ULIL ALBAB : Jurnal Ilmiah Multidisiplin
Vol.3, No.10, September 2024

Penerapan Algoritma Convolutional Neural Network Pada Identifikasi Foto Hero Game Mobile Legends

Moch. Anas Toybah[1], Daniel Swanjaya[2]
[1,2] Fakultas Teknik dan Ilmu Komputer (FTIK), Universitas Nusantara PGRI Kediri
E-mail: anas.toybah123@gmail.com, daniel@unp.ac.id

Article History:
Received: 16 Agustus 2024
Revised: 30 Agustus 2024
Accepted: 03 September 2024

Keywords: *CNN, Classification, Hero, Mobile Legends*

Abstract: *This study aims to develop a hero photo identification system in the Mobile Legends game using the Convolutional Neural Network (CNN) algorithm. The background of this study is based on the need to speed up and simplify the hero identification process which often requires time and special knowledge. The CNN algorithm was chosen because of its superior ability in pattern recognition and image classification. The research method includes several stages, namely collecting Mobile Legends hero image data, data preprocessing including resizing and normalization, and dividing the data into training, validation, and testing data. The CNN model used consists of several convolution and pooling layers for feature extraction, and a fully connected layer for the final classification. Model training is carried out using a dataset that has been processed with augmentation techniques to increase data variation. The results of this study are that the data used amount to 600 image data divided into 30 classes. By implementing the CNN method, researchers have succeeded in creating a system that can recognize images of mobile legends heroes. Based on the scenario created by the researcher, the highest accuracy is a combination of ReLu activation, Dropout 0.2 and using epoch 20, resulting in an accuracy of 62.17%.*

PENDAHULUAN

Menurut sebuah lembaga penelitian, dari 83,7 juta orang dalam populasi negara ini, Indonesia termasuk dalam 20 besar pengguna internet di dunia (Katona, 2019). Dengan evolusi waktu, angka tersebut diperkirakan akan bertambah karena akses internet menjadi lebih terjangkau dan mudah. Segala macam aktivitas perorangan atau bersama membutuhkan konektivitas internet, termasuk di dalamnya adalah bermain game. Game adalah aktivitas interaktif di mana satu atau lebih pemain mengikuti aturan untuk menyelesaikan tantangan guna menghasilkan penilaian akhir (Sena, 2023). Dalam hal pengembang, para pemain juga memanfaatkan situasi ini untuk menjadi pemain profesional dalam sebuah game. Banyak perusahaan game yang berlomba-lomba untuk mengembangkan tim *E-Sports* mereka untuk

turnamen game.

Akronim "E-Sport" merupakan singkatan dari "electronic sport" yang merujuk pada berbagai cabang olahraga yang menggunakan perangkat elektronik sebagai sarana partisipasinya (Bayulianto, 2023). E-Sport di Indonesia sedang berkembang pesat, khususnya di kalangan anak muda, yang tampaknya tidak dapat hidup tanpa ponsel mereka. Mobile Legends Bang Bang merupakan e-sport yang populer dan dipertandingkan secara aktif di Indonesia. Studio Tiongkok Moonton menciptakan Mobile Legends Bang Bang yang termasuk dalam genre MOBA (Multiplayer Online Battle Arena). Menghancurkan markas lawan merupakan tujuan dari setiap pertempuran dalam mode permainan 5 lawan 5 ini. Lebih dari seratus hero yang dapat dimainkan tersedia di Mobile Legends Bang Bang, yang lebih dikenal sebagai MLBB (Listijo, 2020). Saat ini terdapat seratus dua puluh tiga Hero yang tersedia. Namun, ketika sebuah hero menggunakan skin, mungkin akan sangat sulit bagi sebagian orang yang bukan gamer untuk membedakannya dari banyak hero yang tersedia. Oleh karena itu, hasil yang dapat diandalkan memerlukan metode untuk mengidentifikasi hero Mobile Legends. Hal ini memotivasi saya untuk membuat sistem yang memilah hero Mobile Legends ke dalam kategori menggunakan Algoritma Jaringan Syaraf Konvolusional, menilai keberhasilan pendekatan saat ini, dan membuat aplikasi seluler untuk membantu pemain baru menemukan hero mereka.

Convolutional Neural Network adalah desain multilapis yang telah menunjukkan hasil yang baik dalam sejumlah aplikasi visi komputer, termasuk identifikasi objek, segmentasi gambar, dan NL (Saha, 2023). Maurício (2023) menjelaskan bahwa desain ini dimulai dengan lapisan konvolusi yang menggunakan filter atau kernel untuk mengekstrak informasi dari gambar yang bergerak secara horizontal dari kiri ke kanan. Ekstraksi fitur oleh manusia tidak diperlukan bagi CNN untuk mempelajari pola dan atribut dari gambar secara langsung (Taye, 2023). Kemampuan metode untuk memberikan prediksi yang tepat menjadikannya alat yang ampuh untuk kategorisasi gambar. Komponen utama dari pendekatan ini adalah kemampuannya untuk meniru pengenalan gambar manusia (Rahmadhani, 2023).

Dari beberapa penelitian terdahulu, metode *Convolutional Neural Network* terbukti efektif dan dapat digunakan untuk klasifikasi berdasarkan data gambar yang ada. Namun, belum jelas sejauh mana metode ini dapat diaplikasikan untuk klasifikasi jenis *hero* pada *Game Mobile Legends*. Oleh karena itu peneliti memilih metode *Convolutional Neural Network* dalam proses klasifikasi *hero* pada *Game Mobile Legends*.

METODE PENELITIAN

Penelitian ini menggunakan pendekatan deskriptif kualitatif dan deskriptif kuantitatif untuk menggambarkan dan mengukur kinerja klasifikasi *Hero* di *game mobile legends*. Prosedur penelitian akan mengikuti model *Waterfall* untuk pengembangan sistem deteksi

1. Identifikasi Masalah: Dalam sesi ini peneliti mencari pokok dari permasalahan yang dipilih, kemudian peneliti akan menganalisa kegiatan apa saja yang akan dilakukan selama penelitian.
2. Pemilihan Metode: Peneliti akan memilih metode apa yang akan digunakan sebagai solusi terbaik dalam mengatasi permasalahan. Dalam sesi ini peneliti menganalisa beberapa jurnal terkait penelitian terdahulu dengan berbagai metode dan berbagai objek yang berbeda.
3. Pengambilan Data: Data yang digunakan pada penelitian ini adalah data gambar yang didapat dari *website Mobile Legends Bang-Bang* dan beberapa *website* lain yang

membahas tentang *game mobile legends*. Data yang digunakan pada penelitian ini berupa gambar atau foto *hero mobile legends* dengan format jpg

4. Pengelompokan Data: data dikumpulkan dan dikelompokan sesuai dengan nama dan *role* dari setiap *hero* antara lain, *marksman, tank, fighter, support, assassin,* dan *mage*
5. Rancangan dan Desain Sistem: Peneliti akan membuat seebuah rancangan dari sistem yang akan dibuat. Rancangan sistem ini seperti pembuatan diagram dan lain sebagainya. Harapannya dengan pembuatan rancangan sistem, peneliti dapat mengimplementasikan proses yang telah disusun sedemikian rupa sehingga mendapatkan hasil yang memuaskan.
6. Implementasi: Data gambar akan diolah menggunakan metode *Convulotional Neural Network* untuk klasifikasi nama *hero* di *game mobile legends*. Implementasi ini dilakukan dengan membuat program sesuai dengan rancangan sistem yang telah dibuat.
7. Hasil dan Evaluasi: Pada tahap ini peneliti akan menganalisa hasil yang didapatkan setelah proses implementasi. Ketika hasil yang didapat kurang memuaskan peneliti akan melakukan evaluasi sehingga hasil yang didapatkan optimal.

HASIL DAN PEMBAHASAN

Implementasi Lembar Kerja

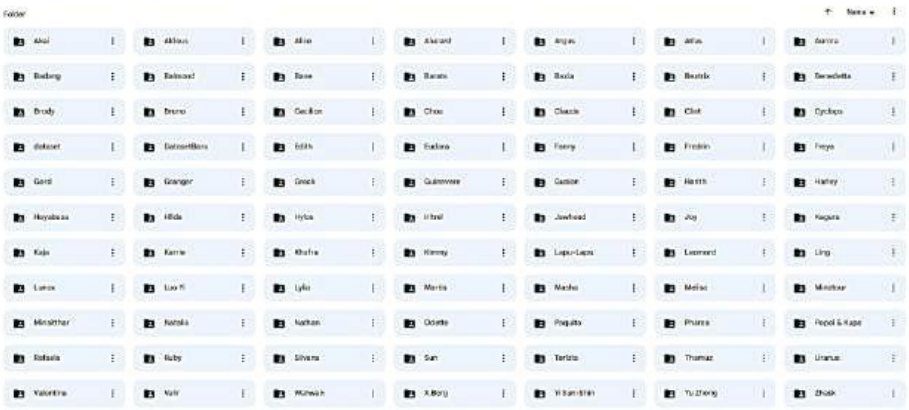
Dalam mengembangkan aplikasi identifikasi gambar *hero mobile legends* menggunakan metode CNN ini menggunakan 3 lembar kerja dalam mendukung alur kerja sistem, yaitu halaman *Landing Page,* halaman *input hero*, dan juga halaman hasil prediksi.

1. *Landing Page,* di halaman ini adalah halaman pertama yang di lihat oleh *user* pada saat pertama kali membuka aplikasi ini
2. Halaman *Input,* di halaman *input* ini, *user* bisa untuk meng-*upload* gambar *hero* ataupun memfoto *hero* tersebut secara langsung. Lalu gambar itu akan di proses oleh sistem dan dilakukannya proses identifikasi
3. Halaman Hasil Klasifikasi, *user* akan melihat dan menemukan hasil identifikasi gambar *hero* yang dia inputkan. Lalu *user* juga bisa melihat nama dari *hero* tersebut. Serta juga bisa melihat berapa persen akurasinya

Implementasi Program *(Development)*

Pada bagian implementasi program berisikan tahapan dalam membuat aplikasi. Dalam penelitian ini menggunakan data yang berjumlah 600 gambar *hero mobile legends*. Data diperoleh dari *website* resmi *Mobile Legends Bang-Bang* dan juga hasil dari *screenshot* yang dilakukan oleh peneliti pada aplikasi *game mobile legends*. Lalu data itu akan di bagi

menjadi dua bagian yaitu: data *training* dan data validasi.



Gambar 1. Dataset Gambar Hero

(Sumber: https://drive.google.com/drive/folders/1Njb-Fav5CPxbEViW6FmIsZlRfGrJvZOB)

Sesuai pada gambar 1 diatas masing-masing dataset akan dilakukan pembagian menjadi 2 bagian yaitu data *training* dan data validasi. Masing masing untuk pembagian datanya yaitu 80% *training* dan 20% validasi. Dataset uji tersebut akan diimplementasikan ke dalam model CNN sesuai skenario uji coba. Dalam melakukan tahapan pembuatan sistem dapat di lihat pada kode program di bawah ini:

1. Mengakses Data

   Data yang ada pada *Google Drive* atau di dalam komputer akan diakses dan di proses ke dalam kode program.

```
[2] !pip install tensorflow scikit-learn

# Mount Google Drive
from google.colab import drive
drive.mount('/content/drive')

[4] # Import necessary libraries
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.metrics import classification_report, confusion_matrix, roc_curve, auc, roc_auc_score

[5] # Define paths
dataset_path = '/content/drive/MyDrive/Anas/dataset'
model_save_path = '/content/drive/MyDrive/Anas/my_model.h5'
tflite_model_save_path = '/content/drive/MyDrive/Anas/my_model.tflite'
```

Gambar 2. Mengakses Data

Pada gambar 2 merupakan kode program untuk mengakses data yang berada pada *Google Drive* yang dimana dataset tesebut akan diolah pada fase *Preprocessing*.

2. *Preprocessing* Data

Pada fase *Preprocessing* data akan dilakukan rescale serta pembagian data menjadi dua bagian. Yaitu data *training* dan data validasi

```
# ImageDataGenerator for train and validation
train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    validation_split=0.2
)

train_generator = train_datagen.flow_from_directory(
    dataset_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='training'
)

validation_generator = train_datagen.flow_from_directory(
    dataset_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='validation'
)
```

Gambar 3. Preprocessing Data

Pada gambar 3 diatas bisa dilihat pembagian data dilakukan dengan format 80% *training* dan 20% *validation*.

3. Model CNN

Data yang sudah menjalali *Preprocessing* akan di implementasikan ke dalam model CNN untuk dilakukan proses klasifikasi. Di dalam proses ini akan ada tahap Operasi Konvolusi, ReLu, Max-pooling serta Flatten.

```
# Build CNN model
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Conv2D(128, (3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(30, activation='softmax')
])

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Gambar 4. Model CNN

Pada gambar 4 diatas program akan menjalankan dan mengimplementasikan model CNN. Penggunaan layer dalam model menyesuaikan dengan kebutuhan.

4. *Training* (Pelatihan)

Setelah model dibentuk, maka fase selanjutnya adalah *training* atau pelatihan data.

Dalam proses ini berfungsi untuk melatih model dalam pembelajaran mesin.

```
[8] # Train the model
    history = model.fit(
        train_generator,
        steps_per_epoch=train_generator.samples // train_generator.batch_size,
        validation_data=validation_generator,
        validation_steps=validation_generator.samples // validation_generator.batch_size,
        epochs=20
    )

[9] # Save the model
    model.save(model_save_path)
    print(f"Model saved to {model_save_path}")

    # Convert the model to TFLite with optimization
    converter = tf.lite.TFLiteConverter.from_keras_model(model)
    converter.optimizations = [tf.lite.Optimize.DEFAULT]
    tflite_model = converter.convert()

    # Save the TFLite model
    with open(tflite_model_save_path, 'wb') as f:
        f.write(tflite_model)

    print(f"TFLite model saved to {tflite_model_save_path}")
```

Gambar 5. Proses Training

Pada gambar 5 diatas merupakan proses *training* data model CNN menggunakan

method fit. *Method fit* ini digunakan dalam melatih model dengan data pelatihan dan menguji kinerjanya dengan data validasi. Data yang sudah dilatih kemudian akan di simpan di dalam *file*.

5. *Confusion Matrix*

Untuk pemahaman yang lebih mendalam tentang kinerja model klasifikasi, perhitungan matriks kebingungan sangat membantu. Untuk perhitungan akurasi penambangan data, matriks kebingungan adalah alat pilihan. Nilai untuk mengingat, presisi, dan akurasi akan dihasilkan dengan mengevaluasi matriks kebingungan. Setelah menilai hasil klasifikasi, nilai akurasi dalam klasifikasi adalah persentase catatan data yang dikategorikan dengan benar. Sebagai ukuran akurasi atau bias, keyakinan didefinisikan sebagai rasio positif asli terhadap positif yang diharapkan dalam data historis. Persentase kejadian positif yang diantisipasi secara akurat dibandingkan dengan jumlah total kasus disebut mengingat.

```
[12] print('Confusion Matrix')
     cm = confusion_matrix(y_true, y_pred_classes)
     plt.figure(figsize=(10, 8))
     sns.heatmap(cm, annot=True, fmt='d', xticklabels=list(test_generator.class_indices.keys()),
                 yticklabels=list(test_generator.class_indices.keys()))
     plt.xlabel('Predicted')
     plt.ylabel('True')
     plt.show()
```

Gambar 6. Confusion Matrix

Dalam kode gambar 6 diatas, merupakan perintah untuk menampilkan model *Confusion Matrix*.

6. *Testing*

Proses *testing* ini adalah proses terakhir untuk menguji sistem dengan data *testing* apakah sistem dapat berjalan dengan baik dan akurat untuk mengidentifikasi data gambar.

```
# Model testing on folder of images with accuracy calculation
def predict_images_from_folder(folder_path):
    from tensorflow.keras.preprocessing import image

    class_labels = list(test_generator.class_indices.keys())
    correct_predictions = 0
    total_predictions = 0

    for img_name in os.listdir(folder_path):
        img_path = os.path.join(folder_path, img_name)
        img = image.load_img(img_path, target_size=(224, 224))
        img_array = image.img_to_array(img)
        img_array = np.expand_dims(img_array, axis=0) / 255.0
        prediction = loaded_model.predict(img_array)
        predicted_class = np.argmax(prediction)

        true_class = img_name.split('.')[0]  # Assuming filename is like 'Gusion.jpg'
        if true_class in class_labels:
            true_class_index = class_labels.index(true_class)
            if predicted_class == true_class_index:
                correct_predictions += 1
            total_predictions += 1

        print(f'Image: {img_name} - Predicted class: {class_labels[predicted_class]} - True class: {true_class}')

    accuracy = correct_predictions / total_predictions
    print(f'Accuracy: {accuracy * 100:.2f}%')

# Example usage
folder_path = '/content/drive/MyDrive/TestingBaru/Hero.Basic'
# folder_path = '/content/drive/MyDrive/TestingBaru/Hero.Elite/'
predict_images_from_folder(folder_path)
```

Gambar 7. Proses Testing

Pada gambar 7 diatas merupakan kode program untuk melakukan proses *testing* data yang sudah melalui proses klasifikasi dan menampilkan hasil klasifikasi dan akurasi yang dihasilkan.

Pengujian Sistem

Dalam pembuatan sebuah sistem patutnya sistem harus diuji terlebih dahulu, agar dapat mengetahui seberapa bagus kinerja dari sistem yang sudah kita buat. Dalam sekenario uji coba ini menggunakan dataset dari 30 *hero* yang terdiri dari 6 *role* yang ada pada *game mobile legends.* Setiap *role* akan di isi oleh 5 *hero most picked* (Pemeilihan terbanyak) di *rank* pada bulan Mei 2024. Setiap *hero* memiliki 20 gambar *hero* sebagai dataset. Data tesebut akan dibagi menjadi dua yaitu data *training* sebanyak 80% dan data validasi sebanyak 20%.

Tabel 1. Jumlah Data

	Gambar Hero
Data Training	480
Data Validasi	120
Total	600

Ketika data sudah melalui *Preprocessing*. Pengujian ini dilakukan dengan menggunakan 3 macam epoch (10 20, 30), fungsi aktivasi (Tanh, Sigmoid, Relu,), Dropout sebesar (0.2, 0.4, 0.8) dan hanya menggunakan optimasi Adam. Pengujian ini bertujuan untuk mengetahui kombinasi terbaik dari beberapa parameter diatas.

Hasil

Setalah melakukan skenario uji coba, di bab ini akan berisikan hasil dari sekenario uji coba yang telah dipaparkan pada sub bab pengujian sistem. Dari uji coba yang telah dilakukan oleh peneneliti didapatkan hasil sebagai berikut:

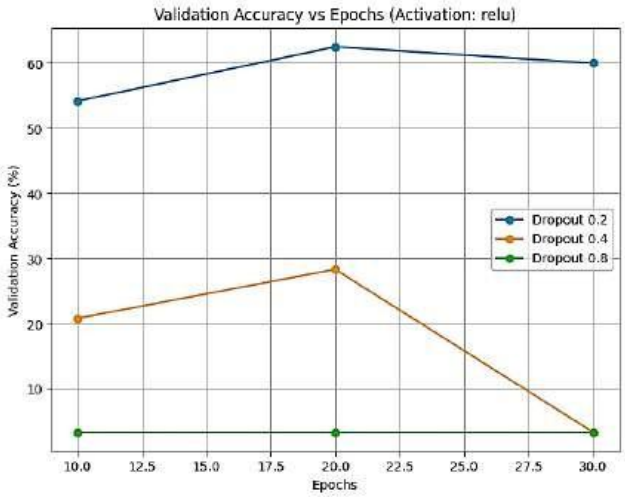
Pada tahap awal penelitian melakukan uji coba dengan menggunakan fungsi aktivasi Elu dengan epoch 10, 20, 30 serta dropout 0.2, 0.4, 0.8 dan optimasi Adam. Tabel 2 menunjukkan hasil percobaan dalam menggunakan fungsi aktivasi dengan ReLu.

Tabel 2. Pengujian menggunakan aktivasi Relu dengan epoch 10, 20, 30 serta dropout 0.2,
0.4, 0.8 dan optimasi adam

No	Activation	dropout	Epoch	Optimation	Val Acuraccy
1	ReLu	0.2	10	Adam	54,17%
2	ReLu	0.2	20	Adam	62,50%
3	ReLu	0.2	30	Adam	60,00%
4	ReLu	0.4	10	Adam	20,83%
5	ReLu	0.4	20	Adam	28,33%
6	ReLu	0.4	30	Adam	3,33%
7	ReLu	0.8	10	Adam	3,33%
8	ReLu	0.8	20	Adam	3,33%
9	ReLu	0.8	30	Adam	3,33%

Berdasarkan pengujian pada Tabel 2 akan dipilih berdasarkan nilai akurasi terbesar yang dihasilkan. Dari data table diatas bisa dilihat akurasi terbesar ada pada kombinasi dropout 0.2, epoch 20, dan optimasi adam dengan presentase 62,50%. Dan berikut ini adalah gambar grafik

dari aktivasi ReLu.



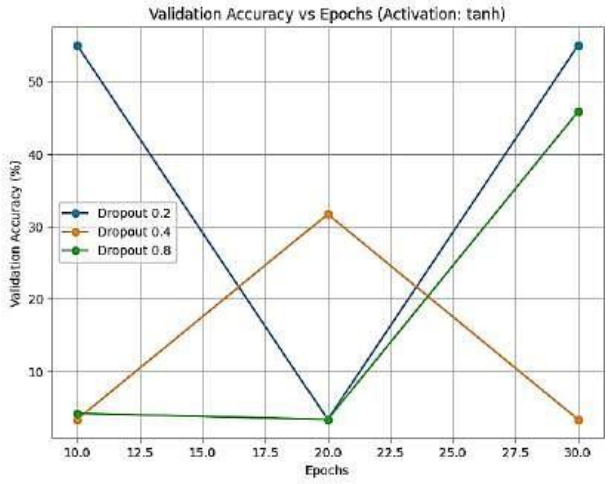
Gambar 8. Grafik dari aktivasi ReLu

Kemudian dilakukan pengujian lagi dengan menggunakan aktivasi Tanh dengan epoch, dropout, dan optimasi yang sama dengan Tabel 2. Tabel 3 menunjukkan hasil dari percobaan menggunakan aktivasi Tanh

Tabel 3. Pengujian menggunakan aktivasi Tanh dengan epoch 10, 20, 30 serta dropout 0.2, 0.4, 0.8 dan optimasi adam.

No	Activation	dropout	Epoch	Optimation	Val Acuraccy
1	Tanh	0.2	10	Adam	55,00%
2	Tanh	0.2	20	Adam	3,33%
3	Tanh	0.2	30	Adam	5,005%
4	Tanh	0.4	10	Adam	3,33%
5	Tanh	0.4	20	Adam	31,67%
6	Tanh	0.4	30	Adam	3,33%
7	Tanh	0.8	10	Adam	4,17%
8	Tanh	0.8	20	Adam	3,33%
9	Tanh	0.8	30	Adam	45,83%

Berdasarkan pengujian pada Tabel 3 akan dipilih dengan akurasi yang terbesar. Dari data table diatas bisa dilihat akurasi terbesar ada pada kombinasi dropout 0.2, epoch 10, dan optimasi adam dengan presentase 55,00%. Dan berikut ini adalah gambar grafik dari aktivasi tanh.



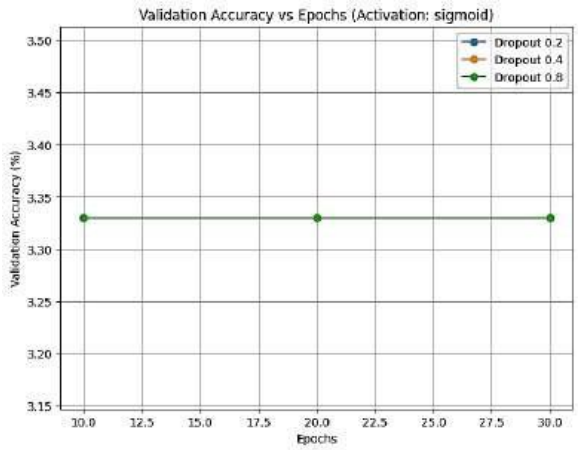
Gambar 9. Grafik dari aktivasi Tanh

Kemudian juga dilakukan pengujian menggunakan aktivasi Sigmoid dengan menggunakan epoch, dropout, serta optimasi yang sama dengan Tabel 3. Disajikan hasil dari pengujian menggunakan aktivasi Relu dalam Tabel 4 berikut:

Tabel 4. Pengujian menggunakan aktivasi Sigmoid dengan epoch 10, 20, 30 serta dropout 0.2, 0.4, 0.8 dan optimasi adam.

No	Activation	dropout	Epoch	Optimation	Val Acuraccy
1	Sigmoid	0.2	10	Adam	3,33%
2	Sigmoid	0.2	20	Adam	3,33%
3	Sigmoid	0.2	30	Adam	3,33%
4	Sigmoid	0.4	10	Adam	3,33%
5	Sigmoid	0.4	20	Adam	3,33%
6	Sigmoid	0.4	30	Adam	3,33%
7	Sigmoid	0.8	10	Adam	3,33%
8	Sigmoid	0.8	20	Adam	3,33%
9	Sigmoid	0.8	30	Adam	3,33%

Berdasarkan pengujian pada Tabel 4 akan dipilih dengan akurasi yang terbesar. Dari data table diatas bisa dilihat akurasi mengalami kegagalan. Dengan hasil presentase yang jauh diluar harapan penulis dengan 3,33%. Dan berikut ini adalah gambar grafik dari aktivasi Sigmoid.



Gambar 10. Grafik dari aktivasi Sigmoid

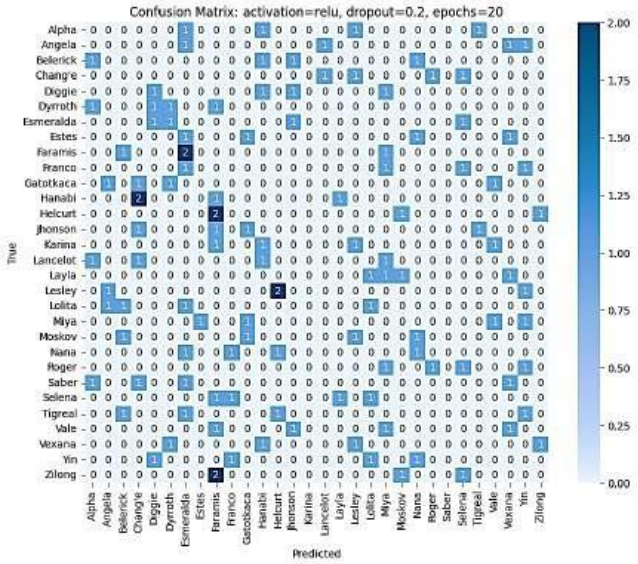
Setelah dilakukan beberapa pengujian dengan menggunakan aktivasi relu, sigmoid, dan tanh dan menggunakan epoch (10, 20, 30) serta dropout (0.2, 0.3, 0.4). Berdasarkan percobaan, hasil terbaik dari penelitian adalah menggunakan jumlah epoch 20, fungsi aktivasi relu dengan nilai dropout 0.2 serta menggunakan optimasi Adam yang menghasilkan akurasi tertinggi 62,17%. Jumlah epoch 20 dipilih karena dengan beberapa pengujian hasil dari akurasi yang besar. Sedangkan nilai dropout menggunakan 0.2, karena untuk mencegah adanya overfitting pada model.

Evaluasi Hasil

Prosedur penilaian menggunakan Laporan Klasifikasi dan Matriks Kebingungan. Nilai-nilai seperti True Positive (TP), False Negative (FN), dan True Negative (TN) disertakan dalam matriks kebingungan. Tidak ada satu pun kejadian yang mungkin negatif (N) atau positif (P). Akurasi, ingatan, dan presisi dapat ditunjukkan dalam laporan klasifikasi. Anda dapat melihat persentase terbaik dan terburuk penulis dari skenario pengujian pada gambar di bawah ini.

1. Hasil Terbaik

   Hasil terbaik dari penelitian adalah menggunakan jumlah epoch 20, fungsi aktivasi relu dengan nilai dropout 0.2 serta menggunakan optimasi Adam yang menghasilkan akurasi tertinggi 62,17%. Berikut adalah gambar confusion matrix dari kombinasi ini :

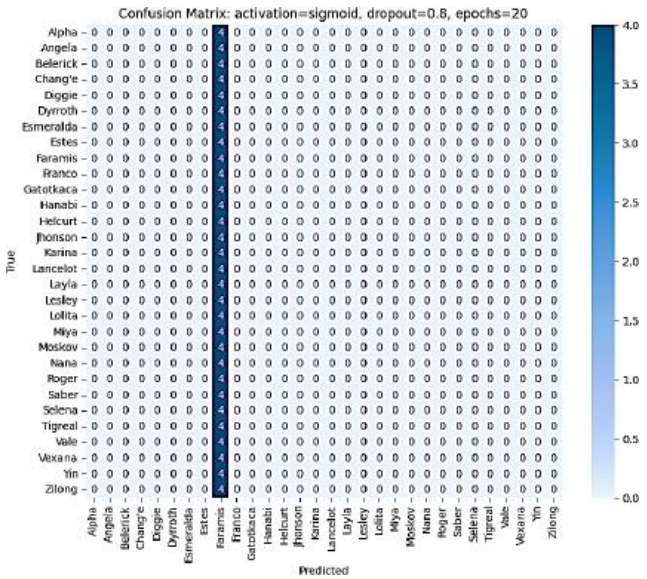


Gambar 11. Confusion matrix (relu, dropout 0.2, epoch 20)

Bedasarkan gambar 11 diatas Secara keseluruhan, *confusion matrix* ini menunjukkan bahwa model CNN memiliki performa yang cukup bervariasi pada masing-masing kelas hero. Hasil akurasi diperoleh dengan akurasi dia angka 62.17.

2. Hasil Terburuk

Hasil terburuk dari penelitian adalah menggunakan jumlah epoch 20, fungsi aktivasi sigmoid dengan nilai dropout 0.8 serta menggunakan optimasi Adam yang menghasilkan akurasi 3,33%. Berikut adalah gambar *confusion matrix* dari kombinasi ini :



Gambar 12. Confusion matrix (sigmoid, dropout 0.8, epoch 20)

Berdasarkan gambar diatas Secara keseluruhan, *confusion matrix* ini menunjukkan bahwa model CNN memiliki performa yang cukup bervariasi pada masing-masing kelas hero.

Beberapa kelas dapat diklasifikasikan dengan baik, sementara beberapa kelas lainnya masih menunjukkan banyak kesalahan prediksi. Ini menandakan bahwa ada kebutuhan untuk peningkatan lebih lanjut dalam model, baik melalui parameters, peningkatan dataset, atau perubahan arsitektur model untuk mencapai hasil klasifikasi yang lebih akurat.

KESIMPULAN

Dalam implemantasi yang dilakukan di pembuatan sistem identifikasi *hero mobile legends*. Berdasarkan uji coba dan pengujian yang dilakukan, peneliti menarik kesimpulan yaitu membangun sebuah aplikasi mobile yang menggunakan algoritma *Convolutional Neural Network* untuk mengklasifikasi jenis *Hero Mobile Legends.* Peneliti berhasil untuk merealisasikan aplikasi tersebut dengan baik dan benar. Berdasarkan sekenario uji coba yang dilakukan, sistem dapat bekerja cukup baik dalam hal identifikasi hero di game mobile legends ini. Menguji dan mengukur performa metode *Convulotional Neural Network* dalam hal klasifikasi untuk mengetahui seberapa akurat metoted ini dapat bekerja. Peneliti berhasil menguji metode CNN ini dengan cukup baik, berdasarkan uji coba yang dilakukan, dapat disimpulkan bahwa algoritma ini dapat bekerja dengan baik pada identifikasi gambar hero mobile legends. Akan tetapi untu keakuratanya masih belum cukup baik. Akurasi yang kurang baik tersebut mungkin saja disebabkan oleh jumlah dataset yang sedikit dan juga kualitas gambar yang digunakan dalam dataset penelitian ini, selain itu masih belum sempurnanya pengetahuan peneliti tentang algoritma ini sehingga algoritma CNN ini tidak tereksplore dengan baik.

DAFTAR REFERENSI

Andono, P. N., & Sutojo, T. (2018). *Pengolahan citra digital*. Penerbit Andi.

Bayulianto, S., Purnamasari, I., & Jajuli, M. (2023). Prediksi Tingkat Kemenangan Mobile Legends Profesional League Indonesia Season 9 Dengan Menggunakan Algoritma Naïve Bayes. JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika), 8(2), 538-550.

Hutagaol. B, “Apa itu Mobile Legends: Bang Bang?, “. 2018.

https://esportsnesia.com/game/mobile-legends/apa-itu-mobile-legends/

Katona, A., Spick, R., Hodge, V. J., Demediuk, S., Blok, F., Drachen, A., & Walker, J. A. "Time to Die: Death Prediction in Dota 2 Using Deep Learning," Conf. Proc. - IEEE Conference on Computatonal Intelligence and Games, CIGKatona, A., Spick, 2019, doi: 10.1109/CIG.2019.8847997.

Listijo, S. M., Purwani, T., & Galih, S. T. (2020). PREDIKSI KEMENANGAN DAN SUSUNAN TIM PADA GAME MOBILE LEGENDS BANG BANG MENGGUNAKAN ALGORITMA NAÏVE BAYES. *KOMPUTAKI*, *6*(1).

Maurício, J., Domingues, I., & Bernardino, J., “Comparing Vision Transformers and Convolutional Neural Networks for Image Classification: A Literature Review,” *Appl. Sci.*, vol. 13, no. 9, hal. 5521, 2023.

Peryanto, A., Yudhana, A., & Umar, R. (2020). Rancang bangun klasifikasi citra dengan teknologi deep learning berbasis metode convolutional neural network. *Format J. Ilm. Tek. Inform*, *8*(2), 138.

Putro, A. C. (2018). *Sistem Prediksi Kemenangan Tim Pada Game Mobilelegends Dengan Metode Naïve Bayes* (Doctoral Dissertation, Universitas 17 Agustus 1945).

Rahmadhani, U. S., & Marpaung, N. L. "Klasifikasi Jamur Berdasarkan Genus Dengan Menggunakan Metode CNN," *J. Inform. J. Pengemb. IT*, vol. 8, no. 2, hal. 169–173, 2023.

Saha, S. "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way," [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional- neural-networks-the-eli5-way-3bd2b1164a53. [Accessed 23 12 2023].

Sena, I. G. W., & Emanuel, A. W. (2023). Mobile Legend Game Prediction Using Machine Learning Regression Method. JURTEKSI (Jurnal Teknologi dan Sistem Informasi), 9(2), 221-230.

Syahid, D., Jumadi, & Nursantika, D. "Sistem Klasifikasi Jenis Tanaman Hias Daun Philodendron Menggunakan Metode K-Nearest Neighboor (KNN) Berdasarkan Nilai Hue, Saturation, Value (HSV)," JOIN, vol. I, no. 1, pp. 20–23, 2016.

Syaputra, H., Supratman, E., & Purnamasari, S. D. (2022). Klasifikasi Jenis Burung Lovebird Menggunakan Algoritma Convolutional Neural Network. *Journal of Computer and Information Systems Ampera*, *3*(2), 133-140.

Taye, M. M. "Theoretical understanding of convolutional neural network: concepts, architectures, applications, future directions," *Computation*, vol. 11, no. 3, hal. 52, 2023.