Klasifikasi Citra Gerakan Olahraga Dalam *Gym* Menggunakan *Graph Convolutional Networ*

Affan Rifqy Kurniadi[1)], Akmal[2)], Deni Setiana[3)]

[1,2,3)] Teknik Informatika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Padjadjaran
Jl. Raya Bandung Sumedang KM.21, Hegarmanah, Kec. Jatinangor, Kabupaten Sumedang, Jawa Barat 45363
Email: affan20001@mail.unpad.ac.id
Email: akmal@unpad.ac.id
Email: deni@unpad.ac.id

Abstract: *The participation rate in sports activities among Indonesians remains low, with the Sport Development Index (SDI) in 2022 recording only 30.93%, a decline from 32.80% in the previous year. Simple kind of sport can be followed is gym. This study aims to introduce and promote basic gym movements such as bench press, squat, and deadlift to encourage greater engagement in sports activities. This research utilizes Deep Learning technology based on Graph Convolutional Network (GCN) to classify gym movement images into three classes: benchpress, squat, and deadlift. The study focuses on comparing various hyperparameters, including model type, batch size, and dropout, to determine the optimal configuration with the best performance.The results indicate that the GCN model achieved an F1 Score of 0.8667, demonstrating strong performance in classifying gym movement images. A simple web-based application was developed as an implementation to facilitate automatic gym movement classification.*
Keywords: *gym, deep learning, graph convolutional network, F1-Score, web*

Abstrak: *Tingkat partisipasi olahraga masyarakat Indonesia masih rendah, dengan Sport Development Index (SDI) tahun 2022 mencatat hanya 30,93%, turun dari 32,80% pada tahun sebelumnya. Olahraga sederhana yang dapat diikuti salah satunya adalah olahraga gym. Penelitian ini bertujuan untuk mengenalkan dan mempromosikan gerakan dasar gym seperti benchpress, squat, dan deadlift guna mendorong peningkatan partisipasi olahraga. Penelitian menggunakan teknologi Deep Learning berbasis Graph Convolutional Network (GCN) untuk mengklasifikasikan citra gerakan gym ke dalam tiga kelas tersebut. Fokus penelitian adalah membandingkan berbagai hyperparameter, termasuk model, ukuran batch, dan dropout, untuk menemukan konfigurasi optimal dengan performa terbaik. Hasil penelitian menunjukkan bahwa model GCN mencapai nilai F1 Score sebesar 0,8667, menunjukkan performa yang baik dalam klasifikasi citra gerakan gym. Sebagai implementasi, aplikasi berbasis web sederhana dikembangkan untuk memfasilitasi klasifikasi gerakan gym secara otomatis.*
Kata kunci: *gym, deep learning, graph convolutional network, F1-Score, web*

I. PENDAHULUAN

Tingkat partisipasi olahraga masyarakat Indonesia mengalami penurunan menjadi 25,4% pada tahun 2023, lebih rendah 3% dibandingkan tahun sebelumnya, yang berdampak pada penurunan kebugaran jasmani sebesar 0,6% (Sport Development Index, 2023). Kurangnya aktivitas fisik dapat mengurangi kekebalan tubuh dan meningkatkan risiko penyakit [1]. Oleh karena itu, edukasi mengenai pentingnya aktivitas fisik, termasuk latihan di gym, diperlukan untuk meningkatkan kebugaran masyarakat.

Gerakan seperti *benchpress*, *squat*, dan *deadlift* memiliki peran penting dalam meningkatkan kekuatan otot dan daya tahan tubuh karena melibatkan aktivasi otot tubuh atas maupun bawah [2][3][4]. Untuk membantu masyarakat memahami dan melakukan gerakan ini dengan benar, sistem klasifikasi berbasis *Deep Learning* dikembangkan guna mengenali gerakan secara lebih akurat.

Convolutional Neural Network (CNN) telah lama menjadi standar dalam klasifikasi citra karena efisiensi parameter dan kemampuannya melakukan generalisasi yang baik [5]. Namun, dalam beberapa tahun terakhir, *Graph Neural Network* (GNN) mulai mendapatkan perhatian karena kemampuannya dalam melakukan klasifikasi citra.

II. METODE PENELITIAN

A. *Benchpress*

Benchpress dilakukan dengan posisi berbaring telentang di bangku, barbel diangkat dari rak di atas dada bagian atas menggunakan pegangan *overhand*, lalu diturunkan ke dada bagian tengah sebelum didorong kembali ke atas hingga lengan lurus. Latihan ini bermanfaat untuk menilai dan meningkatkan kekuatan, tenaga, dan daya tahan tubuh bagian atas, serta mendukung pengembangan otot untuk kinerja atletik dan pekerjaan fungsional [4][6].

B. *Squat*

Squat adalah latihan yang dilakukan dengan menurunkan tubuh dari posisi tegak hingga kedalaman tertentu, lalu kembali naik ke posisi awal. Squat dapat dilakukan menggunakan berat badan, barbel, atau mesin. Latihan ini efektif dalam meningkatkan kekuatan (*strength*), hipertrofi otot tungkai, dan pertumbuhan otot, menjadikannya bagian penting dari program penguatan otot [2][7].

C. *Deadlift*

Deadlift tradisional atau biasa disebut *deadlift* konvensional dimulai dengan posisi jongkok, barbel diangkat dari lantai ke posisi setengah paha melalui gerakan meluruskan sendi pinggul dan lutut. Latihan ini melibatkan aktivasi simultan banyak kelompok otot tubuh, yang memberikan tekanan signifikan pada sistem muskuloskeletal dan mendorong adaptasi kekuatan, terutama di pinggul, paha, dan punggung [3][7][8].

D. *Graph Convolutional Network* (GCN)

GCN adalah varian *Convolutional Neural Network* (CNN) yang dirancang khusus untuk data berbentuk graf. Dalam GCN, representasi tersembunyi dari sebuah node dihitung dengan mengambil rata-rata fitur node tersebut dan tetangganya. Tidak seperti data gambar yang memiliki struktur tetap, data graf memiliki tetangga yang tidak berurutan dan jumlah yang bervariasi.

Pada 2D *convolutional*, setiap piksel dianggap sebagai *node* yang terhubung dengan tetangga tetap berdasarkan ukuran filter. Sebaliknya, pada *graph convolution*, *node* dalam graf terhubung berdasarkan struktur graf yang fleksibel, di mana nilai fitur tetangga dihitung tanpa urutan tetap [9].

E. MobileViG

MobileViG adalah arsitektur berbasis graf yang lebih efisien dibandingkan Vision GNN (ViG) dan cocok untuk perangkat *mobile*. MobileViG menggunakan *Sparse Vision Graph Attention* (SVGA) yang membangun graf terstruktur tetap untuk setiap gambar masukan, mengurangi beban komputasi yang dibutuhkan oleh metode *k-nearest neighbors* (KNN).

KNN *graph attention* membangun koneksi antar piksel berdasarkan hubungan jarak tertentu yang perlu dihitung ulang setiap kali. SVGA, di sisi lain, menggunakan koneksi tetap antara piksel yang konsisten untuk semua gambar, menghilangkan kebutuhan perhitungan ulang graf dan membuatnya lebih efisien [10].

E. *Confusion Matrix*

Confusion Matrix adalah tabel evaluasi yang membandingkan prediksi model dengan label sebenarnya untuk setiap kelas. Tabel ini memungkinkan evaluasi performa model berdasarkan sampel yang diklasifikasikan benar (*True Positive, True Negative*) dan salah (*False Positive, False Negative*). Dari Confusion Matrix, metrik evaluasi seperti *Precision, Recall*, dan *F1 Score* dapat dihitung sebagai berikut:

- *Precision* digunakan mengukur akurasi prediksi kelas positif dengan rumus berikut:

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

- *Recall* digunakan untuk mengukur sensitivitas atau rasio deteksi kelas positif dengan rumus berikut:

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

- *F1 Score* merupakan rata-rata harmonik dari *precision* dan *recall*, dihitung dengan rumus berikut:

$$F1\ Score = 2\ x\ \frac{Precision\ x\ Recall}{Precision+Recall} \quad (3)$$

Metrik ini memberikan gambaran menyeluruh tentang performa model, khususnya untuk *dataset* yang tidak seimbang, dengan mempertimbangkan keseimbangan antara kemampuan model dalam mendeteksi

setiap kelas secara akurat dan menyeluruh, serta menghindari bias terhadap kelas yang lebih dominan [11][12].

F. Fase Analisis (*Software & Hardware*)

1. Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak dalam penelitian ini meliputi beberapa *tools* penting, yaitu Sistem Operasi Microsoft Windows 10 sebagai *platform* utama, Python 3.10 untuk pengembangan dan implementasi model, GitHub sebagai alat manajemen versi dan kolaborasi, Visual Studio Code sebagai *Integrated Development Environment* (IDE), serta Figma untuk kebutuhan desain antarmuka dan visualisasi.

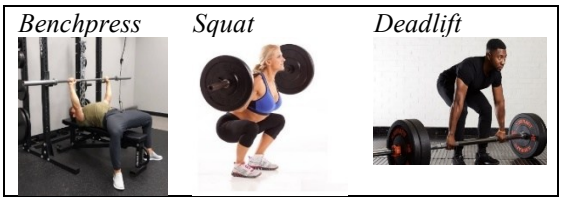
2. Kebutuhan Perangkat Keras

Kebutuhan perangkat keras yang digunakan dalam penelitian ini mencakup prosesor AMD Ryzen 5 3550H untuk mendukung komputasi, penyimpanan SSD berkapasitas 512 GB dan 256 GB untuk data dan aplikasi, RAM 16 GB 2400 MHz untuk kelancaran proses *multitasking*, serta GPU AMD Radeon RX 560X untuk mendukung pemrosesan grafis dan pelatihan model berbasis AI

G. Fase Penelitian (Data & Model)

1. Pengumpulan Data

Pengumpulan data diambil dari GitHub proyek "*Using Human Pose Detection to Identify and Give Feedback on Workout Form*" yang diakses pada 1 Maret 2024. *Dataset* terdiri dari 546 citra gym yang dibagi ke tiga kelas yaitu *benchpress* (154 citra), *squat* (195 citra), dan *deadlift* (197 citra), dengan variasi sudut pandang.



Gambar 1 Sampel Citra Gerakan

Gambar 1 menunjukan sampel dari tiap kelas dalam *dataset*. *Dataset* ini digunakan untuk melatih model dalam mengenali gerakan gym dan diproses lebih lanjut untuk memastikan kualitasnya.

2. *Data Preprocessing*

Data preproccessing dilakukan untuk meningkatkan kualitas dan relevansi data agar model dapat mengenali fitur dengan lebih baik. Langkah pertama adalah pemilahan data, di mana citra yang tidak sesuai dengan kriteria kelasnya, seperti gerakan *front squat* pada kelas *squat* atau citra yang salah kategori, dihapus atau dipindahkan ke kelas yang benar. Selanjutnya, dilakukan perbaikan pada citra dengan latar belakang yang kompleks dengan cara *crop* agar fokus hanya pada individu yang melakukan gerakan. Untuk meningkatkan jumlah data, dilakukan penambahan citra melalui *scraping* menggunakan *bing_image_downloader*, menambahkan masing-masing 100 citra per kelas. Setelah itu, data tambahan ini kembali melalui proses pemilahan dan perbaikan untuk memastikan konsistensi. Tahap berikutnya adalah perubahan ukuran citra menjadi 224x224 piksel agar sesuai dengan standar *input* model. Terakhir, dilakukan augmentasi data untuk meningkatkan variasi citra menggunakan teknik seperti random *horizontal flip*, rotasi hingga 15 derajat, serta penyesuaian *brightness*, *contrast*, *saturation*, dan *hue* sebesar 20%. Tahap-tahap ini memastikan *dataset* memiliki kualitas yang optimal untuk pelatihan model.

3. *Data Splitting*

Data splitting pada *dataset* akhir yang berjumlah 300 citra, dengan pembagian merata masing-masing 100 citra untuk *benchpress*, *squat*, dan *deadlift*. Data dipisahkan menjadi 270 citra untuk pelatihan (*training*) dan 30 citra untuk pengujian (*testing*). Pemisahan dilakukan otomatis menggunakan fungsi berbasis Python dengan *train_test_split* dari *sklearn*,

memastikan data terorganisasi sesuai kebutuhan pelatihan.

4. Pelatihan Model

Dataset yang telah dibagi selajutnya akan dilatih dengan model GCN yang menggunakan dua arsitektur, yaitu MobileViG-Ti dan MobileViG-S, dengan tiga skenario *batch size* (16, 9, 7) dan enam skenario pelatihan menggunakan variasi *dropout* (0, 0.3, 0.5).

Tabel 1 Skenario Pelatihan Model

Skenario	*Model*	*Optimizer*	*Dropout Rate*
1	Mo.Vig-Ti	AdamW	*No Dropout*
2	Mo,Vig-Ti	AdamW	*0,3*
3	Mo,Vig-Ti	AdamW	*0,5*
4	Mo.Vig-S	AdamW	*No Dropout*
5	Mo.Vig-S	AdamW	*0,3*
6	Mo.Vig-S	AdamW	*0,5*

Tabel 1 menunjukan skenario dari penerapan model. Setiap skenario bertujuan mengevaluasi pengaruh arsitektur, *dropout*, dan *batch size* terhadap performa model ber- dasarkan *F1 Score* yang dihasilkan. Model dilatih dengan *learning rate* 0.0001 selama 50 *epoch*, meng- gunakan matriks pengukuran seperti *training accuracy*, *testing accuracy*, *training loss*, dan *testing loss*.

5. Evaluasi Model

Evaluasi performa akan menggunakan metode *confusion matrix,* berdasarkan hasil ini akan terlihat keseluruhan nilai *True Positive, True Negative, False Positive,* dan *False Negative* untuk kelas *trash* pada *dataset*. Berdasarkan hasil tersebut maka nilai *Precision, Recall* dan *F1 Score* dapat dihitung.

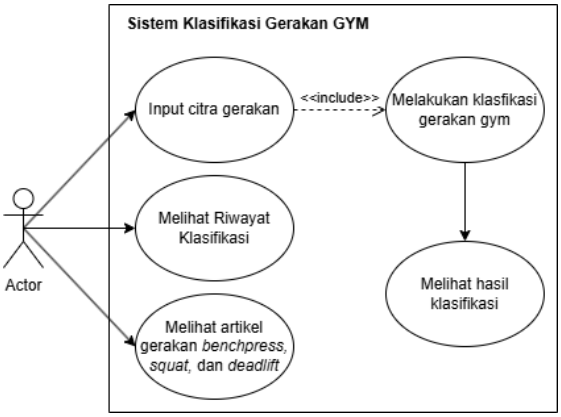
6. Konversi Model

Model terbaik dari semua skenario dikonversi ke format pth untuk diintegrasikan ke aplikasi berbasis *web* menggunakan Flask, sehingga dapat memproses dan memprediksi *input* citra secara langsung.

H. Fase Perancangan Aplikasi

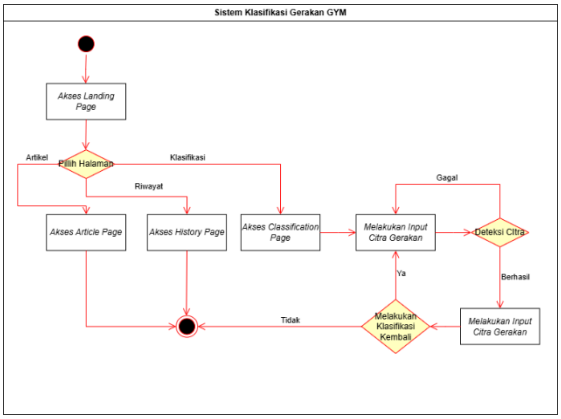
1. *Use Case Diagram*

Pada Gambar 2, aktor user memiliki peran utama dalam sistem. User dapat meng klasifikasikan citra gerakan gym dengan terlebih dahulu mengunggah citra gerakan. Setelah itu, user dapat melihat hasil klasifikasi, mengakses riwayat klasifikasi yang pernah dilakukan, serta membaca artikel terkait gerakan gym yang dapat dikenali oleh aplikasi.



Gambar 2 *Use Case Diagram*

2. *Activity Diagram*

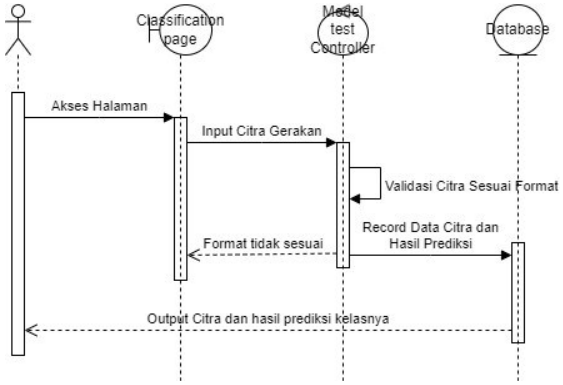


Gambar 3 *Activity Diagram*

Pada Gambar 3, aktivitas utama yang dapat dilakukan oleh *user* adalah klasifikasi citra gerakan gym, melihat riwayat hasil klasifikasi, dan membaca artikel tentang gerakan yang dapat dikenali oleh aplikasi. Untuk melakukan klasifikasi, *user* dapat mengakses halaman *classification page* dan mengunggah citra gerakan gym. Jika citra yang diunggah berhasil terdeteksi, model akan memproses citra tersebut dan menampilkan prediksi kelas gerakan beserta citra hasil klasifikasinya. Setelah itu, *user* dapat memilih untuk melakukan klasifikasi citra lain atau kembali ke menu utama. Selain melakukan klasifikasi, *user* juga dapat melihat riwayat hasil klasifikasi yang pernah dilakukan pada aplikasi ini. Selain itu, *user* dapat mengakses halaman artikel untuk membaca informasi terkait gerakan gym

yang dapat dikenali oleh aplikasi, memberikan wawasan tambahan mengenai gerakan yang telah diklasifikasikan.

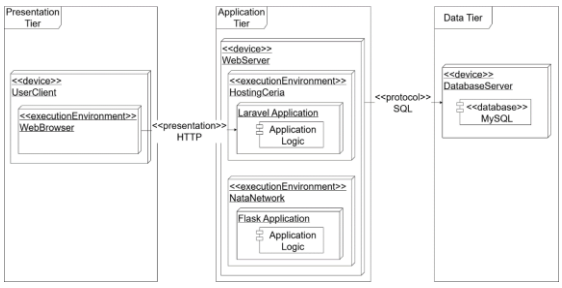
3. *Sequence Diagram*



Gambar 4 *Sequence Diagram*

Pada Gambar 4, terdapat aktor yaitu *user* yang dapat meng-*input* citra melalui halaman klasifikasi. Setelah citra di-input, *Model Test Controller* akan memvalidasi format citra, memastikan bahwa format yang diunggah sesuai dengan ketentuan yang telah ditetapkan (misalnya, ukuran atau ekstensi *file*). Jika formatnya sesuai, citra akan diklasifikasikan oleh model menggunakan algoritma yang telah dilatih sebelumnya, dan hasil prediksi beserta citra tersebut akan disimpan dalam *database* untuk keperluan pencatatan dan analisis lebih lanjut. Selanjutnya, sistem akan memproses hasil prediksi dan citra tersebut untuk ditampilkan kembali kepada *user* melalui halaman prediksi, lengkap dengan informasi tambahan seperti tingkat kepercayaan model terhadap hasil klasifikasi. Hal ini bertujuan untuk memberikan transparansi serta pengalaman interaksi yang optimal bagi pengguna.

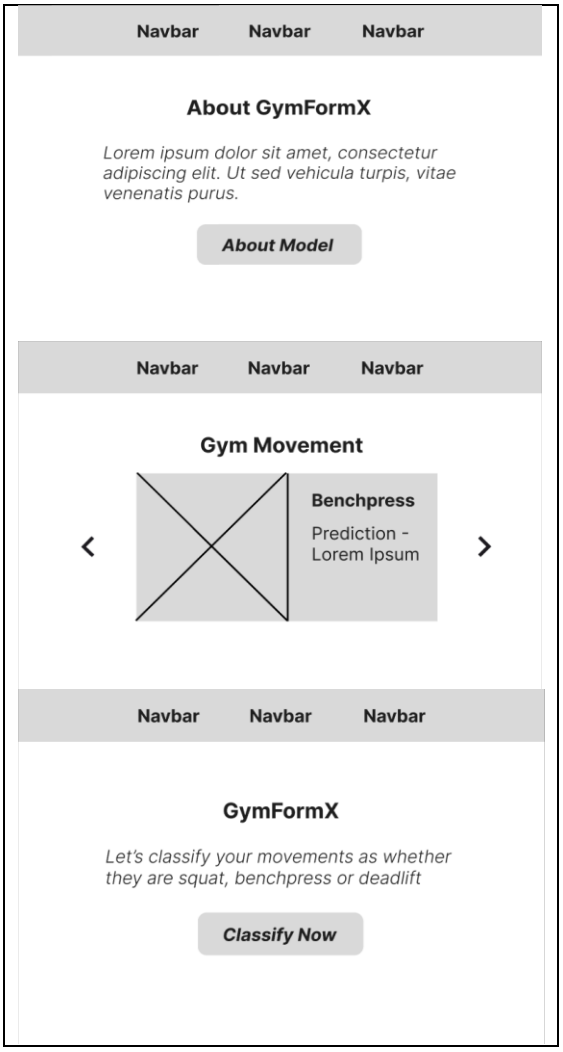
4. *Sequence Diagram*



Gambar 5 *Deployment Diagram*

Pada Gambar 5, terdapat tiga tier yaitu *presentation tier, application tier,* dan *data tier*. *Presentation tier* terdiri dari *web browser* yang memungkinkan *user* mengakses aplikasi. *Application tier* mencakup *web server* dengan dua komponen utama yaitu Laravel Application untuk *hosting* aplikasi dan Flask Application untuk memprediksi citra *input*. *Data tier* mencakup *database server* dengan MySQL untuk menyimpan hasil aplikasi. *Web browser* berhubungan dengan Laravel Application untuk permintaan *user*, sementara Laravel Application berkomunikasi dengan Flask Application untuk klasifikasi citra dan dengan *database server* untuk menyimpan data. Setiap *tier* saling terhubung secara terintegrasi untuk memastikan alur kerja yang efisien dan pengalaman pengguna yang optimal.

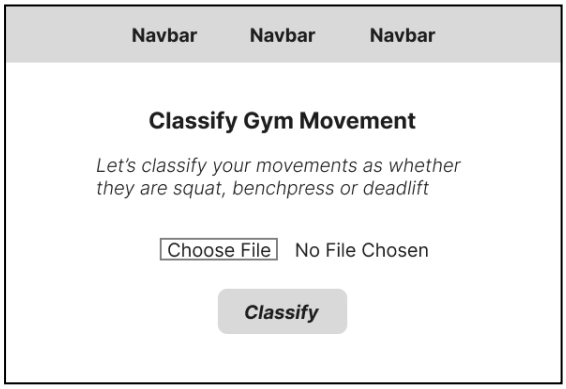
5. *Wireframe* Aplikasi



Gambar 6 *Wireframe Landing Page*

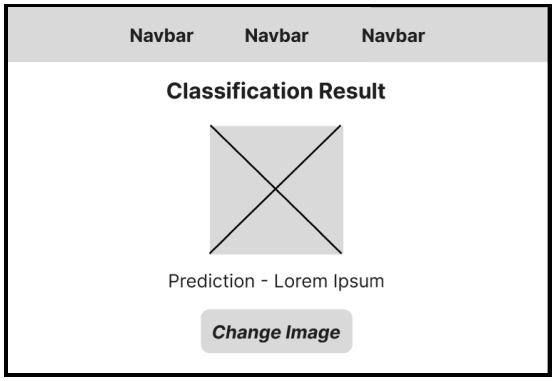
Gambar 6 menunjukkan *wireframe* untuk *landing page*. Terdapat tiga *section* untuk *landing page*, yaitu *section* paling atas

merupakan *section about model*, yang menjelaskan informasi singkat mengenai model AI yang digunakan, termasuk keunggulan dan cara kerjanya secara ringkas. Kemudian, di bawahnya terdapat *section* untuk artikel mengenai gerakan dalam gym, yang memberikan wawasan dan edukasi tentang berbagai gerakan latihan serta manfaatnya bagi pengguna. Lalu, *section* paling bawah merupakan *section* untuk tombol menuju klasifikasi, yang mengarahkan *user* langsung ke halaman klasifikasi untuk mulai mengunggah citra. Ketiga *section* ini disusun secara berurutan untuk memberikan alur navigasi yang logis dan pengalaman *user* yang informatif, sekaligus mempermudah akses ke fitur utama aplikasi.



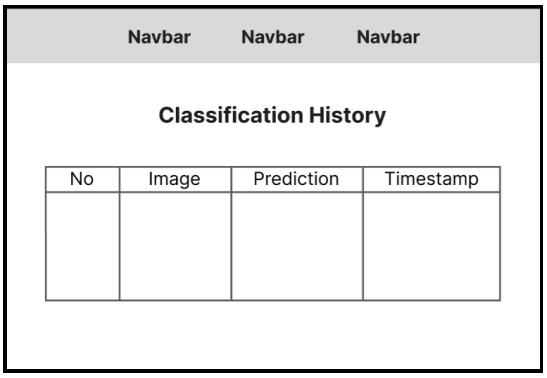
Gambar 7 Wireframe Classification Page

Pada Gambar 7, Terdapat sebuah *form* yang berisi *input* yang memungkinkan pengguna untuk mengunggah gambar. Pengguna dapat mengirimkan gambar yang sudah diunggah melalui tombol di bawah *form*.



Gambar 8 Wireframe Prediction Page

Pada Gambar 8, hasil prediksi dari gambar yang diunggah oleh pengguna akan ditampilkan di halaman ini, termasuk kelas yang diprediksi oleh model. Pengguna dapat mengganti gambar dengan kembali ke halaman *classification page* melalui tombol yang berada di bawah hasil prediksi.



Gambar 9 Wireframe History Page

Pada Gambar 9, citra gerakan yang telah diunggah oleh pengguna beserta hasil prediksinya dapat dilihat dalam bentuk tabel. Tabel ini berisi nomor, citra yang diunggah, hasil prediksi, serta *timestamp* atau waktu dan tanggal ketika pengguna mengunggah gambar pada *form*.

III. HASIL DAN PEMBAHASAN

A. Analisis Perhitungan Performa (*Testing Result*)

Tahapan ini menganalisis dan menghitung performa model berdasarkan evaluasi skenario pelatihan. Performa diukur menggunakan persamaan pada bab II, dengan hasil disajikan dalam tabel untuk setiap percobaan di bab III. Analisis dilakukan berdasarkan *Confusion Matrix* dari setiap skenario, menggunakan F1 Score dari hasil *testing* untuk menilai akurasi model. Hasil *Precision, Recall,* dan *F1 Score* dari percobaan 1 (*batch size* 16) untuk skenario 1 hingga 6 dirangkum pada Tabel 2.

Tabel ini memberikan gambaran komparatif tentang performa model pada setiap skenario, sehingga memudahkan analisis terhadap pengaruh parameter seperti penggunaan *dropout* dan arsitektur model yang berbeda. Data pada tabel ini juga membantu dalam mengevaluasi konsistensi performa model pada berbagai kondisi pengujian.

Tabel 2 *Precision*, *Recall*, *F1 Score* Percobaan 1

Skenario	*Hyper Parameter*	*Precision*	*Recall*	*F1 Score*
1	MobileVig-Ti; AdamW; No Dropout:	0,8214	0,7667	0,7410
2	MobileVig-Ti; AdamW; Dropout(0,3);	0,7228	0,6667	0,6458
3	MobileVig-Ti; AdamW; Dropout(0,5);	0,7778	0,7667	0,7538
4	MobileVig-S; AdamW; No Dropout;	0,8069	0,7667	0,7701
5	MobileVig-S; AdamW; Dropout(0,3);	0,8371	0,8333	0,8307
6	MobileVig-S; AdamW; Dropout(0,5);	0,8387	0,8333	0,8347

Percobaan 2 (*batch size* sebesar 9) untuk nilai *Precision, Recall,* dan *F1 Score* dari hasil testing dari skenario 1 hingga 6 dapat dilihat pada Tabel 3.

Tabel 3 *Precision*, *Recall*, *F1 Score* Percobaan 2

Skenario	*Hyper Parameter*	*Precision*	*Recall*	*F1 Score*
1	MobileVig-Ti; AdamW; No Dropout:	0,6587	0,6333	0,6250
2	MobileVig-Ti; AdamW; Dropout(0,3);	0,6766	0,6667	0,6613
3	MobileVig-Ti; AdamW; Dropout(0,5);	0,6774	0,6667	0,6646
4	MobileVig-S; AdamW; No Dropout;	0,7348	0,7333	0,7302
5	MobileVig-S; AdamW; Dropout(0,3);	0,7781	0,7667	0,7620
6	MobileVig-S; AdamW; Dropout(0,5);	0,7283	0,7333	0,7296

Percobaan 3 (*batch size* sebesar 7) untuk nilai *Precision*, *Recall*, dan *F1 Score* dari hasil *testing* dari skenario 1 hingga 6 dapat dilihat pada Tabel 4.

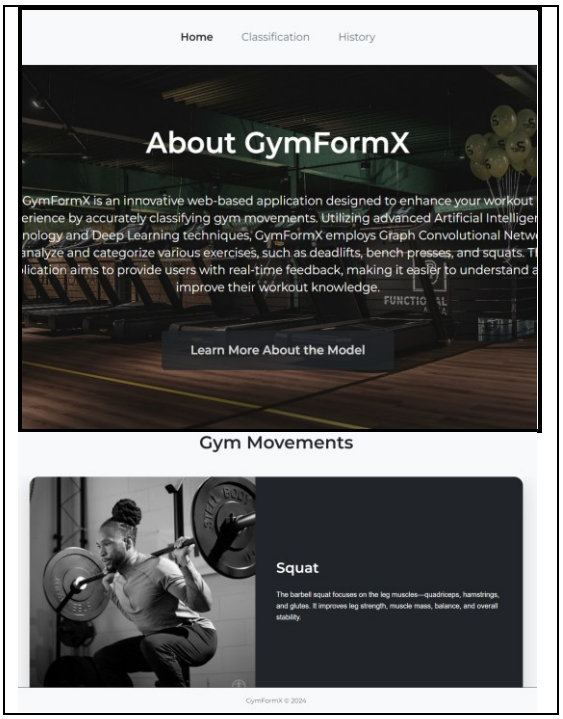
Tabel 4 *Precision*, *Recall*, *F1 Score* Percobaan 3

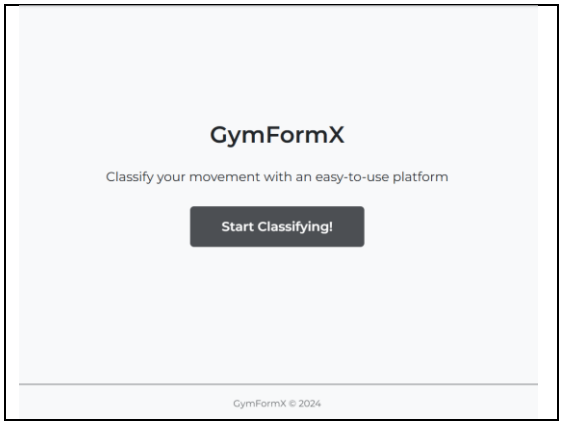
Skenario	*Hyper Parameter*	*Precision*	*Recall*	*F1 Score*
1	MobileVig-Ti; AdamW; No Dropout:	0,8083	0,8000	0,7987
2	MobileVig-Ti; AdamW; Dropout(0,3);	0,8778	0,8667	0,8660
3	MobileVig-Ti; AdamW; Dropout(0,5);	0,8667	0,8667	0,8667
4	MobileVig-S; AdamW; No Dropout;	0,6329	0,6333	0,6095
5	MobileVig-S; AdamW; Dropout(0,3);	0,7313	0,7333	0,7312
6	MobileVig-S; AdamW; Dropout(0,5);	0,7197	0,7000	0,6811

Pada seluruh percobaan dapat terlihat model terlatih pada percobaan 3 skenario 3 memiliki *F1 Score* paling optimal di antara percobaan dan skenario lainnya dengan *F1 Score* sebesar 0,8667. Model terlatih tersebut akan dijadikan sebagai model utama untuk klasifikasi pada implementasi website. Pemilihan ini tidak hanya didasarkan pada nilai *F1 Score* yang tinggi, tetapi juga pada konsistensi performa model terhadap metrik evaluasi lainnya, seperti *precision* dan *recall*. Dengan memilih model ini, diharapkan aplikasi dapat memberikan hasil prediksi yang lebih akurat, relevan, dan mampu memenuhi kebutuhan pengguna di lingkungan operasional sebenarnya.

B. Implementasi Pembuatan Aplikasi (*Website*)

1. *Landing Page*

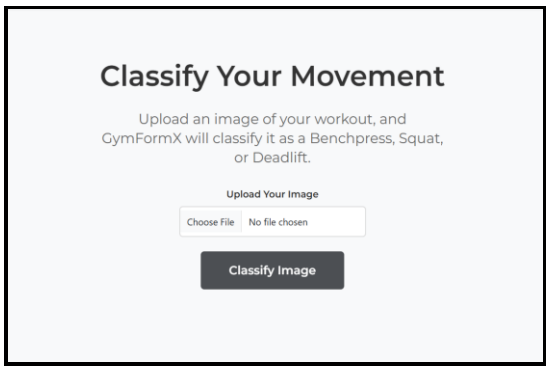




Gambar 10 *Landing Page*\

Gambar 10 menunjukkan *landing page* dengan *navigation bar* yang menyediakan tombol *Home* untuk kembali ke *landing page*, *Classification* untuk mengakses halaman klasifikasi, dan *History* untuk melihat riwayat klasifikasi. Halaman ini memiliki tombol *About Model* di bagian atas untuk informasi detail model, diikuti dengan *section* artikel mengenai gerakan gym. Selain itu, terdapat judul aplikasi, *tag line*, dan tombol menuju halaman klasifikasi pada *section* paling bawah. Desain ini memastikan bahwa pengguna dapat menemukan fitur utama dengan cepat.

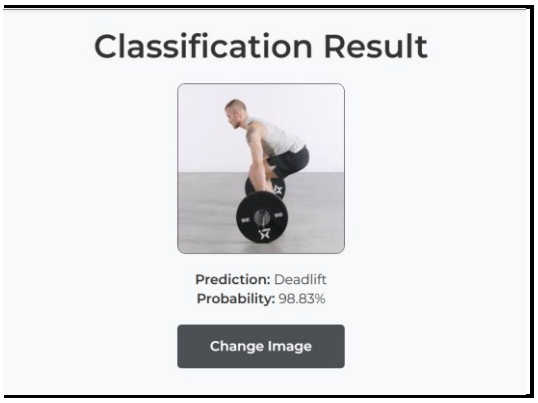
2. *Classification Page*



Gambar 11 *Classification Page*

Gambar 11 menunjukkan *classification page* yang dirancang untuk mengklasifikasikan citra gerakan gym ke dalam kelas yang tersedia. Halaman ini menyediakan *form input file* untuk mengunggah gambar. Setelah *file* gambar berhasil diunggah, pengguna dapat menekan tombol *Classify Image* untuk mengirimkan gambar ke model untuk diprediksi.

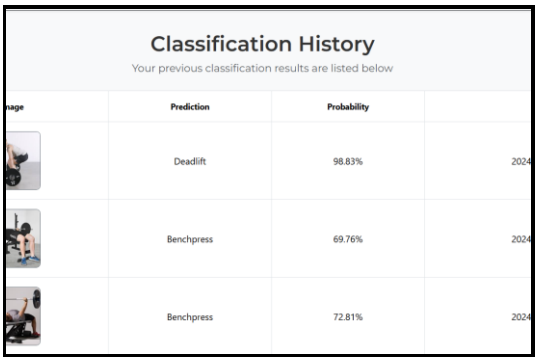
3. *Prediction Page*



Gambar 12 *Prediction Page*

Gambar 12 menunjukkan *prediction page* yang menampilkan hasil klasifikasi citra yang sebelumnya diunggah oleh pengguna. Sebagai contoh, model memprediksi citra tersebut sebagai kelas *deadlift* dengan probabilitas sebesar 98,83%. Jika pengguna ingin mengganti gambar untuk klasifikasi ulang, mereka dapat menekan tombol *Change Image*, yang akan mengarahkan kembali ke halaman *Classification Page*.

4. *History Page*



Gambar 13 *History Page*

Gambar *13* menunjukan *history page* yang dirancang untuk menampilkan riwayat klasifikasi citra yang telah dilakukan oleh pengguna melalui aplikasi ini. Pada halaman tersebut, terdapat sebuah tabel yang berisi beberapa kolom penting, yaitu nomor untuk mengidentifikasi urutan klasifikasi, gambar yang menunjukkan citra yang diunggah, hasil prediksi yang merepresentasikan kelas yang diprediksi oleh model, probabilitas yang mencerminkan tingkat keyakinan model terhadap prediksi tersebut, serta timestamp yang mencatat tanggal dan waktu saat klasifikasi. Halaman ini memungkinkan

pengguna untuk dengan mudah melacak dan meninjau kembali citra-citra yang telah diklasifikasikan beserta hasilnya secara detail.

IV. SIMPULAN

1. Model *Graph Convolutional Network* (GCN) berhasil diterapkan untuk klasifikasi gerakan olahraga gym menggunakan framework Flask dan *library* PyTorch. *Dataset* terdiri dari 3 kelas, yaitu *Benchpress*, *Squat*, dan *Deadlift*, yang diambil dari platform GitHub pada proyek *Using Human Pose Detection to Identify and Give Feedback on Workout Form*. Model ini mampu mengenali pola gerakan secara efektif.
2. Konfigurasi *hyperparameter* yang optimal ditemukan melalui 18 skenario pelatihan, memadukan arsitektur MobileViG, Batch Size, dan Dropout layer. Hasil terbaik diperoleh dengan arsitektur MobileViG-Ti, Batch Size 7, dan Dropout 0,5, menghasilkan F1 Score 0,8667.
3. Skenario pelatihan terbaik menunjukkan keseimbangan performa model dengan nilai Precision, Recall, dan F1 Score masing-masing sebesar 0,8667, menegaskan kemampuan model dalam mendeteksi dan mengklasifikasikan citra menjadi kelas Benchpress, Squat, atau Deadlift.
4. Penerapan model GCN pada aplikasi berbasis *web* dilakukan melalui Flask Application. Model dengan performa terbaik dikonversi ke format pth untuk menerima *input* citra, memprediksi kelas, dan menampilkan hasil melalui antarmuka berbasis *website*.

DAFTAR RUJUKAN

[1] Hearing, C. M., Chang, W. C., Szuhany, K. L., Deckersbach, T., Nierenberg, A. A., & Sylvia, L. G. (2016). Physical exercise for treatment of mood disorders: a critical review. *Current behavioral neuroscience reports*, *3*, 350-359.

[2] Aryadi Rachman. (2014). Pengaruh Latiahn Squat Dan Leg Press Terhadap Strength Dan Hypertrophy Otot. Jurnal Pendidikan Jasmani Dan Olahraga, 13(2).

[3] Braidot, A. A., Brusa, M. H., Lestussi, F. E., & Parera, G. P. (2007). Biomechanics of front and back squat exercises. Journal of Physics: Conference Series, 90, 012009.

[4] Ronai, P. (2018). The Bench Press Exercise. ACSM'S Health & Fitness Journal, 22(6), 52–57.

[5] Esaki Muthu Pandara Kone, S., Yatsugi, K., Mizuno, Y., & Nakamura, H. (2022). Application of convolutional neural network for fault diagnosis of bearing scratch of an induction motor. *Applied Sciences*, *12*(11), 5513.

[6] García-López, D., Izquierdo, M., Rodríguez, S., González-Calvo, G., Sainz, N., Abadía, O., & Herrero, A. J. (2010). Interset Stretching Does Not Influence the Kinematic Profile of Consecutive Bench-Press Sets. Journal of Strength and Conditioning Research, 24(5), 1361–1368.

[7] Escamilla, R. F., Fleisig, G. S., Zheng, N., Barrentine, S. W., Wilk, K. E., & Andrews, J. R. (1998). Biomechanics of the knee during closed kinetic chain and open kinetic chain exercises. Medicine &amp Science in Sports &amp Exercise, 30(4), 556–569.

[8] Earl, J. E., Schmitz, R. J., & Arnold, B. L. (2001). Activation of the VMO and VL during dynamic mini-squat exercises with and without isometric hip adduction. Journal of Electromyography and Kinesiology, 11(6), 381–386.

[9] Kipf, T. N., & Welling, M. (2017). Semi-Supervised Classification with Graph Convolutional Networks.

[10] Munir, M., Avery, W., & Marculescu, R. (2023). MobileViG: Graph-Based Sparse Attention for Mobile Vision Applications. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2211–2219.

[11] Joshi, P. (2017). Artificial Intelligence with Python. Packt Publishing Ltd.

[12] Géron, A. (2022). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (3rd ed.). O'Reilly Media, In