BAB II
LANDASAN TEORI

2.1 KAJIAN PUSTAKA

Penelitian [1] merupakan penelitian yang membahas masalah analisis citra jerawat melalui *Label Distribution Learning* (LDL). Data yang digunakan pada penelitian ini sebanyak 1457 sampel data citra dengan 4 kelas. Penelitian ini membuat kode dan kumpulan data yang tersedia untuk umum di https://github.com/xpwu95/ldl. Arsitektur model yang digunakan adalah ResNet-50. Metode penelitian ini mempelajari representasi fitur berkelanjutan dari gambar jerawat dari tugas penghitungan lesi, kemudian mempelajari distribusi label tingkat keparahan untuk menilai citra jerawat secara efisien. Hasil akurasi yang didapat pada pengujian terhadap data baru sebesar 84%.

Penelitian [6] membahas sistem klasifikasi bekas luka jerawat otomatis yang baru diusulkan berdasarkan model Jaringan Syaraf Konvolusional (CNN) yang mendalam. Data yang digunakan berisi 250 gambar dari lima kelas berbeda dikumpulkan dan diberi label oleh empat dokter kulit berpengalaman. Penelitian ini menggunakan arsitektur ScarNet, untuk ekstraksi peta fitur yang mendalam. Hasil eksperimen menunjukkan kelayakan metode yang diusulkan dengan akurasi, spesifisitas, dan skor kappa masing-masing sebesar 92,53%, 95,38%, dan 76,7%

Penelitian [7] membahas fokus dalam pengembangan keakurasian metode menggunakan metode *hough circle transform* & *Convolutional Neural Network* (CNN). Data yang didapatkan dari dataset jurnal atau *website database* untuk kulit berjerawat ada 3 kelas yaitu *fungal acne, acne nodules* dan *acne fulminans*. Algoritma *Convolutional Neural Network* (CNN) pada penelitian ini dimaksudkan untuk mengklasifikasi citra Jerawat. Berbeda dengan algoritma klasifikasi biasa, jika pada algoritma klasifikasi biasanya melakukan proses ekstraksi fitur dan klasifikasi secara terpisah maka model algoritma dari cabang bidang deep learning ini akan mengekstraksi fitur lalu mengklasifikasi citra dalam satu proses. Dengan kata lain, ekstraksi fitur pada algoritma CNN juga ikut *me-learning*. Hasil dari proses learning didapatkan model CNN dengan akurasi 99,8% hingga 100%.

Penelitian [8] merupakan penelitian mengenai pengembangan aplikasi visi komputer berbasis web untuk melakukan pendeteksian jerawat pada foto wajah megnggunakan algoritma *deep learning* berbasis *convolution neural network* (CNN) dengan *Tensorflow*. Pengumpulan data yang dikakukan menggunakan smartphone mendapatkan hasil sejumlah lima belas gambar dari lima orang reponden. Pengumpulan data dari Google sejumlah tiga puluh lima gambar wajah orang yang mengalami masalah jerawat. Total gambar yang terkumpul sejumlah lima puluh gambar. Model menggunakan metode transfer learning dari model SSD ResNet50 V1 FPN 640x640. Hasil dari model pelatihan memiliki nilai sensitivity 77,3%, specificity 47,3%, dan accuracy 63,2%. Aplikasi yang dikembangkan sudah dapat mendeteksi dan menandai jerawat yang ada pada foto wajah dengan kecepatan rata-rata 2,77 detik untuk setiap gambar.

Penelitian [9] merupakan *implementasi* dari pendeteksian jerawat menggunakan *image processing* dan secara *realtime*, lalu sistem akan mengklasifikasikan jerawat yang ada pada wajah. Jerawat yang dapat dikenali oleh sistem ini yaitu jerawat, bekas, dan pus. Sistem deteksi dan klasifikasi ini dibuat dengan metode *deep learning* dengan menggunakan bahasa pemrograman *Python*, yang dibantu dengan menggunakan *framework TensorFlow* dengan model *Faster* R-CNN. Data yang digunakan pada penelitian ini sebanyak 3 kelas yaitu jerawat, pus, dan bekas. Pengujian mendapatkan hasil nilai akurasi yang berbeda, masing-masing yaitu 71,4% di dalam ruangan pada siang hari dengan bantuan cahaya matahari, 76,1% di luar ruangan pada siang hari dengan bantuan cahaya matahari, dan 57,1% di dalam ruangan pada malam hari dengan bantuan cahaya lampu.

Penelitian [10] menyajikan metode diagnosis otomatis baru untuk *acne vulgaris* wajah berdasarkan jaringan saraf *convolutional*. Data yang digunakan pada penelitian ini sebanyak 7 kelas yaitu *blackhead, whitehead, papul, pustula, kista, nodul*, kulit normal. Penelitian ini membandingkan keefektifan jaringan saraf konvolusional kami dan jaringan saraf pra-pelatihan VGG16 pada kumpulan data *ImageNet*. Hasil yang didapat pada pengujian yaitu *accuracy* 0,943, *sensitivity* 0,959, dan *specificity* 0,961.

Penelitian [11] mengusulkan jaringan deteksi jerawat yang terdiri dari tiga komponen, yaitu: Perbaikan Fitur Komposit, Peningkatan Konteks Dinamis, dan

Multi-Perhatian Masker-*Aware*. Eksperimen dilakukan pada dataset citra jerawat ACNE04 dan dataset citra natural PASCAL VOC 2007. Penelitian ini mengusulkan pendeteksi jerawat yang disebut ACNet untuk mendeteksi jerawat di bawah perubahan warna, variasi skala, dan susunan padat. Metode ini terdiri dari penyempurnaan fitur komposit, peningkatan konteks dinamis, dan *multi-attention mask*-aware. Hasil akurasi yang didapat pada pengujian sebesar 81,8 mAP.

Penelitian [12] merupakan penelitian yang membahas masalah analisis citra jerawat melalui *Label Distribution Learning* (LDL). Data yang digunakan dataset CelebAMask-HQ yang menyediakan 19 kelas berbeda (kulit, hidung, mata (kiri dan kanan), alis (kiri dan kanan), telinga (kiri dan kanan), mulut, bibir, rambut, topi, kacamata, anting-anting, kalung, leher , dan kain, latar belakang). Penelitian ini mengekstrak hanya dua kelas yang berbeda (kulit dan latar belakang yang terkena jerawat), kulit dan hidung telah digabungkan menjadi satu dan semua kelas lainnya telah menjadi kelas latar belakang. Arsitektur yang digunakan yaitu tipe DenseNet121. Hasil model klasifikasi terlatih mencapai skor rata-rata akhir f1 sebesar 60,84% dalam membedakan antara wajah yang terkena jerawat dan wajah yang tidak terpengaruh.

Penelitian [13] mengembangkan sistem AI yang disebut *AcneDet* untuk deteksi objek jerawat otomatis dan penilaian tingkat keparahan jerawat menggunakan gambar wajah yang diambil oleh *smartphone*. *AcneDet* menyertakan dua model untuk dua tugas: (1) model pembelajaran mendalam berbasis R-CNN, (2) model pembelajaran mesin LightGBM. Dataset yang terdiri dari 1572 gambar wajah berlabel yang diambil oleh *smartphone* iOS dan Android digunakan untuk pelatihan. Hasil menunjukkan bahwa model *Faster* R-CNN mencapai mAP 0,54 untuk deteksi objek jerawat. Akurasi rata-rata tingkat keparahan jerawat dengan model LightGBM adalah 0,85.

Penelitian [14] telah mengembangkan pendekatan baru dan menggunakan model *Deep Residual Neural Network* untuk mengklasifikasikan lima kelas acnes. Dataset yang digunakan berupa gambar untuk 5 kelas penyakit jerawat yaitu jerawat komedo tertutup, jerawat kista, jerawat keloidalis, jerawat terbuka komedo, dan jerawat pustular. Jumlah total gambar untuk percobaan mencapai 1800 dan per kelas adalah 360 gambar. Hasil dari model telah mencapai akurasi perkiraan

sebanyak 99,44% untuk satu kelas, dan sisanya juga diatas 94% dengan presisi yang cukup tinggi dan skor ingatan.

Tabel 2.1 Penelitian sebelumnya

NO	Judul	Penulis	Data	Metode	Perbedaan
1	Joint acne image grading and counting via label distribution learning	Wu, Xiaoping Wen, Ni Liang, Jie Lai, Yu Kun She, Dongyu Cheng, Ming Ming Yang, Jufeng	4 kelas: ringan, sedang, berat, sangat parah	Resnet 50	Metode Googlenet
2	ScarNet: Development and Validation of a Novel Deep CNN Model for Acne Scar Classification with a New Dataset	Junayed, Masum Shah Islam, Md Baharul Jeny, Afsana Ahsan Sadeghzadeh, Arezoo Biswas, Topu Shah, A. F.M.Shahen	5 kelas: bekas jerawat Hypertrophic, Keloidal, Icepick, Rolling, dan Boxcar	Arsitektur ScarNet	5 kelas dengan arsitektur scarnet
3	Deteksi Otomatis Jerawat Wajah Menggunakan Metode Convolutional Neural Network (CNN)	Sudana Putra, Fajar Kusrini Kurniawan, Mei P	3 kelas: fungal acne, acne nodules dan acne fulminans	Convolutional Neural Network (CNN)	3 kelas dengan metode CNN
4	Aplikasi Web Pendeteksi Jerawat Pada Wajah Menggunakan Algoritma Deep Learning dengan TensorFlow	Arifianto, July Muhimmah, Izzati	50 gambar	SSD ResNet50 V1 FPN 640x640	Dengan metode SSD ResNet50 V1 FPN 640x640

5	Implementasi Deep Learning Menggunakan Framework Tensorflow Dengan Metode Faster Regional Convolutional Neural Network Untuk Pendeteksian Jerawat	Hasma, Yunita Aulia Silfianti, Widya	3 kelas klasifikasi yaitu jerawat, pus, dan bekas	Model Faster R-CNN	3 kelas dengan metode Faster R-CNN
6	An Automatic Diagnosis Method of Facial Acne Vulgaris Based on Convolutional Neural Network	Shen, Xiaolei Zhang, Jiachi Yan, Chenjun Zhou, Hong	7 kelas: blackhead, whitehead, papul, pustula, kista, nodul, kulit normal	VGG16	7 kelas dengan arsitektur VGG16
7	ACNet: Mask-Aware Attention with Dynamic Context Enhancement for Robust Acne Detection	Min, Kyungseo Lee, Gun Hee Lee, Seong Whan	dataset ACNE04 4 kelas	Arsitektur ACNet	4 kelas dengan metode ACNet
8	A Deep Learning-Based Facial Acne Classification System	Quattrini, Andrea Boër, Claudio Leidi, Tiziano Paydar, Rick	5 Kelas (0-Tidak Berjerawat, 1-Bersih, 2-Hampir Jelas, 3-Ringan, 4-Sedang, dan 5-Berat)	Arsitektur DenseNet 121	5 kelas dengan arsitektur DenseNet121
9	Automatic Acne Object Detection and Acne Severity Grading Using	Huynh, Quan Thanh Nguyen, Phuc Hoang Le, Hieu Xuan	4 kelas: komedo / komedo putih, papula / pustula, nodul / kista,	Faster R-CNN dan LightGBM	5 kelas dengan 2 model: Faster R-CNN dan LightGBM

	Smartphone Images and Artificial Intelligence	Ngo, Lua Thi Trinh, Nhu Thuy Tran, Mai Thi Thanh Nguyen, Hoan Tam Vu, Nga Thi Nguyen, Anh Tam Suda, Kazuma Tsuji, Kazuhiro Ishii, Tsuyoshi Ngo, Trung Xuan Ngo, Hoan Thanh	dan bekas jerawat		
10	AcneNet - A deep CNN based classification approach for acne classes	Junayed, Masum Shah Jeny, Afsana Ahsan Atik, Syeda Tanjila Neehal, Nafis Karim, Asif Azam, Sami Shanmugam, Bharanidharan	5 kelas: Komedo Tertutup, Kistik, Keloidalis, Terbuka Komedo dan Pustular	AcneNet	5 kelas dengan arsitektur AcneNet

2.2 JERAWAT

Kulit merupakan bagian tubuh terluar yang membatasi dari lingkungan manusia. Kulit memiliki struktur yang sangat kompleks, dan juga bervariasi sesuai dengan iklim, usia, jenis kelamin, ras, dan lokasinya pada tubuh. Terdapat tiga lapisan utama pada kulit yang terdiri dari lapisan epidermis, dermis, dan subkutis. Selain itu, kulit juga mempunyai kelenjar pada kulit, rambut, dan kuku yang terdapat kelenjar minyak atau *glandula sebasea*. Kelenjar tersebut memiliki fungsi menjaga keseimbangan dari kelembaban kulit, yang pada masa pubertas berfungsi secara aktif dan menjadi lebih besar. Hal tersebut dapat menyebabkan gangguan pada kulit, salah satunya adalah *acne vulgaris* atau jerawat[15]. Dapat dilihat

contoh sempel jerawat pada gambar 2.1 berupa *dataset* yang diperoleh secara *online* melalui *https://github.com/xpwu95/LDL*[1].

Gambar 2.1 Jerawat[1]

Acne vulgaris adalah gangguan inflamasi pada unit *pilosebasea*, yang berlangsung secara kronis dan dapat sembuh sendiri (*self-limited disease*). *Acne vulgaris* dipicu oleh *Cutibacterium acnes* (sebelumnya dikenal sebagai *Propionibacterium acnes*) pada masa remaja, di bawah pengaruh sirkulasi normal *dehydroepiandrosterone* (DHEA)[16]. *Acne vulgaris* merupakan kelainan kulit yang sangat umum serta dapat muncul dengan lesi inflamasi dan non-inflamasi terutama di wajah tetapi juga dapat terjadi pada lengan atas, dada, dan punggung. *Acne vulgaris* adalah penyakit yang bisa ditemukan pada semua umur. Ini adalah peradangan kronis pada unit *folikel* kelenjar *sebaceous*. Penyebabnya adalah ciri klinis yang multifaktorial berupa komedo, papula, pustula, nodul, dan kista. Jerawat adalah penyakit kulit karena adanya penumpukan minyak yang menyebabkan pori-pori kulit wajah tersumbat sehingga memicu aktivitas bakteri dan peradangan pada kulit[15].

2.3 *DEEP LEARNING*

Deep learning merupakan subbidang *machine learning* yang algoritmanya terinspirasi dari struktur otak manusia. Struktur tersebut dinamakan *Artificial Neural Networks* atau disingkat ANN. Pada dasarnya, ANN merupakan jaringan saraf yang memiliki tiga atau lebih lapisan ANN. ANN mampu belajar dan beradaptasi terhadap sejumlah besar data serta menyelesaikan berbagai permasalahan yang sulit diselesaikan dengan algoritma *machine learning* lainnya.

Deep Learning bagian dari pembelajaran mesin berkaitan dengan model berdasarkan jaringan saraf berlapis banyak yang merupakan pengembangan dari *neural network multiple layer* untuk memberikan ketepatan tugas seperti deteksi objek, pengenalan suara, terjemahan bahasa dan lain–lain. *Deep Learning* berbeda dari teknik *machine learning* yang membedakan *deep learning* adalah bahwa operasi dipelajari pada masing-masing dari banyak lapisan representasi dipelajari bersama dari data[17].

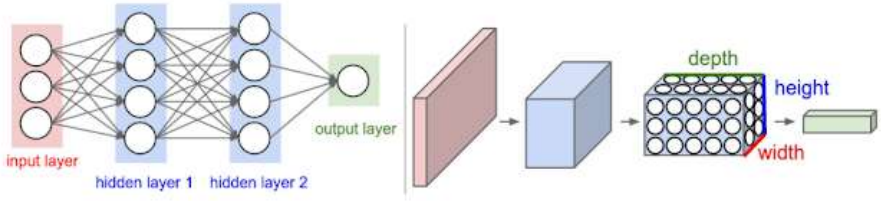
2.4 *CONVOLUTIONAL NEURAL NETWORK* (CNN)

Convolutional Neural Network (CNN) adalah metode *neural network* yang dikhususkan untuk data dimensi tinggi seperti gambar dan video. Metode ini menggunakan operasi matematika dari sebuah operasi linear atau yang biasa disebut konvolusi[18]. Secara umum, CNN disusun oleh beberapa *layer*, yaitu:

1. *Input Layer*, merupakan sebuah *layer* yang berisi sebuah citra 3 dimensi yang memiliki kedalaman 3 *layer*, sebagai representasi *layer* R, G, dan B.
2. *Convolution Layer*, merupakan *layer* yang berfungsi untuk menghitung output dari neuron yang terhubung di dalam suatu *local region*. *Layer* ini berbentuk 3 dimensi dengan kedalaman sesuai dengan jumlah *filter* yang dipakai.
3. *Activation Layer*, *layer* ini berfungsi untuk melakukan perhitungan fungsi aktivasi terhadap hasil output dari *convolution layer*.
4. *Pooling Layer*, berfungsi untuk melakukan *downsampling* terhadap citra hasil konvolusi. Contohnya adalah melakukan *downsampling* dari citra berukuran [64x64] menjadi citra berukuran [32x32].
5. *Fully-Connected Layer*, *layer* ini berfungsi untuk menghitung skor *class* yang ada terhadap citra yang telah diolah.

CNN memiliki fungsi untuk melakukan ekstraksi fitur. Fitur-fitur perlu didapatkan guna proses atau tugas seperti klasifikasi, *clustering* ataupun *regresi*. Pada *machine learning* konvensional dilakukan ekstraksi fitur manual, artinya ditentukan terlebih dahulu fitur-fitur yang diekstraksi. Sedangkan CNN melakukan ekstraksi fitur secara otomatis pada *convolutional layer*, *pooling layer* dan juga aktivasi *Rectified Linear Unit* (ReLU). Selanjutnya fitur-fitur dilakukan proses klasifikasi pada *Fully Connected layer* (FCL) dan aktivasi *softmax*.

Neural network yang umum biasanya mengubah input dengan meletakkannya melalui rangkaian *hidden layer*. Setiap *layer* terdiri dari sekumpulan neuron, dimana setiap *layer* terhubung secara penuh dengan semua neuron pada *layer* sebelumnya. Terakhir, lapisan yang sudah terhubung sepenuhnya (*output layer*) digunakan untuk mewakili prediksi. Tidak seperti *neural network* biasa, lapisan pada algoritma CNN memiliki neuron yang diatur dalam 3 dimensi: *width*, *height*, dan *depth*. Dimensi *depth* mengacu pada dimensi ketiga dari fungsi aktivasi, bukan kedalaman *neural network* atau jumlah total layer dalam jaringan. Neuron-neuron dalam satu *layer* tidak terhubung ke semua neuron di *layer* berikutnya tetapi hanya ke sebagian kecil saja. Terakhir, hasil akhir akan direduksi menjadi satu vektor skor probabilitas, yang diatur sepanjang dimensi *depth*. Perbedaan arsitektur ini dapat dilihat pada gambar berikut:



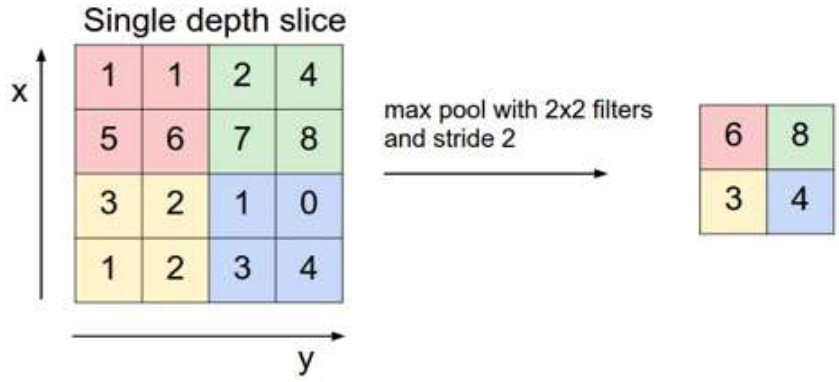
Gambar 2.2 Arsitektur CNN[19].

Dapat dilihat pada gambar 2.2 di sebelah kiri adalah visualisasi layer yang ada pada neural network biasa. Sedangkan sebelah kanan adalah visualisasi layer yang ada di CNN. Dapat dilihat bahwa CNN memiliki dimensi depth yang membuatnya berbentuk 3D.

Setelah *convolution layer*, biasanya ditambahkan lapisan penyatuan (*pooling layer*) di antara layer CNN. Fungsi *pooling* adalah untuk mengurangi dimensi secara terus menerus serta mengurangi jumlah parameter dan komputasi dalam jaringan. Hal ini akan mempersingkat waktu *training* dan mengontrol terjadinya *overfitting*.

Jenis *pooling* yang paling sering digunakan pada CNN adalah *max pooling*. *Pooling* ini mengambil nilai maksimum di setiap *window*/blok. Ukuran *window*/blok ini perlu ditentukan sebelumnya. Operasi ini mengurangi ukuran peta

fitur dan hanya akan menyimpan informasi penting dan signifikan dalam proses klasifikasi nantinya.



Gambar 2.3 *Pooling layer* [20]

Gambar 2.3 adalah ilustrasi kinerja dari *pooling layer*, yang bekerja sesuai dengan jenisnya, dimensi, dan stride. Terdapat tiga jenis *pooling*, yaitu *sum, max,* dan *average pooling*, masing-masing merepresentasikan proses yang berbeda pada *pooling layer* tersebut. Dimensi akan mempengaruhi ukuran matriks yang diproses oleh *pooling layer*. Selain itu, *stride* juga menentukan jarak antara masing-masing *pooling layer*. Dalam contoh Gambar 2.3 *pooling layer* dengan *coloum* 4x4 dengan *max pooling* yang menggunakan filter berukuran 2x2 yang diaplikasikan dengan *stride* sebanyak 2, yang kemudian beroperasi pada setiap irisan. Sehingga didapat hasil *max pooling* mengambil nilai maximum dari irisan berwarna merah bernilai 6, irisan berwarna hijau bernilai 8, irisan kuning bernilai 3, dan irisan biru bernilai 4 dengan *coloum* menjadi 2x2.

Ketika menggunakan CNN, ada 4 *hyperparameter* penting yang perlu ditentukan, yakni:

1. Ukuran kernel/filter, yaitu panjang dan lebar filter yang akan digunakan. Misalnya 3x3
2. Jumlah filter, yaitu seberapa banyak filter yang akan kita gunakan
3. *Stride*, yaitu seberapa jauh step dari filter ketika digeser
4. *Padding,* yaitu parameter yang menentukan jumlah *pixels* (berisi nilai 0) yang akan ditambahkan di setiap sisi dari *input*.

2.5 *HYPERPARAMETER*

Dalam bidang *machine learning/deep learning*, terdapat konsep yang dikenal sebagai *hyperparameter*, yaitu parameter-parameter yang tidak dapat diubah selama proses pelatihan. *Hyperparameter* ini berperan dalam membentuk struktur model, seperti jumlah *hidden layer* dan *activation function*, atau menentukan efisiensi dan akurasi pelatihan model, seperti *optimizer*, *learning rate*, dan *batch size*. Salah satu tahap penting dalam proses pembuatan model *deep learning* adalah proses *hyperparameter optimization* yang bertujuan mencari nilai optimum untuk parameter-parameter struktur *neural network* dan proses pelatihan model[21].

Dalam konteks model *deep learning*, *neural network* memiliki peran sentral. Optimizer juga menjadi faktor penting dalam mengatur bagaimana model disesuaikan dengan data pelatihan. Berikut beberapa optimizer yang biasa digunakan:

1. *Stochastic Gradient Descent* (SGD)

   SGD adalah metode iteratif untuk mengoptimalkan fungsi objektif dengan kompleksitas yang sesuai[22]. Metode ini bekerja dengan menggantikan gradien sebenarnya yang dihitung dari seluruh kumpulan data dengan perkiraan gradien tersebut yang dihitung dari subset data yang dipilih secara acak. Khususnya dalam masalah optimisasi dengan dimensi tinggi, ini mengurangi beban komputasi yang sangat tinggi, sehingga proses iterasi menjadi lebih cepat dengan hasil tingkat konvergensi yang lebih rendah.

$$\omega := \omega - \eta \nabla Q(\omega)^n = \frac{\eta}{n} \sum_{i=1}^{n} \nabla Q_i(\omega)_, \tag{2.1}$$

Keterangan:

w = Nilai yang meminimalisasi Q(w) yang akan diestimasikan

Q(w) = Nilai hasil estimasi

Q(i) = Setiap fungsi dari nilai ke-i dari sebuah dataset untuk data latih

η = *learning rate*

n = Jumlah data latih.

2. Adam

Adam, sebuah metode untuk optimisasi stokastik yang efisien yang hanya memerlukan gradien orde pertama dengan penggunaan memori yang sedikit [23]. Metode ini menghitung tingkat pembelajaran adaptif untuk parameter-parameter yang berbeda dari estimasi momen pertama dan kedua dari gradien. Nama Adam berasal dari *adaptive moment estimation*. Adam menggunakan rata-rata bergerak secara *eksponensial*, dihitung pada gradien yang dievaluasi pada mini-batch.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (2.2)$$
$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

Keterangan:

t = Indeks iterasi pelatihan

m dan v = Perkiraan dari momen pertama dan kedua,

g = Gradien pada *mini-batch* saat ini

β = Tingkat pembelajaran

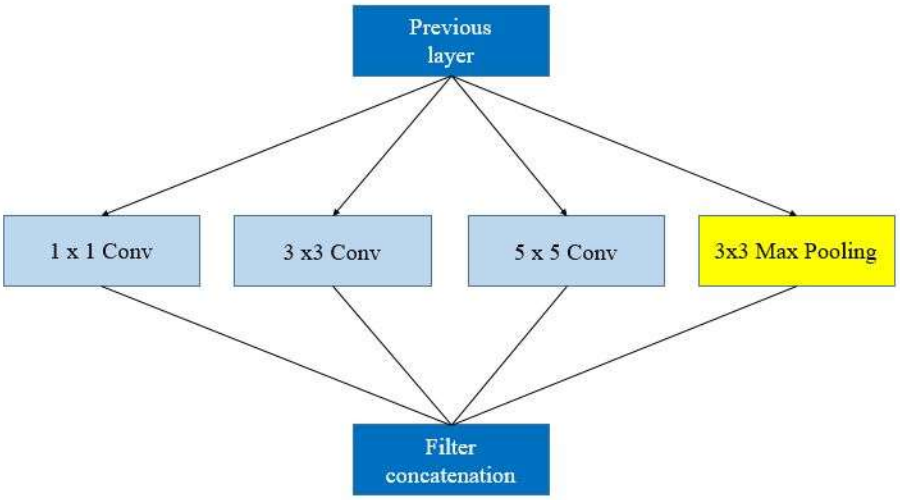
β_1 *dan* β_2 = Tingkat pembelajaran untuk gradien dan momen kedua gradien.

Selain itu, *learning rate* adalah parameter yang menentukan seberapa besar langkah pembelajaran model selama pelatihan. Terakhir, *batch size* mencerminkan jumlah sampel data yang diproses oleh model dalam setiap iterasi pelatihan.

Sebagai kesimpulan, proses *hyperparameter optimization* (HPO) merupakan langkah penting dalam merancang dan melatih model *deep learning* untuk mencapai akurasi dan performa yang optimal. Pengaturan *hyperparameter* ini harus dilakukan dengan cermat berdasarkan pengalaman sebelum memulai proses pelatihan model

2.6 ARSITEKTUR GOOGLENET

GoogLeNet merupakan salah satu arsitektur CNN dengan rangkaian pembangun utamanya merupakan *Inception Module* atau biasa disebut *Inception Network*. Memiliki 22 *layers network*, jauh lebih sedikit dan akurat dari pada arsitektur lainnya seperti Alexnet[24].



Gambar 2.4 Konsep *Inception Module*[25]

GoogLeNet lebih kompleks karena memiliki *network* yang bercabang. Pada Gambar 2.4 menunjukan bahwa konsep *inception module* adalah melakukan beberapa konvolusi dengan ukuran yang berbeda lalu menggabungkan semua hasilnya menjadi satu. Ide yang mendasari *inception module* adalah ukuran *filter* yang berbeda, akan menangani objek yang lebih baik sehingga semua filter dapat dipelajari[25].

2.7 *CONFUSION MATRIX*

Confusion matrix pada dasarnya memberikan informasi dari hasil perbandingan antara hasil klasifikasi yang dibuat dengan hasil sebenarnya dalam bentuk matriks[26]. Dari confusion matrix dapat dihitung berbagai macam parameter performansi sistem sehingga bisa diketahui seberapa baik model sistem dirancang dalam menjalankan tugas klasifikasi.

Pada persamaan 2.1, 2.2, dan 2.3 terdapat istilah-istilah yang digunakan dalam konteks evaluasi performa sistem klasifikasi atau pengujian diagnostik, terutama dalam hal deteksi atau prediksi hasil positif atau negatif dari suatu kondisi atau peristiwa. Istilah-istilah tersebut adalah:

1. *True Positive* (TP): Merujuk pada kasus di mana sistem klasifikasi atau pengujian diagnostik secara benar mengidentifikasi hasil sebagai positif (benar positif). Artinya, sistem berhasil mendeteksi atau memprediksi bahwa kondisi atau peristiwa tersebut memang ada, dan prediksi positifnya tepat.

2. *True Negative* (TN): Merujuk pada kasus di mana sistem klasifikasi atau pengujian diagnostik secara benar mengidentifikasi hasil sebagai negatif (benar negatif). Artinya, sistem berhasil mengklasifikasikan atau memprediksi bahwa kondisi atau peristiwa tersebut memang tidak ada, dan prediksi negatifnya tepat.
3. *False Negative* (FN): Merujuk pada kasus di mana sistem klasifikasi atau pengujian diagnostik salah mengidentifikasi hasil sebagai negatif padahal seharusnya positif. Artinya, sistem gagal mendeteksi atau memprediksi bahwa kondisi atau peristiwa sebenarnya ada, tetapi sistem mengklasifikasikannya sebagai negatif.
4. *False Positive* (FP): Merujuk pada kasus di mana sistem klasifikasi atau pengujian diagnostik salah mengidentifikasi hasil sebagai positif padahal seharusnya negatif. Artinya, sistem keliru memprediksi bahwa kondisi atau peristiwa sebenarnya tidak ada, tetapi sistem mengklasifikasikannya sebagai positif.

Confusion matrix dapat digunakan untuk menghitung berbagai *performance metrics* untuk mengukur kinerja model yang telah dibuat[26]. Berikut penjelasanya:

1. *Accuracy*

   Akurasi digunakan untuk mengukur seberapa baik model dalam melakukan prediksi dengan benar. Lebih tinggi angka akurasi, lebih baik performa modelnya. Maka, *accuracy* merupakan rasio prediksi benar (positif dan negatif) dengan keseluruhan data. Dengan kata lain, *accuracy* merupakan tingkat kedekatan nilai prediksi dengan nilai aktual (sebenarnya). Nilai *accuracy* dapat diperoleh dengan persamaan (2.1).

$$Akurasi = \frac{TP + TN}{TP + TN + FN + FP} \text{ x } 100\% \quad (2.3)$$

   Keterangan:

   TP = Diagnostik secara benar mengidentifikasi hasil sebagai negatif.

   TN = Diagnostik secara benar mengidentifikasi hasil sebagai negatif.

   FP = Diagnostik salah mengidentifikasi hasil sebagai negatif padahal seharusnya positif.

FN = Diagnostik salah mengidentifikasi hasil sebagai positif padahal seharusnya negatif.

2. *Precission*

   Presisi mengukur seberapa baik model dalam mengidentifikasi positif secara benar dari semua prediksi positif yang dibuat. Maka, *precision* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif. Dari semua kelas positif yang telah di prediksi dengan benar, berapa banyak data yang benar-benar positif. Nilai *precision* dapat diperoleh dengan persamaan (2.4).

$$Precission = \frac{TP}{TP + FP} \text{ x } 100\% \quad (2.4)$$

   Keterangan:

   TP = Diagnostik secara benar mengidentifikasi hasil sebagai negatif.

   FP = Diagnostik salah mengidentifikasi hasil sebagai negatif padahal seharusnya positif.

3. *Recall*

   Recall, juga dikenal sebagai Sensitivitas, mengukur seberapa baik model dalam mengidentifikasi semua kasus positif yang ada. Maka, *recall* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif. Nilai *recall* dapat diperoleh dengan persamaan (2.5).

$$Recall = \frac{TP}{TP + FN} \text{ x } 100\% \quad (2.5)$$

   Keterangan:

   TP = Diagnostik secara benar mengidentifikasi hasil sebagai negatif.

   FN = Diagnostik salah mengidentifikasi hasil sebagai positif padahal seharusnya negatif.

4. F-1 *Score*

F-1 *score* merupakan *harmonic mean* dari presisi dan *recall*, memberikan gambaran yang lebih baik tentang keseimbangan antara keduanya[27]. Nilai *recall* dapat diperoleh dengan persamaan (2.6).

$$F-1\ Score = \frac{2 \text{ x } Recall \text{ x } Precission}{Recall\ +\ Precission} \tag{2.6}$$

5. *Loss function*

Loss function digunakan dalam proses pelatihan model untuk mengukur kesalahan prediksi model. Fungsi kerugian yang digunakan dalam penelitian yaitu *Mean Squared Error* (juga disebut kerugian L2) adalah preferensi hampir setiap ilmuwan data dalam hal fungsi kerugian untuk *regresi*. *Mean Squared Error* adalah rata-rata selisih kuadrat antara nilai aktual dan prediksi[28]. Nilai *Loss function* dapat diperoleh dengan persamaan (2.7).

$$MSE = \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - \hat{Y}_i \right)^2 \tag{2.7}$$

Keterangan:

Ŷi = Nilai prediksi

Yi = Titik data

N = Jumlah total titik data dalam kumpulan data