**Bab II**

**Tinjauan Pustaka**

**II.1 Bahasa Arab**

Bahasa Arab adalah salah satu bahasa [Semit Tengah](https://id.wikipedia.org/w/index.php?title=Semit_Tengah&action=edit&redlink=1), yang termasuk dalam rumpun [bahasa Semit](https://id.wikipedia.org/wiki/Bahasa_Semit) dan berkerabat dengan [bahasa Ibrani](https://id.wikipedia.org/wiki/Bahasa_Ibrani) dan [bahasa-bahasa Neo Arami](https://id.wikipedia.org/wiki/Bahasa_Aram#Bahasa_dan_dialek_Aram). Bahasa Arab memiliki lebih banyak penutur daripada bahasa-bahasa lainnya dalam rumpun bahasa Semit. Bahasa Arab dituturkan oleh lebih dari 280 juta orang. sebagai [bahasa pertama](https://id.wikipedia.org/wiki/Bahasa_ibu), yang mana sebagian besar tinggal di [Timur Tengah](https://id.wikipedia.org/wiki/Timur_Tengah) dan [Afrika Utara](https://id.wikipedia.org/wiki/Afrika_Utara). Bahasa ini adalah bahasa resmi dari 25 negara, dan merupakan bahasa peribadatan dalam agama [Islam](https://id.wikipedia.org/wiki/Islam) karena merupakan bahasa yang dipakai oleh [Al-Qur'an](https://id.wikipedia.org/wiki/Al-Qur%27an). Berdasarkan penyebaran geografisnya, bahasa Arab percakapan memiliki banyak variasi (dialek), beberapa dialeknya bahkan tidak dapat saling mengerti satu sama lain. Bahasa Arab modern telah diklasifikasikan sebagai satu makro bahasa dengan 27 sub-bahasa. [Bahasa Arab Baku](https://id.wikipedia.org/w/index.php?title=Bahasa_Arab_Baku&action=edit&redlink=1) atau Bahasa Arab Sastra diajarkan secara luas di sekolah dan universitas, serta digunakan di tempat kerja, pemerintahan, dan media massa.

Bahasa Arab Baku berasal dari [Bahasa Arab Klasik](https://id.wikipedia.org/wiki/Bahasa_Arab_Klasik), satu-satunya anggota rumpun [bahasa Arab Utara Kuno](https://id.wikipedia.org/w/index.php?title=Bahasa_Arab_Utara_Kuno&action=edit&redlink=1) yang saat ini masih digunakan, sebagaimana terlihat dalam [inskripsi](https://id.wikipedia.org/wiki/Prasasti) peninggalan Arab pra-Islam yang berasal dari abad ke-4. Bahasa Arab Klasik juga telah menjadi bahasa kesusasteraan dan bahasa peribadatan [Islam](https://id.wikipedia.org/wiki/Islam) sejak lebih kurang [abad ke-6](https://id.wikipedia.org/wiki/Abad_ke-6). [Abjad Arab](https://id.wikipedia.org/wiki/Abjad_Arab) ditulis dari kanan ke kiri.C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ain_terpisah.png Bahasa Arab telah memberi banyak kosa kata kepada bahasa lain dari dunia Islam, sama seperti peranan [Latin](https://id.wikipedia.org/wiki/Bahasa_Latin) kepada kebanyakan bahasa Eropa. Semasa [Abad Pertengahan](https://id.wikipedia.org/wiki/Abad_Pertengahan) bahasa Arab juga merupakan alat utama budaya, terutamanya dalam sains, matematik adan filsafah, yang menyebabkan banyak bahasa Eropa turut meminjam banyak [kosa kata](https://id.wikipedia.org/wiki/Kosakata) dari bahasa Arab. [15]

C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_mim_diakhir.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_mim_diawal.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_mim_terpisah.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_lam_ditengah.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_lam_terpisah.pngTabel II.1. Huruf Arab dan empat bentuknya

| **No** | **Nama** | **Alih Aksara** | **Bentuk Terisolasi** | **Bentuk Tersambung** | | |
| --- | --- | --- | --- | --- | --- | --- |
| **Awal** | **Tengah** | **Akhir** |
| 1. | alif | - | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_alif_terpisah.png | - | - | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_alif_diakhir.png |
| 2. | ba’ | b | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ba_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ba_diawal.png |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ba_diakhir.png |
| 3. | ta’ | t | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ta_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ta_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ta_ditengah.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ba_ditengah.png |  |
| 4. | tsa | ts |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_tsa_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_tsa_ditengah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_tsa_diakhir.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ta_diakhir.png |
| 5. | jim | j | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_jim_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_jim_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_jim_ditengah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_jim_diakhir.png |
| 6. | ha | h | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ha_terpisah.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_tsa_terpisah.png |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ha_ditengah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ha_diakhir.png |
| 7. | kha | kh | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_kha_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_kha_diawal.png |  |  |
| 8. | dal | d |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ha_diawal.png- | - |  |
| 9. | zal | z | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_dzal_terpisah.png | - | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_kha_ditengah.png- | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_kha_diakhir.png |
| 10. | ra’ | r | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ra_terpisah.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_dal_terpisah.png | - | - | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_dal_ditengah.png |
| 11. | za’ | z |  | - | - | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ra_diakhir.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_dzal_ditengah.png |
| 12. | sīn | s |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sin_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sin_ditengah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sin_diakhir.png |
| 13. | syīn | sy | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_za_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sheen_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sheen_ditengah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_za_ditengah.png |
| 14. | shād | sh | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sin_terpisah.png |  |  |  |
| 15. | dād | d | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sad_terpisah.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sheen_terpisahtr.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sad_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sad_ditengah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sheen_diakhir.png |
| 16. | ta | t |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ta_diawal.png |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_sad_diakhir.png |
| 17. | za | z | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ta_terpisah.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_dhad_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_dhad_diawal.png- | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_dhad_ditengah.png- | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_dhad_diakhir.png |
| 18. | ‘ain | ‘ | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ain_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ain_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ta_ditengah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ta_diakhir.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ain_diakhir.png |
| 19. | gaīn | g | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_za_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ghoin_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ghoin_ditengah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ghoin_diakhir.png C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_za_ditengah.png |
| 20. | fa’ | f |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_fa_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_fa_ditengah.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ain_ditengah.png |  |
| 21. | qāf | q | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ghoin_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_qaf_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_qaf_ditengah.png |  |
| 22. | kāf | k | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_fa_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_kaf_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_kaf_ditengah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_kaf_diakhir.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_fa_diakhir.png |
| 23. | lām | l | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_qaf_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_lam_diawal.png |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_lam_diakhir.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_qaf_diakhir.png |
| 24. | mīm | m | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_kaf_terpisah.png |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_mim_ditengah.png |  |
| 25. | nūn | n |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_nun_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_nun_ditengah.png |  |
| 26. | wāwu | w | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_nun_terpisah.png | - | - | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_waw_ditengah.png |
| 27. | ha’ | h | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_habesar_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_habesar_diawal.png |  | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_habesar_diakhir.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_nun_diakhir.png |
| 28. | ya’ | y | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ya_terpisah.pngC:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_waw_terpisah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ya_diawal.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ya_ditengah.png | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_ya_diakhir.png |
| 29. | hamzah | - | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_hamzah_diakhir.png | - | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_habesar_ditengah.png- | - |
| 30. | lām + alif | - |  | - | - | C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_lamalif_diakhir.png |

C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\timesnewroman_lamalif_terpisah.png

Tulisan Arab memiliki beberapa karakteristik yang berbeda dari tulisan latin. Berikut adalah karakteristik dari tulisan Arab :

1. Huruf arab ditulis dari kanan ke kiri
2. Tidak memiliki huruf kapital
3. Terdiri dari 29 Huruf dasar dan 1 huruf ligatur.
4. Huruf berubah bentuk tergantung pada posisi mereka dalam kata, sehingga memiliki 4 bentuk yakni terisolasi, posisi di awal kata, di tengah kata dan di akhir kata.
5. Karakter bersambung dalam tulisan cetak maupun tulisan tangan
6. Huruf Arab memiliki banyak kemiripan, beberapa huruf memiliki bentuk yang sama, namun dapat dibedakan dengan jumlah titik dan posisi titik.
7. Pada umumnya terdiri dari lekukan dan putaran

**II.2 Pengenalan Huruf Arab**

Konsep pengenalan huruf arab secara umumnya dapat dikategorikan dua bagian, yaitu pengenalan huruf terisolasi dan pengenalan huruf dalam kalimat. Pengenalan huruf terisolasi berkaitan dengan proses mengenali huruf Arab yang terisolasi atau berdiri sendiri. Sedangkan pengenalan huruf dalam kalimat Arab adalah proses mengenali huruf yang terdapat dalam sebuah kalimat Arab. Oleh karena sifat tulisan Arab yang bersambung baik tulisan cetak maupun tulisan tangan, maka proses pengenlahan huruf dalam kalimat Arab membutuhkan proses segmentasi, sehingga tulisan yang bersambung dapat terbagi menjadi huruf – huruf tunggal sesuai dengan posisi-nya pada kalimat, yakni di terisolasi, diawal, ditengah dan diakhir.

**II.3 Praolah Citra**

Pengolahan awal atau praolah (*preprocessing*) citra diperlukan sebelum  
beberapa proses. Tujuan dari pengolahan awal ialah agar pengolahan lebih mudah dilakukan. Pengolahan awal umumnya terdiri dari penghilangan derau (*noise*), penghalusan atau penajaman gambar, dan pengubahan gambar menjadi citra abu-abu atau citra biner.

**II.3.1 Binerisasi**

Binerisasi pada citra adalah proses merubah citra kedalam bentuk biner (0 dan 1). Gambar yang pada mulanya berwarna akan dirubah menjadi hitam dan putih. Citra biner adalah citra yang setiap pikselnya hanya memiliki 2 kemungkinan derajat yakni 0 dan 1. Konversi citra berwarna ke biner ini dilakukan pada saat awal citra dibuka oleh sistem.

Cara yang dilakukan adalah dengan melakukan pengambangan (*threshold*) pada  
masing-masing kanal warna. Ambang yang digunakan adalah 150. Jika kanal warna bernilai kurang dari 150 maka akan diubah menjadi warna hitam, dan jika wanra lebih dari 150 akan diubah menjadi warna putih

## II.3.2 Penipisan

Penipisan adalah proses pengambilan tulang dari suatu pola, salah satu penggunaan penipisan adalah dalam aplikasi pengenalan pola. Citra yang digunakan adalah citra yang telah dilakukan binerisasi terlebih dahulu sehingga citra tersebut menjadi citra biner. Proses ini mengikis piksel sebanyak mungkin tanpa mempengaruhi bentuk umum. Setelah piksel di kikis, pola tersebut tetap harus dikenali. Citra hasil dari algoritpa penipisan disebut dengan *skeleton.* Pada umumnya suatu algoritma penipisan yang dilakukan terhadap citra biner memiliki kriteria sebagai berikut :

1. *Skeleton* tetap menjaga struktur keterhubungan yang sama dengan citra awal.
2. *Skeleton* dari citra kira-kira berada di bagian tengah dari citra awal sebelum dilakukan penipisan.
3. *Skeleton* melikiki bentuk yang mirip dengan citra awal.
4. *Skeleton* mengandung jumlah oiksel yang setipis mungkin namun tetap memenuhi kriteria sebelumnya.



Gambar II.2 Pola huruf dan hasil penipisan

Penipisan berguna apabila tidak diperlukan pada ukuran dari pola melainkan pada posisi relatif goresan dari pola. Ada beberapa algoritma yang dirancang untuk tujuan ini. Dalam penelitian ini penipisan dilakukan dengan algorima Stentiford.

Pada algoritma Stentiford ada 4 buah template yang dipakai , template 3 x 3 yaitu :



Gambar II.3 Template algoritma Stentiford

Berikut adalah langkah – langkah yang dilakukan untuk mendapatkan *skeleton* dari suatu citra dengan algoritma Stentiford

1. Cari lokasi piksel (i, j) yang cocok dengan *template* T1. Dengan *template* T1 semua piksel di bagian atas dari citra akan dihapus. Pencocokkan *template* ini bergerak dari kiri ke kanan dan dari atas ke bawah.
2. Bila piksel tengah bukan merupakan *endpoint* dan mempunyai jumlah konektivitas 1, maka tandai *pixel* untuk kemudian dihapus.
3. Endpoint adalah piksel yang merupakan batas akhir dan hanya terhubung 1 piksel saja. Artinya, jika piksel hitam hanya memiliki satu tetangga hitam dari delapan kemungkinan tetangga.
4. Jumlah konektivitas adalah ukuran berapa banyak objek yang terhubung dengan piksel tertentu. Berikut adalah rumus untuk menghitung jumlah konektivitas.

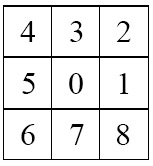


Dimana:

Nk merupakan nilai dari 8 tetangga di sekitar *pixel* yang akan dianalisa, dan nilai S = {1,3,5,7}

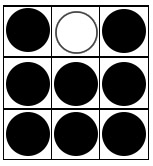
N0 adalah nilai dari piksel tengah.

N1 adalah nilai dari piksel pada sebelah kanan *central pixel* dan sisanya diberi nomor berurutan dengan arah berlawanan jarum jam



Gambar II.4 Penomoran nilai k pada algoritma Stentiford

1. Ulangi langkah 1 dan 2 untuk semua piksel yang cocok dengan Template T1.
2. Ulangi langkah 1–3 untuk *template* T2, T3 dan T4.
   1. Pencocokkan *template* T2 akan dilakukan pada sisi kiri dari obyek dengan arah dari bawah ke atas dan dari kiri ke kanan.
   2. Pencocokkan *template* T3 akan dilakukan pada sisi bawah dari obyek dengan arah dari kanan ke kiri dan dari bawah ke atas.
   3. Pencocokkan *template* T4 akan dilakukan pada sisi kanan dari obyek dengan arah dari atas ke bawah dan dari kanan ke kiri.
3. Piksel yang ditandai untuk dihapus diubah menjadi putih.



Gambar II.5 Jumlah Konektivitas algoritma Stentiford

Sebagai contoh jumlah konektivitas pada gambar II.5 adalah 1, berikut perhitungan jumlah konektivitas dari algoritma Stentiford:

Rumus menghitung jumlah konektifitas:



Saat k = 1 : Cn = N1 – (N1 . N2 . N3)

Cn =1 – (1 . 1 . 0)

Cn =1 – 0

Cn =1

Saat k = 3 : Cn = N3 – (N3 . N4 . N5)

Cn =0 – (0 . 1 . 1)

Cn =0 – 0

Cn = 0

Saat k = 5 : Cn = N5 – (N5 . N6 . N7)

Cn =1 – (1 . 1 . 1)

Cn =1 – 1

Cn = 0

Saat k = 7 : Cn = N7 – (N7 . N8 . N1)

Cn =1 – (1 . 1 . 1)

Cn =1 – 1

Cn = 0

Cn = 1 + 0 + 0 + 0

Cn = 1

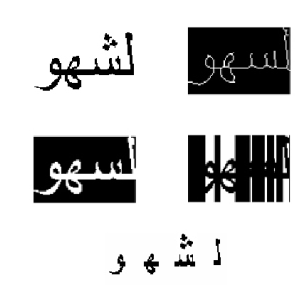
**II.3.3 Segmentasi**

Segmentasi atau proses pemotongan adalah proses yang penting dalam sistem pengenalan huruf khususnya huruf Arab, dikarenakan kalimat bahasa Arab yang ditulis secara bersambung. Segmentasi memiliki tujuan untuk memisahkan baris, kata dan huruf dalam citra. Setelah terpisah, masing-masing huruf dapat diidentifikasi lebih lanjut.

Segmentasi adalah berusaha memotong kata menjadi huruf per huruf khususnya pada tulisan bersambung seperti tulisan Arab. Dalam pengenalan huruf latin proses segmentasi internal dapat dilakukan dengan mudah karena huruf – huruf terpisah satu sama lain (kecuali tulisan tangan). Berbeda dengan tulisan Arab yang memiliki sifat bersambung, baik ditulis dengan tangan maupun tulisan cetak. Oleh karena itu masalah segmentasi dalam tulisah arab perlu diperhatikan.

Zidouri (2010) telah memberikan metode baru untuk segmentasi huruf tulisan Arab cetak. Langkah awal dari metode Zidouri ialah melakukan penipisan terhadap citra upakata. Algoritma Zidouri memperkirakan kemungkinan titik potong antar huruf dengan mencari garis horizontal pada citra tulang. Setiap baris pada citra dipindai dari kiri ke kanan untuk mencari deretan piksel horizontal atau pita horizontal dengan lebar lebih dari *Ls*. Nilai *Ls* adalah lebar terpendek dari huruf terkecil.

Setelah deretan piksel horizontal ditemukan, dilakukan pengecekan piksel pada  
bagian atas dan bawah pita piksel tersebut. Jika tidak ditemukan piksel, batas  
deretan piksel tersebut dianggap sebagai kandidat pita pandu segmentasi.  
Kandidat-kandidat pita pandu ini digambar pada citra sesuai lebar dan lokasi  
ditemukannya.



II.13 Contoh segmentasi upakata menjadi huruf (Zidouri, 2010)

Dari kandidat pita pandu vertikal, dilakukan ekstraksi beberapa fitur untuk  
memilih pita yang akan digunakan untuk memotong upakata menjadi huruf. Fitur  
yang diekstrak adalah lebar pita, jarak antar pita, urutan pita, posisi terhadap lini  
basis, dan titik tengah pita. Terdapat beberapa aturan untuk memilih pita  
pemotongan yang akan digunakan dari pita-pita yang ada. Misalnya, apakah  
rentetan piksel yang menyebabkan pita ditemukan berada di atas atau di bawah  
lini basis, apakah jarak pita dengan pita di kanannya kurang dari lebar maksimum  
huruf, dll. Tahapan pemotongan dapat diamati pada Gambar II.13 dan metode  
lebih rinci dapat ditelusuri pada (Zidouri, 2010)

**II. 4 Tahap Ekstraksi Fitur**

Pada penelitian ini terdapat 3 tahap ekstraksi fitur, yakni fitur posisi titik, fitur jumlah titik dan chaincode. Berikut adalah penjelasan dari masing – masing fitur :

**II.4.1 Chaincode**

1

8

2

3

4

5

6

7

8 Arah Chain code

Pada pengenalan pola, chaincode merupakan suatu teknik untuk menggambarkan suatu struktur dari suatu objek. Chain code diperoleh dengan cara menelusuri piksel batas objek berdasarkan arah-arah yang telah ditetapkan. Hasil dari chain code adalah angka – angka yang menunjukkan arah yang mewakili batas objek. Pencarian chain code hanya bisa dilakukan pada citra biner.

Berikut adalah cara pengambilan chain code dari suatu objek di dalam sebuah citra :

1. Cari piksel hitam yang hanya memiliki 1 tetangga dengan cara menelusuri pixel pada citra dimulai dari pojok kiri atas sampai menemukan piksel yang berwarna hitam, jika tidak ditemukan piksel hitam yang hanya memiliki 1 tetangga maka ambil piksel hitam pertama yang ditemui.
2. Lakukan iterasi pada gambar
3. Ubah piksel saat ini menjadi 0
4. Ikuti prioritas arah 1 sampai 8
5. Pindah posisi piksel
6. Tambahkan arah ke chain code

Panjang chain code dari suatu objek dapat berubah – ubah sesuai bentuk dari suatu objek tersebut. Pada penelitian ini akan dilakukan klasifikasi dengan Neural Network. Input dari neural network harus tetap, tidak dapat berubah – ubah. Oleh karena itu pada penelitian ini akan dilakukan normalisasi chain code. Yaitu membuat panjang chain code dari suatu citra tetap jumlahnya dan tidak berubah-ubah.

**II.4.1.1 Normalisasi Chaincode**

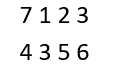
Normalisasi suatu chain code dilakukan agar jumlah chain code yang pada mulanya berubah – ubah sesuai bentuk dari suatu objek dapat dijadikan tetap. Pada penelitian ini panjang chain code dari setiap citra huruf Arab akan ditetapkan menjaidi 10.

Berikut adalah langkah – langkah dari normalisasi chain code :

1. Chain code diubah menjadi matriks 2 dimensi. Baris pertama adalah nilai dari chain code. Baris kedua adalah frekuensi terjadinya setiap angka dalam chain code.

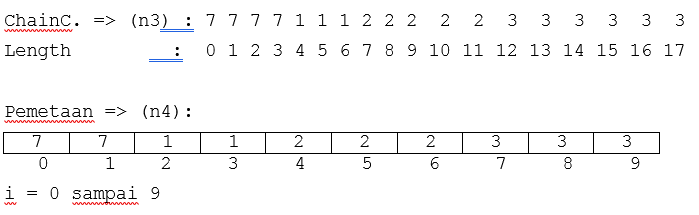
Seperti chain code berikut : 7777311122222583353333, setlah dilakukan tahap pertama dari normalisasi chain code akan menjadi 2 x 9 matriks :



1. Hilangkan semua nilai yang memiliki frekuensi hanya 1.
2. Tampilkan chaincode sesuai dengan frekuensi terjadinya :

777711122222333333

1. Lakukan pemetaan terhadap chaincode



n4[0] = n3[round(i/9 x n3length-1)]

= n3[0/9x17]

= n3[0] = 7

n4[0] = 7

n4[1] = n3[1/9x17] = n3[2] = 7

n4[2] = n3[2/9x17] = n3[4] = 1

n4[3] = n3[3/9x17] = n3[6] = 1

n4[4] = n3[4/9x17] = n3[8] = 2

n4[5] = n3[5/9x17] = n3[9] = 2

N4[6]= n3[6/9x17] = n3[11] = 2

n4[7] = n3[7/9x17] = n3[13] = 3

n4[8] = n3[8/9x17] = n3[15] = 3

N4[9]= n3[9/9x17] = n3[17] = 3

7 7 1 1 2 2 2 3 3 3

**II.4.2 Fitur Jumlah Titik**

Fitur jumlah titik merupakan fitur yang penting dalam huruf Arab, karena beberapa huruf Arab memiliki bentuk yang sama namun hanya dibedakan berdsarkan jumlah titik. Seperti huruf ب, ث, dan ت.

Jumlah titik diperoleh dengan cara melakukan iterasi pada gambar huruf dari sudut kiri atas ke kanan, kemudian kebawah, jika ditemukan titik hitam pertama hitung chaincode dari titik hitam tersebut. Huruf yang memiliki titik akan memiliki chaincode lebih dari 1. Kemudian akan diperiksa, jika chaincode yang ditemukan kurang dari 7 maka akan dihitung sebagai chaincode titik, dan dilakukan penjumlahan terhadap jumlah titik. Jika chaincode memiliki panjang lebih dari 7 maka akan dihitung sebagai chaincode dari *body* huruf.

Seperti pada huruf, ث yang memiliki 4 chaincode. Chaincode pertama adalah “60” , chaincode kedua adalah “60”, chaincode ketiga adalah “60” dan chaincode keempat adalah “66666667654545444488881818178787880”. Dari 4 chaincode tersebut didapat 3 chaincode yang memiliki panjang kurang dari 7. Kemudian setiap kemunculan chaincode yang kurang dari 7 akan dilakukan perhitungan jumlah titik. Dari 4 chaincode pada huruf ث didapat jumlah titik adalah 3.

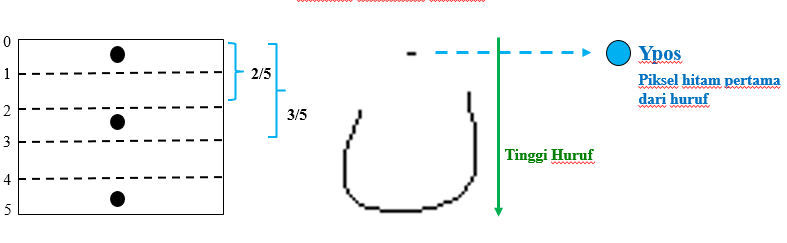
C:\Users\ainawind27\AppData\Local\Microsoft\Windows\INetCache\Content.Word\tahoma_tsa_terpisah_zhangsuen.png

Hasil thinning huruf Tsa

**II.4.3** **Fitur Posisi Titik**

Fitur posisi titik merupakan hal yang penting pada huruf Arab. Beberapa huruf Arab memiliki bentuk dan jumlah titik yang sama, namun dibedakan berdasarkan posisi dari ttik tersebut.

Posisi titik diperoleh dengan cara menghitung piksel hitam pertama yang ada pada gambar yang disebut dengan YPos, kemudian gambar akan dibagi menjadi menjadi 5 bagian. Jika YPos berada pada posisi kurang dari 2/5 tinggi gambar maka posisi titik adalah diatas yang diwakili dengan angka 0. Jika YPos berada pada posisi kurang dari 3/5 tinggi gambar maka posisi titik berada ditengah yang diwakili dengan angka 1. Jika YPos berada pada posisi lebih dari 3/5 tinggi gambar maka posisi titik adalah dibawah yang diwakili dengan angka 2.



Cara menentukan posisi titik

**II.5 Klasifikasi dan Penelitian Terkait**

Penelitian tentang pengenalan huruf Arab lebih sedikit dibandingkan penelitian tentang huruf latin. Beberapa penelitian dalam bidang ini di antaranya adalah oleh Rana, dkk. (2015), Mehmood (2007), dan Albadr (2013).

Rana, dkk. (2015) mengembangkan sistem pengenalan karakter huruf Arab dari tulisan tangan. Ekstraksi fitur yang digunakan adalah jumlah komponen yang terhubung, jumlah lubang, jumlah titik, posisi dari titik dan kerapatan atas dan bawah huruf. Metode klasifikasi yang digunakan adalah Artificial Neural Network (ANN). ANN dilatih menggunakan algoritma Hopfield. Hasil penelitian menunjukkan sistem dapat mengenali 8 tulisan tangan huruf Arab **(ا ب ش س خ ض ع و)** dengan rata – rata pengenalan 77.25% Hal ini dapat dicapai karena sistem hanya mengenali karakter huruf Arab tunggal tanpa melalui proses segmentasi huruf.

Mehmood (2007) mengembangkan sistem pengenalan huruf Arab. Metode klasifikasi yang digunakan adalah Support Vector Machine (SVM), masukan dari SVM adalah ekstraksi fitur *Momment invariant*, jumlah dan posisi titik, dan jumlah lubang dari setiap huruf. Pada klasifikasi digunakan *Multiclass* SVM dengan 58 kelas (bentuk dari karakter). Metode ini menghasilkan tingkat mengenalan 98,34%. Sama dengan penelitian Rana, dkk(2015) penelitian ini dapat mencapai tingkat akurasi yang tinggi karena tidak melalui tahap segmentasi huruf.

Albadr (2013) mengembangkan sistem pengenalan huruf arab dalam rangkaian kata. Adapun tahap yang digunakan adalah pemrosesan awal, segmentasi, penulangan, ekstraksi fitur dan klasifikasi. Pada tahap segmentasi huruf dilakukan dengan algoritma Zidouri. Pada penelitian ini di ekstrak 12 fitur. Tiap fitur dibedakan menjadi 3 bagian, yakni bagian utama, garis keliling tulang dan objek sekunder (titik, hamzah dan zig – zag) pada setiap huruf. Penelitian ini mencapai tingkat akurasi pengenalan 48,3%.

**II.6 Machine Learning**

Menurut Tom M. Mitchel (1997), defenisi fomal tentang machine learning sebagai berikut :

“Sebuah program komputer dikatakan belajar dari pengalaman E yang bergabung pada target T dan ukuran kinerja program P jika kinerja pada target T, menggunakan ukuran P, ditingkatkan oleh pengalaman E.”

Dengan kata lain, *machine learning* meningkatkan mesin untuk memberikan hasil prediksi pada data baru setelah mendapat pengalaman dari sekumpulan data (data training). Berdasarkan input yang diberikan pada data training, *machine learning* dikelompokkan menjadi dua jenis, yaitu :

**II.6.1 *Supervised Learning***

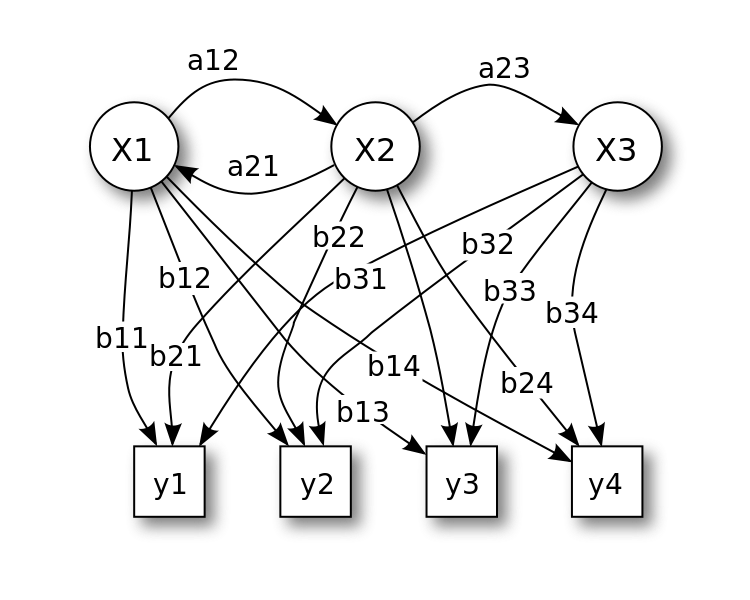
Pada *supervised learning*, *data training* disertai target pada setiap datanya, {xi,ti}, i = 1…N, dengan **x** adalah vector input dan t adalah target. Tujuan dari *supervised learning* adalah membangun model yang dapat memberikan hasil/output secara benar untuk suatu data input. *Supervised learning* digunakan untuk *classification, regression, ordinal regression, ranking,* dll.

**II.6.2 *Unsupervised Learning***

Pada *unsupervised Learning,* data training tidak disertai target pada setiap datanya, {xi}, i = 1…N, dengan **x** adalah vector input. Tujuan dari *unsupervised learning* adalah membangun model yang dapat menemukan variable tersembunyi dari *data training*. *Unsupervised learning* digunakan untuk *clustering, concept extraction, 0-97recommendation, density estimation, dimensionality reduction,* dll.

## II.7 Hidden Markov Model (HMM)

*Hidden Markov Model* (HMM) adalah perluasan dari rantai Markov dimana *state*nya tidak dapat diamati secara langsung (tersembunyi). Perubahan *hidden state* hanya dapat diobservasi melalui pengamatan variable lain.



Gambar II.14 *Hidden Markov Model* (HMM) [16]

Terdapat tiga masalah utaa yang dapat diselesaikan oleh Hidden Markov Model.

**II.7.1 Evaluasi**

Operasi evaluasi dalam HMM adalah perhitungan probabilitas dari urutan nilai observasi yang diberikan oleh *Hidden Markov Models.* Algoritma yang banyak digunakan untuk penyelesaian masalah observasi adalah algoritma maju (*Forward Algorithm*) yang keadaannya mengalir kedepan. Algoritma mundur (*Backward Algorithm*) yang keadaannya mengalir kebelakang dari observasi terakhir pada saat *T,* dan algoritma maju – mundur (*forward-backward algorithm*) yang merupakan gabungan dari algoritma *forward – backward.* .[10]

1. Algoritma maju

Jika variable forward , pada saat *t* dan state *i*, maka persamannya

(II.1)

Dengan *O* = indeks matrik terobservasi

Penyelesain dengan n keadaan dan observasi sampai T secara iterasi

* Inisialisasi :

, (II.2)

dimana = matrik state awal dan = matriks pertama yang terobservasi.

* Induksi :

… (II.3)

dimana :

= jumlah state

= matrik transisi.

* Terminasi :

(II.4)

1. Algoritma mundur

Keadaan mengalir ke belakang dari observasi terakhir saat *t.* Persamaan probabilitas mundur sebagai berikut:

(II.5)

dan dianalogikan dengan prosedur *forward* dengan langkah:

* Inisialisasi

(II.6)

* Induksi

, (II.7)

, (II.8)

**II.7.2 Pengkodean (*Decoding*)**

Permasalahan *decoding* ini yaitu menemukan barisan *state* terbaik (optimal) yang berasosiasi dengan barisan observasi dari sebuah model yang juga telah diketahui. Barisan *state* yang mempunyai probabilitas tertinggi dalam menghasilkan barisan observasi yang telah diketahui sebelumnya. Untuk menentukan keadaan tersembunyi dari suatu barisan observasi perlu digunakan suatu metode yang mempertimbangkan probabilitas transisi *state* pada proses pencarian barisan *state* yang paling optimal. Metode yang digunakan untuk penyelesaian masalah ini antara lain algoritma Viterbi. Langkah – langkah pengkodean sebagai berikut.

* Inisialisasi :

, (II.9)

* Rekursi :

, (II.10)

* Terminasi :

(II.11)

Dimana :

*P* = Probabilitas

**II.7.3 Learning**

Operasi learning dalam HMM adalah melatih parameter HMM jika diberikan data set barisan – barisan tertentu agar dapat menemukan himpunan transisi state yang paling mungkin beserta probabilitas output. Untuk menyelesaikan permasalahan terakhir pada HMM ini, biasanya digunakan algoritma *Baum-Welch* yang merupakan kasus khusus dari algoritma EM (Ekspektasi Maksimum). Algoritma EM sendiri merupakan algoritma yang digunakan untuk mempelajari model – model probabilistik dalam suatu kasus yang melibatkan keadaan – keadaan tersembunyi.

1. Parameter A

Parameter A ditunjukan dengan kumpulan status transisi yang membentuk matrik tertentu yaitu

(II.12)

bentuk

(II.13)

Merupakan peluang ketika state *j* pada waktu *t+1* jika pada waktu *t*  berada di state *i*

= Probabilitas transisi dari state i ke state j

*P*  = probabilitas

*n* = banyaknya hidden state dalam model

1. Parameter B

Parameter B ditunjukkan bahwa

(II.14)

yang merupakan

(II.15)

dan (II.16)

= Probabilitas distribusi matriks observasi

*m* = banyaknya simbol observasi yang berbeda pada tiap state

Pada *continuos density* HMM sering dikarakterisasi oleh fungsi kerapatan (*density function*) atau campuran fungsi kepadatan tertentu di setiap *state* . Dengan asumsi penggunaan *Gaussian* *Mixture*, kepadatan emisi *state* 𝑗 didefinisikan sebagai:

(II.17)

Dimana :

K = *number of mixture*

= *mixing coefficient*

untuk *Gaussian* pada *state* j dengan batasan stokastik dengan persamaan berikut.

(II.18)

Dimana :

= *Gaussian density function* dengan mean dan matrik kovarian untuk *campuran*.

1. Matriks awal di state *i*

Diperlukan inisialisasi matriks awal status yang ditunjukan oleh.

dengan :

(II.19)

Sehingga HMM dapat dilambangkan dengan .

## II.8 Support Vector Machine

Support Vector Machine (SVM) merupakan salah satu metode *machine learning* untuk *pattern recognition*, yang dalam penelitian ini adalah *image recognition*. Algoritma SVM pertama kali dikembangkan oleh Vladimir Vapnik (Bishop,2006). Konsep dasar SVM ialah mengklasifikasikan data menjadi dua kelas yang berbeda dengan cara membuat membuat *hyperplane* atau suatu bidang yang merupakan fungsi klasifikator antara dua kelas dengan menggunakan konsep memaksimumkan margin.

Bentuk umum klasifikator SVM ialah sebagai berikut :

y = *sign*(wTx + b) (II.20) Keterangan :

y merupakan nilai target pada setiap vector baris

x Rn merupakan vector yang dimensinya bergantung dari *n* banyaknya vitur

w Rn vector yang menjadi parameter bobot

b bias atau eror berupa skalar

Hyperplane yang dihasilkan SVM dapat menklasifikasikan data menjadi dua kelas, yaitu kelas positf dan kelas negative yang dimodelkan sebagai berikut :

wTxi + b ≥1, untuk yi bernilai 1 (II.21)

wTxi + b ≥1, untuk yi bernilai -1 (II.22)

dapat dirumuskan sebagai berikut

yi (wTxi + b) ≥1 (II.23)

Misal dibuat dua bidang yang sejajar dengan hyperplane, untuk kelas B+ dapat dirumuskan sebagai berikut,

B+ : wTxi + b = 1 (II.24)

Untuk kelas B- dapat dirumuskan sebagai berikut,

B- : wTxi + b = -1 (II.25)

Margin adalah jarak antara bidang B+ dan B-

X1

Hyperplane

yi = 1

B+ : wTxi + b = 1

B- : wTxi + b = -1

yi = -1

||W||

X2

Gambar II.15 *Hyperplane*

**II.8.1 Fungsi Kernel**

Pada umumnya masalah dalam domain dunia nyata jarang bersifat linear separable. Kebanyakan bersifat non linear. Oleh karena itu untuk menyelesaikan masalah non linear, SVM dapat dimodifikasikan dengan memasukkan fungsi kernel. Dalam non linear SVM, data dipetakan oleh fungsi Ф () ke ruang vector yang berdimensi lebih tinggi. Pada ruang vector yang baru ini, hyperplane yang memisahkan dua kelas tersebut dapat di konstruksikan.

Hal ini sejalan dengan teori Cover yang menyatakan “Jika suatu transformasi bersifat non linear dan dimensi dari feature space cukup tinggi, maka data pada input space dapat dipetakan ke feature space yang baru, dimana pattern-pattern tersebut pada probabilitas tinggi dapat dipisahkan secara linear” (Cover, 1965)

Hyperplane

Gambar II.16 Kernel pada SVM

Pada gambar diatas dapat dilihat bahwa pada kelas kuning dan pada kelas merah yang berada pada input space berdimensi dua tidak dapat dipisahkan secara linear. Selanjutnya ditunjukkan bahwa fungsi Ф memetakan tiap data pada input space tersebut ke ruang vector baru yang berdimensi lebih tinggi (dimensi 3), dimana kedua kelas dapat dipisahkan secara linear oleh sebuah hyperplane. Pemetaan ini dilakukan dengan cara menjaga topologi data, dalam artian dua data yang berjarak dekat pada input space, sebaliknya dua data yang berjarak jauh pada feature space.

**II.8.2 Multi class SVM**

*Hyperplane* yang dihasilkan SVM hanya bisa mengklasifikasikan dua kelas. Sedangkan pada kenyataannya akan banyak ditemukan kasus yang lebih dari dua kelas. Oleh karena itu dapat digunakan *Multiclass* SVM untuk mengklasifikasikan permasalahan yang memiliki lebih dari dua kelas. Terdapat beberapa pendekatan yang dapat menyelesaikan masalah *Multiclass* dengan menggunakan SVM, diantaranya dalah *One Vs All* (OVA) dan *One Vs One* (OVO).

**II.8.2.1 *One Vs All* (OVA)**

Memisahkan permasalahan yang ditemui dari *N* kelas. Sehingga akan dibuat *N decision boundary. Decision boundary* yang dihasilkan merupakan hasil dari pencarian *hyperplane* dari kelas ke *i* dengan kelas sisa yang lainnya. Berikut dilakukan simulasi dengan menggunakan tiga kelas.

A

A

B

B

A

A

B

B

C

C

C

C

Gambar II.17 Contoh pendekatan OVA

Pada gambar diatas terdapat tiga kelas yaitu kelas A, kelas B, dan kelas C. karena menggunakan metode OVA akan dibuat tiga *decision boundary* dari tiga kelas tersebut. Setiap kelas dicari *decision boundary* dengan kelas sisanya.

A

A

B

B

D1

B

A

A

B

C

C

C

C

Gambar II.18 *Decision boundary D1*pada pendekatan OVA

Pada gambar 2.5 dibuat *decision boundary* dari kelas A dengan kelas sisanya (kelas B dan C). Kelas A dilabelkan sebagai positif, sedangkan kelas sisanya dilabelkan menjadi negative, sehingga D1 merupakan decision boundary yang dihasilkan dari kelas A dan kelas sisanya (kelas B dan kelas C).

A

A

A

A

B

B

B

D2

B

C

C

C

C

Gambar II.19 *Decision boundary D2* pada pendekatan OVA

Selanjutnya, sesuai dengan Gambar 2.6, dibuat *decision boundary* dari kelas B dengan kelas sisanya (kelas A dan kelas C). Kelas B dilabelkan sebagai positif, sedangkan kelas sisanya dilabelkan menjadi negatif, sehingga D2 merupakan *decision boundary* yang dihasilkan dari kelas B dan kelas sisanya (kelas A dan kelas C).

B

B

A

A

A

A

D3

B

B

c

c

c

c

Gambar II.20 *Decision boundary D3* pada pendekatan OVA

Lalu yang terakhir adalah mencari *decision boundary* yang terakhir. Sesuai dengan gambar 2.7, dibuat *decision boundary* dari kelas C dengan kelas sisanya (kelas A dan kelas B). kelas C dilabelkan sebagai positif, sedangkan kelas sisanya dilabelkan menjadi negative, sehingga D3 merupakan *decision boundary* yang dihasilkan dari kelas C dan kelas sisanya (kelas A dan kelas B).

A

A

A

A

B

B

B

D2

D1

B

D3

C

C

C

C

Gambar II.21 *Decision boundary D3* pada pendekatan OVA

**II.6.2.2 *One Vs One* (OVO)**

Misalkan permasalahan yang ditemui *N* kelas. Sehingga akan dibuat *N(N-1)/2 Decision Boundary, decision boundary* yang dihasilkan merupakan hasil pencarian *hyperplane* dari setiap kelas yang lainnya. Berikut dilakukan simulasi dengan menggunakan tiga kelas.

C

C

C

B

B

B

A

A

B

A

A

C

Gambar II.22 contoh pendekatan OVO

Pada Gambar 2.8 terdapat tiga kelas, yaitu kelas A, elas B dan kelas C. Karena menggunakan metode OVO dibuat tiga *decision boundary* dari tiga kelas tersebut. Setiap kelas dicari  *decision boundary* dengan masing-masing kelas lainnya.

D1

B

B

A

A

B

A

A

B

Gambar II.22 *Decision Boundary D1* pada pendekatan OVO

Sesuai dengan Gambar 2.9, dibuat *decision boundary* dari kelas A dengan kelas B. kelas A dilabelkan sebagai positif, sedangkan kelas B dilabelkan menjadi negative, sehingga *D1*merupakan *decision boundary* yang dihasilkan kelas A dan elas B.

B

B

B

B

D2

C

C

C

C

Gambar II.12 *Decision Boundary D2* pada pendekatan OVO

Lalu selanjutnya, sesuai Gambar 2.10, dibuat *decision boundary* dari kelas B dengan kelas C. Kelas B dilabelkan sebagai positif, sedangkan kelas C dilabelkan menjadi negative, sehingga *D2* merupakan *decision boundary* yang dihasilkan dari kelas B dan kelas C.

A

A

C

C

C

A

A

D3

Gambar II.23 *Decision Boundary D3* pada pendekatan OVO

Lalu yang terakhir sesuai dengan Gambar 2.11, dibuat *decision boundary* dari kelas A dengan kelas C. Kleas A dilabelkan sebagai positif, sedangkan kelas C dilabelkan menjadi negative *D3*merupakan *decision boundary* yang dihasilkan dari kelas A dan kelas C.

A

D1

B

B

A

A

B

B

A

D3

C

C

D3

C

C

Gambar II.24 Hasil pendekatan OVO

Sehingga metode OVO menghasilkan area seperti Gambar 2.12, yang dibentuk oleh *decision boundary D1, D2 dan D3.*

**II.7 Neural Network**

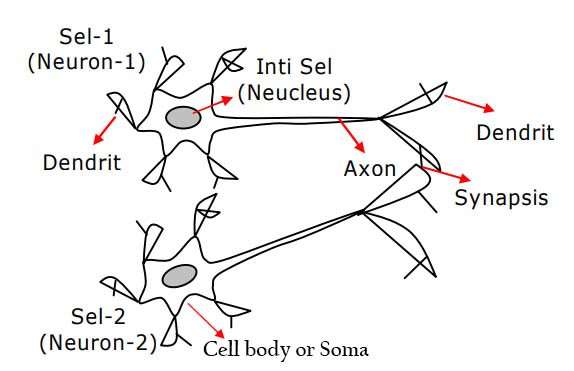
Jaringan Syaraf Bilogis

Pembuatan jaringan syaraf tiruan dibuat berdasarkan struktur jaringan syaraf bilogis otak manusia. Struktur otak manusia sangat kompleks, yakni terdiri dari sel – sel syaraf (neuron) dan penghubung (sinapsis). Cara kerja neuron adalah meneruskan impuls/sinyal yang diberikan padanya ke neuron lain.

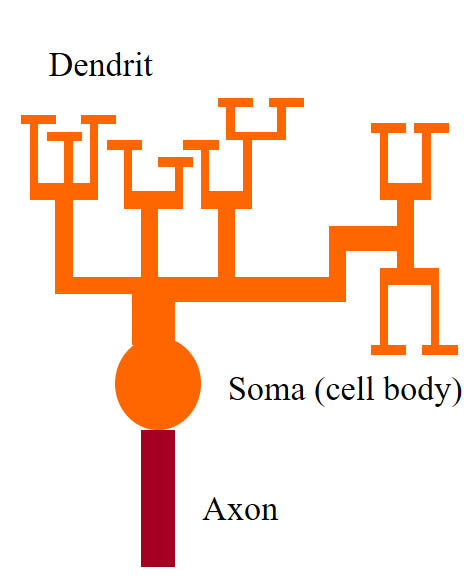
Otak manusia memiliki kemampuan diantaranya :

1. Memproses informasi
2. Melakukan perhitungan matematika
3. Belajar dan beradaptasi, seperti mengenali pola dan mengenali wajah
4. Mengontrol organ-organ tubuh

Otak manusia melakukan kemampuan – kemampuan tersebut dengan kecepatan yang lebih tinggi jika dibandingkan dengan komputer digital. Otak manusia akan bisa mengenali wajah seseorang yang sedikit berubah seperti saat seseorang mengenakan topi atau kumis.

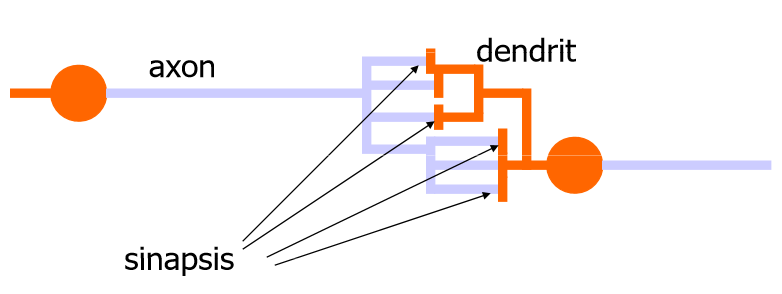


Susunan syaraf manusia



Komponen Penting Neuron

1. Dendrit adalah adalah serat yang berasal dari sel tubuh yang berfungsi menerima aktivasi dari neuron lain.
2. Soma adalah badan dasar struktur sel saraf yang berfungsi untuk tempat pengolahan informasi, seperti menjumlahkan semua sinyal yang masuk. Jika hasil penjumlahan sinyal cukup kuat atau melebihi  *threshold*, sinyal akan diteruskan ke sel lain melalui Axon.
3. Axon berfungsi jalur transmisi yang mengirimkan impuls – impuls atau sinyal ke neuron lain



Koneksi antar neuron

Titik pertemuan antara akson dan dendrit disebut dengan sinapsis. Pada sinapsis terjadi pengiriman sinyal atau informasi. Sinapsis juga berfungsi untuk memperkuat atau memperlemah sinyal yang akan dikirimkan.

**Jaringan Syaraf Tiruan**

Jaringan syaraf tiruan (Artificial neural network) merupakan suatu sistem komputasi yang struktur jaringannya meniru sistem syaraf manusia agar dapat menghasilkan respon dan prilaku seperti dengan jaringan syaraf biologis.

Sistem pemrosesan informasi pada jaringan syaraf tiruan memiliki ciri – ciri yang mirip jaringan syaraf bilogi.

Cara kerja sederhana Jaringan Syaraf Tiruan dibandingkan dengan Jaringan syaraf biologi :

1. Pemrosesan sinyal atau informasi terjadi pada neuron
2. Sinyal dikirimkan diantara neuron melalui suatu penghubung, yakni dendrit dan akson
3. Penghubung antar neuron memiliki bobot yang akan memperkuat atau memperlemah sinyal
4. Setiap neuron memiliki fungsi aktivasi yang berfungsi untuk menentukan keluaran suatu neuron, apakah sinyal akan diteruskan ke neuron lain atau tidak.

**Model Neuron**

x1

x2

Y

w1

w2

w3

x3

X4

w4

z

**Neuron** adalah unit pemproses informasi utama dari jaringan syaraf tiruan yang bekerja berdasarkan impuls/sinyal yang diterimanya dan diteruskan ke neuron lain.

Neuron terdiri dari 3 elemen utama, yaitu :

1. Kummpulan unit-unit yang dihubungkan dengan jalur hubungan
2. Unit penjumlah yang menjumlahkan sinyal masukan yang sudah dikalikan dengan bobotnya.
3. Fungsi aktivasi yang digunakan untuk menentukan keluaran dari suatu neuron, yakni menentukan apakah sinyal dari masukan neuron akan diteruskan ke neuron lain atau tidak.

net=(x1w1+x2w2+x3w3+x4w4)

y =f(net)=f(x1w1+x2w2+x3w3+x4w4)

x1

x2

Y

w1

w2

w3

x3

X4

w4

Keterangan :

1. Y menerima masukan dari neuron x1, x2, dan x3.
2. Neuron – neuron tersebut akan melakukan transformasi pada informasi yang diterima melalui sambungan keluarnya menuju neuron-neuron lain. Pada jaringan syaraf tiruan hubungan ini disebut dengan bobot. Setiap neuron memiliki bobot yang ditandai dengan w1, w2 dan w3. Informasi tersebut disimpan pada suatu nilai tertentu pada bobot tersebut.
3. Informasi masukan akan dikirim ke neuron dengan nilai bobot tertentu. Masukan akan di proses oleh suatu fungsi yang menjumlahkan nilai – nilai semua bobot yang datang. Pada gambar ketiga impuls neuron dijumlahkan : net=(x1w1+x2w2+x3w3),
4. Hasil penjumlahan akan dibandingkan dengan suatu nilai ambang (*threshold*) melalui fungsi aktivasi setiap neuron. Besarnya impuls yang diterima Y mengikuti fungsi aktivasi y=f(net)
5. Jika nilai fungsi aktivasi cukup kuat, yakni melewati nilai ambang tertentu, sinyal akan diteruskan ke neuron lain.

<https://zazha.wordpress.com/2009/10/20/jaringan-syaraf-tiruan/>

Arsitektur jaringan

Pola hubungan antar neuron pada JST terbagi menjadi 2 yakni jaringan layer tunggal dan jaringan layer jamak

1. Jaringan layer tunggal (single layer network)

w11

x1

y1

wj1

xi

wm1

yj

wp1

ym

xk

Wm

wpk

wpn

yp

xn

Jaringan layer tunggal memiliki satu layer bobot terhubung. Pada layer ini sekumpulan neuron masukkan dihubungkan langsung dengan sekumpulan neuron keluaran. Dimana unit masukan merupakan unit yang menerima sinyal sedangkan unit output adalah unit dimana respon dari jaringan terlihat. Arsitektur jaringan tersusun dari n unit input dan m unit output. Semua unit input terhubung ke semua unit output dengan bobot berbeda. bobot wp1 menyatakan bobot antara unit 1 pada input dengan unit p pada output.

1. Jaringan layer jamak (multi layer network)

x1

w11

w11

y1

z1

xi

zs

yj

ym

xk

zt

wpt

wtn

yp

xn

Jaringan layer jamak memiliki satu atau lebih diantara lapisan masukan dan keluaran, lapisan ini disebut lapisan tersembunyi (hidden layer). Jaringan layer jamak memiliki kelebihan dapay menyelesaikan masalah yang lebih kompleks jika dibandingkan dengan jaringan layer tunggal. Namun memeiliki proses pelatihan yang lebih lama dan lebih sulit.

<https://hamdanalwafi.wordpress.com/jaringan-syaraf-tiruan-jst/>

http://okkyibrohim.com/index.php/2016/10/19/arsitektur-jaringan-syaraf-tiruan/

**Training**

Training adalah proses pembelajaran yang dilakukan dengan cara mencari nilai dari bobot – bobot yang menjadi penghubung seluruh neuron, sehingga data input dapat menghasilkan keluaran yang diinginkan. Training dilakukan dengan cara berulang – ulang hingga mendapatkan bobot yang paling optimal.

Training dapat dibagi menjadi dua kategori

1. Training dengan Supervisi
2. Training tanpa supervise

**Supervisied Learning**

Supervised learning adalah metode pembelajaran yang sudah terdapat data latih sebagai masukan dan terdapat target keluaran. Data masukan dan target digunakan untuk melatih jaringan sehingga didapatkan bobot yang diinginkan. Sehingga jaringan dapat melakukan pemetaan dari input ke output sesuai yang diinginkan.

Tahap – tahap yang dilakukan setiap pelatihan pada Supervised Learning :

1. Suatu masukan diberikan ke jaringan
2. Jaringan memproses dan mengeluarkan keluaran
3. Selisih antara keluaran jaringan dan target adalah kesalahan yang terjadi, disebut dengan loss. Semakin kecil nilai dari loss maka proses training semakin bagus
4. Jaringan memodifikasi bobot sesuai dengan loss tersebut, ulangi proses di atas

**Unsupervised Learning**

Pada unsupervised learning tidak ada data target yang mengarahkan proses pelatihan. Perubahan bobot jaringan dilakukan berdasarkan parameter tertentu. Jaringan dimodifikasi menurut ukuran parameter tersebut

**Fungsi aktivasi**

Fungsi aktivasi digunakan untuk menentukan keluaran suatu neuron.

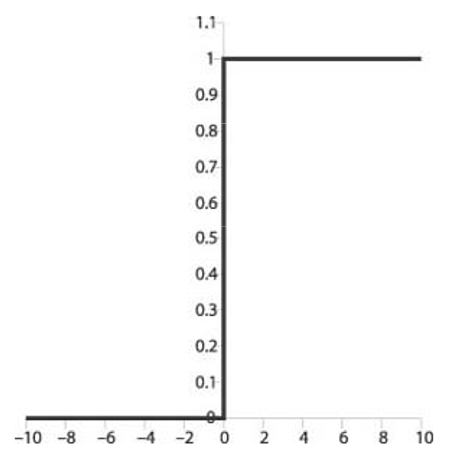
Jika net Σ xiwi, maka fungsi aktivasinya adalah *f*(net) = *f*(Σ xiwi)

Fungsi aktivasi step/threshold :

* Persamaan fungsi aktivasi step :

*F*(x) =

* Digunakan pada awal pengembangan JST
* Tidak Dapat menyelesaikan masalah yang tidak liniesr

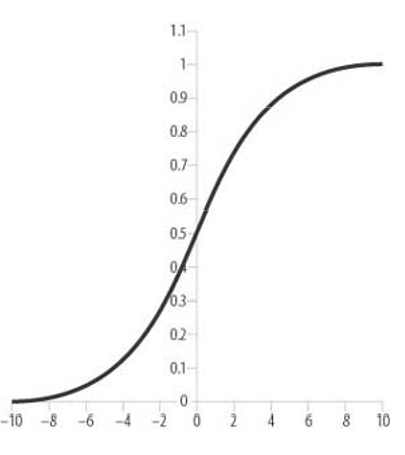


Fungsi Sigmoid

* Persamaan fungsi aktivasi sigmoid :

*F*(x) =

* Cocok untuk menyelesaikan masalah yang tidak linier

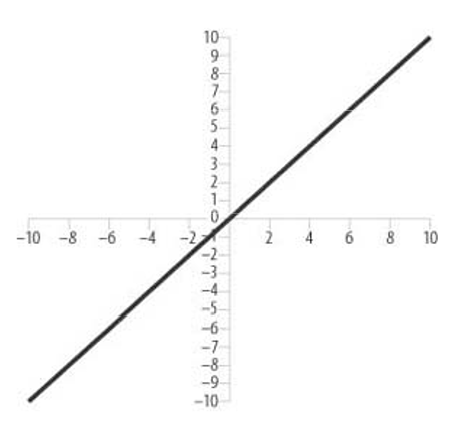


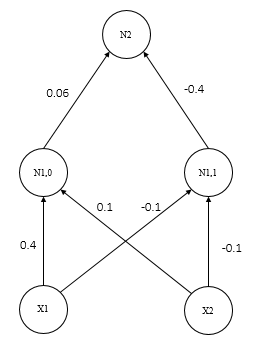
**Fungsi Identitas**

* Persamaan fungsi aktivasi identitas :

*F*(x) = x

* Seringkali digunakan pada neuron keluaran





δk **←** ok (1 – ok) (tk – ok)

untuk setiap unit hidden h

δh **←** oh (1 – oh)

Update setiap jaringan bobot wi,j

wi,j **←** wi,j + Δ wi,j

dimana Δ wi,j = *nδjxi*