai pembentuk desain website yang mengatur setiap elemen html agar tampil menarik pada Browser, seperti warna, ukuran teks, jenis font, dan banyak lagi.

Cara kerja CSS dalam memodifikasi html cukup sederhana yaitu dengan memilih elemen html yang akan diatur, kemudian memberikan property yang sesuai dengan tampilan yang diinginkan. Fungsinya dalam pengembangan web dapat dijelaskan sebagai berikut:

1. Memisahkan tampilan dan konten.
2. Mengatur tampilan dan gaya.
3. Menyediakan responsivitas.

2.9 XAMPP

Xampp adalah suatu bundel web server yang populer digunakan untuk coba-coba di windows karena kemudahan instalisasinya. Xampp merupakan perangkat lunak bebas yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai server yang berdiri sendiri (localhost) yang terdiri atas Apache HTTP Server. Mysql database dan penerjemahan bahasa yang ditulis dengan bahasa pemrograman PHP. Nama Xampp merupakan singkatan dari Cross-Platform (X), Apache (A), MySQL (M), PHP (P), dan Perl (P) (Sarwindah, 2018).

2.10 *Flowchart*

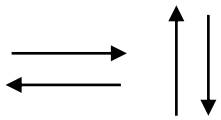
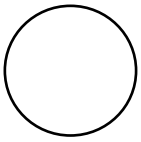
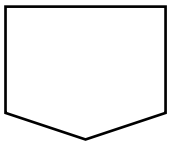
Flowchart atau bagan alur adalah diagram yang menampilkan langkah-langkah dan keputusan untuk melakukan sebuah proses dari suatu program. Dimana

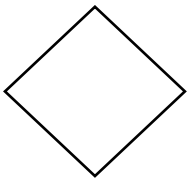
pada setiap Langkah digambarkan dalam bentuk diagram dan dihubungkan dengan garis atau arah panah. *Flowchart* berfungsi untuk memberikan gambaran jalannya sebuah program dari satu proses ke proses lainnya sehingga alur suatu program menjadi mudah dipahami oleh semua orang (Rony Setiawan, 2021).

2.10.1 Simbol *Flowchart*

Pada dasarnya symbol-simbol dalam *flowchart* memiliki arti yang berbeda-beda. Berikut adalah simbol-simbol yang sering digunakan dalam proses pembuatan *flowchart* (Rony Setiawan, 2021).

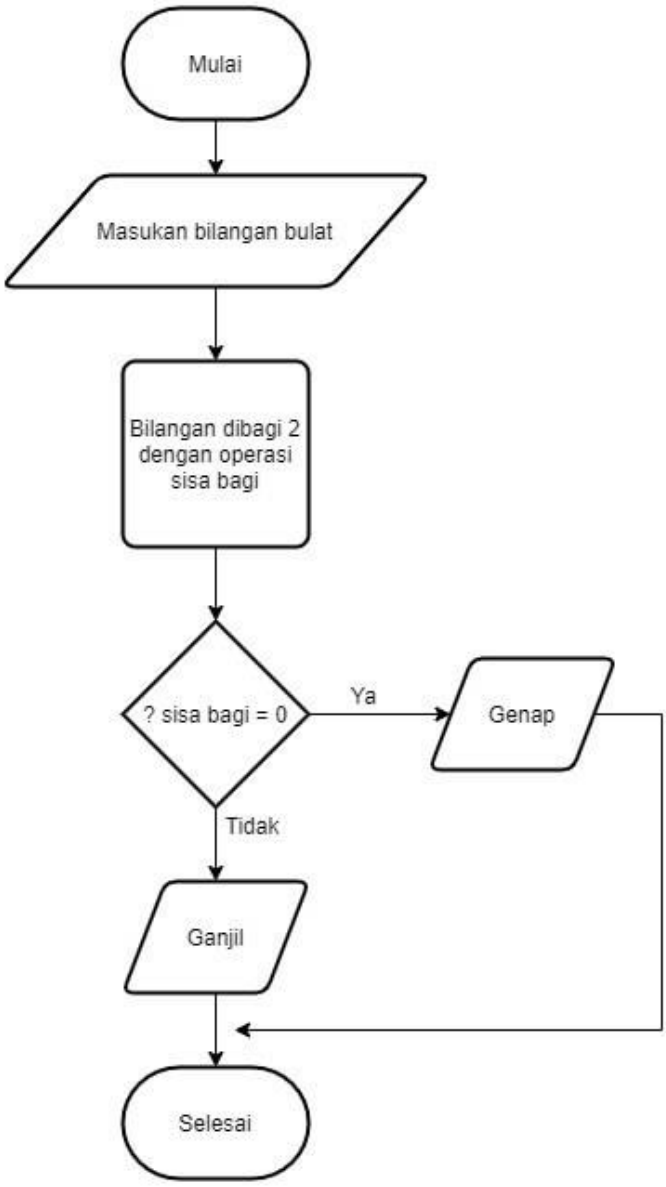
Tabel 2.1 Simbol-Simbol *Flowchart*

No.	Simbol	Nama Simbol	Fungsi
1.		Arus (*Flow Direction*)	Untuk menunjukkan garis alir dari proses.
2.		Penghubung (*On-Page Connector*)	Sebagai penyambungan proses dalam halaman yang sama.
3.		Penghubung (*Off-Page Connector*)	Sebagai penyambungan proses dalam halaman yang berbeda.
4.		Proses (*Processing*)	Untuk menunjukkan proses pengolahan data yang dilakukan oleh komputer.

No.	Simbol	Nama Simbol	Fungsi
5.		Operasi Manual (*Manual Operation*)	Untuk menunjukkan proses pengolahan yang tidak dilakukan oleh komputer.
6.		Titik Terminal (*Terminal Point*)	Untuk menunjukkan suatu permulaan (*start*) atau akhir (*end*) dalam suatu alur proses.
7.		Keputusan (*Decision*)	Untuk memilih proses berdasarkan kondisi yang ada.
8.		Proses Terdefinisi (*Predefined Process*)	Sebagai kumpulan langkah-langkah.
9.		Persiapan (*Preparation*)	Untuk pelaksanaan suatu bagian (sub-program) atau prosedur.
10.		Keluar-Masuk (*Input-Output*)	Untuk menyatakan proses *input* dan *output* pada alur proses.
11.		Dokumen (*Document*)	Untuk menyatakan *input* yang berasal dari dokumen dalam bentuk kertas atau *output* dicetak ke kertas.

2.10.2 Contoh *Flowchart*

Menurut (Rony Setiawan, 2021) untuk mengetahui sebuah *flowchart* sederhana diharuskan untuk mengetahui setiap symbol dan juga fungsinya. Berikut ini adalah sebuah contoh *flowchart* sederhana untuk menentukan apakah bilangan yang dimasukkan merupakan bilangan ganjil atau genap.



Gambar 2.7 Contoh *Flowchart* Sederhana untuk menentukan bilangan ganjil atau genap.