Deteksi dan Klasifikasi Tingkat Keparahan Jerawat: Perbandingan Metode *You Only Look Once*

http://dx.doi.org/10.28932/jutisi.v10i3.9414

Riwayat Artikel
Received: 12 Juli 2024 | *Final Revision*: 30 Agustus 2024 | *Accepted*: 15 September 2024

Giezka Veby Agustin[✉ #1], Mewati Ayub[*2], Swat Lie Liliawati.[#3]

[#] *Program Magister Ilmu Komputer, Universitas Kristen Maranatha*
Jl. Prof. drg. Surya Sumantri, M.P.H. No. 65 Bandung-40164, Indonesia
[1]2279001@maranatha.ac.id
[2]mewati.ayub@maranatha.ac.id
[3]swat.ll@maranatha.ac.id
[✉]Corresponding author: 2279001@maranatha.ac.id

Abstrak **— Jerawat (*Acne vulgaris*) adalah salah satu penyakit kulit yang paling umum terjadi, terutama pada wajah. Diagnosa dan penanganan yang tepat penting untuk hasil perawatan yang maksimal dan meningkatkan keakuratan deteksi serta klasifikasi tingkat keparahan jerawat. YOLO (*You Only Look Once*) merupakan metode *deep learning* yang digunakan untuk deteksi objek dalam gambar. Penelitian ini membandingkan hasil dan kinerja dari YOLOv5 dan YOLOv8 dalam mendeteksi jerawat pada wajah. Pada penelitian ini juga dilakukan beberapa eksperimen dengan proses *pre-processing* data, ukuran model, hingga penggunaan *hyperparameters* dasar yang berbeda pada kedua model, untuk mengetahui pengaruh serta perbedaan pada model YOLOv5 dan YOLOv8. Dari hasil penelitian yang dilakukan menunjukkan bahwa YOLOv5 secara keseluruhan memiliki nilai performa lebih tinggi dalam mendeteksi jerawat dibanding YOLOv8 yang membutuhkan beberapa nilai *hyperparameter* serta ukuran model yang lebih besar untuk mendapatkan hasil paling optimal. *Hyperparameter* konservatif (dengan nilai atau ukuran yang relatif lebih kecil) pada YOLOv5 berkontribusi pada performa yang lebih baik.**

Kata kunci*—*Deep Learning; Deteksi objek; Klasifikasi; Tingkat keparahan jerawat; YOLO.

Acne Severity Detection and Classification: Comparing You Only Look Once Methods

Abstract **— Acne (Acne vulgaris) is one of the most common skin diseases, especially on the face. Accurate diagnosis and proper treatment are important for optimal care results and improving the accuracy of detection and classification of acne severity. YOLO (You Only Look Once) is a deep learning method used for object detection in images. This study compares the results and performance of YOLOv5 and YOLOv8 in detecting acne on the face. Several experiments were also conducted with data pre-processing, model size, and the use of different basic hyperparameters on both models to understand the impact and differences between YOLOv5 and YOLOv8. The results show that YOLOv5 overall has higher performance in detecting acne compared to YOLOv8, which requires larger hyperparameter values and model sizes to achieve the most optimal results. Conservative hyperparameters (with relatively smaller values or sizes) on YOLOv5 contribute to better performance.**

Keywords— ***Acne Severity Level; Classification; Deep Learning; Object Detection; YOLO.***

I. PENDAHULUAN

Dermatologi adalah salah satu cabang ilmu kedokteran yang berkembang pesat, dan perhatian terhadap perawatan kulit semakin meningkat di kalangan masyarakat. Salah satu masalah kulit yang umum dihadapi oleh banyak orang adalah jerawat. Jerawat (dalam istilah medis dikenal juga dengan *acne vulgaris)* merupakan penyakit kulit umum yang dapat terjadi pada usia remaja hingga dewasa. Penyebab utama dari jerawat ini adalah penyumbatan folikel rambut oleh sebum, sel kulit mati, dan bakteri, yang dapat terjadi terutama di area seperti wajah, dada, leher, bahu, dan punggung [1]. Terdapat dua jenis utama lesi jerawat, yaitu non-inflamasi (komedo putih dan komedo hitam) dan inflamasi (papula, pustula, nodul, dan kistik [1]. Jerawat dan bekas jerawat dapat berdampak negatif selain secara fisik tetapi juga pada aspek psikologis, seperti perasaan malu, ketidakpuasan dengan penampilan, kurangnya kepercayaan diri, dan masalah sosial [2]. Diagnosis dan penanganan jerawat umumnya dilakukan oleh profesional seperti dokter kulit, dengan tingkat keparahan menjadi faktor penentu dalam pemilihan perawatan. Meskipun banyak yang ingin mencari bantuan profesional, keterbatasan jumlah tenaga profesional tersebut dapat menjadi penghambat penanganan yang diperlukan. Penting untuk menilai tingkat keparahan jerawat secara akurat, dan meskipun metode manual dapat digunakan, hal ini memiliki keterbatasan seperti kecepatan, akurasi, dan ketergantungan pada pengalaman serta pengetahuan dari tenaga profesional seperti dokter.

Penerapan teknologi ini pada bidang dermatologi, khususnya dalam mendeteksi dan mengklasifikasikan tingkat keparahan jerawat, dapat menjadi solusi inovatif dan efisien. Ketersediaan *dataset* yang cukup banyak dan kemajuan dalam arsitektur *Neural Networks* telah membuka peluang baru untuk mengembangkan sistem otomatis yang dapat membantu dokter dalam menganalisis dan mengelola masalah kulit seperti jerawat. Pendekatan menggunakan *Deep Learning* telah banyak digunakan dalam berbagai aspek seperti klasifikasi dan deteksi objek dalam kehidupan nyata. Dalam bentuk pemodelan data untuk melatih model, *Deep learning* menggunakan pemodelan data untuk mengatasi berbagai tugas dan masalah termasuk pada bidang medis yang membantu untuk melakukan diagnosa ataupun analisis terhadap gambar-gambar medis [3]. Metode *Deep learning* ini kemudian dapat dimanfaatkan untuk membantu dalam proses diagnosis hingga pemantauan selama perawatan pengobatan jerawat dengan membangun model *Deep Learning* untuk mendeteksi jerawat dan melakukan klasifikasi tingkat keparahan jerawat dari gambar wajah sehingga dapat membantu efisiensi dalam penanganan yang lebih tepat terhadap jerawat ini. Saat ini terdapat berbagai penelitian yang telah dilakukan terkait pemanfaatan algoritma atau metode *Deep Learning* untuk melakukan deteksi jerawat dan juga klasifikasi tingkat keparahan jerawat pada area wajah dari gambar wajah. Penelitian-penelitian tersebut dilakukan dengan *dataset*, pendekatan atau teknik, proses, metode atau algoritma *Deep Learning*, hal-hal yang menjadi fokus penelitian, hingga penggunaan skala penilaian tingkat keparahan jerawat (termasuk faktor-faktor penilaiannya) yang berbeda-beda.

Salah satu metode *Deep learning* yang dapat digunakan untuk kasus ini adalah metode YOLO (*You Only Look Once*) yang merupakan jenis pendekatan dalam pemrosesan gambar dan *computer vision* yang digunakan untuk mendeteksi objek. Deteksi objek merupakan tugas penting dalam *computer vision* yang memungkinkan mesin untuk mendeteksi, mengenali, dan menemukan objek dalam gambar atau video. YOLO adalah algoritma yang cukup populer dan banyak digunakan. YOLO dirancang khusus untuk tugas deteksi objek dalam gambar yang terkenal karena kecepatan dan efisiensinya dalam deteksi objek dalam gambar secara *real-time* [4]. YOLO membedakan dirinya dari pendekatan deteksi objek lainnya dengan kemampuannya untuk menghasilkan prediksi secara cepat dan efisien, membuatnya populer di berbagai aplikasi yang memerlukan deteksi objek *real-time*, seperti kendaraan otonom, pengawasan keamanan, dan pengenalan wajah [4]. YOLO secara simultan memprediksi kotak pembatas (*bounding box*) dan kelas objek dalam gambar dengan menggunakan *deep learning*. YOLO adalah pendeteksi objek yang dirancang sebagai masalah regresi terhadap kotak pembatas yang dipisahkan secara spasial dan probabilitas kelas yang terkait. Sebuah jaringan saraf tunggal memprediksi kotak pembatas dan probabilitas kelas langsung dari gambar secara penuh dalam satu evaluasi. Karena seluruh *pipeline* deteksi adalah jaringan tunggal, *pipeline* ini dapat dioptimalkan dari ujung ke ujung langsung pada kinerja deteksi. Model dasar YOLO dapat memproses gambar secara *real-time* yaitu pada 45 *frame* per detik. YOLO dapat melampaui metode deteksi lainnya, termasuk DPM dan R-CNN, ketika digeneralisasi dari gambar alami ke domain lain seperti karya seni [5]. Pada tahun 2015, Redmon *et al.* [5] memperkenalkan versi pertama YOLO. Sejak saat itu YOLO telah mengeluarkan beberapa versi hingga saat ini yang dikembangkan oleh pengembang yang berbeda-beda. Beberapa penelitian terkait telah menggunakan YOLOv5 ini untuk melakukan deteksi jerawat. YOLOv8 dan YOLOv5 adalah dua versi YOLO yang dikembangkan oleh *Ultralytics* dengan menggunakan *framework* PyTorch. YOLOv8 adalah tambahan terbaru dalam keluarga YOLO yang dikembangkan oleh *Ultraytics*, yang membangun kesuksesan versi sebelumnya dan memperkenalkan fitur dan perbaikan baru untuk meningkatkan kinerja dan fleksibilitas [6]. Pada sisi lain YOLOv5 terkenal dan banyak digunakan karena kecepatan, kesederhanaan, dan tingkat akurasinya [6]. Keputusan untuk membandingkan kedua versi ini didasarkan pada kemajuan mutakhir dari YOLOv8 dan presisi serta efisiensi yang terbukti dari YOLOv5 dalam industri *computer vision*. Dengan memusatkan analisis penelitian ini pada dua versi YOLO yang dikembangkan oleh *Ultralytics*, diharapkan dapat diperoleh pemahaman yang lebih mendalam tentang kemampuan dan kontribusinya masing-masing.

Penelitian ini berfokus pada perbandingan hasil kinerja seperti *precision*, *recall*, mAP dan *F1-score* antara versi dari YOLO yang berbeda yaitu YOLOv5 dan YOLOv8 dalam tugas mendeteksi jerawat pada area wajah. Berdasarkan jumlah jerawat terdeteksi akan diklasifikasikan tingkat keparahan jerawatnya. Penilaian tingkat keparahan jerawat yang digunakan adalah berdasarkan penelitian terkait dari *dataset* ACNE04 oleh Wu *et al.* [7] yang akan digunakan pada penelitian kali ini. *Dataset* ACNE04 mencakup anotasi jumlah lesi lokal dan tingkat keparahan jerawat global berdasarkan Kriteria Hayashi. Dari penelitian yang dilakukan oleh Hayashi *et al.* [8], klasifikasi dermatologis berkorelasi dengan jumlah luka inflamasi (papula dan pustula), tetapi tidak dengan jumlah komedo. Perbandingan ini akan memberikan wawasan tentang bagaimana kedua arsitektur YOLO versi berbeda ini mendeteksi objek seperti jerawat pada wajah. Selain itu, akan dilakukan beberapa eksperimen menggunakan proses *pre-processing* yang berbeda, ukuran model, *hyperparameter*s dasar berbeda untuk mengetahui pengaruh hal-hal tersebut terhadap performa model YOLOv5 dan YOLOv8. Dengan cara ini, penelitian dapat mengevaluasi keunggulan dan kelemahan masing-masing dalam konteks deteksi dan klasifikasi tingkat keparahan jerawat pada wajah. Kedua versi YOLO ini memiliki kelebihan masing-masing. Dengan membandingkan keduanya, penelitian ini dapat mengidentifikasi metode yang lebih cocok serta kelebihan dan kekurangan jika menggunakan versi YOLO yang berbeda. Pertimbangan dari hasil perbandingan ini dapat berkaitan dengan kebutuhan komputasi dan sumber daya yang tersedia serta hasil dari kinerja keduanya.

II. METODE PENELITIAN

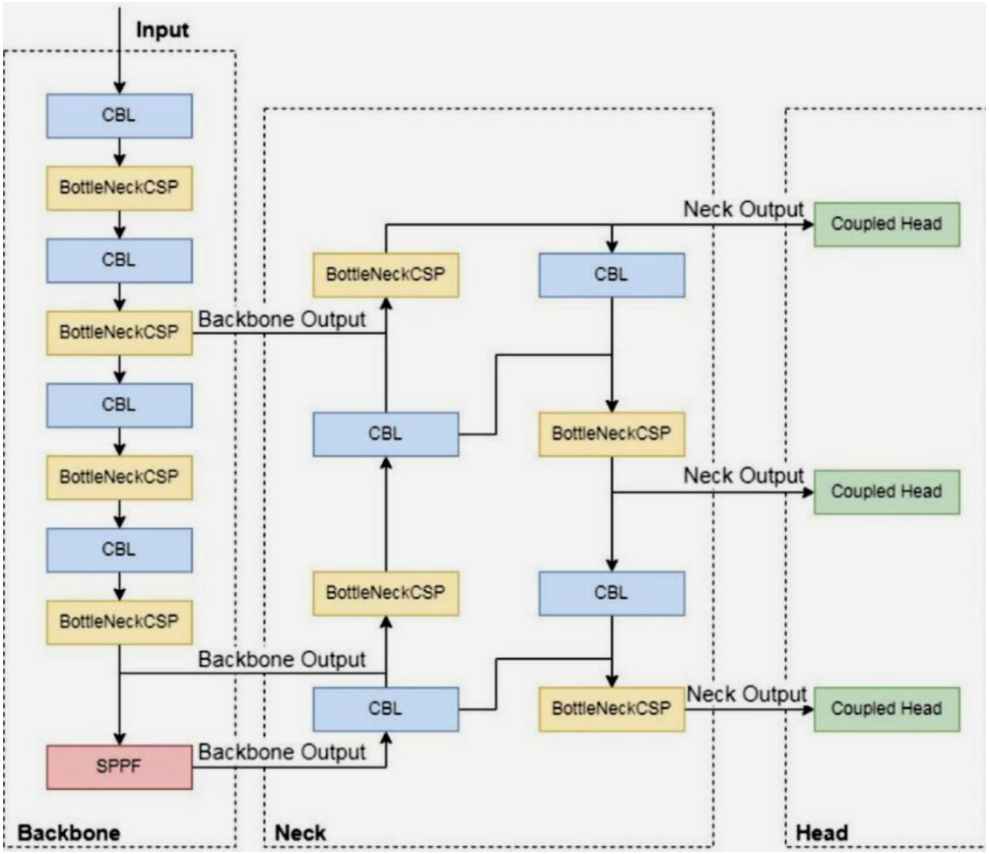
A. *Studi Literatur*

Berbagai penelitian telah dilakukan untuk menggunakan algoritma atau metode *Deep Learning* dalam mendeteksi jerawat dan mengklasifikasikan tingkat keparahannya pada gambar wajah. Penelitian-penelitian ini menggunakan berbagai *dataset*, pendekatan, teknik, metode, dan skala penilaian yang berbeda. Beberapa penelitian fokus pada penggunaan detektor objek seperti YOLO. Sangha *et al.* [9] melatih model menggunakan dataset ACNE04 dengan 1.457 gambar dan 18.983 *bounding box* jerawat, membagi *dataset* menjadi 80% untuk pelatihan dan 20% untuk pengujian. YOLOv5 digunakan dan tercapailah nilai *mean average precision* (mAP) sebesar 37.97 untuk deteksi *single class* dan 26.50 untuk *multi-class*. Model ini menunjukkan *precision* tinggi (96.45%), *recall* (94.73%), dan *true positive rate* yang tinggi (93.59%) untuk deteksi kelas tunggal. Hao Wen *et al.* [10] mengembangkan dan membandingkan model *deep learning* untuk menemukan lesi jerawat pada gambar wajah menggunakan *dataset* ACNE04 dengan 1.222 gambar untuk pelatihan. Pada penelitian tersebut metode seperti SSD, YOLOv4, dan Faster R-CNN diterapkan, sehingga menemukan bahwa Faster R-CNN dengan *backbone* ResNet101 memiliki performa terbaik dengan *average precision* (AP) 0.536 dan *Mean Absolute Error* (MAE) 3.49. Model ini efektif dalam memprediksi tingkat keparahan rendah dan menengah. Zhang *et al.* [11] menyajikan model *ensemble neural networks* yang terdiri dari modul klasifikasi dan lokalisasi menggunakan dataset ACNE04. Dengan ResNet50 dan YOLOv5, model ini mencapai akurasi keparahan 99.31%, akurasi jumlah 84.60%, dan RMSE terendah 2.17, mampu memprediksi tingkat keparahan, jumlah, dan posisi jerawat dengan akurasi tinggi. Nethravathi *et al.* [12] mengembangkan aplikasi *mobile* menggunakan YOLO untuk mengidentifikasi berbagai bentuk jerawat dari 2.000 gambar wajah jerawat. Model ini mencapai nilai mAP 85% dan menggunakan skala *Global Acne Grading System* (GAGS) untuk mengevaluasi keparahan jerawat. Secara keseluruhan, penelitian-penelitian ini menunjukkan upaya besar dalam mengembangkan model *Deep Learning* untuk deteksi dan penilaian tingkat keparahan jerawat berdasarkan gambar wajah dengan berbagai *dataset*. Berdasarkan penelitian terdahulu tersebut, YOLO terbukti dapat permasalah pendeteksian jerawat dan klasifikasi tingkat keparahannya dengan performa yang cukup memuaskan. Penelitian ini membawa kebaruan dengan fokus pada perbandingan performa dua versi model YOLO berbeda, yaitu YOLOv5 dan YOLOv8, khususnya dalam mendeteksi objek seperti jerawat ini yang kemudian hasilnya dapat digunakan untuk mengklasifikasikan tingkat keparahannya.

B. *Konsep Dasar dan Teori*

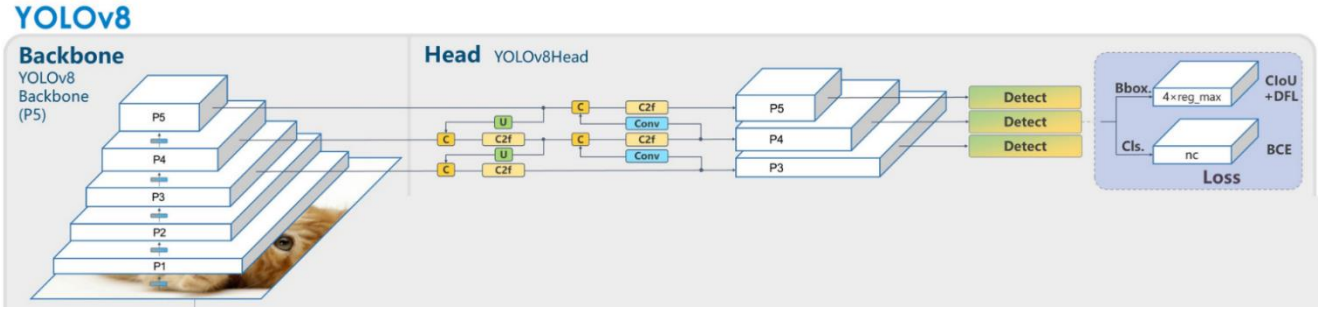
YOLO (*You Only Look Once*) merupakan salah satu arsitektur *Deep Learning* yang berfokus pada objek deteksi menggunakan metode *bounding box*. YOLO pertama kali diperkenalkan oleh Joseph Redmon dan rekan-rekannya pada tahun 2015 [5]. YOLO menggunakan arsitektur R-CNN (*Regional Convolutional Neural Network*) dan *bounding box* untuk melokalisasi dan mengklasifikasi bagian-bagian gambar yang dipilih secara acak. Bagian-bagian dengan nilai tertinggi saat lokalisasi disimpan sebagai hasil deteksi [5]. Telah terjadi perubahan pada arsitektur standar dari susunan model pada YOLOv5. YOLOv5 dibagi menjadi tiga komponen yaitu, *backbone*, *neck*, dan *head*. *Backbone* dari YOLOv5 adalah Darknet 53. Darknet 53 merupakan arsitektur jaringan baru yang berfokus pada ekstraksi fitur yang ditandai dengan jendela penyaring kecil dan koneksi residual. *Neck* dari YOLOv5 berfungsi sebagai penghubung antara *backbone* dan *head*. *Neck* YOLOv5 berfungsi untuk mengumpulkan dan memperhalus fitur-fitur yang diekstraksi oleh *backbone*, dengan fokus pada peningkatan informasi spasial dan semantik pada berbagai skala. *Head* dari YOLOv5 terdiri dari tiga cabang, masing-masing memprediksi fitur pada skala yang berbeda. Setiap head menghasilkan *bounding box*, probabilitas kelas, dan skor *confidence*. Jaringan

tersebut menggunakan *Non-maximum Suppression* (NMS) untuk menyaring *bounding box* yang tumpang tindih. Dapat dilihat pada Gambar 1 merupakan struktur dari YOLOv5 [6].



Gambar 1. Struktur YOLOv5 [6]

YOLOv8 memperkenalkan perbaikan dalam bentuk arsitektur neural network baru. Neural network yang diimplementasikan adalah Feature Pyramid Network (FPN) dan Path Aggregation Network (PAN), serta alat pelabelan baru yang menyederhanakan proses anotasi. Alat pelabelan ini memiliki beberapa fitur berguna, seperti pelabelan otomatis, pelabelan dengan shortcut, dan hotkey yang dapat disesuaikan. Kombinasi fitur-fitur ini memudahkan anotasi gambar untuk pelatihan model. FPN bekerja dengan secara bertahap mengurangi resolusi spasial dari gambar input sambil meningkatkan jumlah saluran fitur. Ini menghasilkan peta fitur yang mampu mendeteksi objek pada skala dan resolusi yang berbeda. Di sisi lain, arsitektur PAN dapat menggabungkan fitur dari berbagai tingkat jaringan melalui skip connections. Akibatnya, jaringan dapat menangkap fitur lebih efektif pada berbagai skala dan resolusi, yang sangat penting untuk mendeteksi objek dengan ukuran dan bentuk yang berbeda secara akurat [6]. Arsitektur dari YOLOv8 dapat dilihat pada Gambar 2. Pada Tabel 1 merupakan perbandingan antara YOLOv5 dan YOLOv8 dalam beberapa aspek termasuk arsitektur.



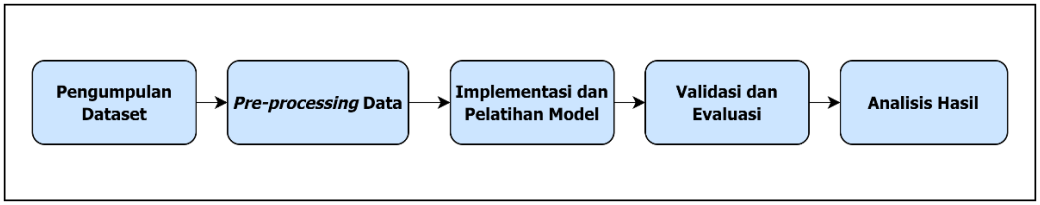
Gambar 2. Arsitektur YOLOv8 [6]

TABEL 1.
PERBANDINGAN YOLOV5 DAN YOLOV8

Aspek	YOLOv5	YOLOv8
Arsitektur	***Backbone***:CSP-Darknet53: Modifikasi dari arsitektur Darknet sebelumnya.	***Backbone:*** *Custom* CSPDarknet53: Menggunakan koneksi *cross-stage* parsial untuk aliran informasi yang lebih baik.
	Neck: SPPF (*Spatial Pyramid Pooling - Fast*) dan CSP-PAN (*Path Aggregation Network*).	***Neck***: C2f *module*: Pengganti FPN tradisional, menggabungkan fitur semantik tinggi dengan informasi spasial rendah untuk akurasi yang lebih baik, terutama pada objek kecil.
	Head: YOLOv3 *Head*: Menghasilkan *output* akhir dari prediksi.	***Head***: Beberapa modul deteksi: Memperkirakan *bounding boxes*, skor objektivitas, dan probabilitas kelas, yang digabungkan untuk menghasilkan deteksi akhir.
Jumlah Parameter	YOLOv5s : 7.2M YOLOv5m: 21.2M YOLOv5l : 46.5M YOLOv5x: 86.7M	YOLOv8s : 11.2M YOLOv8m: 25.9M YOLOv8l : 43.7M YOLOv8x: 68.2M
Kedalaman *Hidden Layer*	Berkisar antara 40 hingga 60 *layers*	Berkisar antara 50 hingga 70 *layers*
Fitur Utama	*Anchor-free* untuk deteksi fleksibel. Keseimbangan kecepatan dan akurasi. Menyediakan berbagai model *pre-trained*.	*Backbone* dan *neck* canggih. *Anchor-free* untuk akurasi tinggi. Keseimbangan optimal kecepatan dan akurasi. Menyediakan berbagai model *pre-trained*.
Strategi Pelatihan	***Multiscale Training***: *Rescaling* acak gambar selama pelatihan. ***Mixed Precision Training***: Mengurangi penggunaan memori dan meningkatkan kecepatan. ***Hyperparameter Evolution***: Menyempurnakan *hyperparameter* secara otomatis.	***Training Resolutions***: Menggunakan resolusi pelatihan yang berbeda. ***Mosaic Data Augmentation***: Menggabungkan beberapa gambar untuk meningkatkan generalisasi.

C. *Alur Penelitian*

Penelitian ini mengadopsi desain penelitian eksperimental komparatif untuk menyelidiki perbedaan kinerja antara dua model dari dua versi YOLO yang berbeda, yaitu YOLOv5 dan YOLOv8, dalam menangani tugas deteksi jerawat yang kemudian jumlah lesi jerawatnya akan digunakan untuk mengklasifikasikan tingkat keparahan jerawat pada satu *dataset* yang sama. Desain eksperimental ini dipilih untuk memberikan pemahaman yang mendalam tentang kemampuan kedua model dari arsitektur berbeda dalam mendeteksi objek berupa jerawat yang ada pada gambar wajah serta menghitung jumlah objek terdeteksi tersebut. Pada tahap pengumpulan data, data yang dibutuhkan untuk penelitian dikumpulkan, yang mungkin berupa gambar wajah yang telah dilengkapi dengan anotasi jerawat. Data yang telah terkumpul kemudian diproses atau dipersiapkan dalam tahap *pre-processing* data, untuk meningkatkan kualitas *dataset* untuk menghasilkan performa model yang optimal. Selanjutnya, dilakukan implementasi dan pelatihan model deteksi jerawat seperti YOLOv5 dan YOLOv8, menggunakan data yang telah dipersiapkan, dan dilatih dengan menggunakan algoritma pelatihan yang sesuai untuk mendapatkan kemampuan deteksi yang optimal. Setelah pelatihan selesai, model dievaluasi dan divalidasi untuk mengukur kinerja dan keakuratannya dalam deteksi jerawat pada wajah. Evaluasi dilakukan menggunakan metrik-metrik yang relevan, seperti *F1-score*, *precision*, *recall* dan mAP. Pengujian juga dilakukan atau diterapkan saat implementasi hasil deteksi untuk mengklasifikasikan tingkat keparahannya berdasarkan kriteria yang telah ditentukan. Hal tersebut dilakukan untuk melihat, bagaimana model yang telah dilatih untuk deteksi dapat digunakan untuk mengklasifikasikan tingkat keparahan jerawat. Terakhir, hasil dari evaluasi dan validasi dianalisis untuk mendapatkan pemahaman yang mendalam tentang kinerja model deteksi, yang akan membantu dalam memahami kekuatan dan kelemahan model yang digunakan. Pada Gambar 3, merupakan diagram alur dari tahapan pelaksanaan penelitian.



Gambar 3. Alur Penelitian

D. *Dataset*

Dataset yang akan digunakan dalam penelitian ini adalah ACNE04 yang memiliki kumpulan gambar wajah dengan jerawat. *Dataset* terdiri dari 1.457 gambar wajah dengan jerawat yang telah diberi label sesuai dengan tingkat keparahannya yaitu dari yang paling ringan hingga paling parah (*level* 0 – *level* 3). Anotasi tersebut disediakan dalam format Pascal VOC untuk masing-masing gambarnya yang berisi informasi mengenai lokasi dari jerawat pada wajah yang telah ditandai oleh dermatologis professional, sehingga pada penelitian ini tidak akan dilakukan pembuatan label dengan *bounding box* atau anotasi untuk jerawat. *Dataset* yang dapat diakses secara publik ini dianggap relevan karena tidak hanya mencakup gambar-gambar jerawat, tetapi juga anotasi jumlah lesi lokal dan tingkat keparahan jerawat global. Selain itu, *dataset* ini telah digunakan oleh beberapa penelitian terkait lainnya yang menambah validitas dan kredibilitasnya. Penggunaan *dataset* yang sama memungkinkan perbandingan hasil dengan studi-studi sebelumnya, membantu mengidentifikasi kelebihan dan kekurangan metode yang digunakan.

Dataset ACNE04 yang dihasilkan dari penelitian yang dilakukan oleh Wu *et al.* [7] mencakup anotasi jumlah lesi lokal dan tingkat keparahan jerawat global berdasarkan Kriteria Hayashi [8]. Penggunaan Kriteria Hayashi pada penelitian ini didasarkan pada *dataset* yang digunakan yang menggunakan Kriteria Hayashi ini untuk menetapkan tingkat keparahan jerawat. Dalam penelitian yang dilakukan Hayashi *et al.* [8], beberapa langkah dilakukan untuk membuat kriteria penilaian keparahan jerawat berbasis bukti yang dapat diterima oleh sebagian besar dokter kulit. Eksperimen menunjukkan bahwa klasifikasi global oleh dokter kulit yang dikonsultasikan sesuai dengan penilaian ahli menggunakan foto dan penghitungan erupsi inflamasi, mendukung penggunaan penilaian global berbasis foto. Foto standar membantu menyesuaikan dan meningkatkan akurasi klasifikasi. Metode penilaian yang tidak konsisten menjadi tantangan, karena beberapa penelitian menggunakan hingga 1.000 foto yang dikategorikan oleh ahli tanpa definisi yang jelas. Jumlah komedo tidak berkorelasi dengan perkiraan keparahan global, sedangkan kista dan nodul berkorelasi meskipun kurang umum. Akibatnya, pada Kriteria Hayashi ini paling baik digunakan khusus hanya untuk kasus dengan papula dan pustula. Komedo, kista, dan nodul memerlukan kriteria penilaian terpisah (tidak digunakan pada penilaian dengan Kriteria Hayashi). Papula dan pustula adalah dua jenis lesi inflamasi yang sering ditemukan pada kulit berjerawat. Papula adalah benjolan kecil dengan diameter kurang dari 5 mm, berwarna merah muda, tidak mengandung nanah, dan biasanya terasa hangat serta sakit saat disentuh. Pustula, dengan ukuran kurang dari 5 mm dalam diameter, memiliki dasar berwarna merah dengan pusat yang kekuningan atau keputihan, mengandung nanah di tengahnya, dan juga terasa hangat serta sakit. Kedua lesi ini menunjukkan adanya inflamasi pada kulit dan merupakan tanda umum dari kondisi jerawat [13]. Berdasarkan penelitian yang dilakukan tersebut maka, Kriteria Hayashi menetapkan bahwa sebagian besar dokter kulit memiliki pemahaman yang serupa tentang keparahan jerawat, yang diformalkan menjadi kriteria (jumlah papula dan pustula pada setengah wajah): 0-5, *Mild* ; 6-20, *Moderate*; 21-50, *Severe*; dan lebih dari 50, *Very Severe.* Pada Gambar 4 merupakan beberapa gambar wajah berjerawat di setiap tingkatan keparahannya dari *dataset* ACNE04.

Gambar 4. Tingkat Keparahan Jerawat pada *Dataset* ACNE04 [7]

E. *Pre-processing Data*

Persiapan data atau *pre-processing* adalah langkah penting dalam penelitian untuk meningkatkan kualitas *dataset*, sehingga model yang dibangun dapat lebih baik dalam mengenali jerawat pada kulit wajah. Penelitian ini menggunakan dataset ACNE04, yang terdiri dari 1.457 gambar wajah dengan jerawat yang diberi label sesuai tingkat keparahannya (*level*

0 – *level* 3). Langkah pertama adalah konversi anotasi gambar, di mana *dataset* memiliki anotasi *bounding box* dalam format Pascal VOC (*file* XML) yang menunjukkan lokasi jerawat pada wajah, dan perlu diubah ke format pelabelan YOLO (*file* TXT) yang berisi kelas objek, koordinat, tinggi, dan lebar kotak pembatas dengan koordinat dinormalisasi dari 0 hingga 1. Langkah kedua adalah *pre-processing* gambar yang meliputi *cleaning data, cropping*, pemberian *padding*, dan *resize* gambar untuk menghasilkan *input* data yang optimal bagi model. Meskipun YOLO dapat menerima gambar dengan rasio atau ukuran yang berbeda, *resize* gambar sebelum pelatihan dilakukan untuk mempercepat proses dan menyesuaikan label anotasi dengan perubahan gambar. *Resize* yang dilakukan ini tidak memotong gambar, jadi tidak ada bagian yang hilang. Gambar diubah ukurannya menjadi ukuran target, yang berarti bisa diperbesar atau diperkecil tergantung pada ukuran asli gambar. *Resize* pada penelitian ini menggunakan OpenCV yang adalah sebuah pustaka *computer vision* yang populer dengan berbagai fungsi untuk pemrosesan gambar. Metode dalam *resize* gambar ini adalah interpolasi bilinear yang merupakan metode interpolasi dalam pemrosesan gambar yang digunakan untuk memperhalus atau memperbesar gambar. Metode ini bekerja dengan mengambil rata-rata tertimbang dari empat piksel terdekat di sekitar piksel yang akan dihitung. Proses ini mempertimbangkan posisi relatif dari piksel baru terhadap empat piksel terdekat untuk menghasilkan nilai piksel yang lebih halus dan lebih akurat.

Pada penelitian ini dieksplorasi beberapa metode *pre-processing* data untuk mengetahui bagaimana data *input* yang digunakan akan mempengaruhi performa model untuk YOLOv5 dan YOLOv8, serta apa perbedaannya antara kedua model tersebut dalam skenario hasil data *pre-processing* berbeda tersebut. Oleh karena itu, dilakukan beberapa proses *pre-processing* data berbeda terhadap *dataset* yang sama ini. Hasil dari *pre-processing* yang berbeda ini diberi nama yaitu, Dataset 1, Dataset 2, dan Dataset 3. Pada Dataset 1 ini hanya dilakukan satu metode *pre-processing* yaitu adalah melakukan *resize* gambar menjadi 640x640 piksel sehingga rasio gambar menjadi *square* atau 1:1. Pemilihan ukuran tersebut didasari oleh *default image size* dari YOLOv5 dan YOLOv8 yang merupakan 640. Gambar pada *dataset* ACNE04 ini memiliki ukuran serta rasio yang berbeda-beda, hal ini dilakukan untuk membantu pelatihan agar seragam dan mengurangi kompleksitas komputasi. Hal tersebut menyebabkan beberapa gambar menjadi terdistorsi, terlebih untuk gambar yang rasionya adalah *landscape.* Pada Dataset 2, gambar juga di *resize* menjadi ukuran 640 x 640 piksel (rasio 1:1 atau *square*). Perbedaan dengan *pre-processing* sebelumnya adalah data diseleksi secara manual setelah dilakukan *resize*, yaitu dengan membuang beberapa gambar berkualitas terlalu rendah (dibawah 100kb), buram, atau jerawat tidak terlihat jelas. Sehingga menghasilkan 1.358 data yang akan digunakan. Pada Dataset 3 ini gambar juga di *resize* tetapi kali ini menjadi ukuran yang lebih besar yaitu 1280 x 1280 piksel (rasio 1:1 atau *square*). Peningkatan ukuran ini dilakukan untuk mendapatkan manfaat dari pelatihan dengan gambar resolusi lebih tinggi karena objek yang dideteksi adalah termasuk objek kecil. Model YOLO dirancang untuk fleksibilitas dalam ukuran *input*, tetapi menggunakan ukuran yang lebih besar seperti 1280 dapat meningkatkan performa deteksi, terutama untuk *dataset* dengan banyak objek kecil. Seleksi data juga dilakukan pada tahap ini setelah *resize* untuk menghasilkan data dengan kualitas gambar yang baik. Hal yang dilakukan adalah membuang gambar yang berkualitas terlalu rendah (dibawah 100kb), buram, terdapat objek yang menutupi jerawat (seperti teks), atau jerawat tidak terlihat jelas. Namun kali ini langkah tambahannya adalah pencegahan gambar terdistorsi. Gambar berorientasi *landscape* diolah agar tetap jelas dengan memotong gambar dan hanya menyisakan gambar wajah saja dan membuang bagian lainnya. Selain itu, pada gambar *landscape* yang terlalu dekat dan miring diberikan *padding* pada bagian samping atau atas dan bawah untuk menjaga kualitas gambar yang jelas pada rasio 1:1. Dihasilkan 1163 data yang akan digunakan.

Langkah ketiga adalah pembagian dataset menjadi subset untuk pelatihan (70-80%), validasi (10-15%), dan pengujian (10-15%). Pembagian ini memastikan model dapat dievaluasi secara adil dan dapat melakukan generalisasi terhadap data yang belum pernah dilihat sebelumnya. Struktur folder khusus digunakan untuk memisahkan gambar dan *file* anotasi untuk setiap subset, dengan konfigurasi data disimpan dalam *file* data.yaml yang berisi *path* direktori data pelatihan dan validasi, jumlah kelas, dan nama setiap kelas. Penelitian ini menekankan pentingnya *pre-processing* data untuk mengoptimalkan kinerja model YOLO dalam mendeteksi jerawat pada kulit wajah.

F. *Implementasi dan Pelatihan Data*

Model YOLO akan menjalani tahap pelatihan menggunakan dataset yang telah dipersiapkan. Evaluasi kinerja model akan dilakukan pada set validasi untuk memastikan kemampuannya dalam mendeteksi jerawat secara efektif. Langkah-langkah ini akan memastikan bahwa dataset siap digunakan dalam arsitektur YOLO. Sebelum pelatihan dimulai hal yang harus dipersiapkan adalah file YAML yang berisi informasi mengenai direktori dari dataset yang akan digunakan. Pelatihan model YOLO melibatkan beberapa langkah penting. Hal yang harus dilakukan sebelum pelatihan adalah melakukan clone repository YOLO dan menyiapkan lingkungan serta memastikan seluruh persyaratan yang diperlukan telah terinstal. YOLOv5 dan YOLOv8 ditulis dalam bahasa pemrograman Python dan menggunakan framework PyTorch, sehingga perlu menginstal dependensi yang diperlukan juga. Pada pelatihan perlu diperhatikan juga konfigurasi hyperparameter yang menentukan hyperparameter pelatihan seperti image size, batch size, jumlah epoch, dan lainnya. Kedua, pemilihan backbone, di mana YOLO menyediakan beberapa arsitektur *backbone* atau variasi yang dapat dipilih sesuai dengan kebutuhan dan kapasitas komputasi. Selama pelatihan, model belajar mengenali objek-objek dalam gambar dengan meminimalkan loss function yang telah ditentukan. Pada penelitian ini akan dilakukan beberapa kali percobaan pelatihan yaitu seperti dengan variasi model

YOLOv5 dan YOLOv8 serta beberapa hyperparameter dasar berbeda untuk melihat perbandingan dengan skenario optimasi hasil yang berbeda tersebut.

G. *Validasi dan Evaluasi*

Setelah tahap pelatihan, model YOLO divalidasi dan dievaluasi menggunakan *dataset* yang terpisah dari *dataset* pelatihan. Secara otomatis, YOLO menjalankan validasi setelah pelatihan selesai, meskipun parameter validasi dapat diubah dan dijalankan terpisah. Validasi ini menghasilkan matrik evaluasi seperti *precision*, *recall*, mAP-50, dan mAP50-95 secara otomatis melalui skrip validasi dari YOLO. Dalam kasus deteksi jerawat, akurasi tidak digunakan sebagai metrik karena kurang tepat untuk menangkap kompleksitas tugas ini, seperti mendeteksi banyak jerawat dalam satu gambar dengan berbagai ukuran dan lokasi. Metrik seperti mAP, *precision*, *recall*, dan *F1-score* dianggap lebih relevan karena mempertimbangkan keseimbangan antara *false positives* dan *false negatives*, memastikan bahwa semua jerawat terdeteksi dengan benar tanpa mendeteksi area kulit normal sebagai jerawat. *F1-Score* dihitung dari nilai *precision* dan *recall*. Metrik-metrik ini digunakan untuk mengevaluasi kemampuan model dalam mendeteksi jerawat. Pengujian tambahan dilakukan menggunakan beberapa data gambar dari dataset pengujian untuk menguji kemampuan model terhadap data yang belum dikenali. Hasil evaluasi memberikan gambaran holistik tentang kinerja model di luar dataset pelatihan. Persamaan untuk menghitung metrik dijelaskan, meskipun tidak semua metrik dihitung secara manual dalam penelitian ini. Berikut ini adalah penjelasan dan cara mendapatkan masing-masing metrik :

Precision ini digunakan untuk mengukur ketepatan prediksi model. Ini merupakan rasio antara jumlah prediksi yang benar (*True Positives*) dengan jumlah semua prediksi yang diidentifikasi sebagai positif (*True Positives* + *False Positives*). Pada penelitian ini *precision* digunakan untuk memastikan bahwa setiap deteksi jerawat yang dilakukan oleh model adalah benar-benar jerawat. Persamaan (1) digunakan untuk menghitung nilai *precision* :

$$precision = \frac{TP}{TP + FP} \tag{1}$$

Recall mengukur kemampuan model untuk menemukan semua *instance* positif. *Recall* merupakan rasio antara jumlah prediksi yang benar (*True Positives*) dengan jumlah semua *instance* yang sebenarnya positif (*True Positives* + *False Negatives*). Pada penelitian ini *recall* memastikan bahwa model mendeteksi sebanyak mungkin jerawat yang ada dalam gambar, meskipun itu berarti menerima beberapa kesalahan deteksi. Persamaan (2) ini digunakan untuk menghitung nilai *recall* :

$$recall = \frac{TP}{TP + FN} \tag{2}$$

F1-Score merupakan metrik yang menggabungkan *precision* dan *recall* menggunakan rata-rata harmonik. Hal ini adalah nilai keseimbangan antara *precision* dan *recall.* Nilai *F1-Score* dihitung sebagai hasil perkalian *precision* dan *recall*, dibagi oleh jumlah *precision* dan *recall*, dan kemudian dikalikan dua. Persamaan (3) ini digunakan untuk menghitung nilai dari *F1-Score* :

$$F1\ Score = 2\ X \frac{recall \text{ x } precision}{recall + precision} \tag{3}$$

Metrik mAP50 ini digunakan untuk mengukur *mean Average Precision* ketika *Intersection over Union* (IoU) *threshold* ditetapkan pada 0.5. Ini adalah metrik yang mengukur akurasi keseluruhan model deteksi objek. Pada persamaan 3.4, N adalah jumlah kelas objek yang dievaluasi sedangkan AP i adalah *Average Precision* untuk kelas objek ke- i. Persamaan (4) ini digunakan untuk menghitung nilai dari mAP50 :

$$mAP50 = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{4}$$

Metrik mAP50-95 adalah metrik yang *mengukur mean Average Precision* di berbagai *threshold* IoU, mulai dari 0.5 hingga 0.95 dengan interval 0.05. Ini memberikan gambaran yang lebih komprehensif tentang seberapa baik model dapat

mengidentifikasi objek dengan akurat pada berbagai tingkat tumpang tindih. Persamaan (5) ini digunakan untuk menghitung nilai dari mAP50-95 :

$$mAP50.95 = \frac{1}{10}\sum_{k=1}^{10} mAP_{50+k.0,05} \quad (5)$$

H. Klasifikasi Tingkat Keparahan Jerawat

Setelah pelatihan selesai dilakukan maka selanjutnya adalah melakukan klasifikasi tingkat keparahan jerawat dengan mengambil jumlah objek yang disini adalah jerawat yang berhasil terdeteksi. Klasifikasi disini tidak akan menggunakan metode ataupun algoritma *Machine Learning.* Model yang telah dilatih akan dimuat dan digunakan untuk mendeteksi jerawat pada sebuah gambar. Hasil deteksi tersebut kemudian dimasukkan kedalam sebuah fungsi yang akan menentukan tingkat keparahan jerawat.

III. HASIL DAN PEMBAHASAN

Pada penelitian ini, model YOLO yang digunakan adalah versi *small* dari YOLOv5 dan YOLOv8 karena lebih ringan dan memungkinkan pelatihan lebih cepat serta efisien dalam penggunaan sumber daya. Penelitian ini dilakukan menggunakan Google Colab dengan GPU Tesla T4 (memori 15102MiB). YOLOv5 versi 7.0-331 dan YOLOv8 versi 2.45 digunakan dengan konfigurasi Python 3.10.12 dan PyTorch 2.3.0+cu121. Lingkungan untuk eksperimen ini dilengkapi dengan 2 CPU, 12.7 GB RAM sistem, dan ruang *disk* 31.4/78.2 GB. Beberapa eksperimen dilakukan dengan variasi model YOLOv5 dan YOLOv8 serta *hyperparameter* dasar yang berbeda untuk menemukan model terbaik untuk deteksi jerawat, dan untuk memahami bagaimana perubahan *hyperparameter* mempengaruhi performa model.

A. *Perbandingan Metode Data Pre-processing pada YOLOv5 dan YOLOv8*

Setelah pelatihan selesai dilakukan maka selanjutnya adalah melakukan klasifikasi tingkat keparahan jerawat dengan mengambil jumlah objek yang disini adalah jerawat yang berhasil terdeteksi. Klasifikasi disini tidak akan menggunakan metode ataupun algoritma *Machine Learning.* Model yang telah dilatih akan dimuat dan digunakan untuk mendeteksi jerawat pada sebuah gambar. Hasil deteksi tersebut kemudian dimasukkan kedalam sebuah fungsi. *Pre-processing* data dilakukan untuk menghasilkan *input* yang membantu performa model. Tiga *dataset* dengan metode *pre-processing* berbeda dihasilkan. Setelah *pre-processing*, dilakukan pelatihan, validasi, dan evaluasi performa model menggunakan *dataset* tersebut, serta anotasi disesuaikan agar akurat. Pada penelitian ini, tiga dataset dibuat dengan pendekatan pre-processing yang berbeda. Dataset 1 melibatkan *resize* gambar menjadi 640 x 640 piksel dengan rasio 1:1, tanpa kehilangan bagian gambar namun menghasilkan distorsi pada gambar landscape. Dataset 2 juga menggunakan ukuran yang sama, tetapi dengan proses seleksi yang membuang gambar dengan kualitas rendah atau tidak jelas, menghasilkan total 1.358 data yang digunakan untuk eksperimen. Dataset 3, dengan ukuran 1280 x 1280 piksel, mengalami proses *resize* serupa dan seleksi ketat untuk memastikan kualitas gambar yang baik. Gambar *landscape* dipotong agar hanya wajah yang tetap jelas terlihat, sementara gambar yang terlalu dekat atau miring diberi padding agar mempertahankan rasio 1:1, menghasilkan 1.163 data yang digunakan dalam penelitian ini. Pembagian data dilakukan dengan urutan data yang dicari terlebih dahulu, perbandingan pembagiannya adalah, 80% untuk data pelatihan, 10% untuk data validasi, dan 10% untuk data pengujian. Pada Tabel 2 merupakan hasil matrik dari model YOLOv5 dan YOLOv8 dengan langkah *pre-processing* berbeda pada setiap *dataset*.

TABEL 2.
HASIL METRIK MODEL YOLOV5 DAN YOLOV8 DATASET BERBEDA

Model	Dataset	*Precision*	*Recall*	*F1-Score*	mAP50	mAP50-95	*Training Time (hours)*
YOLOv5	1	0,459	0,409	0,433	0,343	0,086	0,383
	2	0,478	0,443	0,460	0,380	0,096	0,330
	3	0,497	0,460	0,478	0,406	0,104	0,313
YOLOv8	1	0,399	0,394	0,396	0,308	0,078	0,633
	2	0,387	0,406	0,396	0,322	0,085	0,584
	3	0,431	0,403	0,417	0,350	0,093	0,875

Secara umum untuk YOLOv5 dan YOLOv8 berdasarkan peningkatan nilai seluruh metrik, dapat dikatakan semakin *pre-processing* dan *data cleaning* dioptimalkan maka performa semakin meningkat. Berdasarkan eksperimen ini, untuk optimasi deteksi jerawat, metode *pre-processing* yang diterapkan selanjutnya adalah pada Dataset 3 yang memberikan hasil terbaik, terutama pada model YOLOv5. Meskipun YOLOv8 menunjukkan peningkatan performa dengan Dataset 3, YOLOv5 tetap unggul.

B. *Perbandingan Ukuran Model YOLOv5 dan YOLOv8*

Penelitian ini membandingkan performa model YOLOv5 dan YOLOv8 dengan berbagai ukuran. YOLOv5s menonjol dengan precision, recall, F1-Score, dan mAP50 tertinggi serta waktu pelatihan yang singkat (0.313 jam), mengindikasikan efisiensi yang baik. YOLOv5m mendekati performa YOLOv5s, sementara YOLOv5l dan YOLOv5x menunjukkan penurunan performa meskipun waktu pelatihan lebih lama, mungkin karena *overfitting*. YOLOv8x memiliki recall dan F1-Score tertinggi namun memerlukan waktu pelatihan yang lama (1.850 jam), sementara YOLOv8s menawarkan waktu pelatihan lebih singkat dengan performa rendah. Secara keseluruhan, YOLOv5s menawarkan keseimbangan terbaik antara performa dan efisiensi waktu pelatihan untuk deteksi jerawat dalam lingkungan komputasi yang terbatas. Pada Tabel 3 merupakan hasil matrik dari model YOLOv5 dan YOLOv8 dengan beberapa ukuran model yang berbeda.

TABEL 3.
HASIL METRIK MODEL YOLOV5 DAN YOLOV8 UKURAN MODEL

Model	Ukuran	*Precision*	*Recall*	*F1-Score*	mAP50	mAP50-95	*Training Time (hours)*
YOLOv5	Yolov5s	0,497	0,460	0,478	0,406	0,104	0,313
	Yolov5m	0,492	0,450	0,470	0,411	0,101	0,454
	Yolov5l	0,485	0,416	0,448	0,390	0,101	0,705
	Yolov5x	0,469	0,417	0,441	0,383	0,098	1,140
YOLOv8	Yolov8s	0,431	0,403	0,417	0,350	0,093	0,875
	Yolov8m	0,430	0,428	0,429	0,367	0,099	1,100
	Yolov8l	0,426	0,440	0,433	0,360	0,095	1,368
	Yolov8x	0,429	0,443	0,436	0,363	0,095	1,850

C. *Perbandingan Hyperparameters pada YOLOv5 dan YOLOv8*

Penelitian ini mengadakan percobaan pelatihan dengan berbagai hyperparameter dasar secara manual untuk model YOLOv5 dan YOLOv8 *small* menggunakan Dataset 3, yang telah memberikan hasil terbaik pada eksperimen sebelumnya. Eksperimen fokus pada penyesuaian *epoch*, *batch size*, dan ukuran gambar untuk mengevaluasi dampaknya terhadap performa model. Karena keterbatasan sumber daya komputasi, pelatihan dilakukan dengan hyperparameter yang dapat dijalankan dengan baik pada sumber daya yang terbatas. Tujuan utama percobaan ini adalah untuk memahami bagaimana *tuning hyperparameter* dasar mempengaruhi performa deteksi jerawat pada model YOLOv5 dan YOLOv8. Pada Tabel 4 merupakan *hyperparameter* beserta dengan nilainya yang menjadi uji coba untuk penelitian ini.

;

TABEL 4.
HYPERPARAMETER DAN NILAI UJI COBA

Hyperparameter	Nilai
Epoch	25 | 50 | 75 | 100
Batch Size	8 | 16 | 32
Image Size	640 | 832 | 896

Pada penelitian ini, dilakukan eksperimen menggunakan Dataset 3 dengan batch size 16 dan image size 640 untuk model YOLOv5s dan YOLOv8s dengan variasi jumlah epoch. Hasil pelatihan menunjukkan bahwa jumlah epoch berpengaruh signifikan terhadap performa kedua model. Pada YOLOv5, precision mengalami fluktuasi setelah epoch tertentu, menunjukkan potensi overfitting pada data pelatihan. Epoch 25 menunjukkan keseimbangan baik antara precision dan recall, sementara epoch 75 memberikan sedikit peningkatan pada precision dengan keseimbangan yang tetap baik. Pada YOLOv8, epoch 75 memberikan performa terbaik dengan precision yang lebih tinggi dan recall yang masih stabil, menunjukkan konsistensi dalam deteksi jerawat pada wajah. Epoch 100 menunjukkan penurunan performa, mungkin disebabkan oleh overfitting atau konvergensi dini. Secara keseluruhan, YOLOv5s memerlukan epoch lebih sedikit yaitu 25 untuk performa yang baik, sedangkan YOLOv8s memerlukan lebih banyak epoch yaitu 75 untuk mencapai hasil optimal dengan waktu

pelatihan yang lebih panjang. Pada Tabel 5 merupakan hasil matrik dari model YOLOv5 dan YOLOv8 dengan beberapa ukuran epoch yang berbeda.

TABEL 5.
HASIL METRIK MODEL YOLOV5 DAN YOLOV8 EPOCH

Model	*Epoch*	*Precision*	*Recall*	F1-Score	mAP50	mAP50-95	*Training Time (hours)*
YOLOv5	25	0,477	0,481	0,479	0,412	0,102	0,166
	50	0,497	0,460	0,478	0,406	0,104	0,313
	75	0,487	0,460	0,473	0,416	0,105	0,465
	100	0,494	0,439	0,465	0,401	0,104	0,626
YOLOv8	25	0,386	0,412	0,399	0,323	0,085	0,917
	50	0,429	0,443	0,436	0,363	0,095	1,850
	75	0,450	0,428	0,439	0,362	0,094	1,366
	100	0,384	0,419	0,401	0,286	0,072	0,936

Pada penelitian ini, eksperimen dilakukan dengan menggunakan Dataset 3, *epoch* 50, dan *image size* 640 untuk YOLOv5s dan YOLOv8s dengan variasi *batch size*. YOLOv5s menunjukkan performa optimal pada *batch size* 32 dengan keseimbangan *precision*, *recall*, dan *F1-Score* yang baik. Sedangkan YOLOv8s mencapai hasil terbaik pada *batch size* 16 dengan mAP50 tertinggi, meskipun memerlukan waktu pelatihan yang lebih lama. Batch size 32 pada YOLOv8s memberikan *precision* tertinggi dengan waktu pelatihan lebih efisien, meski mengalami sedikit penurunan pada *recall* dan *F1-Score*. Dengan demikian, seleksi *batch size* yang tepat penting untuk mencapai performa dan efisiensi pelatihan yang optimal pada deteksi jerawat dengan YOLOv5s dan YOLOv8s. Pada Tabel 6 merupakan hasil matrik dari model YOLOv5 dan YOLOv8 dengan beberapa *batch size* yang berbeda.

TABEL 6.
HASIL METRIK MODEL YOLOV5 DAN YOLOV8 BATCH SIZE

Model	*Batch*	*Precision*	*Recall*	*F1-Score*	mAP50	mAP50-95	*Training Time (hours)*
YOLOv5	8	0,524	0,436	0,476	0,409	0,107	0.390
	16	0,497	0,460	0,478	0,406	0,104	0.313
	32	0,511	0,458	0,483	0,424	0,106	0.306
YOLOv8	8	0,422	0,421	0,421	0,357	0,096	0,915
	16	0,429	0,443	0,436	0,363	0,095	1.850
	32	0,447	0,423	0,435	0,358	0,095	0,862

Pada percobaan dengan Dataset 3, *epoch* 50, dan *batch size* 16 menggunakan YOLOv5s dan YOLOv8s, variasi *image size* (640, 832, 896) diteliti. YOLOv5s menunjukkan stabilitas dengan precision tertinggi pada ukuran 640 dan 832, serta *recall* terbaik pada ukuran 896. Meskipun ukuran 896 memberikan sedikit peningkatan performa, waktu pelatihan lebih lama dibandingkan ukuran 640 atau 832. YOLOv8s menunjukkan *precision* tertinggi pada ukuran 832 dan *recall* tertinggi pada ukuran 640, dengan waktu pelatihan yang bervariasi. Secara keseluruhan, YOLOv5s lebih efisien dan memiliki performa metrik yang lebih baik dibandingkan YOLOv8s pada ukuran gambar yang sama. Pada Tabel 7 merupakan hasil matrik dari model YOLOv5 dan YOLOv8 dengan beberapa *image size* yang berbeda.

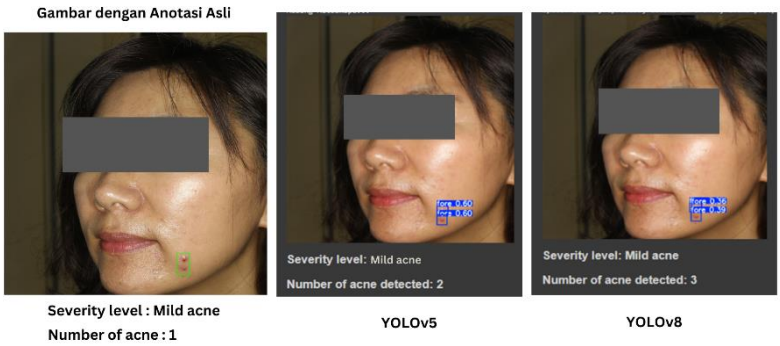
TABEL 7.
HASIL METRIK MODEL YOLOV5 DAN YOLOV8 IMAGE SIZE

Model	*Image Size*	*Precision*	*Recall*	*F1-Score*	mAP50	mAP50 -95	*Training Time (hours)*
YOLOv5	640	0,497	0,460	0,478	0,406	0,104	0,313
	832	0,496	0,446	0,470	0,402	0,106	0,498
	896	0,482	0,461	0,471	0,409	0,104	0,551
YOLOv8	640	0,429	0,443	0,436	0,363	0,095	1,850
	832	0,457	0,418	0,437	0,356	0,096	1,199
	896	0,445	0,413	0,428	0,359	0,096	1,267

D. *Implementasi Klasifikasi Tingkat Keparahan Jerawat*

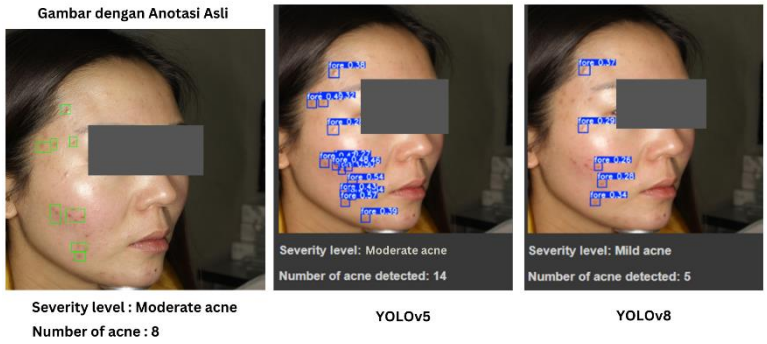
Setelah beberapa percobaan pelatihan model dilakukan selanjutnya adalah melakukan implementasi model tersebut dengan beberapa gambar berbeda untuk pendeteksian jerawat serta klasifikasi tingkat keparahannya berdasarkan jumlah jerawat. Model yang akan digunakan adalah yang didapatkan menghasilkan performa yang paling optimal diantara percobaan yang telah dilakukan, dilihat dari matrik evaluasi masing-masing untuk YOLOv5 dan YOLOv8. Matrik evaluasi yang akan dijadikan metrik utama untuk menentukan model terbaik ini adalah *F1-score. F1-Score* dapat digunakan sebagai metrik utama untuk tugas deteksi jerawat karena menggabungkan *precision* dan *recall* menjadi satu nilai yang seimbang, memberikan gambaran lengkap tentang kemampuan model dalam mendeteksi jerawat secara akurat tanpa terlalu banyak *false positives* atau *false negative*. Berdasarkan hal tersebut maka ditentukan model yang digunakan untuk implementasi dengan YOLOv5 yaitu dengan YOLOv5s, *epoch* 50, *batch size* 32, dan *image size* 640. Sedangkan untuk YOLOv8, model yang digunakan adalah YOLOv8s dengan *epoch* 75, *batch size* 16, dan *image size* 640. Gambar yang digunakan untuk implementasi ini adalah dari data pengujian. Implementasi dilakukan untuk setiap level tingkat keparahan jerawat, untuk menguji model pada setiap tingkat keparahan.

Pada Gambar 5 merupakan hasil dari implementasi deteksi dan klasifikasi tingkat keparahan jerawat dengan YOLOv5 dan YOLOv8 pada gambar ‘levle0_522’ yang termasuk dalam tingkat keparahan *mild* (0-5 jerawat) dengan jumlah jerawat 1. Pada YOLOv5 terdeteksi 2 jerawat dengan tingkat keparahan jerawat *mild*, sedangkan YOLOv8 mendeteksi 3 jerawat dengan tingkat keparahan *mild.* Tingkat keparahan untuk kedua model sudah sesuai, tetapi jumlah jerawat yang terdeteksi masih belum tepat dengan jumlah yang lebih banyak. Posisi jerawat terdeteksi masih dalam area yang sama dengan aslinya.



Gambar 5. Deteksi Jerawat dan Klasifikasi Tingkat Keparahan *Mild*

Pada Gambar 6 adalah hasil dari implementasi deteksi dan klasifikasi tingkat keparahan jerawat dengan YOLOv5 dan YOLOv8 pada gambar ‘levle1_634’ yang termasuk dalam tingkat keparahan *moderate* (6-20 jerawat) dengan jumlah jerawat 8. Pada YOLOv5 terdeteksi 14 jerawat dengan tingkat keparahan jerawat *moderate*, sedangkan YOLOv8 mendeteksi hanya 5 jerawat dengan tingkat keparahan *mild.* Tingkat keparahan YOLOv5 adalah benar, tetapi jumlah terdeteksi lebih banyak dibandingkan aslinya. Sebaliknya dari hasil YOLOv8, jerawat terdeteksi lebih sedikit dibandingkan yang seharusnya serta tingkat keparahan jerawat tidak tepat.



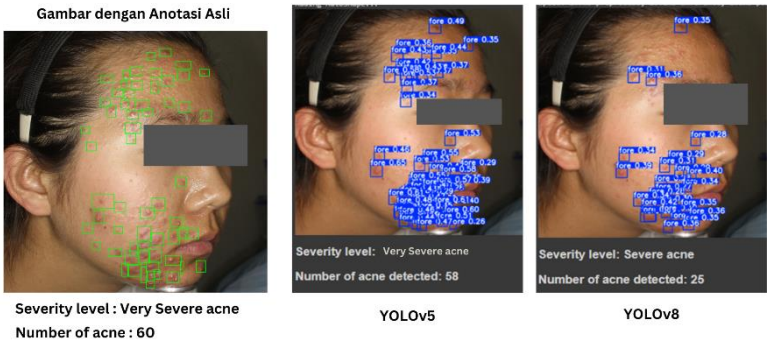
Gambar 6. Deteksi Jerawat dan Klasifikasi Tingkat Keparahan *Moderate*

Pada Gambar 7 merupakan hasil dari implementasi deteksi jerawat dan klasifikasi tingkat keparahan dengan YOLOv5 dan YOLOv8 pada gambar ‘levle2_165 yang termasuk dalam tingkat keparahan *severe* (21-50 jerawat) dengan jumlah jerawat 21. YOLOv5 mendeteksi 31 jerawat dengan tingkat keparahan jerawat *severe*, sedangkan YOLOv8 mendeteksi 10 jerawat

dengan tingkat keparahan *moderate.* Tingkat keparahan YOLOv5 adalah benar, tetapi jumlah terdeteksi jauh lebih banyak dibandingkan aslinya yaitu 10 lebih banyak. Sedangkan dari hasil YOLOv8, jerawat terdeteksi lebih sedikit dibandingkan yang seharusnya yaitu kurang 10, serta tingkat keparahan jerawat juga tidak tepat.

Gambar 7. Deteksi Jerawat dan Klasifikasi Tingkat Keparahan *Severe*

Pada Gambar 8, hasil dari implementasi deteksi jerawat dan klasifikasi tingkat keparahan dengan YOLOv5 dan YOLOv8 pada gambar 'levle3_89' yang termasuk dalam tingkat keparahan *very severe* (lebih dari 50 jerawat) dengan jumlah jerawat 60. Model YOLOv5 berhasil mendeteksi 58 jerawat dengan tingkat keparahan jerawat *very severe*. Model YOLOv8 mendeteksi 25 jerawat dengan tingkat keparahan *severe*. Tingkat keparahan YOLOv5 termasuk benar, tetapi jumlah terdeteksi tidak sesuai dengan yang seharusnya. Hasil YOLOv8, jerawat terdeteksi jauh lebih sedikit dibandingkan yang seharusnya sehingga tingkat keparahan jerawat juga tidak tepat.



Gambar 8. Deteksi Jerawat dan Klasifikasi Tingkat Keparahan *Very Severe*

E. *Diskusi*

Dalam penelitian ini, telah dilakukan perbandingan performa deteksi jerawat antara model YOLOv5 dan YOLOv8 dengan variasi *hyperparameter* dasar, ukuran model, dan proses data *pre-processing* yang berbeda. Eksperimen ini menunjukkan bahwa metode *pre-processing* yang tepat sangat mempengaruhi performa deteksi, di mana Dataset 3 memberikan hasil terbaik untuk kedua model. YOLOv5 menunjukkan adaptasi yang lebih baik terhadap variasi data input, menghasilkan *precision*, *recall*, dan nilai *mAP50* tertinggi. Namun, YOLOv5 juga mengalami tantangan berupa *false positives* yang lebih tinggi dibandingkan YOLOv8. *False positives* ini adalah dimana model memprediksi bahwa ada jerawat tetapi sebenarnya tidak ada jerawat. Sedangkan YOLOv8 pada kasus tersebut justru mengalami *false negative*, yang merupakan keadaan saat model memprediksi bahwa tidak ada jerawat tetapi sebenarnya ada jerawat. Pada sisi ukuran model, YOLOv5s mengungguli versi lainnya dalam hal *precision*, *recall*, *F1-Score*, dan *mAP50* dengan waktu pelatihan yang relatif singkat. Meskipun YOLOv5x memiliki potensi untuk mendeteksi lebih banyak objek, terdapat penurunan dalam *precision*. YOLOv8x menunjukkan performa yang lebih baik dalam *recall* dan *F1-Score* dibandingkan versi YOLOv8 lainnya, tetapi memerlukan waktu pelatihan yang lebih lama. Pemilihan ukuran model harus mempertimbangkan antara akurasi deteksi, efisiensi waktu pelatihan, dan kebutuhan sumber daya komputasi.

Eksperimen *hyperparameter* mengungkapkan bahwa *epoch*, *batch size*, dan ukuran gambar sangat mempengaruhi performa model. YOLOv5 mencapai performa stabil pada *epoch* 25 dan 75, sedangkan YOLOv8 memerlukan lebih banyak *epoch* untuk mencapai performa optimal. *Batch size* optimal untuk YOLOv5 adalah 32, sedangkan untuk YOLOv8 adalah 16. Ukuran gambar juga mempengaruhi performa, dengan YOLOv5 menunjukkan stabilitas pada ukuran 640 dan 832,

sementara YOLOv8 mencapai performa tertinggi pada ukuran 832. Pemilihan *hyperparameter* yang tepat harus mempertimbangkan *trade-off* antara akurasi deteksi dan efisiensi waktu pelatihan. Dalam hal deteksi dan klasifikasi tingkat keparahan jerawat, YOLOv5 mendeteksi lebih banyak jerawat dengan tingkat *false positives* yang lebih tinggi, sementara YOLOv8, meskipun mendeteksi lebih sedikit jerawat, memberikan hasil deteksi yang lebih akurat. Evaluasi menunjukkan bahwa kedua model memiliki kelebihan dan kekurangan masing-masing, dan *fine-tuning* lebih lanjut diperlukan untuk meningkatkan performa dan akurasi deteksi. Penelitian ini menyoroti pentingnya seleksi yang cermat terhadap metode *pre-processing*, ukuran model, dan *hyperparameter* untuk mengoptimalkan performa deteksi jerawat menggunakan YOLOv5 dan YOLOv8.

IV. SIMPULAN

Dari penelitian ini didapatkan bahwa YOLOv5 secara keseluruhan memiliki metrik performa lebih tinggi dalam mendeteksi jerawat dibandingkan YOLOv8. Penggunaan *hyperparameter* konservatif pada YOLOv5 sudah cukup berkontribusi pada performa yang lebih baik. Penelitian ini menemukan bahwa proses *pre-processing* yang efektif, termasuk *resize*, *data cleaning*, penyesuaian gambar, konfigurasi *hyperparameter* serta pemilihan ukuran model, penting untuk mempersiapkan data input dan memperhatikan konfigurasi atau *hyperparameter* yang digunakan pelatihan model YOLOv5 dan YOLOv8. Matrik evaluasi yang digunakan meliputi *F1-Score*, *Precision*, *Recall*, dan *mAP*. Hasil perbandingan menunjukkan YOLOv5 cenderung memberikan nilai metrik lebih tinggi dan mendeteksi lebih banyak jerawat meskipun dengan *false positive* lebih banyak dibandingkan *false negative*, sebaliknya YOLOv8 mendeteksi lebih sedikit jerawat dengan *false positive* lebih sedikit namun memberikan lebih banyak *false negative*. Dari hal tersebut dapat dikatakan, YOLOv5 lebih sensitif dalam mendeteksi jerawat tetapi mungkin membuat lebih banyak kesalahan dengan mendeteksi sesuatu yang tidak ada, sedangkan YOLOv8 lebih berhati-hati dan membuat kesalahan lebih sedikit dalam mendeteksi area yang bukan jerawat, tetapi juga mungkin melewatkan beberapa jerawat yang sebenarnya ada. Perbedaan dalam penggunaan *hyperparameter* dasar seperti jumlah *epoch*, *batch size*, dan ukuran gambar mempengaruhi performa deteksi jerawat pada kedua model secara signifikan, dengan YOLOv5 yang menggunakan *hyperparameter* lebih konservatif memberikan performa yang lebih baik. Saran untuk penelitian selanjutnya adalah mempertimbangkan peningkatan teknik *pre-processing* untuk mengurangi *false positive* pada YOLOv5 serta performa untuk kedua model, serta eksplorasi penggunaan *fine-tuning* atau *hyperparameters tuning* yang lebih mendalam menggunakan *optimizer* khusus untuk meningkatkan akurasi deteksi jerawat pada YOLOv5 dan YOLOv8. Eksplorasi lebih dalam untuk YOLOv8 yang memiliki parameter lebih banyak dan kompleks dibandingkan YOLOv5, untuk meningkatkan kinerjanya. Selain itu, evaluasi lebih lanjut terhadap pengaruh variasi *hyperparameter* terhadap performa model juga dapat menjadi fokus untuk meningkatkan kinerja deteksi jerawat secara lebih mendalam.

DAFTAR PUSTAKA

[1] H. Alsulaimani, A. Kokandi, S. Khawandanh and R. Hamad, "Severity of Acne Vulgaris: Comparison of Two Assessment Methods," *Clinical, Cosmetic and Investigational Dermatology*, vol. 13, pp. 711-716, 2020.

[2] P. Magin, J. Adams, G. Heading, D. Pond and W. Smith, "Psychological sequelae of acne vulgaris: results of a qualitative study," *Can Fam Physician*, vol. 52, no. 8, pp. 978-979, 2006.

[3] Y. LeCun, K. Kavukcuoglu and C. Farabet, "Convolutional Networks and Applications in Vision," *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, p. 253–256, 2010.

[4] P. Jiang, D. Ergu and F. Liu, "A Review of Yolo Algorithm Developments," *Procedia Computer Science*, pp. 1066-1073, 2022.

[5] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You only look once: Unified, real-time object detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[6] E. Casas, L. Ramos, E. Bendek and F. Rivas-Echeverria, "YOLOv5 vs. YOLOv8: Performance Benchmarking in Wildfire and Smoke Detection Scenarios," *Journal of Image and Graphics*, vol. 12, no. 2, 2024.

[7] X. Wu, N. Wen, J. Liang, Y.-K. Lai, D. She and M.-M. Cheng, "Joint acne image grading and counting via label Distribution Learning," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[8] N. Hayashi, H. Akamatsu and M. Kawashima, "Establishment of grading criteria for acne severity," *The Journal of Dermatology*, vol. 35, no. 5, pp. 255-260, 16 April 2008.

[9] A. Sangha and M. Rizvi, "Detection of acne by Deep Learning Object Detection," *medRxiv*, 11 December 2021.

[10] H. Wen, W. Yu, Y. Wu, J. Zhao, X. Liu, Z. Kuang and R. Fan, "Acne detection and severity evaluation with interpretable convolutional neural network models," *Technology and Health Care*, vol. 30, pp. 143-153, 25 February 2022.

[11] H. Zhang and T. Ma, "Acne Detection by Ensemble Neural Networks," *Sensors*, vol. 22, no. 18, p. 6828, 2022.

[12] B. Nethravathi, C. Aradita, S. Veeranna, V. Patil, S. Nagaraj and S. Kulkarni, "Acne vulgaris severity analysis application," *Research Square Platform LLC*, 29 June 2023.

[13] M. Vasam, S. Korutla and R. A. Bohara, "Acne vulgaris: A review of the pathophysiology, treatment, and recent nanotechnology based advances," *Biochemistry and Biophysics Reports*, vol. 36, 2023.