Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

Analisis Pengaruh Hyperparameter terhadap Kinerja MobileNetV2 dan InceptionV3 pada Klasifikasi Retakan Beton

Akfi Rozada, Nurul Baroroh, Muhammad Ivan Khoirur Rizky , Ricardus Anggi Pramunendar*

Fakultas Ilmu Komputer, Teknik Infomatika, Universitas Dian Nuswantoro, Semarang, Indonesia
Email: [1]111202214724@mhs.dinus.ac.id, [2]111202214802@mhs.dinus.ac.id ,[3,]111202214778@mhs.dinus.ac.id,
[4,*] ricardus.anggi@dsn.dinus.ac.id
Email Penulis Korespondensi: ricardus.anggi@dsn.dinus.ac.id*
Submitted: **29/11/2025**; Accepted: **12/12/2025**; Published: **31/12/2025**

Abstrak- Deteksi retakan pada permukaan beton merupakan langkah penting dalam menjaga keandalan dan keselamatan struktur infrastruktur. Metode inspeksi visual masih memiliki keterbatasan karena dipengaruhi kondisi lingkungan, subjektivitas operator, serta potensi kesalahan identifikasi. Untuk mengatasi hal tersebut, penelitian ini membandingkan performa dua arsitektur Convolutional Neural Network (CNN), yaitu MobileNetV2 dan InceptionV3, dalam melakukan klasifikasi citra retakan beton. Dataset yang digunakan adalah NYA-Crack-DATA yang terdiri dari dua kelas, yaitu crack dan no-crack, dengan total 5.026 citra. Seluruh citra diproses melalui tahapan pra-pemrosesan dan augmentasi untuk menghasilkan data yang seragam, lebih variatif, serta mendukung proses pelatihan yang stabil pada kedua model modern tersebut.Penelitian ini berfokus pada analisis pengaruh hyperparameter terhadap performa kedua arsitektur CNN tersebut. Empat hyperparameter utama diuji secara bertahap, meliputi learning rate, dropout, batch size, dan epoch. Evaluasi setiap konfigurasi dilakukan menggunakan Stratified 5-Fold Cross-Validation agar hasil yang diperoleh lebih stabil, konsisten, dan tidak bias. MobileNetV2 menunjukkan performa terbaik pada kombinasi learning rate 0.0005, dropout 0.2, batch size 128, dan 30 epoch, dengan akurasi 0.981, presisi 0.979, recall 0.988, dan F1-score 0.984. Sementara itu, InceptionV3 mencapai akurasi tertinggi sebesar 0.966 pada konfigurasi learning rate 0.0003, dropout 0.8, batch size 128, dan 40 epoch.Hasil penelitian menunjukkan bahwa MobileNetV2 lebih unggul dalam akurasi, stabilitas, serta efisiensi komputasi dibandingkan InceptionV3, sehingga lebih sesuai untuk implementasi nyata pada perangkat dengan keterbatasan sumber daya komputasi modern.

Keywords: CNN; Hyperparameter; InceptionV3; Klasifikasi Retakan; MobileNetV2.

Abstract- Crack detection on concrete surfaces is an essential step in maintaining the reliability and safety of infrastructure structures. Manual visual inspection methods still have limitations due to environmental conditions, operator subjectivity, and potential identification errors. To address these issues, this study compares the performance of two Convolutional Neural Network (CNN) architectures, namely MobileNetV2 and InceptionV3, for classifying concrete crack images. The dataset used is NYA-Crack-DATA, consisting of two classes crack and no crack with a total of 5,026 images. All images were processed through pre-processing and augmentation steps to produce consistent, more diverse data and support stable training on both modern models. This study focuses on analyzing the influence of hyperparameters on the performance of both CNN architectures. Four key hyperparameters were examined gradually, including learning rate, dropout, batch size, and epoch. Each configuration was evaluated using Stratified 5-Fold Cross-Validation to ensure results that are more stable, consistent, and unbiased. MobileNetV2 achieved the best performance with a learning rate of 0.0005, dropout of 0.2, batch size of 128, and 30 epochs, resulting in an accuracy of 0.981, precision of 0.979, recall of 0.988, and F1-score of 0.984. Meanwhile, InceptionV3 reached its highest accuracy of 0.966 with a learning rate of 0.0003, dropout of 0.8, batch size of 128, and 40 epochs. The findings indicate that MobileNetV2 outperforms InceptionV3 in terms of accuracy, stability, and computational efficiency, making it more suitable for real implementation on devices with limited computational resources.

Keywords: Crack Classification; CNN; Hyperparameters; InceptionV3; MobileNetV2.

1. PENDAHULUAN

Pembangunan infrastruktur memiliki peran penting dalam mendukung pertumbuhan ekonomi dan meningkatkan kesejahteraan masyarakat[1]. Infrastruktur memadai mampu mempercepat mobilitas, distribusi barang dan jasa, serta berbagai aktivitas ekonomi[2]. Dalam proses pembangunannya, kebutuhan terhadap material konstruksi yang kuat, tahan lama, dan efisien menjadi aspek yang sangat penting. Beton menjadi salah satu material yang paling banyak digunakan karena memiliki kekuatan tekan tinggi, mudah dibentuk, dan relatif tahan terhadap perubahan cuaca[3]. Hampir seluruh proyek besar seperti jembatan, jalan raya, gedung bertingkat, dan bendungan menggunakan beton sebagai bahan utama. Namun dibalik keunggulannya, beton juga memiliki kelemahan berupa ketahanan tarik yang rendah sehingga mudah mengalami keretakan, yang berpotensi menurunkan kualitas, keamanan, dan umur pakai struktur[4].Di Indonesia, permasalahan keretakan pada infrastruktur seperti jalan dan jembatan masih sering dijumpai[5]. Proses keretakan pada struktur beton dipercepat oleh iklim tropis yang penuh dengan kelembapan, perubahan suhu, dan beban lalu lintas yang signifikan[6]. Menurut data terbaru pada Statistik Infrastruktur PUPR Tahun 2023, tingkat kemantapan jalan nasional meningkat dari 92,20% pada tahun 2022 menjadi 94,18% pada tahun 2023. Jalan provinsi juga mengalami kenaikan dari 70% menjadi 71,33%, sedangkan jalan kabupaten/kota naik dari 56% menjadi 57,90%, sebagaimana tercantum pada situs resmi PUPR [7]-[8]. Meskipun demikian, kesenjangan kualitas antara jalan nasional dan jalan daerah masih terlihat jelas, terutama pada infrastruktur daerah yang membutuhkan perhatian lebih karena lebih rentan terhadap kerusakan struktural seperti retakan pada lapisan beton. Berdasarkan permasalahan tersebut, dibutuhkan sistem yang mampu memantau

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

kondisi beton secara cepat dan akurat agar kerusakan dapat terdeteksi lebih dini[9]. Metode manual seperti inspeksi visual atau pengujian *Nondestruktif (NDT)* masih banyak digunakan, namun cara ini memakan waktu, bergantung pada keahlian petugas, dan berisiko menimbulkan kesalahan yang dapat membahayakan struktur[10]. Teknologi kecerdasan buatan *(Artificial Intelligence/AI)* menawarkan solusi yang lebih objektif, efisien, dan terukur, karena mampu menganalisis citra permukaan beton secara otomatis dan konsisten[11]. Dalam hal ini, Convolutional Neural Network (CNN) menjadi salah satu pendekatan paling efektif dalam mendeteksi pola, tekstur, dan bentuk retakan pada permukaan beton dengan tingkat akurasi yang tinggi[11].

Berbagai penelitian sebelumnya telah berkontribusi menyelesaikan permasalahan keretakan beton dengan mengembangkan metode deteksi retak beton berbasis *deep learning*, khususnya dengan arsitektur Convolutional Neural Network (CNN). Arafin et al. menguji beberapa model CNN seperti VGG19, ResNet50, dan InceptionV3 pada dataset retak dan spalling beton dengan variasi *optimizer* dan *learning rate*, di mana InceptionV3 mencapai akurasi terbaik 91,98% untuk segmentasi spalling, sedangkan EfficientNetB3–U-Net unggul pada segmentasi retakan dengan F1-score 95,66%[12]. Zadeh et al. membandingkan model transfer learning (VGG19, ResNet50, InceptionV3, EfficientNetV2) untuk klasifikasi retak permukaan beton, dengan EfficientNetV2 memperoleh hasil tertinggi 99,6%[13]. Penelitian lain oleh Luqman Ali et al. (2021) dalam jurnal Sensors mengevaluasi performa VGG16, VGG19, ResNet50, dan InceptionV3 pada delapan dataset berbeda dengan pengaturan *hyperparameter* seperti *learning rate*, *dropout*, dan *epoch* 20, dan menemukan bahwa ukuran dan variasi data lebih memengaruhi akurasi dibanding kompleksitas model, dengan InceptionV3 unggul pada dataset menengah[14]. Sementara itu, Nguyen et al. (2024) mengoptimasi CNN untuk deteksi retak beton pada Jetson Nano, menguji ResNet50, MobileNetV2, DenseNet121, dan EfficientNetB0 dengan variasi *batch size*, *learning rate*, dan *epoch*, menghasilkan MobileNetV2 dengan akurasi 97,5% serta waktu inferensi tercepat dan konsumsi daya paling efisien[15]. Penelitian serupa yang menggunakan GLCM yang dikombinasikan dengan klasifikasi jaringan neural sederhana dari [16] menunjukkan hasil yang cukup baik dengan akurasi puncak sebesar 95% pada data berjumlah sedikit. Hasil-hasil tersebut membuktikan bahwa teknologi *deep learning* sangat efektif dalam mendeteksi retakan pada beton[12]-[15]. Meskipun demikian, sebagian besar studi masih berfokus pada pengembangan arsitektur model dan perbandingan performa antar jaringan Convolutional Neural Network (CNN). Kajian mengenai pengaruh pengaturan parameter pelatihan terhadap hasil klasifikasi masih terbatas, padahal parameter seperti *learning rate*, *dropout*, *batch size*, dan jumlah *epoch* memiliki peran penting dalam mengoptimalkan proses pembelajaran model serta mencegah *overfitting*[17].Berdasarkan celah tersebut, penelitian ini berfokus pada optimasi kombinasi hyperparameter pada dua model CNN, yaitu InceptionV3 dan MobileNetV2, untuk klasifikasi biner citra retakan beton. InceptionV3 dipilih karena mampu mengekstraksi fitur visual secara kompleks dan memiliki kemampuan generalisasi yang baik[18], sementara MobileNetV2 dipilih karena ringan dan efisien sehingga cocok untuk implementasi di perangkat dengan sumber daya terbatas[19]. Proses pelatihan model dilakukan dengan metode *k-fold cross-validation* agar hasil yang diperoleh lebih stabil dan akurat, serta dapat mengurangi risiko *overfitting*[20].

Kontribusi utama penelitian ini terletak pada belum adanya studi yang secara komparatif menguji proses *tuning hyperparameter* pada dua model CNN tersebut menggunakan dataset NYA-Crack-DATA secara sistematis. Penelitian sebelumnya juga belum membandingkan perilaku *learning rate*, *dropout rate*, *batch size*, dan jumlah *epoch* pada InceptionV3 dan MobileNetV2 dalam konteks dataset yang sama, sehingga belum tersedia kajian yang menjelaskan bagaimana konfigurasi pelatihan dapat memengaruhi performa model. Penelitian ini kemudian menawarkan sudut pandang baru dengan menyediakan pendekatan yang lebih terarah dan berbasis bukti untuk mengoptimalkan klasifikasi retakan beton berbasis *deep learning*. Secara akademik, penelitian ini menyusun proses *tuning hyperparameter* secara terstruktur untuk memperoleh konfigurasi optimal pada kedua model, sekaligus memberikan bukti empiris mengenai pengaruh variasi *hyperparameter* terhadap performa klasifikasi retakan beton. Secara praktis, penelitian ini menghasilkan model yang lebih akurat, stabil, dan siap diterapkan untuk mendukung pemantauan kondisi infrastruktur secara lebih cepat dan tepat. Berdasarkan tujuan tersebut, penelitian ini merumuskan tiga pertanyaan utama: pengaruh variasi *hyperparameter* terhadap performa model, perbandingan kemampuan InceptionV3 dan MobileNetV2 pada dataset NYA-Crack-DATA, serta identifikasi kombinasi *hyperparameter* yang paling optimal dan stabil untuk klasifikasi citra retakan beton. Rumusan masalah tersebut menjadi landasan dalam merancang eksperimen dan analisis, sehingga hasil penelitian dapat memberikan kontribusi metodologis maupun praktis dalam pengembangan sistem deteksi kerusakan beton berbasis citra.

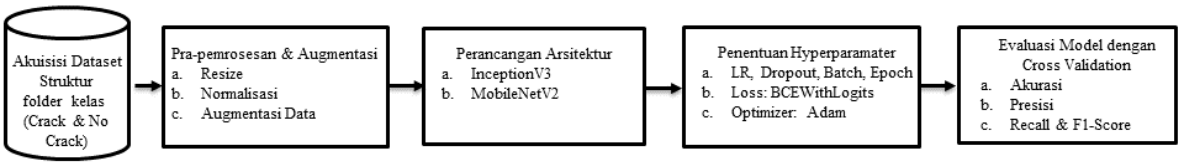
2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Tahapan penelitian ini divisualisasikan pada Gambar 1 sebagai representasi runtutan proses yang dilakukan mulai dari akuisisi dataset hingga evaluasi performa model. Proses dimulai dari pengumpulan dataset

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

serta penyusunan struktur folder kelas. Selanjutnya dilakukan pra-pemrosesan melalui *resize*, normalisasi, dan augmentasi data untuk memastikan kualitas input yang seragam. Pada tahap perancangan model, dilakukan pemilihan arsitektur *InceptionV3* dan *MobileNetV2* yang telah dipretrain pada *ImageNet*. Tahap berikutnya adalah penentuan hyperparameter yang meliputi *learning rate*, *dropout*, *batch size*, dan jumlah *epoch*, serta penggunaan *loss function* BCEWithLogits dan optimizer Adam. Evaluasi model dilakukan menggunakan *k-fold cross-validation* untuk memperoleh nilai akurasi, presisi, recall, dan F1-score secara lebih stabil.

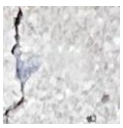


Gambar 1. Alur Kerja Penelitian

2.2 Akuisisi Dataset

Dataset yang digunakan dalam penelitian ini adalah NYA-Crack-DATA, yaitu dataset publik dari Mendeley Data [21]. Dataset ini berisi citra permukaan beton yang terbagi menjadi dua kelas utama, yaitu Crack (retak) dan No Crack (tidak retak), dengan total 5026 citra yang terdiri atas 2167 citra Crack dan 2859 citra No Crack. Contoh visual dari kedua kelas ditampilkan pada Tabel 1 sebagai gambaran karakteristik dataset yang digunakan. Pemilihan NYA-Crack-DATA dilakukan karena ketersediaannya sebagai dataset publik dengan jumlah citra yang memadai serta keberagaman visual yang meliputi variasi tekstur permukaan, kondisi pencahayaan, ukuran retakan, dan tingkat kehalusan beton. Kualitas citra yang baik dan anotasi kelas yang jelas menjadikan dataset ini representatif untuk tugas deteksi retakan beton. Variasi kondisi visual ini juga memberikan dasar yang kuat bagi model untuk mempelajari pola retakan secara lebih komprehensif dan meningkatkan potensi generalisasi pada berbagai lingkungan nyata.

Tabel 1. Distribusi Dataset Berdasarkan Kelas Crack dan No Crack

Label	Jumlah Data	Data ke-1	Data ke-2	Data ke-3	Data ke-4
Crack	2167				
No Crack	2859				

2.3 Pra-pemrosesan dan Augmentasi Data

Pra-pemrosesan dilakukan untuk memastikan seluruh citra memiliki karakteristik input yang seragam, baik dari segi ukuran, intensitas piksel, maupun distribusi kanal warna. Standarisasi ini diperlukan karena model *CNN* sangat sensitif terhadap variasi yang tidak relevan dan membutuhkan input yang konsisten agar proses ekstraksi fitur berlangsung stabil[21]. Tahap pertama adalah *resize*, yaitu mengubah dimensi citra ke ukuran tetap sesuai arsitektur yang digunakan. *MobileNetV2* memerlukan citra berukuran 224×224 piksel[19], sedangkan *InceptionV3* membutuhkan 299×299 piksel . Setelah itu, citra dikonversi ke format tensor dengan menskalakan nilai piksel dari rentang 0–255 menjadi 0–1 untuk meningkatkan stabilitas numerik selama proses komputasi [9]. Tahap selanjutnya adalah normalisasi RGB menggunakan nilai *mean* dan *standard deviation* yang digunakan pada dataset ImageNet, sehingga distribusi piksel berada pada rentang yang sesuai dengan karakteristik data pra-latih model dan dapat mempercepat proses konvergensi[9].Langkah pra-pemrosesan kemudian dilanjutkan dengan augmentasi data, yang diterapkan pada data pelatihan untuk meningkatkan keragaman sampel secara artifisial sehingga model memperoleh variasi yang lebih luas dan tidak mudah mengalami overfitting[23]. Augmentasi yang digunakan meliputi *rotation* acak hingga ±20° untuk memberikan variasi orientasi citra melalui matriks rotasi dua dimensi [24]. Selain itu, digunakan pula transformasi affine yang mencakup *translation*, *shear*, dan *scaling*, yang masing-masing menggeser posisi citra, memberikan efek kemiringan, serta mengubah skala objek melalui penerapan matriks transformasi linear[24]. Variasi kondisi pencahayaan diperkenalkan melalui *brightness jitter*, yaitu penyesuaian intensitas piksel berbasis faktor acak [24]. Untuk menambah keragaman orientasi, digunakan pula *horizontal flip* dan *vertical flip* yang membalik citra pada sumbu horizontal maupun vertikal tanpa mengubah struktur objek [25].

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

2.4 Perancangan Arsitektur Model

2.4.1 Arsitektur InceptionV3

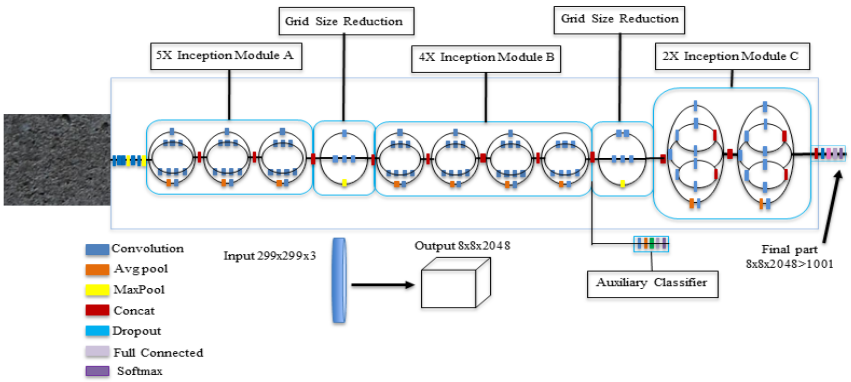
InceptionV3 merupakan arsitektur *CNN* lanjutan yang dirancang untuk meningkatkan efisiensi dan kedalaman representasi melalui penggunaan *factorized convolutions*, reduksi dimensi yang agresif, serta pemrosesan multi-jalur paralel[26]. Pendekatan ini memungkinkan model menangkap informasi visual pada berbagai skala secara simultan, sehingga efektif untuk mendeteksi variasi bentuk dan orientasi pada pola retakan beton[13]. Secara matematis, salah satu jalur konvolusi 3×3 dalam modul *Inception* dapat dinyatakan sebagai

$$Y_{3\times3}(u,v,k) = \sum_{i=-1}^{1}\sum_{j=-1}^{1}\sum_{c=1}^{C} X(u+i, v+j, c)\; w_{i,j,c,k} + b_k \qquad (1)$$

Pada persamaan (1), menghitung keluaran konvolusi 3×3, di mana $Y_{(3\times3)}(u,v,k)$ adalah nilai output pada posisi (u,v) untuk kanal ke-*k*. Variabel $X(u+i, v+j, c)$ merepresentasikan nilai masukan yang digeser oleh indeks *i* dan *j* pada kanal ke-*c*. Bobot *kernel* ditunjukkan oleh $w_{(i,j,c,k)}$, sedangkan *C* adalah jumlah kanal masukan. Nilai b_k merupakan bias untuk kanal keluaran ke-*k*. Persamaan ini menggambarkan proses penjumlahan hasil perkalian antara nilai masukan dan bobot dalam jendela konvolusi 3x3[27].Seluruh jalur yang bekerja secara paralel kemudian digabungkan melalui operasi konkatenasi kanal:

$$Y_{\text{Inception}}(u,v,:) = \text{Concat}\left(Y_{1\times1},\ Y_{3\times3},\ Y_{5\times5},\ Y_{\text{pool\_proj}}\right). \qquad (2)$$

Persamaan (2) merepresentasikan keluaran blok Inception pada posisi spasial (u,v). Pada persamaan ini, $Y_{\text{Inception}}(u,v,:)$ adalah vektor fitur hasil penggabungan seluruh cabang konvolusi pada koordinat tersebut. Variabel $Y_{1\times1}$, $Y_{3\times3}$, $Y_{5\times5}$masing-masing merupakan keluaran dari operasi konvolusi dengan *kernel* berukuran 1×1, 3×3, dan 5×5. Sementara itu, $Y_{\text{pool\_proj}}$ adalah fitur yang dihasilkan dari cabang *pooling* yang diikuti oleh proyeksi konvolusi 1×1. Operasi $\text{Concat}(\cdot)$menunjukkan bahwa seluruh keluaran dari empat cabang tersebut digabungkan pada dimensi kanal untuk membentuk representasi fitur akhir pada blok Inception.[27].



Gambar 2. Diagram Arsitektur InceptionV3

Struktur umum model pada penelitian ini ditunjukkan pada Gambar 2, yang menggambarkan alur makro *InceptionV3* mencakup lima modul *Inception A*, *grid size reduction* pertama, empat modul *Inception B*, *grid size reduction* kedua, dua modul *Inception C*, serta sebuah *auxiliary classifier*[26]. Diagram tersebut bersifat ilustratif dan tidak menampilkan *stem network* serta detail internal seperti *factorized convolution* (1×7, 7×1), meskipun seluruh komponen tersebut tetap aktif dalam implementasi model *pretrained PyTorch* yang digunakan.Dalam penelitian ini, *InceptionV3* diinisialisasi menggunakan bobot *ImageNet* dan dimodifikasi pada bagian *classifier* akhir menjadi *Dropout → Linear (2048→1)* agar menghasilkan satu *logit* sesuai kebutuhan klasifikasi biner[22]. *Auxiliary classifier (AuxLogits)* juga disesuaikan menjadi keluaran tunggal. Walaupun diagram arsitektur menampilkan *softmax*, implementasi ini menggunakan *BCEWithLogitsLoss* yang lebih stabil untuk tugas biner, sehingga *softmax* tidak digunakan secara eksplisit. Modifikasi ini memungkinkan model mempertahankan kemampuan representasi fitur mendalam sembari menyesuaikannya dengan karakteristik tekstur retakan beton.

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

2.4.2 Arsitektur MobileNetV2

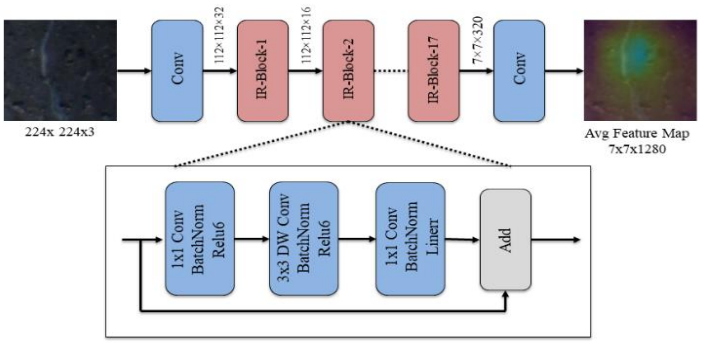
MobileNetV2 dirancang sebagai arsitektur *CNN* yang sangat efisien melalui pemanfaatan *depthwise separable convolution* dan *inverted residual block*. *Depthwise separable convolution* memecah konvolusi standar menjadi dua tahap, yaitu *depthwise convolution* untuk memproses setiap kanal secara terpisah dan *pointwise convolution* (1×1) untuk menggabungkan kembali hasilnya[28]. Pemisahan ini menurunkan biaya komputasi secara signifikan dibandingkan konvolusi standar dengan kompleksitas

$$\text{Cost}_{\text{std}} = H\,W\,K^2\,C_{\text{in}}\,C_{\text{out}}, \qquad (3)$$

Pada persamaan (3), *H* dan *W* merepresentasikan tinggi dan lebar *feature map*, sedangkan *K* menunjukkan ukuran *kernel* konvolusi yang digunakan. Nilai C_{in}merupakan jumlah kanal masukan, dan C_{out} merupakan jumlah kanal keluaran yang dihasilkan oleh lapisan konvolusi. Susunan variabel ini menggambarkan total beban komputasi yang harus dihitung untuk setiap lokasi piksel pada satu lapisan konvolusi.[28], biaya ini berkurang menjadi

$$\text{Cost}_{sep} = H\,.\,W(K^2C_{\text{in}} + C_{\text{in}}C_{\text{out}}) \qquad (4)$$

Pada Persamaan, variabel *H* dan *W* menyatakan tinggi dan lebar *feature map*, sedangkan *K* adalah ukuran *kernel* konvolusi. Nilai C_{in} merepresentasikan jumlah kanal masukan, dan C_{out} menunjukkan jumlah kanal keluaran. Rumus ini menunjukkan bahwa beban komputasi terdiri dari konvolusi per kanal sebesar K^2C_{in} dan konvolusi 1×1 sebesar $C_{in}C_{out}$.[28].



Gambar 3. Diagram Arsitektur MobileNetV2

Struktur makro *MobileNetV2* yang digunakan dalam penelitian ini ditunjukkan pada Gambar 3, yang terdiri atas lapisan konvolusi awal, diikuti beberapa *inverted residual block*, dan diakhiri dengan ekstraksi fitur melalui *global average pooling* sehingga menghasilkan *feature map* berukuran 7×7×1280[19]. Diagram tersebut menampilkan alur blok secara konseptual, sedangkan implementasi pada penelitian ini mengikuti konfigurasi bawaan *PyTorch*, termasuk tahap ekspansi kanal, *depthwise convolution*, dan *residual connection* ketika dimensi fitur sesuai[19].Dalam penelitian ini, *MobileNetV2* dimuat menggunakan bobot *pretrained ImageNet*. Lapisan *classifier* asli dihapus dan diganti dengan *Dropout → Linear (in_features → 1)* sehingga menghasilkan satu nilai *logit* untuk klasifikasi biner *crack* dan *no-crack*. Sesuai kode eksperimen, sebagian parameter awal dibekukan dan hanya beberapa lapisan akhir yang di-*unfreeze* untuk *fine-tuning*, sehingga model tetap mempertahankan fitur *pretrained* sambil beradaptasi terhadap karakteristik tekstur retakan beton[29].

2.5 Desain Eksperimen untuk Penentuan Hyperparamater

Proses *tuning hyperparameter* dilakukan secara sistematis menggunakan pendekatan *grid search* bertahap pada arsitektur MobileNetV2 dan InceptionV3. Setiap konfigurasi dievaluasi menggunakan nilai akurasi yang dihasilkan dari proses pengujian model, sesuai dengan implementasi pada kode eksperimen. Variasi *hyperparameter* yang diujikan ditampilkan pada Tabel 2.

Tabel 2. Konfigurasi Hyperparameter pada Eksperimen.

Parameter	MobilenetV2	InceptionV3
Learning rate	0.0001-0.001	0.0001-0.001
Drop Out	0.1-1	0.1-1
Batch Size	4,8,16,32,48,64,128	4,8,16,32,48,64,128
Epoch	10,20,30,40,50	10,20,30,40,50

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

Hyperparameter yang diuji mencakup aspek optimisasi, regularisasi, jadwal pelatihan, dan konfigurasi fine-tuning. Seluruh eksperimen dijalankan pada laptop dengan AMD Ryzen 5 6000 series, GPU RTX 3050, menggunakan library NumPy, Pandas, PyTorch CUDA beserta torchvision, scikit-learn, PIL, Matplotlib, serta OpenPyXL pada lingkungan Python 3.11 melalui VSCode 1.97. Pada tahap fine-tuning, backbone awal dibekukan lalu beberapa layer dibuka sesuai fine tune params, sehingga kedalaman fine-tuning menjadi hyperparameter penting [19]. Optimizer Adam diperlakukan sebagai hyperparameter karena memanfaatkan momen pertama dan kedua untuk konvergensi yang stabil [26]. Tuning dilakukan secara progresif, dimulai dari pemilihan learning rate 0.0001–0.001, kemudian pengujian dropout rate 0.1–1.0 [24], diikuti variasi batch size 4–128, dan akhirnya jumlah epoch 10–50 dengan pemilihan bobot terbaik berdasarkan validation loss. Parameter pos weight pada BCEWithLogits juga diuji untuk menangani ketidakseimbangan kelas [29], sementara steps per epoch digunakan untuk menjaga konsistensi iterasi antar konfigurasi. Seluruh kombinasi dievaluasi menggunakan cross validation, dan konfigurasi terbaik dipilih berdasarkan rata-rata akurasi tertinggi dari lima fold tanpa variasi performa ekstrem [24]. Pendekatan bertahap ini memastikan proses optimasi lebih efisien dan terarah.

2.6 Evaluasi Model dengan Cross Validation

Penelitian ini menggunakan *Stratified 5-Fold Cross-Validation* untuk memastikan evaluasi yang stabil, dengan pembagian data ke dalam lima fold yang mempertahankan proporsi kelas Crack dan No Crack[24]. Stratifikasi dilakukan sebelum pra-pemrosesan, sehingga setiap fold memiliki distribusi kelas serupa dengan dataset asli. Setiap fold bergantian menjadi data validasi, sementara empat fold lainnya menjalani training lengkap termasuk *resize*, normalisasi, dan augmentasi, sehingga evaluasi dilakukan secara independen dan konsisten. Pendekatan ini mengurangi bias akibat ketidakseimbangan kelas dan memberikan gambaran performa model yang lebih dapat digeneralisasikan. Kinerja model dievaluasi melalui *confusion matrix* yang mencakup empat kategori prediksi yaitu TP (*True Crack*), TN (*True No-Crack*), FP (*False Crack*), dan FN (*False No-Crack*), dengan FN sebagai kesalahan paling kritis karena berisiko menyebabkan retakan tidak terdeteksi. Berdasarkan nilai TP, TN, FP, dan FN tersebut, metrik evaluasi kemudian dihitung menggunakan rumus berikut.

$$[\text{Accuracy} = [\frac{TP + TN}{TP + TN + FP + FN}] \quad (5)$$

$$[\text{Precision} = [\frac{TP}{TP + FP}] \quad (6)$$

$$[\text{Recall} = [\frac{TP}{TP + FN}] \quad (7)$$

$$[\text{F1-Score} = 2 \times [\frac{\text{Precision} \times \text{Recall}}{Precision + \text{Recall}}] \quad (8)$$

Persamaan (5) digunakan untuk mengukur ketepatan keseluruhan model, persamaan (6) menunjukkan ketepatan model dalam memprediksi kelas crack, persamaan (7) menunjukkan kemampuan model menemukan seluruh citra retakan, dan persamaan (8) memberikan penilaian seimbang antara precision dan recall. Rata-rata nilai dari lima fold digunakan sebagai hasil akhir, dan durasi pelatihan setiap konfigurasi turut dicatat untuk menilai efisiensi komputasi. Seluruh hasil disimpan otomatis dalam file Excel untuk memudahkan analisis performa secara konsisten pada setiap konfigurasi.

3. HASIL DAN PEMBAHASAN

3.1 Analisis Karakteristik Dataset

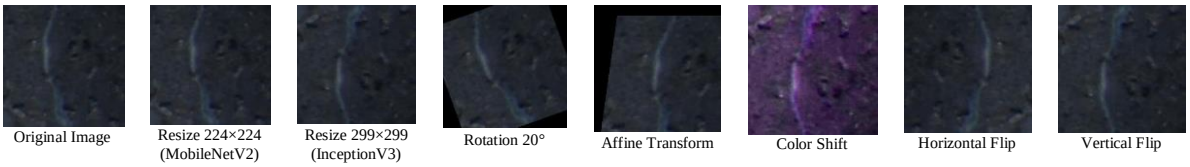
Analisis terhadap dataset NYA-Crack-DATA menunjukkan bahwa variasi visual pada citra memiliki pengaruh signifikan terhadap performa model. Dataset ini terdiri dari 2167 citra Crack dan 2859 citra No Crack, di mana perbedaan jumlah tersebut berpotensi menimbulkan bias prediksi sehingga diperlukan strategi seperti augmentasi dan validasi yang seimbang. Variasi tekstur permukaan beton serta perbedaan pencahayaan menuntut model untuk mengenali pola retakan meskipun citra dipengaruhi faktor eksternal. Beberapa citra dengan latar belakang kompleks atau tekstur mirip retakan pada kelas No Crack juga memperlihatkan tantangan bagi model dalam melakukan generalisasi. Secara keseluruhan, model mampu mengenali pola retakan pada sebagian besar kondisi visual, tetapi performanya menurun pada citra dengan *noise* tinggi atau kontras rendah. Kondisi ini menunjukkan bahwa meskipun dataset cukup representatif, aspek seperti kompleksitas tekstur dan variasi kualitas citra tetap menjadi tantangan yang perlu diperhatikan selama pelatihan.

3.2 Analisis Hasil Pra-pemrosesan dan Augmentasi Data

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

Hasil pra-pemrosesan pada Tabel 3 menunjukkan bahwa proses *resize* berhasil menghasilkan citra dengan ukuran seragam tanpa menyebabkan distorsi pada struktur retakan, sehingga proporsi objek tetap konsisten saat masuk ke model. Citra yang telah dinormalisasi juga memperlihatkan distribusi intensitas piksel yang lebih stabil dan merata dibandingkan citra asli yang memiliki variasi pencahayaan tidak seragam. Pada tahap augmentasi, rotasi dan *affine transform* memberikan variasi sudut dan posisi secara halus tanpa mengubah pola retakan utama, sementara *color shift* menambah variasi pencahayaan yang tetap realistis. Pembalikan horizontal dan vertikal juga menambah keragaman orientasi citra tanpa mengubah konteks tekstur beton. Secara keseluruhan, tahapan pra-pemrosesan berhasil menstandarkan citra dengan baik, sedangkan augmentasi memperkaya variasi dataset tanpa menghilangkan informasi penting, sehingga memberikan dasar yang lebih kuat bagi model dalam mempelajari pola retakan pada berbagai kondisi.

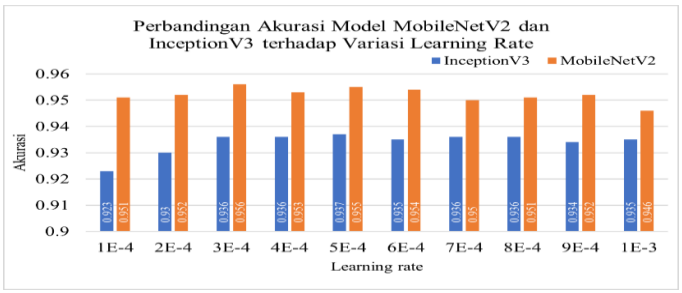
Tabel 3. Gambar Hasil Pra-pemrosesan dan Augmentasi



3.3 Hasil Eksperimen Hyperparamater

3.3.1 Learning rate

Pengujian *learning rate* dari 0.0001 hingga 0.001 pada dua arsitektur, yaitu MobileNetV2 dan InceptionV3, dilakukan untuk mengidentifikasi konfigurasi paling stabil dan akurat. Seluruh *hyperparameter* lain dipertahankan pada nilai default agar pengaruh *learning rate* dapat diamati secara spesifik. Performa masing-masing konfigurasi ditampilkan pada Gambar 4, yang memperlihatkan bagaimana perubahan kecil pada *learning rate* berdampak langsung pada pola konvergensi dan stabilitas pembelajaran.Pada MobileNetV2, *learning rate* 0.0003 memberikan performa tertinggi dengan akurasi 0.956, *precision* 0.945, *recall* 0.979, dan F1-score 0.962. Nilai ini menunjukkan keseimbangan optimal antara kecepatan pembaruan bobot dan stabilitas gradien. Ketika *learning rate* diturunkan ke 0.0001, peningkatan akurasi berjalan lambat karena pembaruan bobot terlalu kecil. Sebaliknya, penggunaan *learning rate* lebih besar dari 0.0003 menyebabkan performa menurun akibat gradien yang berosilasi dan sulit menemukan titik minimum *loss*. Waktu pelatihan yang jauh lebih singkat, yaitu rata-rata 4,5 menit, memperlihatkan bahwa arsitektur ini sangat efisien untuk tugas deteksi retakan.InceptionV3 menunjukkan kecenderungan berbeda. Model ini mencapai hasil terbaik pada *learning rate* 0.0005 dengan akurasi 0.937, *precision* 0.945, *recall* 0.944, dan F1-score 0.944. Karena arsitektur InceptionV3 lebih dalam, nilai *learning rate* kecil seperti 0.0001–0.0002 menghasilkan proses belajar yang lambat, sedangkan *learning rate* di atas 0.0005 memicu fluktuasi akurasi dan ketidakstabilan akibat pembaruan bobot yang terlalu agresif. Waktu pelatihan rata-rata 10–11 menit per konfigurasi mencerminkan kompleksitas arsitektur yang tinggi.



Gambar 4. Grafik Perbandingan Akurasi MobileNetV2 dan InceptionV3 terhadap Learning Rate.

Analisis ANOVA (*Analysis of Variance*) terhadap akurasi menunjukkan bahwa perbedaan rerata antara kedua model tidak terlalu besar, meskipun MobileNetV2 cenderung menghasilkan akurasi lebih tinggi pada sebagian besar nilai *learning rate*. Nilai F memang tinggi, tetapi pola data tidak memperlihatkan perbedaan yang mencolok, sehingga variasi akurasi pada rentang 0.0001–0.001 relatif serupa tanpa dominasi ekstrem dari salah satu arsitektur. Berbeda dengan akurasi, hasil pengujian waktu komputasi menunjukkan kontras yang jauh lebih jelas. MobileNetV2 membutuhkan rata-rata 4.5 menit per konfigurasi, sedangkan InceptionV3 memerlukan lebih dari 11 menit, menghasilkan nilai F yang sangat besar dan menegaskan perbedaan komputasi yang signifikan. Variansi waktu pada InceptionV3 juga lebih tinggi, menunjukkan kebutuhan sumber daya yang lebih besar dan proses pelatihan yang kurang stabil dibandingkan MobileNetV2. Secara keseluruhan, kompleksitas arsitektur memengaruhi sensitivitas terhadap *learning rate*: MobileNetV2 yang ringan stabil pada *learning rate* kecil, sedangkan InceptionV3 memerlukan nilai lebih besar agar propagasi gradien lebih efektif. Berdasarkan evaluasi

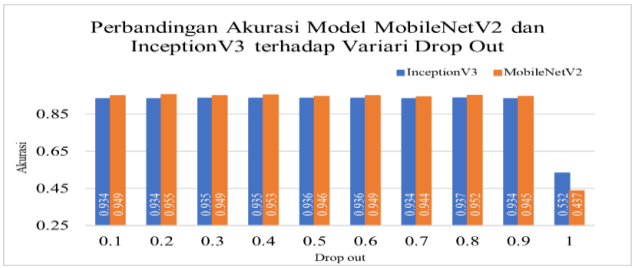
Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

akurasi dan waktu komputasi, *learning rate* terbaik untuk tahap pengujian *dropout* selanjutnya adalah 0.0003 untuk MobileNetV2 dan 0.0005 untuk InceptionV3.

3.3.2 Drop Out

Pada tahap ini, pengujian *dropout* dilakukan menggunakan *learning rate* terbaik dari tahap sebelumnya, yaitu 0.0003 untuk MobileNetV2 dan 0.0005 untuk InceptionV3, dengan hyperparameter lainnya dipertahankan pada nilai default. Evaluasi difokuskan pada akurasi sebagai indikator utama, disertai *precision*, *recall*, dan *F1-score* sebagai metrik pendukung. Hasil pada Gambar 5 menunjukkan bahwa MobileNetV2 mencapai performa terbaik pada *dropout* 0.2 dengan akurasi 0.955, *precision* 0.948, *recall* 0.974, dan *F1-score* 0.961. *Dropout* rendah (0.1–0.2) membantu menjaga stabilitas pembelajaran, sedangkan nilai di atas 0.3 mulai menurunkan performa dan menjadi sangat buruk pada *dropout* ekstrem 0.8–1.0. Temuan ini menunjukkan bahwa arsitektur ringan seperti MobileNetV2 tidak membutuhkan regularisasi besar dan sensitif terhadap *dropout* agresif. Sebaliknya, InceptionV3 mencapai performa terbaik pada *dropout* 0.8 dengan akurasi 0.937, *precision* 0.928, *recall* 0.965, dan *F1-score* 0.946. Kompleksitas arsitektur yang lebih besar menyebabkan model rentan *overfitting* pada *dropout* rendah, sehingga *dropout* tinggi membantu meningkatkan generalisasi. Pada rentang 0.1–0.3, selisih besar antara akurasi *training* dan *validasi* menjadi indikator *overfitting*. Dari sisi efisiensi, MobileNetV2 membutuhkan sekitar 2 menit per *epoch*, jauh lebih cepat dibandingkan InceptionV3 yang memerlukan 4–5 menit. Perbedaan komputasi tersebut sejalan dengan kompleksitas arsitektur, tetapi pola efektivitas *dropout* tetap konsisten: MobileNetV2 optimal pada *dropout* rendah, sedangkan InceptionV3 membutuhkan *dropout* tinggi.

Hasil tersebut kemudian diperkuat melalui analisis statistik ANOVA (*Analysis of Variance*) yang diterapkan pada nilai akurasi dan waktu komputasi dari seluruh variasi dropout. Pengujian ini menunjukkan bahwa penggunaan dropout memiliki pengaruh signifikan terhadap perubahan akurasi pada kedua model karena nilai *p-value* berada jauh di bawah ambang 0.05. Kondisi tersebut menegaskan bahwa perbedaan performa yang muncul tidak bersifat acak, melainkan dipengaruhi langsung oleh variasi tingkat dropout. Pada MobileNetV2, kelompok dropout rendah 0.1–0.2 membentuk kluster akurasi yang berbeda secara signifikan dibandingkan kelompok dropout tinggi 0.6–1.0, sehingga memperlihatkan sensitivitas arsitektur ringan terhadap regularisasi yang terlalu besar. Sementara itu, pada InceptionV3, dropout tinggi memberikan peningkatan stabil yang tidak muncul pada kelompok dropout rendah, menunjukkan kebutuhan regularisasi yang lebih kuat pada arsitektur yang kompleks.Analisis yang sama juga menunjukkan bahwa waktu komputasi tidak mengalami perbedaan signifikan antar variasi dropout, mengindikasikan bahwa perubahan durasi pelatihan lebih dipengaruhi oleh karakteristik arsitektur masing-masing daripada tingkat dropout. Berdasarkan seluruh rangkaian eksperimen, baik dari sisi performa maupun validitas statistik, konfigurasi terbaik ditentukan berdasarkan akurasi tertinggi untuk masing-masing model. Oleh karena itu, dropout 0.2 dipilih sebagai konfigurasi optimal untuk MobileNetV2, sedangkan dropout 0.8 ditetapkan sebagai konfigurasi terbaik untuk InceptionV3.



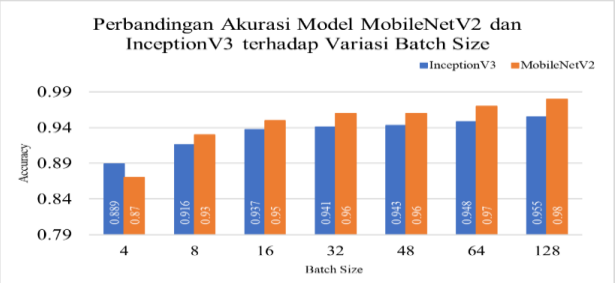
Gambar 5. Grafik Perbandingan Akurasi MobileNetV2 dan InceptionV3 terhadap Drop Out

3.3.3 Batch Size

Tahap ini menguji pengaruh variasi batch size terhadap performa MobileNetV2 dan InceptionV3 dengan menggunakan konfigurasi learning rate dan dropout terbaik dari eksperimen sebelumnya, yaitu 0.0003 dan dropout 0.2 untuk MobileNetV2 serta 0.0005 dan dropout 0.8 untuk InceptionV3. Visualisasi hasil pengujian ditampilkan pada Gambar 6. Variasi batch size diberikan pada rentang 4 hingga 128 yang memengaruhi stabilitas gradien dan pola pembaruan bobot selama proses pelatihan. Pada batch size kecil seperti 4 dan 8, kedua model menunjukkan fluktuasi akurasi akibat gradien yang tidak stabil, dengan efek yang lebih terasa pada InceptionV3 karena arsitekturnya yang kompleks. Ketika batch size dinaikkan pada rentang menengah yaitu 16 dan 32, pola pelatihan menjadi lebih stabil karena lebih banyak sampel yang berkontribusi dalam proses propagasi balik. Konsistensi gradien meningkat sehingga nilai akurasi bergerak lebih teratur, meskipun belum mencapai performa optimum.Performa terbaik tercapai pada batch size 128, di mana MobileNetV2 memperoleh akurasi 0.980 dan InceptionV3 mencapai 0.955.Nilai ini menunjukkan bahwa batch besar memberikan stabilitas gradien yang lebih efektif, memungkinkan model mengenali pola tekstur retakan secara lebih seragam pada setiap pembaruan bobot. Walaupun batch size besar meningkatkan waktu pemrosesan per iterasi, efisiensi total pelatihan tetap meningkat karena model lebih cepat mencapai konvergensi dalam jumlah *epoch* yang lebih sedikit.MobileNetV2 tetap

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

menunjukkan keunggulan efisiensi waktu, sejalan dengan arsitekturnya yang lebih ringan dan jumlah parameternya yang lebih sedikit.

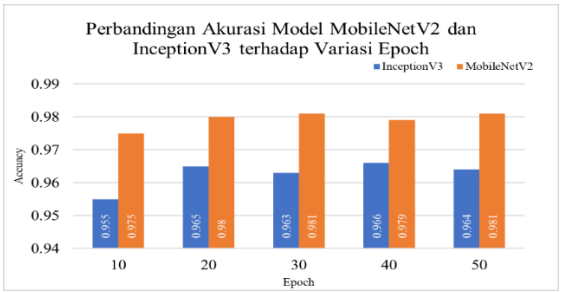


Gambar 6. Grafik Perbandingan Akurasi MobileNetV2 dan InceptionV3 terhadap Batch Size

Hasil tersebut kemudian diperkuat melalui analisis statistik *ANOVA* (Analysis of Variance) yang diterapkan pada akurasi kedua model. Analisis ini menunjukkan bahwa perbedaan rerata performa tidak signifikan secara statistik, sehingga variasi akurasi pada seluruh konfigurasi lebih dipengaruhi oleh stabilitas gradien yang meningkat pada batch besar dibandingkan oleh perbedaan arsitektural antara MobileNetV2 dan InceptionV3. Kondisi ini mengindikasikan bahwa batch size 128 memberikan dampak peningkatan yang relatif seimbang pada kedua CNN.Dari sisi efisiensi, MobileNetV2 menunjukkan waktu komputasi yang jauh lebih rendah pada seluruh variasi batch size, mulai dari 0.90 detik pada batch size 4 hingga 14.31 detik pada batch size 128. Sementara itu, InceptionV3 membutuhkan waktu yang lebih lama, yaitu 2.60 detik pada batch size 4 dan meningkat hingga 30.27 detik pada batch size 128. Meskipun perbedaan waktu komputasi tampak sangat besar, hasil *ANOVA* menunjukkan bahwa rerata kedua kelompok tetap tidak berbeda signifikan, menandakan bahwa varian waktu terutama dipengaruhi oleh kompleksitas internal InceptionV3.Hal ini memperlihatkan bahwa peningkatan batch size memengaruhi kedua model dengan arah yang sama, tetapi InceptionV3 menanggung beban komputasi yang lebih besar akibat kedalaman arsitekturnya.Berdasarkan seluruh hasil tersebut, batch size 128 dipilih sebagai konfigurasi optimal karena memberikan keseimbangan terbaik antara akurasi dan stabilitas pembelajaran. MobileNetV2 memperoleh manfaat paling besar dari penggunaan batch besar tanpa mengorbankan efisiensi, sedangkan InceptionV3 tetap mengalami peningkatan performa meskipun terbebani oleh waktu komputasi yang lebih tinggi. Konfigurasi ini kemudian digunakan pada tahap pengujian jumlah epoch untuk memastikan konsistensi performa pada proses pelatihan berikutnya.

3.3.4 Epoch

Pada tahap ini, kedua model dilatih menggunakan kombinasi *hyperparameter* terbaik dari eksperimen sebelumnya, yaitu *learning rate* 0.0003 dan *dropout* 0.2 untuk MobileNetV2, serta *learning rate* 0.0005 dan *dropout* 0.8 untuk InceptionV3, dengan *batch size* 128 untuk keduanya. Tujuan pengujian ini adalah menentukan titik konvergensi sekaligus mengidentifikasi tanda *underfitting* atau *overfitting*. Pola pelatihan terhadap variasi epoch dapat dilihat pada Gambar 7, yang juga menunjukkan stabilitas gradien dan kapasitas masing-masing model dalam mengekstraksi fitur penting dari citra retakan beton.Secara umum, kedua model menunjukkan peningkatan performa seiring bertambahnya epoch sebelum akhirnya mencapai titik optimum dan stagnasi. Pada InceptionV3, akurasi meningkat dari 0,955 pada epoch ke-10 menjadi 0,966 pada epoch ke-40, dengan *F1-score* tertinggi 0,970, yang menunjukkan bahwa arsitektur lebih dalam memerlukan waktu lebih lama untuk stabilisasi *gradien* dan pembelajaran fitur secara menyeluruh.



Gambar 7. Grafik Perbandingan Akurasi MobileNetV2 dan InceptionV3 terhadap Epoch

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

Analisis ANOVA terhadap akurasi menunjukkan bahwa perbedaan performa kedua model nyata dan konsisten, mengindikasikan keunggulan MobileNetV2 dalam menangkap fitur penting. Seiring penambahan epoch hingga 50, peningkatan performa tidak lagi signifikan dan mulai muncul tanda *overfitting* ringan, sementara waktu pelatihan InceptionV3 meningkat drastis dari sekitar 21 menjadi hampir 97 menit sehingga tambahan epoch tidak memberikan manfaat yang sebanding. MobileNetV2 menunjukkan tren peningkatan lebih cepat dan stabil, dengan akurasi naik dari 0.975 pada epoch ke-10 menjadi 0.981 pada epoch ke-30 serta F1-score tertinggi 0.984. Setelah epoch ke-30, performa kedua model cenderung stagnan, menandakan titik jenuh pembelajaran, sekaligus menunjukkan bahwa arsitektur ringan mencapai konvergensi lebih awal. Waktu pelatihan meningkat hingga lebih dari 70 menit pada epoch ke-50, menegaskan bahwa penambahan epoch tidak lagi efisien. Analisis ANOVA terhadap durasi komputasi menunjukkan bahwa meskipun MobileNetV2 lebih cepat, variasi waktu antar model tidak signifikan secara statistik sehingga memperlihatkan stabilitas proses pelatihan. Dari sisi efisiensi, MobileNetV2 tetap unggul dengan performa tinggi dan waktu pelatihan jauh lebih singkat dibandingkan InceptionV3. Berdasarkan keseluruhan evaluasi, epoch optimal ditetapkan pada 30 untuk MobileNetV2 dan 40 untuk InceptionV3 karena memberikan keseimbangan terbaik antara akurasi, F1-score, dan efisiensi komputasi, sehingga digunakan sebagai konfigurasi akhir dalam tahap evaluasi model.

3.4 Analisis Perbandingan Model

Pada penelitian ini, proses tuning hyperparameter dilakukan secara bertahap untuk menentukan konfigurasi pelatihan terbaik bagi kedua model, sebagaimana ditunjukkan pada Tabel 4. Pengujian dilakukan pada rentang learning rate 0.0001–0.001, dropout 0.1–1.0, batch size 4–128, dan epoch 10–50 untuk memastikan bahwa setiap kombinasi hyperparameter dievaluasi secara komprehensif. Seluruh nilai performa yang dianalisis merupakan rata-rata dari lima fold Stratified 5-Fold Cross-Validation sebagaimana dijelaskan pada metodologi, sehingga hasil yang diperoleh mencerminkan stabilitas performa pada distribusi data yang berbeda. Berdasarkan eksplorasi bertahap tersebut, MobileNetV2 mencapai stabilitas pelatihan terbaik pada learning rate 0.0003, dropout 0.2, batch size 128, dan 30 epoch, sesuai konfigurasi optimal pada Tabel 4. Sementara itu, InceptionV3 menunjukkan performa paling optimal pada learning rate 0.0005, dropout 0.8, batch size 128, dan 40 epoch, di mana arsitektur yang lebih dalam membuat model lebih sensitif terhadap perubahan hyperparameter sehingga memerlukan regularisasi yang lebih tinggi untuk mencegah overfitting. Perbedaan kebutuhan konfigurasi ini menegaskan bahwa karakteristik arsitektur sangat memengaruhi respons model terhadap pengaturan hyperparameter yang diuji.

Pengaruh langsung dari pengaturan tersebut terlihat pada evaluasi kinerja pada Tabel 5. MobileNetV2 memberikan performa terbaik dengan akurasi 0.981, presisi 0.979, recall 0.988, dan F1-score 0.984. Nilai recall yang tinggi menunjukkan bahwa model ini mampu mengidentifikasi hampir seluruh citra retakan, sehingga risiko false negative dapat diminimalkan, sebuah aspek penting dalam pemantauan struktur. Selain itu, waktu pelatihan yang lebih singkat sekitar 70 menit mencerminkan efisiensi komputasi yang unggul dan konsisten dengan karakter arsitektur yang ringan. InceptionV3 menghasilkan akurasi 0.966, presisi 0.962, recall 0.978, dan F1-score 0.970. Walaupun performanya tetap kompetitif, model ini lebih rentan terhadap pola tekstur permukaan yang menyerupai retakan sehingga menghasilkan false positive maupun false negative yang lebih besar, ditambah waktu pelatihan yang lebih lama sekitar 97 menit. Perbedaan performa ini juga sejalan dengan karakteristik arsitektur, di mana struktur MobileNetV2 yang lebih ringan cenderung memberikan respons gradien yang stabil pada dataset dengan karakter visual homogen, sedangkan InceptionV3 memerlukan kompleksitas visual yang lebih tinggi untuk mencapai performa optimal. Temuan ini turut selaras dengan beberapa penelitian terdahulu. Performa MobileNetV2 yang stabil dan efisien konsisten dengan laporan Nguyen et al. (2024) yang melaporkan bahwa model ringan bekerja optimal pada dataset dengan variasi tekstur yang tidak terlalu kompleks[15]. Adapun perbedaan hasil dengan Arafin et al. (2024), yang menempatkan InceptionV3 sebagai model unggul, menegaskan bahwa efektivitas arsitektur CNN sangat dipengaruhi oleh tingkat kompleksitas tekstur dataset yang digunakan[12]. Secara keseluruhan, MobileNetV2 menunjukkan respons pelatihan yang lebih stabil, waktu komputasi yang lebih efisien, serta akurasi validasi yang lebih tinggi dibandingkan InceptionV3, sehingga lebih sesuai dengan karakter dataset retakan yang relatif homogen.

Tabel 4. Performa Hyperparameter Terbaik dari Kedua Model CNN

Model	Learning Rate	Drop Out	Batch Size	Epoch
MobileNetV2	0.0003	0.2	128	30
InceptionV3	0.0005	0.8	128	40

Tabel 5. Perbandingan Kinerja Akhir Model MobileNetV2 dan InceptionV3

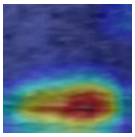
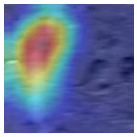
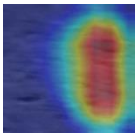
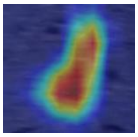
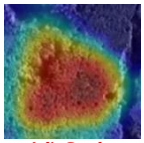
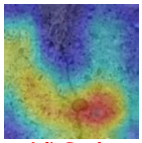
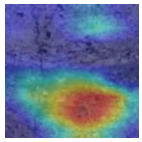
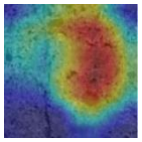
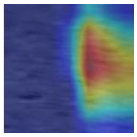
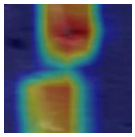
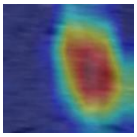
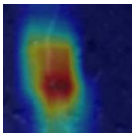
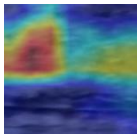
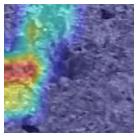
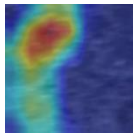
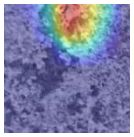
Model	Akurasi	Presisi	Recall	F1-Score	Waktu Pelatihan(menit)
MobileNetV2	0.981	0.979	0.988	0.984	≈ 70
InceptionV3	0.966	0.962	0.978	0.970	≈ 97

Visualisasi pada Tabel 6 menunjukkan perbedaan karakteristik prediksi antara InceptionV3 dan MobileNetV2 berdasarkan contoh klasifikasi benar dan salah. Pada bagian InceptionV3 Klasifikasi Benar, model

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

mengidentifikasi retakan dengan tepat ketika citra menampilkan pola retakan yang jelas dan memiliki kontras kuat. Namun pada bagian InceptionV3 Klasifikasi Salah, citra yang memiliki tekstur beton kasar atau *noise* yang tinggi menyebabkan model memberikan aktivasi yang menyebar sehingga model melakukan *false positive* maupun *false negative*. Kondisi ini menunjukkan bahwa InceptionV3 memiliki sensitivitas tinggi terhadap variasi permukaan beton. Sebaliknya, pada bagian MobileNetV2 Klasifikasi Benar, model menyoroti area retakan secara lebih terfokus dan konsisten, bahkan ketika citra memiliki retakan tipis atau kontras rendah. Pada bagian MobileNetV2 Klasifikasi Salah, jumlah kesalahan yang lebih sedikit menunjukkan bahwa kualitas tekstur yang buruk, pencahayaan rendah, atau permukaan yang sangat tidak rata menjadi faktor utama terjadinya kekeliruan. Aktivasi yang salah juga muncul dengan cakupan yang lebih kecil dibandingkan InceptionV3. Secara keseluruhan, visualisasi tersebut menguatkan hasil evaluasi kuantitatif, di mana MobileNetV2 memberikan prediksi yang lebih stabil dan lebih tahan terhadap variasi tekstur. Mekanisme ekstraksi fitur lokal yang lebih efektif pada MobileNetV2 menjadikannya model yang lebih sesuai untuk sistem deteksi retakan, terutama untuk perangkat dengan keterbatasan komputasi seperti drone inspeksi atau sistem *edge-processing* berbasis Jetson Nano. Temuan ini juga menunjukkan bahwa MobileNetV2 memiliki potensi yang baik untuk digunakan pada aplikasi inspeksi lapangan secara real-time. Selain itu, konsistensi aktivasi pada area retakan menegaskan bahwa model ini mampu mempertahankan kinerja meskipun dihadapkan pada kondisi citra yang lebih menantang.

Tabel 6. Visualisasi Klasifikasi Benar dan Salah dari InceptionV3 dan MobileNetV2

InceptionV3: Klasifikasi Benar				InceptionV3: Klasifikasi Salah			
 Asli: Crack Predict: Crack	 Asli: Crack Predict: Crack	 Asli: Crack Predict: Crack	 Asli: Crack Predict: Crack	 Asli: Crack Predict: No Crack	 Asli: Crack Predict: No Crack	 Asli: Crack Predict: No Crack	 Asli: Crack Predict: No Crack
MobileNetV2: Klasifikasi Benar				**MobileNetV2: Klasifikasi Salah**			
 Asli: Crack Predict: Crack	 Asli: Crack Predict: Crack	 Asli: Crack Predict: Crack	 Asli: Crack Predict: Crack	 Asli: Crack Predict: No Crack	 Asli: No Crack Predict: Crack	 Asli: No Crack Predict: Crack	 Asli: No Crack Predict: Crack

3.5 Pembahasan

Respons kedua model terhadap perubahan hyperparameter memperlihatkan bahwa karakteristik arsitektur berperan besar dalam menentukan stabilitas pembelajaran. MobileNetV2 yang berbasis *inverted residual* dan *depthwise separable convolution* menunjukkan perilaku yang lebih stabil di hampir seluruh konfigurasi, terutama karena desainnya yang ringan membuat model ini tidak terlalu sensitif terhadap variasi regularisasi maupun ukuran batch. Sementara itu, InceptionV3 justru lebih mudah mengalami fluktuasi performa akibat kedalaman jaringan dan jumlah parameternya yang jauh lebih besar, sehingga membutuhkan pengaturan hyperparameter yang lebih hati-hati.Perbedaan kebutuhan dropout dan epoch pada kedua model memperlihatkan bagaimana kompleksitas arsitektur memengaruhi kecenderungan overfitting. Model yang ringan dapat bekerja baik meskipun diberikan regularisasi minimal, sedangkan model yang lebih dalam justru memerlukan keseimbangan antara kapasitas jaringan dan tingkat regularisasi agar tidak kehilangan kemampuan ekstraksi fitur. Temuan ini menjelaskan mengapa performa kedua model cenderung berbeda meskipun diuji pada dataset yang sama dan rentang parameter yang identik.Analisis statistik melalui ANOVA juga menunjukkan bahwa beberapa hyperparameter memberikan pengaruh lebih signifikan dibanding lainnya, khususnya yang berkaitan dengan mekanisme regularisasi dan durasi pelatihan. Hal ini mengonfirmasi bahwa konfigurasi optimal tidak hanya ditentukan oleh besarnya parameter, tetapi oleh kecocokannya dengan struktur internal model. Secara keseluruhan, hasil pembelajaran memperkuat bahwa model yang lebih efisien secara arsitektural dapat memberikan generalisasi lebih baik dan komputasi lebih ringan pada dataset dengan karakter visual yang relatif homogen, seperti citra permukaan beton.

4. KESIMPULAN

Penelitian ini dilakukan untuk menjawab kebutuhan akan sistem pendeteksi retakan beton yang akurat dan efisien, mengingat metode inspeksi visual manual masih rentan terhadap subjektivitas dan kesalahan identifikasi. Melalui serangkaian eksperimen tuning hyperparameter menggunakan *Stratified 5-Fold Cross-Validation*, penelitian ini

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9389

berhasil mengidentifikasi konfigurasi pelatihan yang paling stabil bagi dua arsitektur *CNN*, yaitu *MobileNetV2* dan *InceptionV3*. Temuan utama menunjukkan bahwa *MobileNetV2* memberikan performa paling konsisten dalam mempelajari pola visual retakan, termasuk pada citra dengan tekstur beton yang kompleks maupun kontras rendah, sehingga model ini menjadi pilihan paling efektif untuk sistem deteksi retakan otomatis. Analisis ANOVA memperlihatkan bahwa tidak semua hyperparameter memberikan kontribusi yang sama terhadap performa akhir. *Dropout* dan jumlah *epoch* memiliki pengaruh signifikan terhadap kualitas pembelajaran, sedangkan *learning rate* dan *batch size* tidak menunjukkan perbedaan yang berarti karena stabilitas gradien bawaan *MobileNetV2*. Selain itu, ANOVA pada waktu komputasi menegaskan bahwa perbedaan kecepatan antara kedua model bukan hanya akibat variasi hyperparameter, melainkan karakter arsitektural yang lebih ringan sehingga *MobileNetV2* consistently lebih efisien. Implikasi praktis dari temuan ini adalah bahwa arsitektur ringan seperti *MobileNetV2* memiliki potensi besar untuk diterapkan pada perangkat *edge* berdaya rendah misalnya Jetson Nano, Raspberry Pi, atau modul komputasi serupa yang membutuhkan akurasi tinggi dengan konsumsi komputasi minimal.Untuk pengembangan selanjutnya, penelitian perlu diperluas dengan pengujian pada citra lapangan untuk memastikan ketangguhan model terhadap variasi kondisi visual nyata, seperti pencahayaan tidak merata, tekstur permukaan yang tidak homogen, serta gangguan lingkungan. Selain itu, studi komparatif dengan model yang lebih modern seperti *EfficientNet* atau *Vision Transformer* penting dilakukan untuk mengevaluasi potensi peningkatan akurasi dan efisiensi. Dengan penerapan teknik optimasi seperti *pruning* dan kuantisasi, sistem ini berpotensi dikembangkan menjadi solusi siap pakai yang dapat berjalan secara optimal pada perangkat bergerak atau *edge device*, sehingga benar-benar memberikan dampak praktis dalam inspeksi struktur beton.

REFERENCES

[1] S. N. Anisa, S. Aulia, A. Indah, M. A. K. Dipa, and M. Panorama, "Analisis Peran Infrastruktur Dalam Pertumbuhan Ekonomi Pembangunan Di Kota Palembang," *JUPEA*, vol. 4, no. 1, pp. 36–54, Jan. 2024, doi: 10.51903/jupea.v4i1.2435.

[2] Muhammad Fakhrur Rodzi, "Pembangunan Infrastruktur Dan Pemerataan Ekonomi Di Indonesia," *JMD*, vol. 3, no. 2, pp. 151–163, Dec. 2023, doi: 10.47431/jmd.v3i2.353.

[3] A. Pujianto, R. Faizah, A. Widiyanto, T. A. Putra, H. Prayuda, and F. Firdausa, "Pemanfaatan Limbah Bata Ringan Sebagai Bahan Penyusun Pengganti Pada Beton," *J. Bangunan Teor. Prakt. Penelitian, dan Pengajaran Tek. Bangunan*, vol. 26, no. 2, pp. 1–8, July 2025, doi: 10.17977/um071v26i22021p1-8.

[4] L. H. Ali, "Analisa Retrofit Beton Pasca Kegagalan dan Kelenturan Menggunakan Metode Injeksi Epoxy," *Bull Civ Eng*, vol. 3, no. 1, pp. 31–36, Feb. 2023, doi: 10.18196/bce.v3i1.17671.

[5] M. E. Tosulpa, L. S. Putranto, and R. Sulistyorini, "Analisis Penanganan Pengaruh Kerusakan Jalan Nasional Akibat Muatan Berlebih Di Provinsi Lampung," *JTSC*, vol. 6, no. 2, pp. 1272–1289, July 2025, doi: 10.51988/jtsc.v6i2.348.

[6] M. Chai, L. Wang, C. Hu, and T. Chen, "Investigation into the Impact of Humidity on Early Age Cement Concrete Pavement Behavior in Hot and Humid Regions," *Applied Sciences*, vol. 13, no. 18, p. 10180, Sept. 2023, doi: 10.3390/app131810180.

[7] Setia Dewi Prihapsari *et al.*, "Informasi Statistik Infrastruktur PUPR 2023," Kementerian Pekerjaan Umum dan Perumahan Rakyat, Jakarta Selatan, Mar. 2024. [Online]. Available: https://data.pu.go.id/sites/default/files/BIS%20PUPR%202023.pdf

[8] Setia Dewi Prihapsari *et al.*, "Informasi Statistik Infrastruktur PUPR 2022," Kementerian Pekerjaan Umum dan Perumahan Rakyat, Jakarta Selatan, Laporan Statistik Infrastruktur, Nov. 2022. [Online]. Available: https://data.pu.go.id/sites/default/files/Buku%20Informasi%20Statistik%20Infrastruktur%20PUPR%202022%20ISBN.pdf

[9] V. P. Golding, Z. Gharineiat, H. S. Munawar, and F. Ullah, "Crack Detection in Concrete Structures Using Deep Learning," *Sustainability*, vol. 14, no. 13, p. 8117, July 2022, doi: 10.3390/su14138117.

[10] G. Świt, A. Krampikowska, and P. Tworzewski, "Non-Destructive Testing Methods for In Situ Crack Measurements and Morphology Analysis with a Focus on a Novel Approach to the Use of the Acoustic Emission Method," *Materials*, vol. 16, no. 23, p. 7440, Nov. 2023, doi: 10.3390/ma16237440.

[11] P. N. Hadinata, D. Simanta, and L. Eddy, "Deep Convolutional Neural Network untuk Mendeteksi Retak pada Permukaan Beton yang Memiliki Void," *JoSC*, vol. 1, no. 1, pp. 45–55, Oct. 2021, doi: 10.26593/josc.v1i1.5151.

[12] P. Arafin, A. M. Billah, and A. Issa, "Deep learning-based concrete defects classification and detection using semantic segmentation," *Structural Health Monitoring*, vol. 23, no. 1, pp. 383–409, Jan. 2024, doi: 10.1177/14759217231168212.

[13] Sara Shomal Zadeh, Sina Aalipour birgani, Meisam Khorshidi, and Farhad Kooban, "Concrete Surface Crack Detection with Convolutional-based Deep Learning Models," Nov. 2023, doi: 10.5281/ZENODO.10061654.

[14] L. Ali, F. Alnajjar, H. A. Jassmi, M. Gocho, W. Khan, and M. A. Serhani, "Performance Evaluation of Deep CNN-Based Crack Detection and Localization Techniques for Concrete Structures," *Sensors*, vol. 21, no. 5, p. 1688, Mar. 2021, doi: 10.3390/s21051688.

[15] C. L. Nguyen, A. Nguyen, J. Brown, T. Byrne, B. T. Ngo, and C. X. Luong, "Optimising Concrete Crack Detection: A Study of Transfer Learning with Application on Nvidia Jetson Nano," *Sensors*, vol. 24, no. 23, p. 7818, Dec. 2024, doi: 10.3390/s24237818.

[16] E. Wahyudi, B. Imran, A. Subki, Z. Zaeniah, L. D. Samsumar, and S. Salman, "Crack Detection of Concrete Surfaces with A Combination of Feature Extraction and Image-Based Backpropagation Artificial Neural Networks," *Ilk. J. Ilm.*, vol. 16, no. 3, pp. 228–235, Dec. 2024, doi: 10.33096/ilkom.v16i3.2249.228-235.

[17] M. A. Mohammed, Z. Han, and Y. Li, "Exploring the Detection Accuracy of Concrete Cracks Using Various CNN Models," *Advances in Materials Science and Engineering*, vol. 2021, no. 1, p. 9923704, Jan. 2021, doi: 10.1155/2021/9923704.

[18] W. Qayyum, R. Ehtisham, A. Bahrami, C. Camp, J. Mir, and A. Ahmad, "Assessment of Convolutional Neural Network Pre-Trained Models for Detection and Orientation of Cracks," *Materials*, vol. 16, no. 2, p. 826, Jan. 2023, doi: 10.3390/ma16020826.
[19] F. Yang, "Enhancing Concrete Crack Image Detection Using MobileNetV2 and Transfer Learning," *FSE*, vol. 4, no. 4, pp. 110–119, Apr. 2024, doi: 10.54691/ynk4nf60.
[20] K. Shibano, N. Morozova, Y. Shimamoto, N. Alver, and T. Suzuki, "Improvement of crack detectivity for noisy concrete surface by machine learning methods and infrared images," *Case Studies in Construction Materials*, vol. 20, p. e02984, July 2024, doi: 10.1016/j.cscm.2024.e02984.
[21] Nyathi, Mthabisi Adriano, Bai, Jiping, and Wilson, Ian David, "NYA-Crack-Data: A High Variability Concrete Crack Dataset for Enhanced Model Generalisation." Mendeley data. doi: doi:%2010.17632/z93rb2m4fk.1.
[22] W. Setiawan, M. M. Suhadi, Y. D. Pramudita, and Mulaab, "Inception-v3 with reduce learning rate for optimization of lung cancer histopathology classification," *ISI*, vol. 29, no. 2, pp. 561–570, Apr. 2024, doi: 10.18280/isi.290217.
[23] L. Yu, S. He, X. Liu, S. Jiang, and S. Xiang, "Intelligent Crack Detection and Quantification in the Concrete Bridge: A Deep Learning-Assisted Image Processing Approach," *Advances in Civil Engineering*, vol. 2022, no. 1, p. 1813821, Jan. 2022, doi: 10.1155/2022/1813821.
[24] Md. M. Islam, Md. B. Hossain, Md. N. Akhtar, M. A. Moni, and K. F. Hasan, "CNN Based on Transfer Learning Models Using Data Augmentation and Transformation for Detection of Concrete Crack," *Algorithms*, vol. 15, no. 8, p. 287, Aug. 2022, doi: 10.3390/a15080287.
[25] H. K. Kapadia, P. V. Patel, and J. B. Patel, "Convolutional Neural Network Based Improved Crack Detection In Concrete Cubes," *IJCDS*, vol. 13, no. 1, pp. 341–352, Jan. 2023, doi: 10.12785/ijcds/130127.
[26] S. Cong and Y. Zhou, "A review of convolutional neural network architectures and their optimizations," *Artif Intell Rev*, vol. 56, no. 3, pp. 1905–1969, Mar. 2023, doi: 10.1007/s10462-022-10213-5.
[27] M. Krichen, "Convolutional Neural Networks: A Survey," *Computers*, vol. 12, no. 8, p. 151, July 2023, doi: 10.3390/computers12080151.
[28] G. Statnikov, S. Karthik, Y. Boumhaout, and A. Elfikky, "A Comprehensive Survey of Convolutions in CNNs: Evolution, Architectures, and Applications," Oct. 14, 2025, *Computer Science and Mathematics*. doi: 10.20944/202510.1050.v1.
[29] L. Hui, A. Ibrahim, and R. Hindi, "Computer Vision-Based Concrete Crack Identification Using MobileNetV2 Neural Network and Adaptive Thresholding," *Infrastructures*, vol. 10, no. 2, p. 42, Feb. 2025, doi: 10.3390/infrastructures10020042.