

# TEMU KEMBALI CITRA BERBASIS ISI BERBASIS *FEATURE* *LEARNING* DENGAN MENGGUNAKAN *SELF SUPERVISED* *CONVOLUTIONAL AUTENCODER*

Wahyu Akbar Wibowo  
(16.04.1.1.1.00112)

Pembimbing I : Dr. Indah Agustien S., S.Kom., M.Kom.  
Pembimbing II : Moch. Kautsar Sophan, S.Kom., M.MT.

## ABSTRAK

Temu Kembali Citra berbasis Isi (TKCBI) merupakan proses mendapatkan beberapa citra berdasarkan *content* dan karakteristik dari sebuah citra kueri yang dimasukan. Pada TKCBI terdapat proses ekstraksi fitur dari citra kueri untuk mendapatkan fitur yang merepresentasikan karakteristik dari citra. Pada umumnya, TKCBI menggunakan *handcrafted feature* seperti warna, bentuk, tekstur citra untuk proses ekstraksi fitur. Hal ini memerlukan waktu tersendiri untuk menentukan *handcrafted feature* yang sesuai dengan citra di basis data. Pada penelitian ini diusulkan metode *feature learning* untuk mengekstraksi fitur dari citra kueri maupun citra basis data menggunakan *Self Supervised Convolutional Autoencoder*. Penggunaan *feature learning* tidak memerlukan penentuan fitur buatan tangan karena *feature learning* mempelajari secara langsung fitur-fitur penting dari citra basis data. Sedangkan, *Self Supervised Convolutional Autoencoder* digunakan untuk meningkatkan kinerja dari *feature learning* dalam proses pelatihan. Pada *Convolutional Autoencoder* terdapat dua *layer*, yaitu *layer Encoder* dan *layer Decoder*. *Layer Encoder* digunakan untuk mengurangi dimensi citra dan menentukan fitur-fitur yang paling penting dalam citra masukan. *Layer Decoder* digunakan untuk merekonstruksi fitur-fitur yang penting hasil dari *Encoder* menjadi sebuah citra yang mirip citra masukan. Untuk pengukuran kemiripan citra secara umum menggunakan metode *Ecludian Distance*. Uji coba dilakukan untuk membandingkan akurasi TKCBI menggunakan *feature learning* dan TKCBI tanpa *feature learning*. Uji coba dilakukan pada dataset *GHIM-10K Dataset* dari *Guangxi Normal University, China*. Dari hasil uji coba yang telah dilakukan TKCBI dengan *feature learning* untuk dua ratus citra kueri dari dua puluh kelas menggunakan *Self Supervised Convolutional Autoencoder* mendapatkan nilai *Average Precision* yaitu 39,6%. Jumlah fitur yang diekstraksi adalah 64 fitur. Sedangkan, TKCBI tanpa menggunakan *feature learning* dengan jumlah citra kueri yang sama namun jumlah fiturnya adalah 64x64x3 fitur mendapatkan nilai *Average Precision* yaitu 41,2%. Namun, waktu *retrieve* yang dibutuhkan untuk mendapatkan citra dari penggunaan *feature learning* lebih cepat yaitu 0,2 detik daripada waktu komputasi dari TKCBI tanpa *feature learning*.

**Kata Kunci:** *Feature Learning, Self Supervised Convolutional Autoencoder, Temu Kembali Citra berbasis Isi, Euclidean Distance*

# CONTENT BASED IMAGE RETRIEVAL BASED ON FEATURE LEARNING USING *SELF SUPERVISED CONVOLUTIONAL AUTENCODER*

Wahyu Akbar Wibowo  
(16.04.1.1.1.00112)

Pembimbing I : Dr. Indah Agustien S., S.Kom., M.Kom.  
Pembimbing II : Moch. Kautsar Sophan, S.Kom., M.MT.

## ***ABSTRACT***

*Content-based Image Retrieval (CBIR) is the process of obtaining multiple images based on the content and characteristics of an entered query image. In CBIR there is a feature extraction process from the query image to get features that represent the characteristics of the image. In general, CBIR uses handcrafted features such as color, shape, image texture for the feature extraction process. This requires a separate time to determine the handcrafted features that match the image in the database. In this research, a feature learning method is proposed to extract features from query images and database images using Self Supervised Convolutional Autoencoder. The use of feature learning does not require the determination of handmade features because feature learning directly learns important features of database images. Meanwhile, the Self Supervised Convolutional Autoencoder is used to improve the performance of feature learning in the training process. In Convolutional Autoencoder there are two layers, the Encoder layer and the Decoder layer. The Layer Encoder is used to reduce the dimensions of the image and determine the most important features in the input image. The Layer Decoder is used to reconstruct important features resulting from the Encoder into an image that is similar to the input image. For general image similarity measurements using the Ecludian Distance method. Experiment was conducted to compare the accuracy of CBIR using feature learning and CBIR without feature learning. The trial was conducted on the dataset GHIM-10K Dataset from Guangxi Normal University, China. From the results of experiments that have been conducted by CBIR with feature learning for two hundred query images from twenty classes using Self Supervised Convolutional Autoencoder, the average precision is 39.6%. The number of features extracted is 64 features. Meanwhile, CBIR without using feature learning with the same number of query images, but the number of features is 64x64x3 features get an Average Precision of 41.2%. However, the retrieve time needed to get an image from using feature learning is faster, that is, 0.2 seconds than the computing time of CBIR without feature learning.*

**Keywords:** *Feature Learning, Self Supervised Convolutional Autoencoder, Content Based Image Retrieval, Euclidean Distance*

## DAFTAR ISI

|                                                 |      |
|-------------------------------------------------|------|
| HALAMAN JUDUL .....                             | ii   |
| HALAMAN PENGESAHAN .....                        | iii  |
| HALAMAN PERNYATAAN ORISINALITAS .....           | iv   |
| ABSTRAK .....                                   | v    |
| ABSTRACT .....                                  | vi   |
| KATA PENGANTAR .....                            | vii  |
| DAFTAR ISI .....                                | viii |
| DAFTAR GAMBAR .....                             | xi   |
| DAFTAR TABEL .....                              | xv   |
| BAB I PENDAHULUAN .....                         | 1    |
| 1.1. Latar Belakang .....                       | 1    |
| 1.2. Perumusan Masalah .....                    | 3    |
| 1.2.1. Permasalahan .....                       | 3    |
| 1.2.2. Usulan Solusi .....                      | 3    |
| 1.2.3. Pertanyaan Penelitian .....              | 4    |
| 1.3. Tujuan & Manfaat .....                     | 4    |
| 1.3.1. Tujuan Penelitian .....                  | 4    |
| 1.3.2. Manfaat .....                            | 4    |
| 1.4. Batasan-batasan .....                      | 4    |
| 1.5. Sistematika Penulisan .....                | 5    |
| BAB II TINJAUAN PUSTAKA .....                   | 6    |
| 2.1. Temu Kembali Citra berbasis Isi .....      | 6    |
| 2.2. <i>Convolutional</i> Neural Network .....  | 7    |
| 2.2.1. Lapisan Konvolusi .....                  | 9    |
| 2.2.2. Lapisan <i>Pooling</i> .....             | 11   |
| 2.2.3. Fungsi Aktivasi ( <i>Neurons</i> ) ..... | 12   |
| 2.3. <i>Convolutional Autoencoder</i> .....     | 13   |

|                                                               |    |
|---------------------------------------------------------------|----|
| 2.4. <i>Euclidian Distance</i> .....                          | 16 |
| 2.5. <i>Precision dan Recall</i> .....                        | 17 |
| 2.6. Penelitian Terkait.....                                  | 18 |
| BAB III METODOLOGI PENELITIAN .....                           | 22 |
| 3.1. Prosedur Penelitian .....                                | 22 |
| 3.2. <i>Dataset</i> .....                                     | 23 |
| 3.3. Arsitektur Sistem .....                                  | 24 |
| 3.3.1. Rancangan Arsitektur Sistem .....                      | 24 |
| 3.3.2. Algoritma Sistem.....                                  | 34 |
| 3.4. Pengujian dan Evaluasi .....                             | 35 |
| BAB IV HASIL DAN PEMBAHASAN .....                             | 38 |
| 4.1. Lingkungan Uji Coba.....                                 | 38 |
| 4.2. Tahapan Pembuatan Program.....                           | 39 |
| 4.2.1. Praproses Dataset Citra.....                           | 39 |
| 4.2.2. Pembuatan Model <i>Convolutional Autoencoder</i> ..... | 40 |
| 4.2.3. Perhitungan Jarak Fitur.....                           | 49 |
| 4.2.4. Perhitungan <i>Precision dan Recall</i> .....          | 50 |
| 4.2.5. Tampilan Sistem.....                                   | 52 |
| 4.3. Skenario Uji Coba.....                                   | 52 |
| 4.4. Hasil Uji Coba .....                                     | 57 |
| 4.4.1. Skenario Uji Coba Kelas Kembang Api .....              | 57 |
| 4.4.2. Skenario Uji Coba Kelas Bangunan .....                 | 60 |
| 4.4.3. Skenario Uji Coba Kelas Tembok Cina.....               | 62 |
| 4.4.4. Skenario Uji Coba Kelas Mobil .....                    | 64 |
| 4.4.5. Skenario Uji Coba Kelas Capung.....                    | 67 |
| 4.4.6. Skenario Uji Coba Kelas Gunung .....                   | 69 |

|                                                            |     |
|------------------------------------------------------------|-----|
| 4.4.7. Skenario Uji Coba Kelas Bunga.....                  | 71  |
| 4.4.8. Skenario Uji Coba Kelas Pohon.....                  | 74  |
| 4.4.9. Skenario Uji Coba Kelas Padang Rumput.....          | 76  |
| 4.4.10. Skenario Uji Coba Kelas Pantai dan Teluk.....      | 78  |
| 4.4.11. Skenario Uji Coba Kelas Pesawat .....              | 81  |
| 4.4.12. Skenario Uji Coba Kelas Kupu-Kupu .....            | 83  |
| 4.4.13. Skenario Uji Coba Kelas Bangunan Cina.....         | 85  |
| 4.4.14. Skenario Uji Coba Kelas Matahari Terbenam .....    | 88  |
| 4.4.15. Skenario Uji Coba Kelas Sepeda Motor .....         | 90  |
| 4.4.16. Skenario Uji Coba Kelas Perahu Layar .....         | 92  |
| 4.4.17. Skenario Uji Coba Kelas Kapal Pesiar .....         | 94  |
| 4.4.18. Skenario Uji Coba Kelas Ayam .....                 | 97  |
| 4.4.19. Skenario Uji Coba Kelas Serangga .....             | 99  |
| 4.4.20. Skenario Uji Coba Kelas Kuda .....                 | 101 |
| 4.5. Analisa Hasil Uji Coba.....                           | 104 |
| 4.6. Hasil Uji Coba <i>Convolutional Autoencoder</i> ..... | 114 |
| 4.7. Hasil Uji Coba <i>Dataset Lain</i> .....              | 116 |
| BAB V PENUTUP .....                                        | 119 |
| 5.1. Kesimpulan.....                                       | 119 |
| 5.2. Saran.....                                            | 119 |
| DAFTAR PUSTAKA .....                                       | 121 |
| LAMPIRAN .....                                             | 124 |

## DAFTAR GAMBAR

|                                                                                                  |    |
|--------------------------------------------------------------------------------------------------|----|
| Gambar 2.1 Sistem TKCBI Umum [15] .....                                                          | 7  |
| Gambar 2.2 Arsitektur Dasar CNN .....                                                            | 8  |
| Gambar 2.3 Ilustrasi operasi konvolusi .....                                                     | 10 |
| Gambar 2.4 Ilustrasi operasi <i>max pooling</i> .....                                            | 12 |
| Gambar 2.5 Fungsi ReLU .....                                                                     | 13 |
| Gambar 2.6 Proses Aktivasi ReLU.....                                                             | 13 |
| Gambar 2.7 Arsitektur <i>Convolutional Autoencoder</i> .....                                     | 14 |
| Gambar 2.8. Input dan Output dari Proses Lapisan <i>Encoder</i> .....                            | 15 |
| Gambar 2.9 Operasi <i>upsampling</i> menggunakan <i>nearest neighbor algorithm</i> .....         | 16 |
| Gambar 2.10. Input dan Output dari Proses Lapisan <i>Decoder</i> .....                           | 16 |
| Gambar 3.1 Prosedur Penelitian .....                                                             | 22 |
| Gambar 3.2 Sampel <i>GHIM-10K Dataset</i> .....                                                  | 23 |
| Gambar 3.3 Arsitektur Rancangan Sistem TKCBI menggunakan <i>feature learning</i><br>.....        | 24 |
| Gambar 3.4. Arsitektur Rancangan Sistem TKCBI tanpa menggunakan <i>feature learning</i><br>..... | 25 |
| Gambar 3.5 Model <i>Convolutional Autoencoder</i> .....                                          | 27 |
| Gambar 3.6 Contoh Proses Pengukuran Kemiripan.....                                               | 33 |
| Gambar 3.7 <i>Flowchart</i> TKCBI dengan <i>Feature Learning</i> .....                           | 35 |
| Gambar 3.8 Pengujian dan Evaluasi Sistem.....                                                    | 36 |
| Gambar 4.1. Contoh Pengubahan Ukuran Citra.....                                                  | 40 |
| Gambar 4.2. Tampilan Sistem.....                                                                 | 52 |
| Gambar 4.3. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Kembang Api ...               | 58 |
| Gambar 4.4. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Kembang Api.....                           | 59 |
| Gambar 4.5. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Kembang Api.....                           | 59 |
| Gambar 4.6. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Bangunan .....                | 60 |
| Gambar 4.7. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Bangunan .....                             | 61 |
| Gambar 4.8. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Bangunan.....                              | 62 |
| Gambar 4.9. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Tembok Cina ....              | 63 |
| Gambar 4.10. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Tembok Cina .....                         | 63 |

|                                                                                            |    |
|--------------------------------------------------------------------------------------------|----|
| Gambar 4.11. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Tembok Cina .....                   | 64 |
| Gambar 4.12. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Mobil.....             | 65 |
| Gambar 4.13. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Mobil.....                          | 66 |
| Gambar 4.14. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Mobil.....                          | 66 |
| Gambar 4.15. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Capung .....           | 67 |
| Gambar 4.16. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Capung .....                        | 68 |
| Gambar 4.17. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Capung .....                        | 69 |
| Gambar 4.18. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Gunung.....            | 70 |
| Gambar 4.19. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Gunung.....                         | 70 |
| Gambar 4.20. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Gunung.....                         | 71 |
| Gambar 4.21. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Bunga.....             | 72 |
| Gambar 4.22. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Bunga .....                         | 73 |
| Gambar 4.23. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Bunga .....                         | 73 |
| Gambar 4.24. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Pohon .....            | 74 |
| Gambar 4.25. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Pohon .....                         | 75 |
| Gambar 4.26. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Pohon.....                          | 76 |
| Gambar 4.27. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Padang Rumput .....    | 77 |
| Gambar 4.28. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Padang Rumput .....                 | 77 |
| Gambar 4.29. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Padang Rumput.....                  | 78 |
| Gambar 4.30. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Pantai dan Teluk ..... | 79 |
| Gambar 4.31. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Pantai dan Teluk .....              | 80 |
| Gambar 4.32. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Pantai dan Teluk .....              | 80 |
| Gambar 4.33. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Pesawat .....          | 81 |
| Gambar 4.34. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Pesawat.....                        | 82 |
| Gambar 4.35. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Pesawat.....                        | 83 |
| Gambar 4.36. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Kupu-Kupu.....         | 84 |
| Gambar 4.37. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Kupu-Kupu .....                     | 84 |
| Gambar 4.38. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Kupu-Kupu .....                     | 85 |
| Gambar 4.39. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Bangunan Cina.....     | 86 |
| Gambar 4.40. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Bangunan Cina.....                  | 87 |

|                                                                                                                                                                                           |     |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| Gambar 4.41. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Bangunan Cina.....                                                                                                                 | 87  |
| Gambar 4.42. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Matahari Terbenam.....                                                                                                | 88  |
| Gambar 4.43. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Matahari Terbenam.....                                                                                                             | 89  |
| Gambar 4.44. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Matahari Terbenam.....                                                                                                             | 90  |
| Gambar 4.45. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Sepeda Motor ..                                                                                                       | 91  |
| Gambar 4.46. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Sepeda Motor.....                                                                                                                  | 91  |
| Gambar 4.47. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Sepeda Motor.....                                                                                                                  | 92  |
| Gambar 4.48. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Perahu Layar ..                                                                                                       | 93  |
| Gambar 4.49. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Perahu Layar.....                                                                                                                  | 93  |
| Gambar 4.50. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Perahu Layar.....                                                                                                                  | 94  |
| Gambar 4.51. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Kapal Pesiar ...                                                                                                      | 95  |
| Gambar 4.52. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Kapal Pesiar.....                                                                                                                  | 96  |
| Gambar 4.53. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Kapal Pesiar.....                                                                                                                  | 96  |
| Gambar 4.54. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Ayam.....                                                                                                             | 97  |
| Gambar 4.55. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Ayam.....                                                                                                                          | 98  |
| Gambar 4.56. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Ayam.....                                                                                                                          | 99  |
| Gambar 4.57. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Serangga.....                                                                                                         | 100 |
| Gambar 4.58. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Serangga.....                                                                                                                      | 100 |
| Gambar 4.59. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Serangga.....                                                                                                                      | 101 |
| Gambar 4.60. Grafik <i>Precision-Recall</i> Skenario Uji Coba Kelas Kuda.....                                                                                                             | 102 |
| Gambar 4.61. Hasil <i>Retrieve</i> Citra Kueri 1 Kelas Kuda.....                                                                                                                          | 103 |
| Gambar 4.62. Hasil <i>Retrieve</i> Citra Kueri 2 Kelas Kuda.....                                                                                                                          | 103 |
| Gambar 4.63. Contoh Hasil <i>Retrieve</i> yang Tidak Bagus .....                                                                                                                          | 112 |
| Gambar 4.64. Contoh Keberagaman Karakteristik Citra pada <i>Database</i> .....                                                                                                            | 112 |
| Gambar 4. 65. Contoh Hasil <i>Retrieve</i> yang Bagus .....                                                                                                                               | 113 |
| Gambar 4.66. Grafik Kinerja <i>Precision-Recall</i> dari TKCBI dengan Model <i>Encoder</i> 1, Model <i>Encoder</i> 2, Model <i>Encoder</i> 3 dan TKCBI tanpa <i>Feature Learning</i> .... | 114 |
| Gambar 4.67. Hasil Ekstraksi Fitur dan Hasil Rekontruksi Fitur dengan <i>Convolutional Autoencoder</i> .....                                                                              | 115 |
| Gambar 4.68. Grafik <i>Precision-Recall</i> Skenario Uji Coba Citra <i>Corel</i> 1 .....                                                                                                  | 116 |
| Gambar 4.69. Hasil <i>Retrieve</i> Citra <i>Corel</i> 1 .....                                                                                                                             | 117 |

|                                                                                                     |     |
|-----------------------------------------------------------------------------------------------------|-----|
| Gambar 4.70. Grafik <i>Precision-Recall</i> Skenario Uji Coba Citra <i>Corel 2</i> .....            | 117 |
| Gambar 4.71. Hasil <i>Retrieve</i> Citra <i>Corel 1</i> .....                                       | 117 |
| Gambar 4.72. Hasil Ekstraksi Fitur dan Hasil Rekontruksi Fitur Beberapa <i>Dataset Corel</i> . .... | 118 |

## DAFTAR TABEL

|                                                                                                                            |     |
|----------------------------------------------------------------------------------------------------------------------------|-----|
| Tabel 2.1 Hyperparameter pada <i>Convolutional Layer</i> .....                                                             | 10  |
| Tabel 2.2 Rangkuman Penelitian Terkait.....                                                                                | 20  |
| Tabel 3.1 Daftar Lapisan pada Model <i>Convolutional Autoencoder 1</i> .....                                               | 29  |
| Tabel 3.2. Daftar pada Lapisan Model <i>Convolutional Autoencoder 2</i> .....                                              | 30  |
| Tabel 3.3. Daftar Lapisan pada Model <i>Convolutional Autoencoder 3</i> .....                                              | 32  |
| Tabel 3.4 Hasil Uji Coba Sistem .....                                                                                      | 37  |
| Tabel 4.1. Spesifikasi Perangkat Keras.....                                                                                | 38  |
| Tabel 4.2. Spesifikasi Kebutuhan Perangkat Lunak.....                                                                      | 38  |
| Tabel 4.3. Skenario Uji Coba .....                                                                                         | 52  |
| Tabel 4.4. Citra Kueri .....                                                                                               | 54  |
| Tabel 4.5. Hasil Rata-Rata Kombinasi <i>Precision-Recall</i> Model <i>Encoder 1</i> pada Semua Skenario .....              | 105 |
| Tabel 4.6. Hasil Rata-Rata Kombinasi <i>Precision-Recall</i> Model <i>Encoder 2</i> pada Semua Skenario .....              | 106 |
| Tabel 4.7. Hasil Rata-Rata Kombinasi <i>Precision-Recall</i> Model <i>Encoder 3</i> pada Semua Skenario .....              | 107 |
| Tabel 4.8. Hasil Rata-Rata Kombinasi <i>Precision-Recall</i> TKCBI Tanpa <i>Feature learning</i> pada Semua Skenario ..... | 108 |
| Tabel 4.9. Rangkuman <i>Precision-Recall</i> Semua Skenario Uji Coba .....                                                 | 110 |

# BAB I PENDAHULUAN

Bab I berisi tentang latar belakang, perumusan masalah, tujuan & manfaat, batasan-batasan dan sistematika penulisan.

## 1.1. Latar Belakang

Semakin pesatnya perkembangan teknologi penyimpanan data digital seperti informasi berupa teks, citra, suara dan video. Berbagai teknik untuk mengolah, mencari, mengambil informasi dari data semakin ditingkatkan. Salah satu teknologi yang ditingkatkan adalah pencarian informasi dari data. Dalam proses untuk mencari informasi atau mendapatkan informasi dari data sangat penting, karena tingkat akurasi pencariannya semakin baik jika semakin cepat dan tepat dalam mendapatkan informasi [1]. Citra merupakan salah satu bentuk data yang sering digunakan untuk mendapatkan informasi. Sistem yang dapat menerapkan pencarian informasi citra adalah Temu Kembali Citra berbasis Isi (TKCBI). Dimana dalam pencarian citra, TKCBI dapat digunakan untuk mengukur kemiripan citra yang terdapat pada *database* dengan citra yang menjadi query [2]. TKCBI bertujuan untuk mencari citra melalui analisis isi visual citra dan demikian representasi citra merupakan inti dari TKCBI [3].

Pada TKCBI tradisional terdapat beberapa hasil pencarian yang tidak sesuai dengan yang diinginkan pengguna dan belum optimal karena dalam pencarian citra menggunakan teks [4]. Umumnya, dalam fitur yang merepresentasikan citra yang digunakan pada TKCBI masih menggunakan *handcrafted feature*, dimana metode yang digunakan dalam ekstraksi fitur bergantung pada bentuk, warna, tekstur citra dan jenis data [5], [6].

Permasalahan ekstraksi fitur, masih merupakan salah satu masalah penting dalam pengenalan pola. Secara umum, representasi fitur bergantung pada jenis data, sehingga sulit untuk menemukan representasi fitur terbaik [5], [7]. Dalam TKCBI yang menggunakan metode histogram warna untuk merepresentasikan citra, citra *database* yang memiliki jumlah atau frekuensi piksel sama dengan citra *query* maka akan ditampilkan [8]. Contoh pencarian citra sapi berwarna putih yang mungkin akan menghasilkan citra sebuah mobil berwarna putih.

Untuk mengatasi beberapa permasalahan tersebut terdapat beberapa metode pada penelitian sebelumnya tentang TKCBI. Pada tahun 2013, Guang-Hai

Liu dan Jing-Yu Yang [8]. Menggunakan metode *color difference histogram* berdasarkan perbedaan warna, orientasi tepi citra dan perbedaan warna yang seragam. Pada tahun 2016, Xiangyuan Zhao dan Brian Nutter [9]. Menggunakan *wavelet transformation* dan *multiple input multiple task deep Autoencoder* untuk TKCBI, dimana citra *query* masuk ke dalam proses *wavelet transform* dan diurai menjadi koefisien *wavelet*. Kemudian koefisien tersebut menjadi masukan untuk *deep Autoencoder*.

Pada metode-metode yang digunakan diatas masih menggunakan *handcrafted feature* atau fitur buatan tangan dalam ekstraksi fitur, sehingga memerlukan waktu tersendiri untuk menentukan fitur mana yang sesuai dengan citra. Pada penelitian ini mengusulkan pembelajaran fitur atau *featured learning* untuk proses ekstraksi fitur, karena pada *featured learning* tidak memerlukan *handcrafted feature*. *Convolutional Neural Network* (CNN) dapat digunakan untuk *featured learning* [6]. CNN dapat mempelajari fitur *dataset* dalam jumlah besar karena memiliki arsitektur yang dalam sehingga dapat meningkatkan performa dalam pengenalan pola [10], [11]. Banyak penelitian yang menggunakan arsitektur CNN, khususnya pengenalan pola, karena keuntungan tersebut. CNN memiliki dua bagian utama, yaitu *featured learning* dan *classification*. Pada TKCBI, CNN dapat diimplementasikan untuk mengekstraksi fitur citra penting melalui kemampuan *featured learning*.

Pada penelitian ini menggunakan *featured learning* dari CNN untuk proses ekstraksi fitur sehingga menghasilkan fitur penting citra dan model arsitektur CNN. Karena dengan CNN, pembelajaran fitur dapat dilakukan secara langsung dengan citra mentah tanpa perlu menentukan *handcrafted feature*. Untuk mengetahui hasil fitur penting citra tersebut merupakan representasi dari citra yang diekstrak, maka dilakukan rekontruksi fitur tersebut menjadi citra semula. [5] Oleh karena itu, penelitian ini menggunakan *Self Supervised Convolutional Autoencoder*. *Convolutional Autoencoder* merupakan pembelajaran *self-supervised* (pengawasan sendiri) dengan menggunakan lapisan yang terdapat pada *Convolutional Neural Network* (CNN), dimana dalam proses pembelajaran, *Convolutional Autoencoder* menyandikan citra masukan dalam satu set fitur sederhana dan kemudian mencoba merekontruksi citra masukan [5]. *Convolutional Autoencoder* mempunyai

kelebihan dalam ekstraksi fitur penting karena terdapat CNN pada lapisan pembelajaran fitur [5]. Disamping itu, *Convolutional Autoencoder* dapat mereduksi dimensi citra masukan dan merekonstruksi fitur tersebut kembali ke ukuran citra masukan [6]. Oleh karena itu, *Convolutional Autoencoder* tidak memerlukan label dalam pembelajaran fitur.

Untuk pengukuran kemiripan citra masukan dengan citra *database* menggunakan *Euclidian Distance* karena umum dan populer digunakan untuk menghitung jarak fitur pada TKCBI [12]. Penggunaan *Euclidean Distance* disarankan untuk dimensi dan nilai pada data yang sama dengan data pembanding, serta data dengan dimensi yang tidak tinggi dan nilai pada data yang tidak bernilai negatif [12]. Dengan menggunakan pembelajaran fitur atau *feature learning* untuk ekstraksi fitur dalam TKCBI dan metode *Self Supervised Convolutional Autoencoder* untuk meningkatkan kinerja pembelajaran fitur, diharapkan meningkatkan akurasi hasil TKCBI sehingga memberikan kepuasan terhadap pengguna dalam mencari citra.

Pengujian sistem pada penelitian ini menggunakan kombinasi *precision* dan *recall*, karena kombinasi *precision* dan *recall* menghasilkan grafik *precision-recall* untuk mengevaluasi kinerja sistem TKCBI [12], [13]. Pada penelitian ini menggunakan *Average Precision* untuk membandingkan kinerja antara TKCBI menggunakan *feature learning* dan TKCBI tanpa menggunakan *feature learning*.

## **1.2. Perumusan Masalah**

Pada perumusan masalah terdapat tiga poin penting yaitu permasalahan, usulan solusi dan pertanyaan penelitian.

### **1.2.1. Permasalahan**

Tahapan ekstraksi fitur sangatlah penting pada TKCBI yang akan dibangun, akan tetapi penggunaan *handcrafted feature* untuk penentuan fitur penting, membutuhkan waktu tersendiri untuk penentuan fitur penting apa saja yang merepresentasikan dataset dan metode ekstraksi fitur yang tepat sesuai dengan fitur penting yang telah ditentukan.

### **1.2.2. Usulan Solusi**

Pada penelitian ini diusulkan metode pembelajaran fitur atau *feature learning* untuk mengekstraksi fitur dari citra. *Self Supervised Convolutional*

*Autoencoder* untuk meningkatkan kinerja *feature learning* dan mengetahui hasil rekontruksi fitur dari *feature learning* menjadi citra semula.

### **1.2.3. Pertanyaan Penelitian**

Berdasarkan latar belakang dan rumusan masalah dapat dihasilkan pertanyaan penelitian yaitu bagaimana kinerja dari TKCBI dengan menggunakan pembelajaran fitur atau *feature learning* untuk ekstraksi fitur dan *Self Supervised Convolutional Autoencoder* untuk meningkatkan kinerja *feature learning* dibandingkan TKCBI tanpa menggunakan *feature learning*?

Kinerja pada penelitian ini merupakan kemampuan kerja dari sistem TKCBI yang dibuat. Kinerja tersebut akan diukur menggunakan kombinasi *precision* dan *recall*, serta untuk membandingkan antara TKCBI dengan *feature learning* dan TKCBI tanpa *feature learning* menggunakan *Average Precision*.

## **1.3. Tujuan & Manfaat**

### **1.3.1. Tujuan Penelitian**

Mengetahui kinerja atau kemampuan kerja dari TKCBI dengan menggunakan *feature learning* untuk ekstraksi fitur. Mengetahui hasil citra dari TKCBI yang menggunakan *feature learning* sebagai proses ekstraksi fitur. Penerapan metode *Self Supervised Convolutional Autoencoder* bertujuan untuk memberikan hasil yang optimal terhadap proses pencarian pada TKCBI.

### **1.3.2. Manfaat**

Manfaat yang diharapkan dari penelitian ini adalah mempercepat hasil pencarian dan memberikan hasil optimal terhadap proses pencarian pada TKCBI. Dan fitur citra buatan tangan / *handcrafted feature* tidak diperlukan lagi dalam proses pencarian pada TKCBI menggunakan metode *Convolutional Autoencoder*.

## **1.4. Batasan-batasan**

Batasan-batasan masalah pada penelitian ini adalah:

1. *Dataset* yang dikumpulkan berupa citra yang berasal dari *GHIM-10k Dataset*. Pada dataset *GHIM-10k Dataset* terdapat 10.000 citra beragam seperti diantaranya Kembang Api, Pesawat, Bangunan Cina, dan lain-lain. Pada dataset tersebut terdapat 20 kategori dan setiap kategori terdiri dari 500 citra dengan ukuran 400x300 dan 300x400, sehingga total semua citra yang digunakan adalah 10.000 citra. Semua citra dari dataset ini berasal

dari web dan kamera, dan dikumpulkan oleh peneliti dari *Guangxi Normal University, China* yaitu *Guang-Hai Liu* [14]. *Dataset* tersebut dapat diunduh pada alamat url <http://www.ci.gxnu.edu.cn/cbir/Dataset.aspx>.

2. Pengukuran kemiripan citra hasil *Autoencoder* dengan citra hasil *Decoder* menggunakan *Euclidean Distance*.
3. Pengukuran akurasi hasil citra TKCBI menggunakan *Precision* dan *Recall*.

### **1.5. Sistematika Penulisan**

Adapun sistematika penulisan penulisan adalah sebagai berikut:

1. Bab I Pendahuluan  
Pada bab I ini berisikan latar belakang penelitian ini, perumusan masalah, pertanyaan penelitian, tujuan & manfaat penelitian, batasan-batasan masalah, dan sistematika penulisan.
2. Bab II Tinjauan Pustaka  
Pada bab II ini berisikan penjelasan dasar teori yang mendasari penelitian ini. Teori yang dibahas yaitu, TKCBI, *Convolutional Autoencoder*, *Euclidean Distance*, *Precision* dan *Recall*.
3. Bab III Metodologi Penelitian  
Pada bab III ini berisikan penjelasan tahapan-tahapan penelitian, arsitektur sistem, pembuatan model *Convolutional Autoencoder*, cara pengujian sistem dan perkiraan jadwal pelaksanaan penelitian.
4. Bab IV Hasil dan Pembahasan  
Pada bab IV ini berisikan implementasi metode dari rancangan yang telah dibuat pada bab III, sistem diuji sesuai dengan skenario pengujian pada bab IV dan penjelasan hasil penelitian.
5. Bab V Penutup  
Pada bab V ini terdapat kesimpulan dari hasil uji coba dan saran untuk pengembangan penelitian

## BAB II TINJAUAN PUSTAKA

Bab II berisi tentang dasar teori yang menjadi penunjang pada penelitian ini yaitu TKCBI, *Convolutional Neural Network*, *Convolutional Autoencoder*, *Euclidean Distance*, *Precision & Recall* dan Penelitian Terkait.

### 2.1. Temu Kembali Citra berbasis Isi

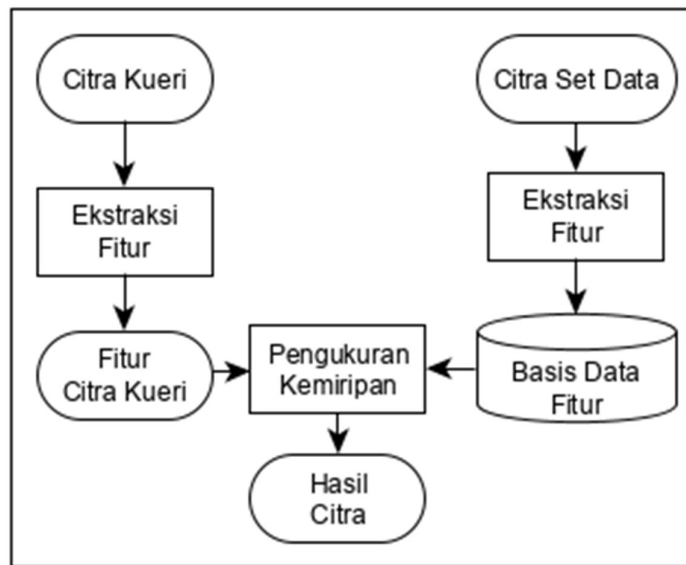
TKCBI merupakan teknik pencarian kembali *image* yang mempunyai kemiripan karakteristik atau *content* dari sekumpulan citra. Proses secara umum dari TKCBI pada citra *query* dilakukan proses ekstraksi fitur. Parameter fitur citra yang dapat digunakan untuk *retrieval* pada sistem ini seperti *histogram*, susunan warna, tekstur, bentuk, tipe spesifik dari obyek, tipe *event* tertentu, nama, lokasi dan emosi [15]. Citra *query* terdiri atas 3 level, yaitu:

- a. Level 1: *retrieval* dengan *primitive feature*, seperti warna, bentuk dan tekstur.
- b. Level 2: *retrieval* dengan *logical feature*, seperti tipe obyek, individu obyek atau orang.
- c. Level 3: *retrieval* dengan abstrak *feature*, seperti nama even, tipe aktifitas, *emotional*, religius.

Penelitian dan pembangunan dalam TKCBI mencakup keseluruhan topik. Beberapa pokok persoalan yang terjadi dalam pembangunan TKCBI dapat diuraikan sebagai berikut:

- a. Pemahaman citra yang dibutuhkan oleh user dan pencarian informasi.
- b. Pengidentifikasian cara yang sesuai dalam konten citra atau karakteristik dari citra.
- c. Proses ekstraksi fitur dari citra.

Ekstraksi fitur merupakan proses penting pada sistem TKCBI karena hasil dari proses ini perbedaan pada setiap citra dapat diketahui berdasarkan cirinya seperti ciri bentuk, warna, tekstur dan lain-lain. Beberapa teknik yang dapat digunakan untuk ekstraksi fitur antara lain: teknik analisis komponen utama, besaran statistik, histogram warna, wavelet transform dan lain-lain.



Gambar 2.1 Sistem TKCBI Umum [15]

Pengukuran kemiripan merupakan proses pencocokan citra untuk memperoleh citra yang mempunyai kemiripan dengan citra kueri dapat dilihat pada Gambar 2.1. Pengukuran kemiripan dilakukan dengan menghitung jarak antara fitur citra kueri dengan kumpulan fitur dari citra yang berada pada *database*. Hasil citra merupakan citra-citra *database* yang mempunyai jarak fitur terpendek dengan fitur citra kueri.

## 2.2. *Convolutional Neural Network*

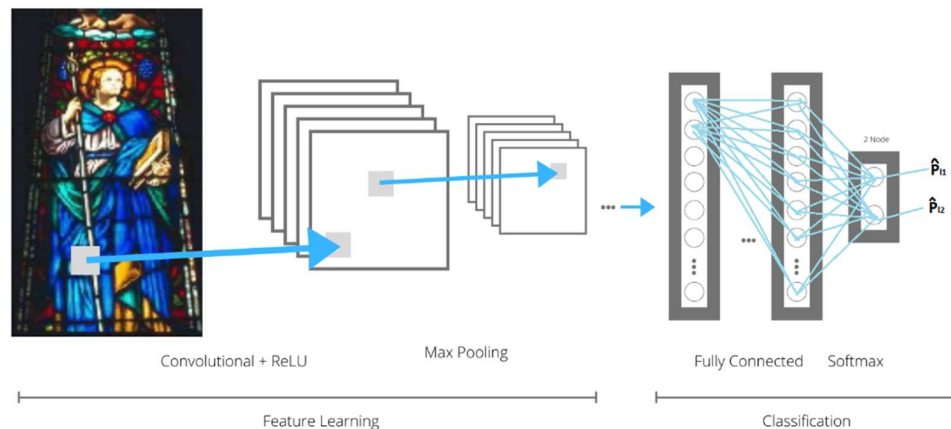
*Convolutional neural network* (CNN) adalah jenis jaringan syaraf tiruan khusus untuk memproses data yang memiliki kemiripan dengan *grid topology* [16], [17]. Misalnya data deret-waktu yang mengambil sampel pada interval waktu yang teratur dianggap *grid* 1D dan data citra dapat dianggap *grid* 2D karena memiliki piksel 2 dimensi yaitu panjang dan lebar. CNN merupakan suatu *layer* yang memiliki susunan *neuron* 3D (lebar, tinggi, kedalaman). Lebar dan tinggi merepresentasikan ukuran *layer* yang digunakan pada CNN sedangkan kedalaman mengacu pada jumlah *layer*. Secara umum *layer* pada CNN dibedakan menjadi dua jenis yaitu:

- a. Layer ekstraksi fitur citra, letak *layer* ini berada pada awal arsitektur tersusun atas beberapa *layer* dan setiap *layer* tersusun atas *neuron* yang terkoneksi pada daerah lokal (*local region*) layer sebelumnya. Pada *layer*

ini terdapat dua bentuk layer yaitu *layer* konvolusi dan *layer pooling*. Setiap *layer* diberlakukan fungsi aktivasi. Posisi untuk *layer* konvolusi dan *layer pooling* disusun berselang-seling. *Layer* ini menerima input citra secara langsung dan memprosesnya hingga menghasilkan keluaran berupa vektor untuk diolah pada *layer* berikutnya.

- b. Layer klasifikasi tersusun atas beberapa *layer* dan setiap *layer* tersusun atas neuron yang terkoneksi secara penuh (*fully connected*) dengan *layer* lainnya. *Layer* menerima input dari hasil keluaran *layer* ekstraksi fitur citra berupa vektor kemudian ditransformasikan seperti *Multi Neural Network* dengan tambahan beberapa tambahan *hidden layer*. Hasil keluaran berupa skoring kelas untuk klasifikasi. Namun untuk *Convolutional Autoencoder* tidak memerlukan *layer* ini.

Dengan demikian CNN merupakan metode untuk mentransformasikan citra original *layer* per *layer* dari nilai piksel citra kedalam nilai skoring kelas untuk klasifikasi. Dan setiap *layer* ada yang memiliki *hyperparameter* dan ada yang tidak memiliki parameter (bobot dan bias pada *neuron*).



Gambar 2.2 Arsitektur Dasar CNN

Berikut penjelasan dari Gambar 2.2 yang merepresentasikan arsitektur dasar CNN:

1. Citra masukan masuk ke lapisan konvolusi pertama. Fitur yang diterapkan di lapisan konvolusi mengekstrak fitur yang relevan dari gambar masukan. Setiap filter akan memberikan fitur berbeda yang dapat meningkatkan

ekstraksi fitur. Operasi konvolusi dan proses aktivasi *ReLU* pada lapisan ini menghasilkan *rectified feature map*.

2. *Rectified feature map* kemudian masuk ke lapisan *pooling*. Lapisan *pooling* berfungsi untuk mengurangi dimensi fitur atau jumlah parameter.
3. Beberapa lapisan konvolusi dan *pooling* dapat ditambahkan sebelum memasuki lapisan *fully-connected*. Lapisan konvolusi berperan dalam melakukan ekstraksi fitur. Semakin dalam arsitektur jaringan, maka fitur yang diekstraksi akan lebih spesifik jika dibandingkan dengan jaringan dangkal dimana fitur yang diekstraksi lebih umum.
4. Setelah melewati lapisan konvolusi dan *pooling*, selanjutnya memasuki lapisan *fully-connected*. Hasil keluaran dari lapisan konvolusi dan *pooling* merepresentasikan fitur tingkat tinggi gambar masukan. Lapisan *fully-connected* menggunakan fitur tersebut untuk tugas klasifikasi gambar. Lapisan *fully-connected* merupakan *multi layer perceptron* biasa yang memiliki 3 bagian yaitu lapisan masukan, lapisan tersembunyi dan lapisan keluaran. Hasil keluaran dari lapisan konvolusi dan *pooling* yang terakhir menjadi masukan pada lapisan ini. Lapisan keluaran merupakan vektor prediksi yang ukurannya sesuai dengan jumlah kelas yang akan diprediksi. Namun pada TKCBI, lapisan ini tidak digunakan karena TKCBI hanya membutuhkan ekstraksi fitur dengan lapisan konvolusi dan lapisan *pooling*.
5. Setelah forward pass selesai, *backpropagation* dimulai untuk memperbaru filter (bobot) dan nilai bias.

### **2.2.1. Lapisan Konvolusi**

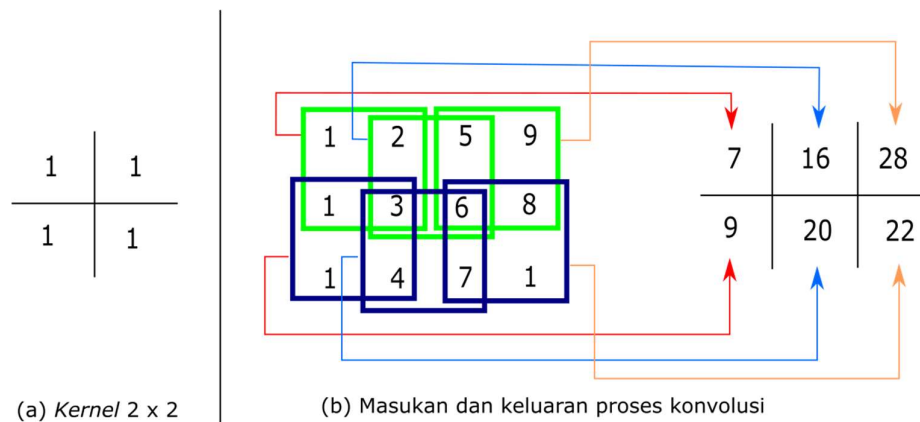
Nama CNN menunjukkan bahwa jaringan tersebut menggunakan operasi matematika yang disebut konvolusi / *convolutional*. CNN merupakan jaringan saraf yang menggantikan penerapan matriks umum menggunakan konvolusi dimana minimal terdapat satu konvolusi di setiap *layer*. *Layer Convolutional* merupakan lapisan yang pertama kali menerima input citra langsung pada arsitektur [18]. Operasi pada *layer* ini sama dengan operasi konvolusi yaitu melakukan operasi kombinasi linear *filter* terhadap daerah lokal. *Filter* merupakan representasi bidang reseptif dari *neuron* yang terhubung kedalam daerah lokal (*local connectivity*) pada

input citra. Bentuk *layer* direpresentasikan sebagai volume  $B \times K \times L$  atau *layer* ukuran  $B \times K$  dengan jumlah sebanyak  $L$ . *Convolutioan layer* memiliki *hyperparameter* dan parameter yang penjelasanya dapat dilihat pada Tabel 2.1.

Tabel 2.1 Hyperparameter pada *Convolutional Layer*

| No | <i>Hyperparameter</i> | Keterangan                                                           |
|----|-----------------------|----------------------------------------------------------------------|
| 1  | <i>Depth</i>          | Kedalaman <i>layer</i> atau jumlah <i>layer</i> konvolusi            |
| 2  | <i>Stride</i>         | Jumlah pergeseran <i>filter</i> pada proses konvolusi                |
| 3  | <i>Zero-padding</i>   | Jumlah penambahan nilai intensitas nol di daerah sekitar input citra |

Ilustrasi operasi konvolusi pada matriks dengan kernel konvolusi tunggal dapat dilihat pada Gambar 2.3.



Gambar 2.3 Ilustrasi operasi konvolusi

Misalkan ukuran citra masukan adalah  $3 \times 4$  dan ukuran *kernel* konvolusi adalah  $2 \times 2$ , seperti yang diilustrasikan pada Gambar 2.3. Jika *kernel* konvolusi diletakkan di atas citra input, hasil kali antar angka pada lokasi yang sama pada *kernel* dan inputan dapat dihitung dan akan didapatkan nilai tunggal dengan menjumlahkan semua hasil kali ini. Misalnya ketika *kernel* diletakkan di bagian atas kiri pada inputan, hasil operasi konvolusi pada daerah ini adalah:  $1 \times 1 + 1 \times 1 + 1 \times 2 + 1 \times 3 = 7$ . Kemudian *kernel* berpindah ke bawah satu piksel dan mendapatkan hasil konvolusi berikutnya:  $1 \times 2 + 1 \times 3 + 1 \times 5 + 1 \times 6 = 16$ . *Kernel* terus dipindahkan hingga mencapai batas bawah matriks (citra) masukan. Kemudian, *kernel* dikembalikan ke atas, dan dipindahkan ke kanan sebanyak satu elemen

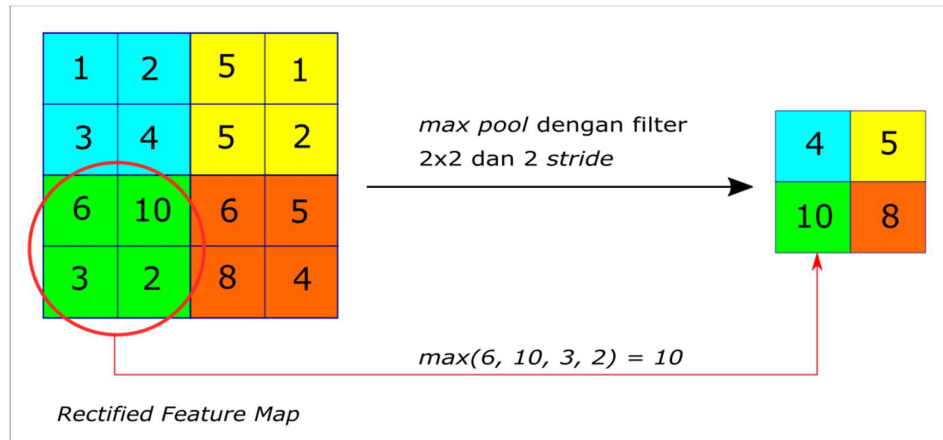
(piksel). Operasi konvolusi dilakukan terus-menerus pada setiap lokasi yang memungkinkan sampai *kernel* berpindah ke sudut kanan bawah citra masukan, seperti yang ditunjukkan pada Gambar 2.3.

Selama ukuran *kernel* konvolusi lebih besar dari  $1 \times 1$ , hasil keluaran operasi konvolusi akan berukuran lebih kecil dari masukan. Terkadang dibutuhkan citra keluaran yang memiliki tinggi dan lebar sama dengan citra masukan. Hal ini bisa dilakukan dengan memanfaatkan *padding*. Jika citra masukan berukuran  $H^l \times W^l \times D^l$ , dan ukuran kernel adalah  $H \times W \times D$ , maka hasil konvolusi berukuran  $(H^l - H + 1) \times (W^l - W + 1) \times D$ . Untuk setiap kanal masukan, jika ditambahkan  $\lfloor \frac{H-1}{2} \rfloor$  baris diatas baris pertama,  $\lfloor \frac{H}{2} \rfloor$  baris dibawah baris terakhir dan ditambahkan  $\lfloor \frac{W-1}{2} \rfloor$  kolom di sebelah kiri kolom pertama,  $\lfloor \frac{W}{2} \rfloor$  kolom di sebelah kanan kolom terakhir, maka hasil keluaran akan berukuran  $H^l \times W^l \times D^l$ , sama seperti ukuran citra masukan. Nilai elemen pada baris dan kolom ditambahkan biasanya bernilai 0.

Konsep penting lainnya pada operasi konvolusi adalah *stride*. Pada citra 1, nilai *stride* yang digunakan  $s = 1$ . *Kernel* konvolusi berpindah di setiap lokasi yang memungkinkan pada citra masukan. Namun, jika  $s > 1$ , setiap perpindahan kernel melewati sejumlah  $s - 1$  lokasi piksel.

### 2.2.2. Lapisan *Pooling*

Lapisan *pooling* akan mereduksi ukuran spasial dan jumlah parameter dalam jaringan serta mempercepat komputasi dan mengontrol terjadinya *overfitting* [18]. Lapisan *pooling* bekerja dengan blok spasial yang bergerak sepanjang ukuran *feature pattern*. Ukuran pergeseran blok pada umumnya adalah ukuran pada dimensi blok ( $H \times H$ ) itu sendiri sehingga tidak ada *overlapping* seperti pada lapisan konvolusi. Pergerakan blok diikuti dengan perhitungan *pooling* pada masukan pola fitur. Pada *layer* ini tidak memiliki parameter karena parameter sudah ditentukan sebelumnya (*fixed*). Lapisan *pooling* memiliki beberapa macam tipe antara lain *average pooling*, *max pooling* dan *Lp pooling*. Contoh ilustrasi operasi *max pooling* dapat dilihat pada Gambar 2.4.

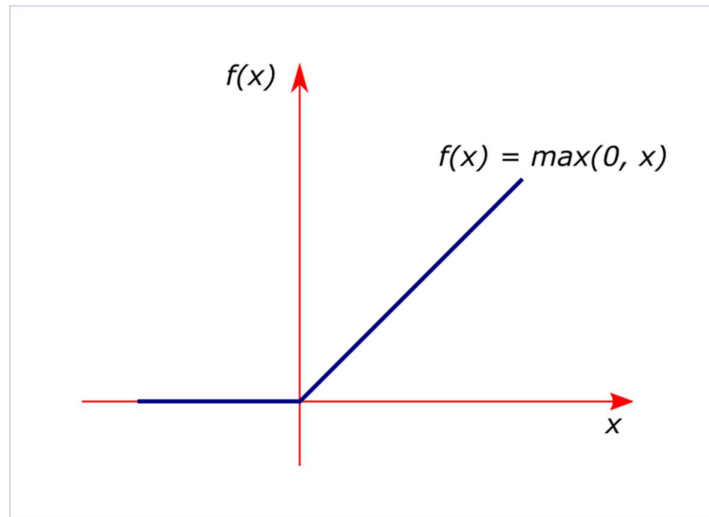


Gambar 2.4 Ilustrasi operasi *max pooling*.

Gambar 2.4 menunjukkan contoh operasi *max pooling* pada *rectified feature map* yang diperoleh setelah operasi konvolusi + ReLu dengan menggunakan *grid* berukuran  $2 \times 2$ . *Grid*  $2 \times 2$  bergeser sebesar 2 piksel (disebut juga “*stride*”) dan diambil nilai maksimal di setiap wilayah. Sehingga dimensi *rectified feature map* berkurang.

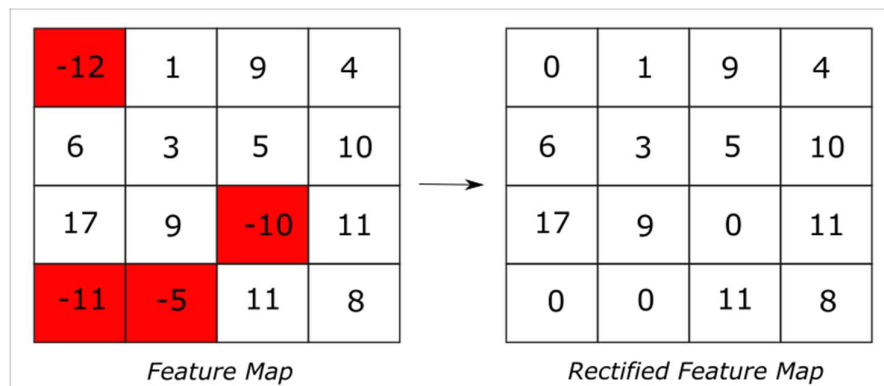
### 2.2.3. Fungsi Aktivasi (*Neurons*)

Fungsi aktivasi atau fungsi transfer merupakan fungsi non-linear yang memungkinkan sebuah jaringan untuk dapat menyelesaikan permasalahan-permasalahan *non trivial* [18]. Setiap fungsi aktivasi mengambil sebuah nilai dan melakukan operasi matematika. Pada arsitektur CNN, fungsi aktivasi terletak perhitungan akhir keluaran *feature map* atau sesudah proses perhitungan konvolusi atau *pooling* untuk menghasilkan suatu pola fitur. Beberapa macam fungsi aktivasi yang sering digunakan dalam penelitian antara lain fungsi *sigmoid*, *tanh*, *Rectified Linier Unit (ReLU)*, *Leaky ReLU(LreLU)* dan *Parametic ReLU*. Fungsi aktivasi ReLU biasanya digabung dengan *layer* konvolusi. Penambahan fungsi aktivasi ReLU memungkinkan CNN mempelajari hubungan non-linier antara nilai piksel masukan dan hasil keluaran jaringan. Operasi konvolusi yang ada pada lapisan konvolusi bersifat linear. Jika tidak ada fungsi non-linear pada jaringan maka akan dihasilkan model linear yang kinerjanya buruk dalam melakukan tugas klasifikasi[16]. Grafik fungsi ReLU yang akan digunakan pada penelitian ini dapat dilihat pada Gambar 2.5.



Gambar 2.5 Fungsi ReLU

Grafik pada Gambar 2.5 menunjukkan bahwa fungsi *ReLU* bersifat non-linear. Fungsi *ReLU* mengambil nilai  $x$  dan mengembalikan 0 jika nilai  $x$  negatif dan mengembalikan nilai  $x$  jika nilai  $x$  positif. Pada fungsi aktivasi ini diterapkan di setiap piksel *feature map*. Hasil keluaran berupa *feature map* tanpa nilai negatif. Ilustrasi proses aktivasi *ReLU* dapat dilihat pada Gambar 2.5.



Gambar 2.6 Proses Aktivasi ReLU

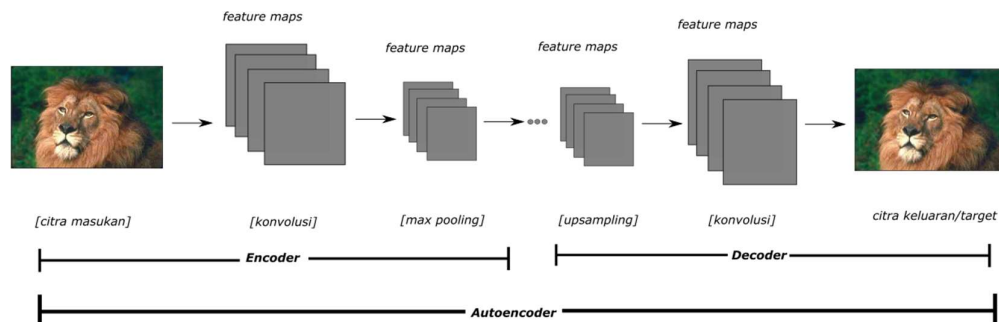
### 2.3. Convolutional Autoencoder

Operator konvolusi memungkinkan untuk menyaring data/citra masukan untuk mengekstrak beberapa bagian dari isi data tersebut. Operator tersebut digunakan oleh *Autoencoder* untuk mendapatkan fitur baru yang mewakili data/citra. *Autoencoder* merupakan model jaringan saraf tiruan yang memiliki data masukan dan data keluaran yang sama [19], [20]. *Autoencoder* mempelajari citra masukan hingga menghasilkan fitur-fitur penting, lalu *Autoencoder* berusaha untuk

melakukan rekontruksi terhadap citra masukan tersebut. Dengan menggunakan *Autoencoder*, dapat mereduksi atau mengurangi fitur citra masukan. Jika citra masukan memiliki dimensi yang sangat tinggi bisa jadi setiap fitur tesebar pada setiap dimensi dari citra sehingga setiap citra yang ada terlihat sangat berbeda.

Dalam formulasi *Autoencoder* tradisional tidak memperhitungkan bahwa sebuah fitur dapat mewakili sejumlah fitur tertentu. Sebaliknya, *Convolutional Autoencoder* menggunakan operator konvolusi untuk mengeksploitasi pengamatan tersebut. *Convolutional Autoencoder* merupakan pembelajaran *self-supervised* (pengawasan sendiri) dengan menggunakan lapisan yang terdapat pada *Convolutional Neural Network* (CNN), dimana dalam proses pembelajaran, *Convolutional Autoencoder* menyandikan citra masukan dalam satu set fitur sederhana dan kemudian mencoba merekontruksi citra masukan[5]. *Convolutional Autoencoder* bertujuan untuk membuat output sedekat mungkin dengan gambar input.

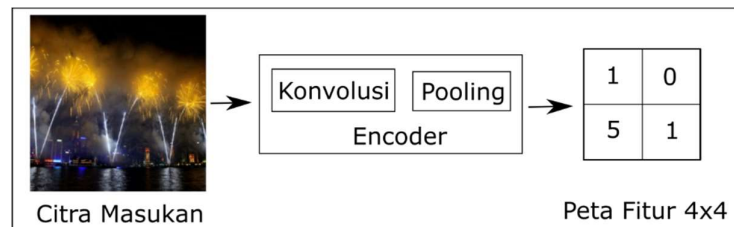
Arsitektur *Convolutional Autoencoder* terbagi menjadi dua lapisan sama seperti *Autoencoder* tradisional, yaitu lapisan *Encoder* dan lapisan *Decoder*. *Encoder* merupakan lapisan yang menyederhanakan fitur citra input ke dalam representasi berdimensi rendah yang berisi konten informatif dari data. Dalam lapisan *Encoder* terdapat operator konvolusi mengekstrak fitur, hal tersebut yang membedakan *Convolutional Autoencoder* dengan *Autoencoder* tradisional. Sedangkan, *Decoder* merupakan lapisan yang dilatih untuk merekontruksi input dari fitur yang diekstrak oleh *Encoder*. Arsitektur *Convolutional Autoencoder* dapat dilihat pada Gambar 2.7.



Gambar 2.7 Arsitektur *Convolutional Autoencoder*

Berikut penjelasan dari Gambar 2.7 yang merepresentasikan arsitektur *Convolutional Autoencoder*:

1. Citra masukan masuk ke lapisan konvolusi. Filter yang diterapkan di lapisan konvolusi mengekstrak fitur yang relevan dari citra masukan. Setiap filter akan memberikan fitur berbeda yang mewakili citra tersebut. Pada lapisan ini terdapat fungsi aktivasi ReLU.
2. Fitur hasil lapisan konvolusi memasuki lapisan *pooling* untuk mengurangi jumlah parameter. Pada lapisan *pooling* menggunakan nilai maksimal dari perwakilan piksel tertentu sesuai *stride* masing-masing.
3. Jumlah lapisan konvolusi dan *pooling* sebanyak sesuai kebutuhan, semakin banyak kedua lapisan tersebut maka semakin spesifik fitur hasil ekstraksi[5]. Lapisan *Encoder* menggunakan citra masukan sebagai input dan lapisan *Encoder* menghasilkan *feature maps* kecil yang merupakan representasi dari citra masukan. Input dan keluaran proses pada lapisan *Encoder* ditunjukkan pada Gambar 2.8.



Gambar 2.8. Input dan Output dari Proses Lapisan *Encoder*

Pada Gambar 2.8 menunjukkan input dari proses pada lapisan *Encoder* adalah citra masukan. Sedangkan, output dari proses pada lapisan *Encoder* adalah peta fitur atau *feature maps* dengan ukuran yang telah ditentukan yang merepresentasikan citra masukan.

4. Setelah melewati lapisan pooling atau berakhirnya bagian *Encoder*, dan menghasilkan fitur penting citra, selanjutnya memasuki bagian *Decoder* dengan melewati lapisan *upsampling*. Pada lapisan *upsampling* melakukan penggandaan fitur penting agar ukuran piksel keluaran sesuai dengan citra masukan. Proses dari *upsampling* dapat dilihat pada Gambar 2.9.