**BAB I**

**PENDAHULUAN**

* 1. **Latar Belakang**

Bahasa Jepang memiliki tiga sistem penulisan yang digunakan sehari – hari, yaitu hiragana, katakana, dan Kanji. Diantara ketiga sistem tersebut, kanji memiliki peran yang penting dalam tata bahasa jepang karena digunakan untuk merepresentasikan makna kata secara lebih tepat dan efisien. Secara keseluruhan terdapat puluhan ribu karakter kanji, namun hanya ada 2.136 kanji yang dikategorikan sebagai Jōyō Kanji, yakni daftar kanji standar yang digunakan dalam kehidupan sehari-hari.

Tata cara penulisan kanji tersusun atas goresan–goresan dasar yang disebut *stroke*. Jumlah dan urutan penulisan stroke harus diperhatikan dengan saksama agar dapat dipahami dengan benar. Beberapa kanji sederhana hanya terdiri atas dua hingga lima *stroke*, sementara kanji kompleks seperti *ryū* yang berarti naga dapat mencapai lebih dari 15 *stroke*. Semakin banyak jumlah *stroke* pada suatu kanji, semakin sulit pula kanji tersebut untuk diingat, mengingat banyaknya kanji yang mirip, para pelajar kanji seringkali salah membedakan maknanya [1]. Kompleksitas ini menjadikan kanji sebagai tantangan besar, baik dalam proses pembelajaran maupun dalam aplikasi komputasi seperti pengenalan tulisan tangan [2], [3], [4], [5].

Untuk mengatasi permasalahan dalam pembelajaran kanji, berbagai penelitian telah dilakukan sebelumnya, beberapa di antaranya meliputi pengenalan tulisan tangan karakter kanji menggunakan Ensemble of Convolutional Neural Network [5] dan penggabungan image processing dan deep learning untuk pengorganisasian ulang buku-–buku Jepang kuno [6]. Berbeda dengan penelitian – penelitian tersebut, penelitian ini berfokus pada pengembangan sebuah model *Artificial Intelligence* (AI) yang mampu mengukur tingkat kemiripan visual antarkarakter kanji. Dengan demikian, model dapat dengan mudah mengidentifikasi karakter kanji lain yang memiliki bentuk serupa dengan suatu karakter kanji tertentu.

Convolutional Neural Network (CNN) dipilih sebagai solusi utama dalam penelitian ini. Dalam beberapa tahun terakhir, perkembangan AI, khususnya di bidang *computer vision*, telah memberikan kontribusi signifikan dalam pengenalan karakter kanji [7]. Model-model CNN telah digunakan secara luas untuk mengenali berbagai karakter bahasa dalam berbagai aplikasi, mulai dari pengenalan tulisan tangan hingga pelabelan otomatis [8], [9]. *Computer vision* berfokus pada pemrosesan dan analisis citra, sehingga memungkinkan komputer memahami struktur stroke pada karakter kanji.

Dataset yang digunakan dalam penelitian ini adalah KanjiVG yang berisi ribuan kanji dari JIS level satu dan dua dengan format *Scalable Vector Graphics* (SVG) dengan jumlah data setiap kanji yang relatif terbatas. KanjiVG berfokus pada *stroke* dari setiap karakter sehingga sesuai untuk penelitian yang menekankan pada struktur goresan. Namun, dataset ini memiliki kekurangan, yaitu jumlah data yang tidak banyak. Untuk mengatasi keterbatasan jumlah data, penelitian ini menggunakan konsep *Few-Shot Learning* (FSL) agar model dapat berjalan sesuai dengan keinginan. Selain itu, *Siamese Neural Network* (SNN) yang berbasis CNN juga digunakan karena dapat mendukung model untuk mencari *similarity* pada kanji. SNN berisi dua *neural network* yang sejenis dan saling berbagi bobot yang sama. Dengan berbagi bobot, kedua jaringan ini mampu menghasilkan representasi vektor yang konsisten, sehingga cocok untuk perhitungan nilai kesamaan (*similarity score*) antara dua input [10].

Penelitian ini menggunakan python sebagai bahasa pemrograman utama karena python memiliki banyak *library* yang mendukung pembelajaran mesin, khususnya di bidang *computer vision*, seperti Pandas, NumPy, PyTorch, dan Matplotlib. Platform Google Colab digunakan sebagai *notebook* untuk mengeksekusi kode, karena menyediakan akses GPU dan *environtmet* yang nyaman, sehingga cocok untuk menjalankan model AI yang memiliki beban komputasi berat, seperti dataset citra.

Algoritma yang digunakan untuk SNN adalah EfficientNet-B0 karena arsitektur ini dikenal memiliki kinerja bagus dan beban komputasi yang tidak terlalu berat. Berdasarkan studi komparatif yang telah dilakukan, EfficientNet-B0 secara konsisten menunjukkan akurasi yang kompetitif dengan jumlah parameter yang jauh lebih sedikit dibandingkan arsitektur konvensional seperti *ResNet50* [11].

* 1. **Rumusan Masalah**

Berdasarkan latar belakang masalah yang sudah dipaparkan, maka dapat dirumuskan beberapa masalah yang perlu diperhatikan:

1. Bagaimana model dapat membedakan pola-pola *stroke* yang ada pada kanji?
2. Mengapa diperlukan penerapan FSL dan arsitektur SNN untuk mengukur kemiripan visual antar karakter kanji di dataset kanjiVG?
3. Bagaimana tingkat akurasi EfficientNet-B0 dengan arsitektur SNN untuk pencarian kemiripan visual pada kanji?
   1. **Tujuan Penelitian**

Adapun tujuan dari penelitian ini adalah:

1. Mengetahui bagaimana model mampu mengidentifikasi perbedaan pola *stroke* pada kanji.
2. Mengembangkan sistem berbasis kecerdasan buatan yang mampu membantu proses pencarian karakter kanji dengan mempertimbangkan tingkat kemiripan visual.
3. Mengetahui tingkat akurasi EfficientNet-B0 untuk mencari kemiripan visual pada karakter kanji.
   1. **Manfaat Penelitian**

Penelitian ini bermanfaat untuk pengembangan metode pencarian kemiripan visual antar karakter kanji menggunakan konsep FSL dengan basis SNN menggunakan algoritma EfficientNet-B0 yang tidak hanya berkontribusi pada pengembangan ilmu *computer vision* tetapi juga dapat membantu para pelajar kanji dalam membedakan serta mempelajari kanji secara lebih mudah.

* 1. **Batasan Penelitian**

Penelitian ini memiliki beberapa batasan, diantaranya:

1. Dataset yang digunakan berasal dari KanjiVG, dengan format citra (hasil konversi dari SVG ke PNG).
2. Fokus penelitian adalah pada kemiripan visual karakter kanji berdasarkan struktur *stroke*, bukan pada aspek semantik atau pelafalan (kunyomi/onyomi).
3. Arsitektur yang digunakan adalah SNN dengan model EfficientNet-B0 sebagai dasarnya.
4. Penelitian dilakukan menggunakan Google Colab dan bahasa pemrograman Python.
   1. **Sistematika Penulisan**

Penulisan Skripsi ini dibagi dalam 5 bab, adapun isi dari masing-masing bab dapat diuraikan secara singkat sebagai berikut:

BAB I: PENDAHULUAN

Bab ini berisi uraian mengenai latar belakang penelitian, rumusan masalah, tujuan penelitian, batasan masalah, manfaat penelitian, serta sistematika penulisan skripsi.

BAB II: TINJAUAN PUSTAKA

Bab ini memuat kajian teori dan penelitian terdahulu yang relevan, meliputi sistem penulisan kanji, konsep computer vision, CNN, FSL, SNN, serta arsitektur *EfficientNet*-B0 yang digunakan sebagai backbone.

BAB III: METODOLOGI PENELITIAN

Bab ini menjelaskan metode penelitian yang digunakan, termasuk deskripsi dataset KanjiVG, tahap pengolahan data, arsitektur model SNN dengan backbone *EfficientNet*-B0, skema pelatihan menggunakan FSL, serta rancangan evaluasi model.

BAB IV: HASIL DAN PEMBAHASAN

Bab ini menyajikan hasil eksperimen yang dilakukan, mulai dari preprocessing dataset, hasil pelatihan model, evaluasi performa menggunakan metrik akurasi maupun top-k retrieval, hingga analisis efektivitas penggunaan SNN berbasis *EfficientNet*-B0 pada pencarian kemiripan visual kanji.

BAB V: PENUTUP

Bab ini berisi kesimpulan yang diperoleh dari hasil penelitian serta saran untuk pengembangan penelitian lebih lanjut, baik dari sisi metode, dataset, maupun aplikasi praktis di bidang pembelajaran kanji dan computer vision.

# **BAB II**

# **TINJAUAN PUSTAKA**

1. **Kanji**

Karakter kanji adalah sistem penulisan digunakan dalam bahasa Jepang yang berasal dari karakter Tionghoa, oleh karena itu kanji memiliki dua cara pelafalan, yaitu kun`yomi (cara baca Jepang) dan on`yomi (cara baca Tionghoa). Kanji tidak hanya merepresentasikan bunyi, melainkan juga unsur makna dan struktur visual yang kompleks [12].

Setiap karakter kanji terdiri dari goresan (*stroke*) dan radikal (komponen dasar) yang mempengaruhi bentuk visualnya. Struktur ini menyebabkan variasi visual yang beragam antar karakter, tetapi berbeda dalam makna atau pelafalan [13]. Karena itu, pengenalan atau pencarian karakter kanji berdasarkan kemiripan visual menjadi tantangan yang menarik dalam *computer vision* dan *deep learning*.

Dalam lingkup penggunaan resmi dan pendidikan di Jepang, karakter kanji diklasifikasikan ke dalam daftar yang diatur pemerintah Jepang. Salah satu daftar yang paling dikenal dan sering dijadikan acuan adalah Jōyō kanji. Daftar ini berisi karakter kanji yang dianggap sebagai karakter yang sering digunakan dalam kegiatan sehari-hari dan wajib diketahui oleh penutur bahasa Jepang dalam konteks literasi umum. Saat ini, jumlah Jōyō kanji adalah 2.136 karakter, setelah revisi tahun 2010. 1.026 karakter diajarkan pada jenjang *sekolah dasar* (kelas 1–6), sedangkan sisanya diajarkan pada jenjang menengah (kelas 7–9) dan sekolah menengah atas [12]. Selain Jōyō kanji, ada pula karakter Jinmeiyō kanji yang digunakan untuk nama pribadi [14].

**2.1.1. Standar Kode dan Representasi Digital Kanji**

Agar karakter kanji dapat diproses secara digital (misalnya dalam sistem komputer atau basis data), diperlukan standar pengkodean karakter. Beberapa standar yang dipakai hingga sekarang adalah Unicode dan *Japanese Industrial Standards* (JIS). Unicode sudah menjadi standar internasional untuk merepresentasikan berbagai karakter, termasuk kanji. Sedangkan JIS adalah standar nasional Jepang untuk representasi huruf dan simbol Jepang secara digital. JIS memiliki beberapa jenis, seperti JIS X 0208 dan JIS X 0213 [14].

1. **KanjiVG Dataset**

KanjiVG merupakan dataset berbasis vektor yang berisi informasi struktur dan urutan goresan (stroke order) dari karakter kanji dalam format Scalable Vector Graphics (SVG). Dataset ini dikembangkan oleh Ulrich Apel sebagai bagian dari proyek *KanjiVG Project* pada tahun 2010 dan dipublikasikan secara terbuka di GitHub di bawah lisensi Creative Commons Attribution-ShareAlike (CC BY-SA)

Setiap file dalam KanjiVG merepresentasikan satu karakter kanji yang diberi nama sesuai dengan kode Unicode-nya. Misalnya, karakter *mizu* yang artinya air disimpan dalam 6c34.svg. File SVG tersebut berisi elemen-elemen <path> dan <g> yang mendefinisikan setiap goresan serta kelompok komponennya. Dengan demikian, KanjiVG tidak hanya menyimpan bentuk akhir suatu kanji, tetapi juga struktur hierarkis dan urutan pembentukannya.

KanjiVG terdiri dari sekitar 6.353 karakter kanji, yang mencakup Seluruh jōyō kanji (2.136 karakter), jinmeiyō kanji, JIS X 0208 Level 1 dan Level 2. KanjiVG juga mencakup beberapa kanji dari standar JIS X 0212/0213, serta sejumlah kecil karakter langka atau varian bentuk historis. Dengan cakupan tersebut, KanjiVG dapat dianggap mewakili hampir seluruh kanji yang digunakan dalam penulisan bahasa Jepang modern.

1. ***Computer Vision***

Perkembangan bidang Computer Vision berawal pada dekade 1960-an, ketika penelitian mulai berfokus pada bagaimana komputer dapat mengekstraksi informasi tiga dimensi dari representasi dua dimensi sebuah objek. Pada tahap awal, penelitian di bidang ini menitikberatkan pada proses pengambilan informasi penting dari citra digital melalui pemodelan komputasional. Secara garis besar, Computer Vision memiliki dua orientasi utama. Dari sisi rekayasa, bidang ini bertujuan mengembangkan sistem otonom yang mampu melakukan tugas-tugas visual layaknya penglihatan manusia. Sementara itu, dari sisi biologis, pendekatan komputasional digunakan untuk meniru mekanisme penglihatan manusia, seperti dalam proses deteksi atau diagnosis kondisi medis melalui citra tubuh [15].

Dalam konteks teknis, Computer Vision telah diterapkan di berbagai sektor, termasuk bidang pendidikan, di mana sistem pengenalan wajah dimanfaatkan untuk mencatat kehadiran secara otomatis melalui kamera. Teknologi ini juga berkembang pesat dalam sistem kendaraan udara tanpa awak (Unmanned Aerial Vehicle atau UAV) yang menggunakan penglihatan komputer untuk melakukan pemantauan situasi secara otomatis tanpa kendali langsung manusia.  
Sementara dalam ranah biomedis, Computer Vision berperan besar dalam pencitraan medis (medical imaging), sebuah bidang yang mengalami perkembangan pesat karena kemampuannya membantu proses diagnosis dan peningkatan mutu pelayanan kesehatan [15].

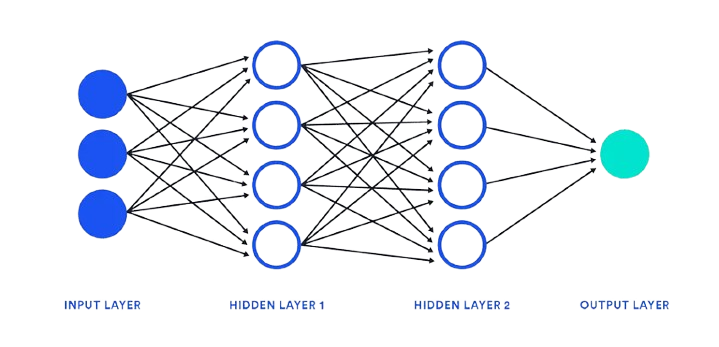
Seiring perkembangan teknologi, Computer Vision menjadi salah satu bidang utama dalam Artificial Intelligence (AI) yang berfokus pada kemampuan sistem komputer untuk menafsirkan, menganalisis, dan memahami informasi visual dari dunia nyata dalam bentuk gambar maupun video. Teknologi ini memungkinkan mesin untuk meniru persepsi visual manusia melalui proses pengenalan objek, analisis citra, pelacakan gerakan, hingga interpretasi konteks visual secara otomatis [16].  
Meskipun demikian, baik penglihatan manusia maupun computer vision sama-sama berupaya menafsirkan data multidimensi secara spasial, namun sistem komputer belum sepenuhnya mampu menandingi kompleksitas dan kepekaan penglihatan manusia karena keterbatasan algoritma, variabel, dan tingkat akurasi. Tantangan utama dalam pengembangannya meliputi efektivitas algoritma, ketepatan hasil analisis, serta kemampuan sistem untuk beradaptasi secara akurat, tangguh, dan skalabel dalam berbagai kondisi. Berbagai penelitian juga telah memperluas penerapan computer vision ke berbagai bidang, seperti bantuan bagi penyandang disabilitas visual, otomasi industri, penginderaan jauh, robotika, hingga komunikasi manusia berbasis komputer, dengan efektivitas sistem yang sangat bergantung pada arsitektur dan rancangan aplikasinya [17].

Dengan demikian, *Computer Vision* dapat dipandang sebagai disiplin ilmu yang berperan penting dalam menjembatani data visual mentah menjadi **representasi** bermakna yang mendukung pengambilan keputusan otomatis dalam berbagai aplikasi — mulai dari kesehatan, transportasi, hingga pengenalan karakter Kanji dalam konteks penelitian ini.

1. ***Deep Learning***

Deep Learning merupakan cabang dari Machine Learning yang menggunakan jaringan saraf tiruan berlapis untuk mengekstrak fitur otomatis dari data mentah, seperti gambar atau video, dan mengaplikasikannya pada tugas-tugas seperti klasifikasi, deteksi objek, segmentasi, dan restorasi citra; CNN (Convolutional Neural Network) misalnya, telah digunakan secara luas dalam visi komputer karena kemampuannya dalam menangkap fitur visual dari lapisan rendah ke lapisan tinggi [18].

Secara struktural, deep learning terdiri dari beberapa lapisan tersembunyi (hidden layers) di antara lapisan input dan output, yang memungkinkan pemodelan hubungan non-linier yang kompleks dalam data; metode seperti forward propagation dan backpropagation digunakan untuk melatih model dan mengoptimalkan bobot antar lapisan agar kesalahan prediksi berkurang [19].



Gambar 1 Ilustrasi *input*, *output*, dan *hidden layer* pada *deep learning*

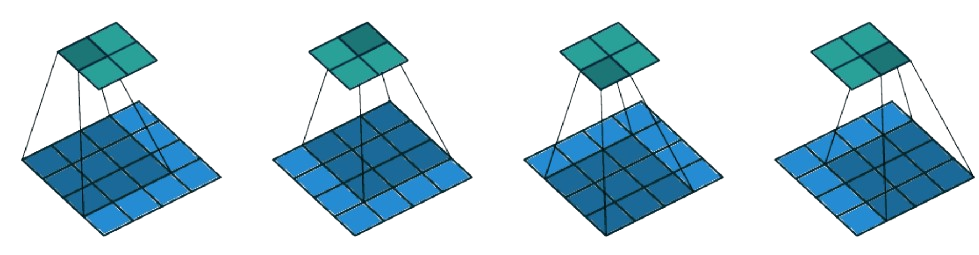
Kebutuhan data yang besar dan berkualitas menjadi faktor krusial pada deep learning. Tanpa dataset yang cukup besar, model-model sering kali mengalami overfitting atau generalisasi yang buruk, teknik augmentasi data dan penggunaan *pretrained models* sering diterapkan untuk mengatasi keterbatasan data. Augmentasi data dapat digunakan untuk memberikan lebih banyak variasi pada data yang ada, seperti merotasi dan memperbesar gambar, menggunakan sinonim untuk menggantikan suatu kata pada data teks, dan memberikan noise pada data [20].

Selain itu, tantangan utama dalam pengembangan deep learning meliputi computational cost yang tinggi, kebutuhan terhadap perangkat keras yang kuat seperti GPU untuk mendukung kelancaran proses pelatihan, kebutuhan untuk arsitektur yang efisien agar model bisa digunakan pada perangkat dengan sumber daya terbatas [21], dan robustness, yaitu bagaimana model tetap mampu melakukan prediksi yang akurat ketika kondisi input berubah, misalnya adanya noise, pencahayaan berbeda, kondisi occlusion atau gangguan adversarial yang membuat input menjadi sangat bervariasi [22].

Berbagai penelitian terkini telah memperluas aplikasi deep learning dalam computer vision, mencakup deteksi objek, segmentasi semantik, pengolahan citra medis, dan deteksi serta klasifikasi dalam data point cloud 3D; aplikasi-aplikasi ini menunjukkan bahwa deep learning telah menjadi dasar untuk banyak sistem *computer vision* modern [23].

1. ***Convolutional Neural Network* (CNN)**

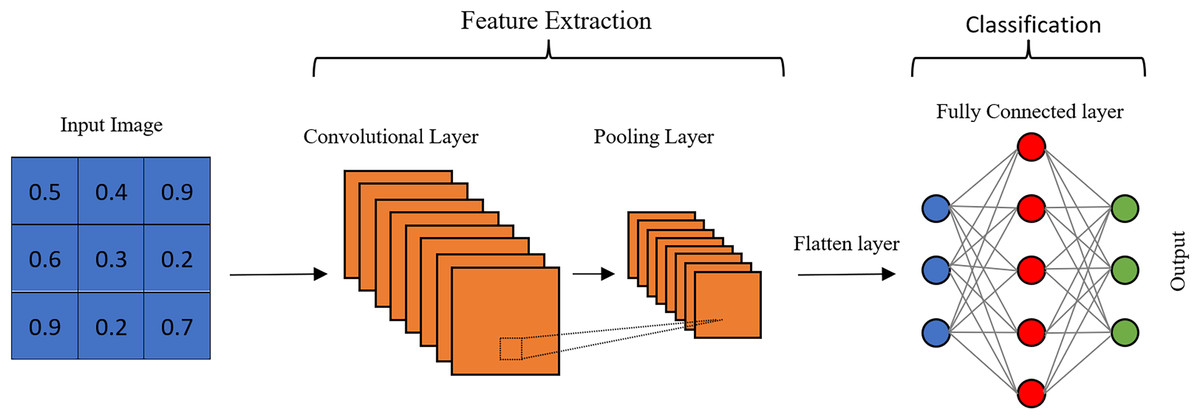
Convolutional Neural Network (CNN) adalah jenis jaringan saraf tiruan dalam deep learning yang dirancang khusus untuk mengolah data yang memiliki struktur *grid* seperti gambar. CNN mampu mengekstrak fitur visual secara otomatis melalui operasi konvolusi, di mana filter atau kernel digeser pada citra input untuk mendeteksi pola yang ada pada suatu citra seperti tepi, garis, tekstur, atau titik-perubahan intensitas. Lapisan-lapisan konvolusi ini diikuti oleh operasi *pooling* atau *downsampling* untuk mengurangi dimensi spasial sambil mempertahankan informasi penting, kemudian diakhiri dengan lapisan fully connected atau output untuk melakukan klasifikasi atau prediksi berdasarkan fitur-fitur yang telah dipelajari, di mana CNN membangun hirarki fitur mulai dari lapisan awal yang mengenali pola sederhana hingga lapisan yang lebih dalam yang menangkap fitur abstrak [24].



Gambar 2 Ilustrasi pemrosesan konvolusi menggunakan filter 3×3 dan stride=1

**2.5.1. Evolusi & Arsitektur CNN Terkini**

Seiring berjalannya riset, arsitektur-CNN menjadi semakin kompleks dan beragam. Misalnya, peneliti menggabungkan modul-modul baru, merancang blok arsitektur yang efisien (seperti residual blocks, dense connectivity, dan blok-blok modul lainnya) agar model menjadi lebih dalam, lebih lebar, namun tetap mengusahakan agar dapat dilatih dengan biaya komputasi yang lebih terkendali [24]. Selain itu, penelitian terbaru juga membahas cara baru dalam memproses gambar resolusi rendah atau objek kecil, salah satunya dengan mengganti pooling atau strided convolution yang biasa digunakan dengan modul yang menjaga detail spasial lebih baik agar tidak kehilangan informasi penting [25].



Gambar 3 Contoh struktur CNN yang terdiri dari *convolutional*, *pooling*, dan *flatten layer*.

**2.5.2. Kelebihan dan Tantangan CNN**

CNN memiliki sejumlah keunggulan yang menjadikannya salah satu arsitektur paling dominan dalam bidang *deep learning* dan *computer vision*:

1. Ekstraksi fitur otomatis, CNN dapat belajar dan mengenali pola visual tanpa memerlukan fitur manual yang rumit, sehingga sangat cocok untuk tugas-citra kompleks.
2. Invarian terhadap transformasi lokal, karena operasi konvolusi dan pooling, CNN relatif tangguh terhadap translasi, rotasi ringan, dan variasi kecil lainnya dalam data gambar [26].
3. Performanya sangat baik dalam berbagai tugas seperti klasifikasi gambar, deteksi objek, segmentasi semantik, dan prediksi video[24].

Meskipun memiliki banyak keunggulan, CNN juga menghadapi sejumlah tantangan yang perlu diperhatikan dalam pengembangannya

1. Overfitting dan kapasitas model, semakin dalam atau kompleks arsitektur CNN, semakin besar kemungkinan terjadi overfitting jika data latih tidak cukup atau kurang bervariasi.
2. Kebutuhan komputasi dan sumber daya: Pelatihan CNN besar biasanya memerlukan GPU/TPU dan sumber daya memori yang besar. Ini menjadi kendala khususnya pada aplikasi yang dijalankan pada perangkat dengan keterbatasan hardware/edge devices [24].
3. Penurunan performa pada gambar resolusi rendah atau objek kecil: Beberapa desain standar CNN (misalnya penggunaan pooling atau convolusi yang “strided”) bisa kehilangan detil penting, yang berdampak buruk pada deteksi objek kecil atau fitur-fitur halus. Penelitian seperti “No More Strided Convolutions or Pooling” mengatasi hal ini dengan modul alternatif yang mempertahankan informasi spasial lebih baik [25].
4. ***Few-Shot learning* (FSL)**

*Few-Shot Learning* adalah pendekatan dalam *machine learning* / *deep learning* yang bertujuan agar model dapat belajar dan beradaptasi pada kelas atau tugas baru dengan hanya beberapa contoh sebagai data pelatihan. FSL sering kali dipandang sebagai metode yang memberikan solusi untuk kebutuhan data besar pada *deep learning* dan kondisi nyata dimana data yang tersedia terbatas untuk suatu proses pelatihan [27].

* + 1. **Metode Utama dalam FSL**

1. Meta-Learning, model dilatih dengan banyak “task” kecil sehingga bisa cepat beradaptasi ke task baru yang hanya memiliki sedikit data. Contohnya episodic training dan pembelajaran cara belajar [28].
2. *Metric-Learning* / *Embedding-based Methods*, model membuat representasi fitur / *embedding* di mana contoh-contoh dari kelas yang sama lebih dekat, dan contoh dari kelas berbeda lebih jauh. Metode ini dapat diterapkan dengan bantuan rumus perhitungan jarak, seperti *Euclidean distance* atau *cosine distance* [29].
3. *Augmentation Data* dan *Transfer Learning*, data augmentasi dapat memperluas dvariasi data yang ada dalam suatu dataset. Transfer Learning adalah menggunakan bantuan model pretrained dari domain lain lalu disesuaikan ke domain baru yang memiliki sedikit data sehingga dapat menutupi kekurangan ini [27].
4. Optimisasi dan Regularisasi: teknik untuk mencegah *overfitting* ketika data sangat terbatas; misalnya *variational autoencoders* untuk *generative augmentation*, *regularizatio*, *dropout*, dan lain-lain [30].
   * 1. **Kelebihan dan Tantangan FSL**

FSL memiliki sejumlah keunggulan yang menjadikannya pendekatan menarik dalam kondisi keterbatasan data. Beberapa kelebihannya antara lain:

* Efisiensi data: model tidak memerlukan dataset besar, sangat membantu ketika label sulit atau mahal diperoleh [27].
* Adaptabilitas: model FSL mampu cepat beradaptasi ke kelas atau tugas baru setelah pelatihan meta atau pra-pelatihan [28].

Meskipun menawarkan beberapa kelebihan, FSL juga menghadapi sejumlah tantangan yang membatasi kinerjanya dalam praktik. Beberapa tantangan utama yang sering ditemui meliputiOverfitting: model mudah mengalami overfitting karena jumlah contoh pelatihan yang sangat sedikit.

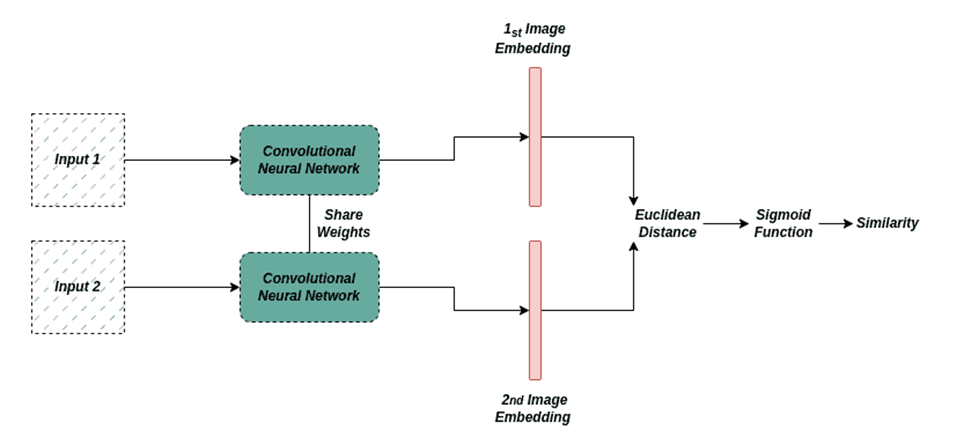
* Representasi yang kurang kuat: embedding atau fitur yang dipelajari mungkin tidak menangkap variasi penting karena keterbatasan data [31].
* Perbedaan domain (domain shift): tantangan ketika data tugas baru berbeda distribusinya dari data pelatihan (sumber), yang bisa menurunkan performa [27].
* Ketidakstabilan dalam performa: kinerja model FSL bisa sangat dipengaruhi oleh pemilihan contoh pelatihan (sampling), ukuran variasi antar kelas, dan kesiapan model pra-latih [29].

1. ***Siamese Neural Network* (SNN)**

*Siamese Neural Network* (SNN) adalah arsitektur jaringan saraf yang bertujuan untuk mempelajari kemiripan antara dua input dengan cara memproyeksikan tiap input ke dalam *embedding* (representasi vektor) dan kemudian menghitung jarak atau kesamaan antar *embeddings* tersebut untuk menentukan apakah kedua input mirip atau tidak [32].

Struktur dasar SNN terdiri dari dua cabang jaringan identik yang berbagi bobot satu sama lain, sehingga kedua input mengalami pemrosesan yang sama hingga menghasilkan dua embedding, misal f(x₁) dan f(x₂). Kedua embedding tersebut kemudian dibandingkan menggunakan metrik jarak seperti *Euclidean distance*, *cosine similarity*, atau fungsi loss seperti *contrastive loss* atau *triplet loss*, yang secara khusus memaksa embedding dari pasangan positif (mirip) menjadi dekat dan embedding dari pasangan negatif menjadi berjauhan [32].

Dalam pelatihan, *contrastive loss* merupakan fungsi loss populer untuk SNN: ketika dua input adalah pasangan “mirip”, loss meminimalkan jarak antar embedding; ketika pasangan “berbeda”, loss memaksimalkan jarak hingga melewati margin tertentu. Untuk kasus di mana digunakan triplet (anchor, positive, negative), *triplet loss* berusaha memastikan jarak (anchor, negative) lebih besar dari jarak (anchor, positive) ditambah margin [33].



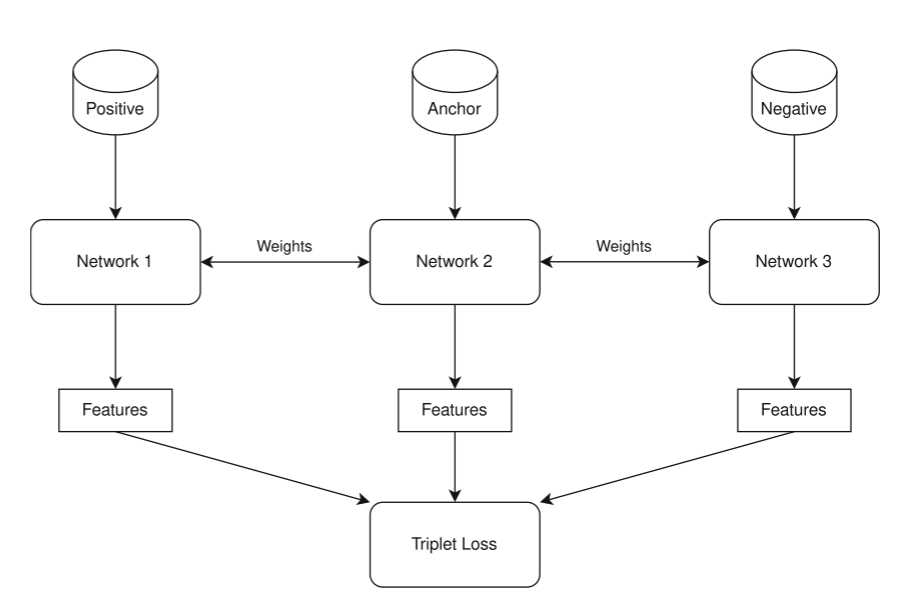
Gambar 4 Arsitektur SNN dengan dua *input*

Dalam aplikasi nyata, SNN telah digunakan untuk pengenalan penyakit tanaman dengan dataset kecil dan imbalanced: sebuah studi membangun framework ringan berbasis SNN dengan contrastive loss untuk mengenali penyakit daun tomat, yang berhasil mempertahankan akurasi meskipun data terbatas [34]. Juga, dalam aplikasi identifikasi hewan (cattle facial recognition) dalam skenario few-shot, SNN dengan loss yang dimodifikasi menunjukkan perbaikan signifikan dibanding model baseline pada jumlah sampel terbatas [33].

1. ***Triplet Network***

Triplet Network adalah arsitektur dalam *metric learning* yang menggunakan tiga *input* sekaligus — *anchor*, *positive*, dan *negative* — untuk belajar representasi fitur (embedding) di mana *anchor-positive* lebih dekat daripada *anchor-negative* dalam ruang embedding. Tujuannya agar model dapat membedakan antar kelas berdasarkan jarak, bukan hanya klasifikasi langsung dengan label kelas. Pendekatan ini sangat berguna untuk tugas verifikasi atau *retrieval* seperti pengenalan wajah, ketika harus membandingkan data baru terhadap referensi dengan contoh terbatas. Arsitektur Triplet Network [35].

Arsitektur dasar *Triplet Network* terdiri dari tiga cabang jaringan identik yang berbagi bobot (weight sharing). Cabang-cabang ini masing-masing memproses input *anchor*, *positive*, dan *negative*, menghasilkan tiga embedding yang berbeda, misal f(xₐ), f(xₚ), dan f(xₙ). Ketiga output ini dibandingkan menggunakan metrik jarak, misalnya Euclidean, dalam fungsi loss agar embedding positif-anchor mendekat dan embedding negatif menjauh. Penggunaan struktur jaringan seperti CNN atau kombinasi CNN dan layer lainnya adalah umum tergantung modaltas data (citra, vis-audio, dsb.) [36].



Gambar 5 Arsitektur SNN dengan input berupa triplet elemen (*anchor, positive, dan negative*) [37]

* 1. **Triplet Loss**

Triplet Loss merupakan fungsi loss inti dari *Triplet Network*:

di mana adalah metrik jarak (misalnya Euclidean), dan adalah margin yang mengatur seberapa jauh *negative* harus lebih jauh dibanding *positive*. Margin ini mencegah model menjadi terlalu longgar dalam pemisahan kelas. Ada pula variasi seperti margin adaptif yang disesuaikan berdasarkan distribusi data agar stabilitas pelatihan meningkat [38].

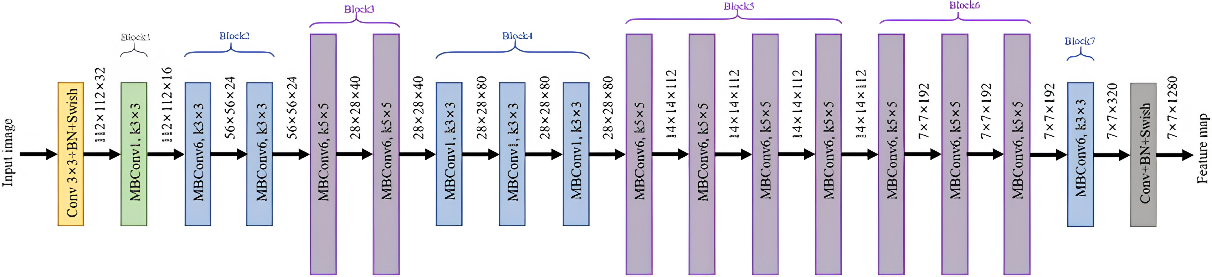
* 1. **Triplet Sampling**

Pemilihan triplet yang tepat sangat penting karena banyaknya kombinasi triplet yang mungkin dari dataset. Beberapa strategi sampling termasuk *hard negative mining* (negatif yang sangat mirip ke anchor), *semi-hard negative* (negatif yang berada diantara anchor-positive dan anchor-negative dalam hal jarak), dan teknik lain seperti *batch hard* dan *batch all*. Selain itu, metode baru mengusulkan sampling adaptif atau menggunakan probabilitas sampling negatif yang berubah seiring pelatihan (negative sampling annealing) agar model lebih efektif belajar dari contoh yang benar-benar sulit [35].

1. ***EfficientNet*-B0**

EfficientNet-B0 adalah model baseline dalam keluarga EfficientNet yang menggunakan metode *compound scaling*, yaitu menskalakan dimensi kedalaman (depth), lebar (width), dan resolusi input secara serentak agar mencapai keseimbangan antara akurasi dan efisiensi komputasi. Model ini menjadi dasar dari varian-varian yang lebih besar (B1 sampai B7) dan sering dipakai sebagai pilihan ringan dalam tugas klasifikasi dan transfer learning modern [39].

Arsitektur EfficientNet-B0 menggabungkan blok *mobile inverted bottleneck convolution* (MBConv blocks), teknik *squeeze-and-excitation* untuk adaptasi kanal, serta penggunaan resolusi input yang relatif rendah dibanding varian lebih besar agar model tetap ringan. Skema scaling-nya didasarkan pada parameter padat yang menyamakan skala perubahan pada depth, width, dan ukuran gambar sebagai bagian dari metode *compound scaling* [39].



Gambar 6 Arsitektur *EfficientNet*-B0

**2.9.1. Keunggulan dan Keterbatasan**

Model ini menawarkan keunggulan berupa keseimbangan yang baik antara ukuran parameter, kecepatan inferensi, dan akurasi, sehingga cocok digunakan pada lingkungan dengan sumber daya terbatas. Dalam penelitian “Deep Learning-Based Waste Classification with Transfer Learning Using EfficientNet-B0 Model” (jurnal.iaii.or.id), EfficientNet-B0 mencapai akurasi sekitar 91,94% untuk klasifikasi enam kelas sampah, dengan nilai presisi dan recall yang hampir serupa [40]. Selain itu, EfficientNet-B0 juga banyak diterapkan dalam bidang medis dan pertanian, seperti pada identifikasi penyakit gandum menggunakan varian EfficientNet-B0 yang dilengkapi dengan modul perhatian (attention mechanism), yang berhasil mencapai akurasi sekitar 98,70% dan F1-score sebesar 98% pada dataset nyata [41].

Walaupun efisien, performa EfficientNet-B0 dapat menjadi kurang optimal ketika digunakan untuk tugas yang memerlukan deteksi objek sangat kecil, detail halus, atau citra dengan resolusi rendah, karena informasi spasial dapat hilang selama proses downsampling atau pooling. Hal ini juga terlihat pada penelitian “Waste Classification - EfficientNet-B0” (jurnal.iaii.or.id), yang menunjukkan bahwa augmentasi dan fine-tuning diperlukan agar model mampu menangani variasi data dengan lebih baik [40]. Selain itu, adaptasi terhadap domain baru sering kali membutuhkan upaya tambahan seperti penerapan teknik augmentasi, fine-tuning pada layer tertentu, serta pemilihan dataset yang representatif agar performa model tidak menurun drastis. Sebagai contoh, pada penelitian deteksi Alzheimer menggunakan citra MRI (Semarak Ilmu), EfficientNet-B0 berbasis transfer learning mencapai akurasi tinggi pada data pelatihan, tetapi menunjukkan penurunan performa pada data validasi, menandakan adanya tantangan dalam kemampuan generalisasi terhadap data di luar distribusi pelatihan [42].

1. ***Visual Similarity***

Visual Similarity merujuk pada tingkat kemiripan antar dua gambar atau objek visual berdasarkan fitur visual seperti warna, tekstur, bentuk, struktur spasial, dan konteks. Tujuannya adalah agar sistem komputer mampu menilai seberapa “mirip” dua gambar bukan hanya berdasarkan label kelas, tetapi berdasarkan persepsi visual yang lebih mendalam—baik secara semantik maupun perseptual. Ini penting dalam tugas-retrieval gambar, verifikasi objek, dan tugas klasifikasi yang mengutamakan kemiripan visual [43].

Beberapa metode populer untuk mengukur *visual similarity* meliputi embedding berbasis *deep learning* (seperti CNN atau arsitektur yang mempelajari representasi visual), *global descriptor*, *metric learning* seperti *contrastive loss* atau *triplet loss*, serta pendekatan baru yang berupaya menciptakan fungsi adaptasi agar pengukuran *similarity* lebih mendekati persepsi manusia. Pendekatan-pendekatan ini dapat bersifat *supervised*, *semi-supervised*, maupun *self-supervised*, tergantung pada ketersediaan data dan anotasi [43]. Sebagai contoh, dalam penelitian “Learning Visual Similarity for Image Retrieval with Global Descriptors and Capsule Networks”, peneliti menggabungkan *global descriptors* dengan *Capsule Networks* untuk mempertahankan informasi spasial objek dan bagian-bagian visual penting, sehingga kinerja *image retrieval* meningkat secara signifikan [44].

Dalam konteks penelitian ini, pengukuran *visual similarity* antar karakter Kanji dilakukan menggunakan cosine similarity, yang terbukti efektif dalam mengukur kesamaan arah antar vektor embedding*. Cosine similarity* bekerja baik dalam menangkap kesamaan semantik pada *embedding space* yang dihasilkan oleh model pembelajaran representasi. Pada kasus pengenalan visual Kanji, embedding yang dihasilkan oleh SNN merepresentasikan arah vektor yang menggambarkan pola bentuk Kanji [45].

1. ***Image Retrieval***

Image Retrieval adalah tugas dalam computer vision dan multimedia yang bertujuan menemukan gambar-gambar yang relevan dari basis data besar berdasarkan sebuah kueri (query), yang bisa berupa gambar atau fitur visual. Proses ini melibatkan perbandingan fitur antara gambar kueri dengan gambar-gambar dalam database untuk mengurutkan hasil berdasarkan tingkat kemiripan. Pendekatan klasik menggunakan *feature engineering* tradisional (warna, tekstur, bentuk), sementara penelitian modern banyak didorong oleh penggunaan deep learning untuk mengekstrak representasi fitur yang lebih kuat dan lebih abstrak [46].

Seiring perkembangan, banyak metode *image retrieval* kontemporer menggabungkan fitur global dan lokal untuk meningkatkan akurasi. Contohnya, metode DOLG (*Deep Orthogonal Local and Global features*) mengintegrasikan fitur lokal dan global dalam satu representasi kompak dalam satu tahap (*single-stage*), sehingga mampu memberikan performa *retrieval* *state-of-the-art* pada dataset seperti Revisited Oxford dan Paris [47]. Selain itu, metode lain seperti *embedding* visual-semantik untuk *zero-shot image retrieval* juga semakin populer, karena memungkinkan sistem melakukan *retrieval* untuk kelas baru tanpa contoh pelatihan langsung, dengan memanfaatkan kecocokan antara gambar dan teks atau atribut semantik [48].

1. **Kajian Pustaka**

Penelitian-penelitian terbaru menunjukkan bahwa pendekatan berbasis Siamese Network memiliki peran penting dalam pengenalan pola visual dan pengukuran kesamaan antar entitas data. Serrano dan Bellogín melakukan tinjauan komprehensif mengenai penerapan Siamese Neural Network (SNN) dalam sistem rekomendasi dan menemukan bahwa arsitektur ini efektif dalam menangkap kemiripan antar item maupun pengguna, meskipun masih terdapat kendala reprodusibilitas hasil [37]. Pada tahun 2022 terdapat penelitian terkait Masked Siamese Networks (MSN) yang menggabungkan self-supervised learning dan masked image modeling untuk menghasilkan representasi visual yang lebih robust tanpa memerlukan label data [9]. Sementara itu, Bandara dan Patel pada tahun 2022 mengembangkan ChangeFormer, arsitektur transformer berbasis Siamese Network untuk mendeteksi perubahan pada citra multi-temporal [49], dan Liu et al. di tahun yang sama menggunakan Siamese Similarity Network dengan multi-scale fusion untuk mengenali karakter kuno melalui pendekatan one-shot learning. Secara keseluruhan, penelitian-penelitian ini menegaskan bahwa model berbasis Siamese efektif dalam memahami relasi semantik antar data, baik pada domain citra maupun teks [50].

Di sisi lain, fokus penelitian pada pengenalan karakter tulisan tangan, khususnya kanji, juga mengalami perkembangan pesat. Pada tahun 2023, Solis et al. mengembangkan CNN-ensemble untuk meningkatkan akurasi pengenalan karakter Jepang, seperti Hiragana, Katakana, dan Kanji [5], sedangkan penelitian yang dilakukan Pandkar et al. menggunakan Capsule Network dengan teknik pruning untuk mengurangi kompleksitas komputasi tanpa kehilangan akurasi [8]. Pada tahun 2025, Rusyn et al. memperkuat arah ini dengan menerapkan transfer learning lintas bahasa dari huruf Cina ke Jepang, yang terbukti meningkatkan performa pengenalan kanji [2]. Dari berbagai studi tersebut, terlihat bahwa meskipun performa model terus meningkat, tantangan seperti keterbatasan dataset spesifik kanji dan efisiensi komputasi masih menjadi celah riset yang perlu dieksplorasi lebih lanjut.

**BAB III**

**METODOLOGI PENELITIAN**

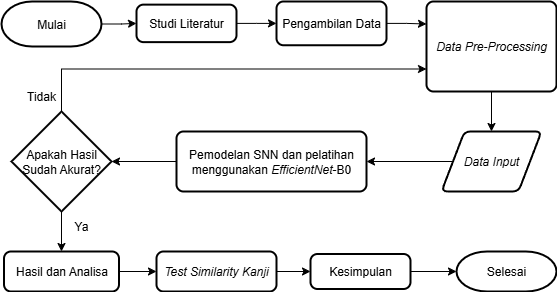
* 1. **Metode Penelitian**

Penelitian ini menggunakan Pendekatan kuantitatif karena data yang dianalisis bersifat numerik, yaitu hasil pengujian model berupa nilai akurasi, *loss*, *similarity score*, atau *mean average precision (mAP)*. Secara umum, tujuan penelitian ini adalah untuk menguji dan mengevaluasi performa Siamese Neural Network dalam pencarian kemiripan visual Kanji, serta menilai sejauh mana model tersebut mampu membedakan karakter Kanji yang mirip secara bentuk.

Tahapan-tahapan penelitian yang dilakukan untuk penelitian ini adalah sebagai berikut:

* 1. Melakukan studi literatur untuk memahami konsep dasar dan penelitian terdahulu yang relevan dengan topik.
  2. Pengumpulan dataset KanjiVG yang berisi informasi struktur, urutan goresan, serta komposisi visual tiap karakter. Dataset ini dipilih karena bersifat *open-source* dan telah digunakan secara luas dalam penelitian pengenalan tulisan Jepang.
  3. Melakukan data *pre-processing* seperti pengubahan format data dari svg ke png, data cleaning, dan data splitting.
  4. Merancang model SNN dengan EfficientNet-B0 sebagai *base model* untuk ekstraksi fitur.
  5. Melakukan pelatihan dengan data yang sudah siap, lalu menganalisis performa model setelah pelatihan, jika sudah akurat maka lanjut, jika tidak maka data dan model akan dievaluasi kembali.
  6. Setelah hasil pelatihan dianggap akurat, lakukan analisa lebih lanjut pada modelnya menggunakan lebih banyak metrik pengukuran, seperti *precision* dan *Recall@K*.
  7. *Test similarity kanji* menggunakan *cosine similarity*. Metrik ini digunakan untuk mengukur kesamaan arah antar vektor embedding.
  8. Buat kesimpulan dari penelitian yang telah dilakukan berdasarkan temuan yang diperoleh.

Secara keseluruhan tahapan dalam melakukan pengerjaan penelitian ini dapat dilihat dalam diagram alir pada Gambar 7 di bawah ini.



Gambar 7 Diagram alir penelitian

* 1. **Instrumen Penelitian**

Instrumen yang digunakan untuk mendukung penelitian ini meliputi perangkat keras dan perangkat lunak ditunjukan pada Tabel 1.

Tabel 1 Instrumen penelitian

|  |  |
| --- | --- |
| Instrumen | Detail |
| Laptop | Lenovo IdeaPad Gaming 3 15ACH6, prosesor AMD Ryzen™ 5 5600H, 3.30 GHz, RAM 24 GB, Windows 11 64-bit, x64-based processor |
| Google Colab | Free Tier, T4 GPU 16 GB VRAM |
| Python | Versi 3.12.11 |
| PyTorch | Versi 2.8.0+cu126 |
| TorchVision | Versi 0.23.0+cu126 |
| NumPy | Versi 2.0.2 |
| Matplotlib | Versi 3.10.0 |
| Scikit-learn | Versi 1.6.1 |
| Pillow | Versi 11.3.0 |
| cairosvg | Versi 2.8.2 |

* + 1. **Dataset Penelitian**

Dataset yang digunakan dalam penelitian ini adalah KanjiVG, yaitu kumpulan data vektor yang merepresentasikan karakter Kanji dalam bentuk grafik vektor skala (SVG — Scalable Vector Graphics). Dataset ini dikembangkan sebagai proyek *open source* oleh Ulrich Apel dan komunitas Wiktionary.

Dalam penelitian ini, seluruh file SVG pada dataset KanjiVG dikonversi menjadi citra PNG beresolusi 128×128 piksel, menghasilkan total 17756 sampel karakter Kanji dan 6391 class kanji yang unik. Setiap citra merepresentasikan satu karakter Kanji, dan data tersebut dibagi menjadi tiga bagian utama, yaitu data latih (training set) sebesar 70%, data validasi (validation set) sebesar 15%, dan data uji (testing set) sebesar 15%. Pembagian ini dilakukan untuk memastikan proses pelatihan model berjalan efektif dan evaluasi dilakukan secara objektif tanpa *data leakage*.

* 1. **Objek Penelitian**

Objek penelitian ini adalah citra karakter Kanji, yaitu huruf logografis yang digunakan dalam sistem penulisan bahasa Jepang. Setiap karakter Kanji tersusun dari sejumlah goresan (strokes) dan radikal (komponen pembentuk Kanji) yang membentuk pola visual kompleks dan unik. Dalam konteks penelitian ini, citra Kanji digunakan sebagai data utama untuk menguji kemampuan sistem dalam mengenali kemiripan visual antar karakter menggunakan pendekatan *deep learning*. Pemilihan karakter Kanji sebagai objek penelitian didasarkan pada kompleksitas bentuknya yang tinggi, keberagaman struktur visual, serta tingkat kemiripan antar karakter yang sering kali sulit dibedakan secara kasat mata, terutama bagi pembelajar bahasa Jepang atau sistem komputer vision tradisional.

Karakter Kanji memiliki variasi bentuk yang sangat luas, baik dari jumlah goresan, gaya penulisan, maupun proporsi visual antar komponennya. Beberapa karakter memiliki kemiripan tinggi karena berbagi radikal yang sama atau memiliki bentuk yang hampir identik, meskipun arti dan pelafalannya berbeda. Kompleksitas visual ini memberikan tantangan tersendiri dalam pengenalan pola, karena sistem harus mampu memahami kemiripan bentuk secara semantik dan struktural, bukan sekadar mencocokkan pola piksel. Oleh karena itu, citra Kanji menjadi objek yang ideal untuk menguji efektivitas algoritma *Visual Similarity Retrieval* berbasis Siamese Neural Network (SNN), yang dirancang untuk membedakan dan mengelompokkan citra berdasarkan kedekatan fitur representatifnya.

Dalam penelitian ini, setiap karakter Kanji diambil dari dataset KanjiVG, yang menyediakan representasi vektor dalam format SVG dan dikonversi menjadi PNG agar dapat diolah oleh model *deep learning*. Citra-citra ini digunakan sebagai input untuk sistem *Siamese Neural Network* yang bertujuan menghasilkan embedding vektor yang merepresentasikan pola visual tiap Kanji. Dengan memanfaatkan citra Kanji sebagai objek penelitian, studi ini diharapkan dapat menunjukkan sejauh mana model SNN mampu mengidentifikasi tingkat kemiripan visual antar karakter, serta memberikan kontribusi terhadap pengembangan sistem pencarian berbasis bentuk (*shape-based retrieval*) dalam konteks penulisan Jepang.

* 1. **Metode Pengumpulan Data**

Dataset yang digunakan dalam penelitian ini adalah KanjiVG, yaitu kumpulan data vektor yang merepresentasikan karakter Kanji dalam bentuk grafik vektor skala (SVG — Scalable Vector Graphics). Dataset ini dikembangkan sebagai proyek *open source* oleh Ulrich Apel dan komunitas Wiktionary.

Dalam penelitian ini, seluruh file SVG pada dataset KanjiVG dikonversi menjadi citra PNG beresolusi 128×128 piksel, menghasilkan total 17756 sampel karakter Kanji dan 6391 class kanji yang unik. Setiap citra merepresentasikan satu karakter Kanji, dan data tersebut dibagi menjadi tiga bagian utama, yaitu data latih (training set) sebesar 70%, data validasi (validation set) sebesar 15%, dan data uji (testing set) sebesar 15%. Pembagian ini dilakukan untuk memastikan proses pelatihan model berjalan efektif dan evaluasi dilakukan secara objektif tanpa *data leakage*.

* 1. **Pra-pemrosesan Data**

Tahap pra-pemrosesan data bertujuan untuk menyiapkan dataset agar dapat digunakan secara optimal dalam proses pelatihan model. Dataset yang digunakan dalam penelitian ini adalah KanjiVG, yang berisi representasi vektor karakter Kanji dalam format SVG. Agar data dapat diproses oleh model EfficientNet-B0 yang membutuhkan masukan berupa citra raster, dilakukan beberapa tahap pra-pemrosesan sebagaimana dijelaskan berikut ini.

* + 1. Konversi svg ke png

Data pada KanjiVG awalnya tersedia dalam format SVG, karena model EfficientNet-B0 bekerja pada data citra berbentuk piksel, maka seluruh file SVG dikonversi menjadi citra Portable Network Graphics (PNG). Proses konversi dilakukan secara otomatis menggunakan skrip Python yang memanfaatkan pustaka CairoSVG.

* + 1. Augmentasi Data

Agar menambahkan variasi pada data dan model tidak mengalami *overfitting* serta mampu mengenali karakter dengan variasi bentuk atau posisi, dilakukanlah tahap augmentasi data. Transformasi ini melakukan pemotongan acak (*random crop*) pada citra, dengan ukuran akhir 224×224 piksel, serta skala area potongan antara 80% hingga 95% dari ukuran asli. Teknik ini membantu model mempelajari representasi yang lebih robust terhadap perbedaan posisi, ukuran, dan proporsi karakter Kanji.

* + 1. Resize Citra

Setelah proses augmentasi, seluruh citra diubah ukurannya menjadi 224×224 piksel, ukuran tersebut dipilih karena menyesuaikan dengan ukuran input standar model *EfficientNet*-B0, yang diadaptasi dari arsitektur CNN konvensional.

* + 1. Normalisasi Data

Langkah selanjutnya adalah normalisasi citra untuk menyamakan rentang nilai piksel dengan distribusi yang digunakan pada *pretrained model* EfficientNet-B0. Format citra yang dipakai adalah RGB, lalu setiap channel pada RGB dinormalisasi menggunakan fungsi *Normalize* dari pustaka TorchVision, dengan kriteria sebagai berikut:

1. Channel merah memiliki rata-rata 0.485 dan simpangan baku 0.229.
2. Channel hijau memiliki rata-rata 0.456 dan simpangan baku 0.224.
3. Channer biru memiliki rata-rata 0.406 dan simpangan baku 0.225.

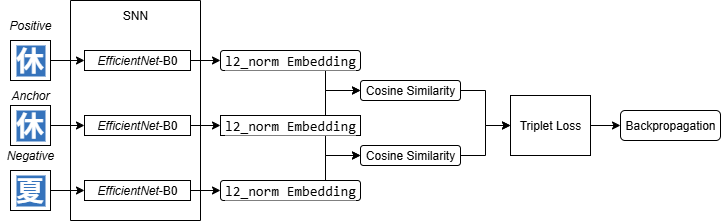
Rata-rata dan simpangan baku tersebut berasal dari dataset ImageNet, yang menjadi dasar pelatihan awal model *EfficientNet*-B0.

* + 1. Pembagian Data

Setelah seluruh citra selesai diproses, dataset dibagi menjadi tiga bagian dengan proporsi 70% untuk data pelatihan (*training set*), 15% untuk data pengujian (*testing set*), dan 15% untuk data validasi (*validation set*). Pembagian dilakukan secara acak namun tetap mempertahankan distribusi kelas yang seimbang pada setiap subset agar representasi data tetap konsisten. Selanjutnya, data pelatihan diubah ke dalam bentuk Triplet Dataset yang terdiri dari tiga elemen, yaitu *anchor*, *positive*, dan *negative*. Setiap triplet digunakan untuk menghitung *triplet loss* pada model SNN berbasis EfficientNet-B0, sehingga model dapat belajar mengenali tingkat kesamaan antar karakter Kanji berdasarkan representasi *embedding* yang dihasilkan.

* 1. **Desain Sistem**

Tahap ini menjelaskan rancangan sistem serta arsitektur model yang digunakan dalam penelitian, yang terdiri dari desain *Siamese Neural Network (SNN)*, pemilihan *backbone* EfficientNet-B0, penerapan fungsi *loss* berupa Triplet Loss, serta mekanisme perhitungan jarak antar *embedding*. Secara umum, model dirancang untuk melakukan pembelajaran kesamaan visual antar karakter Kanji berdasarkan representasi vektor yang dihasilkan pada ruang *embedding*.



Gambar 8 Arsitektur SNN yang digunakan pada penelitian ini

Arsitektur yang digunakan dalam penelitian ini adalah *Siamese Neural Network* (SNN) dengan skema *triplet input*, terdiri atas tiga cabang dengan bobot yang sama (*shared weights*) yang masing-masing menerima citra anchor, positive, dan negative. Citra anchor berfungsi sebagai acuan, positive merupakan citra yang berasal dari kelas yang sama, sedangkan negative berasal dari kelas yang berbeda. Ketiga citra tersebut diproses oleh backbone EfficientNet-B0 yang telah dimodifikasi dengan menghapus lapisan klasifikasi akhir dan menggantinya dengan lapisan embedding berdimensi tetap (misalnya 128 atau 256), sehingga jaringan menghasilkan tiga vektor representasi fitur. Bobot awal EfficientNet-B0 diinisialisasi menggunakan pretrained ImageNet dan kemudian disesuaikan melalui *fine-tuning* menggunakan dataset KanjiVG. Setiap embedding selanjutnya dinormalisasi menggunakan L2 normalization sebelum dibandingkan tingkat kemiripannya menggunakan Cosine Similarity. Perhitungan kemiripan ini kemudian dioptimalkan menggunakan Triplet Loss, yang bertujuan untuk mengecilkan jarak antara anchor dan positive sekaligus memperbesar jarak antara anchor dan negative hingga melebihi nilai margin tertentu. Nilai loss yang dihasilkan digunakan dalam proses backpropagation untuk memperbarui bobot jaringan agar model secara bertahap mampu membentuk ruang embedding yang terstruktur, dimana citra yang serupa berada berdekatan dan citra yang berbeda saling menjauh.

* 1. **Evaluasi Model**

Evaluasi performa model dilakukan menggunakan skema *image retrieval*, di mana sistem diberikan sebuah *query image* dan harus menemukan citra paling mirip dari *gallery* (kumpulan citra referensi yang telah diekstraksi embedding-nya). Hasil pencarian kemudian dinilai menggunakan tiga jenis metrik, yaitu *Recall@K*, *Mean Average Precision (mAP)*, dan *Visual Top-K Retrieval Results*.

* + 1. Recall@K

Recall@K digunakan untuk mengukur seberapa sering citra dengan kelas yang benar muncul di antara *K hasil teratas*. Jika minimal satu dari K hasil retrieval memiliki label yang sama dengan query, maka query tersebut dianggap berhasil. Pengujian dilakukan dengan berbagai nilai K, seperti K = 1, 3, dan 5.

* + 1. Mean Average Precision (mAP)

mAP mengevaluasi kualitas urutan hasil retrieval secara lebih menyeluruh. Untuk setiap query, dihitung Average Precision (AP), yaitu rata-rata presisi di setiap posisi di mana hasil yang benar ditemukan. Nilai mAP kemudian diperoleh dengan merata-ratakan AP dari seluruh query. Metrik ini memberikan gambaran seberapa konsisten sistem meletakkan gambar yang benar di posisi lebih tinggi dalam urutan hasil.

* + 1. Visual Top-K Retrieval Results

Selain evaluasi numerik, dilakukan pula evaluasi kualitatif dengan menampilkan *Top-K retrieval results* untuk beberapa query secara visual. Pendekatan ini berguna untuk memvalidasi apakah sistem benar-benar menangkap kemiripan bentuk secara visual, terutama pada karakter Kanji yang cenderung memiliki struktur serupa meskipun berbeda kelas.

* + 1. Logika Pengujian Query vs Gallery

Pada tahap evaluasi, dataset terlebih dahulu dibagi menjadi dua bagian, yaitu *query set* sebagai citra uji dan *gallery set* sebagai citra referensi. Seluruh citra pada gallery diekstraksi embedding-nya menggunakan model yang telah dilatih dan disimpan sebagai basis pencarian. Selanjutnya, untuk setiap citra pada query set, embedding dihitung dan dibandingkan dengan seluruh embedding di dalam gallery menggunakan *cosine similarity* atau *Euclidean distance*. Hasil perbandingan tersebut kemudian diurutkan berdasarkan tingkat kemiripan tertinggi. Berdasarkan urutan hasil retrieval tersebut, nilai *Recall@K* dan *Mean Average Precision (mAP)* dihitung secara kuantitatif, sedangkan evaluasi kualitatif dilakukan dengan menampilkan *Top-K retrieval results* untuk beberapa contoh query secara visual.

[1] N. T. Danh, “Sustanaible Methods of Improving Kanji Learning Skills for Japanese Language Learners at Basic Level at FPT University,” in *E3S Web of Conferences*, EDP Sciences, Jul. 2021. doi: 10.1051/e3sconf/202129505031.

[2] V. Rusyn, A. Boichuk, and L. Mochurad, “Cross-Language Transfer-Learning Approach via a Pretrained Preact ResNet-18 Architecture for Improving Kanji Recognition Accuracy and Enhancing a Number of Recognizable Kanji,” *Applied Sciences (Switzerland)*, vol. 15, no. 9, May 2025, doi: 10.3390/app15094894.

[3] D. Kurniadi, A. Mulyani, and N. Asih, “Comparative Analysis of Machine Learning Algorithms for Kanji Character Recognition Using HOG Features,” Jul. 24, 2025. doi: 10.20944/preprints202507.2038.v1.

[4] R. J. Rodríguez, B. Abstract, R. Badas, and R. José, “Kanji Recognition with AI,” 2024.

[5] A. I. Solis, J. Zarkovacki, J. Ly, and A. Atyabi, “Recognition of Handwritten Japanese Characters Using Ensemble of Convolutional Neural Networks,” Jun. 2023, [Online]. Available: http://arxiv.org/abs/2306.03954

[6] B. Lyu, H. Li, A. Tanaka, and L. Meng, “The early Japanese books reorganization by combining image processing and deep learning,” *CAAI Trans Intell Technol*, vol. 7, no. 4, pp. 627–643, Dec. 2022, doi: 10.1049/cit2.12104.

[7] P. Annisa, Z. A. Gultom, and Y. Sary, “Analysis and Implementation of CNN in Real- time Classification and Translation of Kanji Characters,” *Sinkron*, vol. 9, no. 1, pp. 296–305, Jan. 2024, doi: 10.33395/sinkron.v9i1.13176.

[8] S. Pandkar, K. Sabane, S. Rathod, P. Bansode, and S. Deore, “Handwritten Japanese Kanji Character Recognition using different Pruning Algorithm.”

[9] M. Assran *et al.*, “Masked Siamese Networks for Label-Efficient Learning,” Apr. 2022, [Online]. Available: http://arxiv.org/abs/2204.07141

[10] Y. Li, C. L. P. Chen, and T. Zhang, “A Survey on Siamese Network: Methodologies, Applications, and Opportunities,” *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 6, pp. 994–1014, Dec. 2022, doi: 10.1109/TAI.2022.3207112.

[11] D. Hartanto and R. Herawati, “COMPARATIVE ANALYSIS OF EFFICIENTNET AND RESNET MODELS IN THE CLASSIFICATION OF SKIN CANCER.”

[12] S. Otsuka and T. Murai, “Cognitive underpinnings of multidimensional Japanese literacy and its impact on higher-level language skills,” *Sci Rep*, vol. 11, no. 1, Dec. 2021, doi: 10.1038/s41598-021-81909-x.

[13] Y. Sakurai, Y. Uchiyama, A. Takeda, and Y. Terao, “On-Reading (Chinese-Style Pronunciation) Predominance Over Kun-Reading (Native Japanese Pronunciation) in Japanese Semantic Dementia,” *Front Hum Neurosci*, vol. 15, Aug. 2021, doi: 10.3389/fnhum.2021.700181.

[14] J. Breen and V. H. Mair, “SINO-PLATONIC PAPERS Kanji and the Computer: A Brief History of Japanese Character Set Standards.” [Online]. Available: www.sino-platonic.org

[15] A. A. Khan, A. A. Laghari, and S. A. Awan, “Machine Learning in Computer Vision: A Review,” *EAI Endorsed Transactions on Scalable Information Systems*, vol. 8, no. 32, pp. 1–11, 2021, doi: 10.4108/eai.21-4-2021.169418.

[16] M. Gao, G. Zou, Y. Li, and X. Guo, “Recent Advances in Computer Vision: Technologies and Applications,” Jul. 01, 2024, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/electronics13142734.

[17] “View of Deep Learning in Computer Vision\_ A Critical Review”.

[18] M. H. M. Noor and A. O. Ige, “A Survey on State-of-the-art Deep Learning Applications and Challenges,” Jul. 2025, doi: 10.1016/j.engappai.2025.111225.

[19] D. Dakshayani Himabindu and S. Praveen Kumar, “A Survey on Computer Vision Architectures for Large Scale Image Classification using Deep Learning.” [Online]. Available: www.ijacsa.thesai.org

[20] N. E. Khalifa, M. Loey, and S. Mirjalili, “A comprehensive survey of recent trends in deep learning for digital images augmentation,” *Artif Intell Rev*, vol. 55, no. 3, pp. 2351–2377, Mar. 2022, doi: 10.1007/s10462-021-10066-4.

[21] P. Mittal, “A comprehensive survey of deep learning-based lightweight object detection models for edge devices,” *Artif Intell Rev*, vol. 57, no. 9, Sep. 2024, doi: 10.1007/s10462-024-10877-1.

[22] J. Liu and Y. Jin, “A comprehensive survey of robust deep learning in computer vision,” Nov. 01, 2023, *KeAi Communications Co.* doi: 10.1016/j.jai.2023.10.002.

[23] P. K. Vinodkumar, D. Karabulut, E. Avots, C. Ozcinar, and G. Anbarjafari, “A Survey on Deep Learning Based Segmentation, Detection and Classification for 3D Point Clouds,” Apr. 01, 2023, *MDPI*. doi: 10.3390/e25040635.

[24] X. Zhao, L. Wang, Y. Zhang, X. Han, M. Deveci, and M. Parmar, “A review of convolutional neural networks in computer vision,” *Artif Intell Rev*, vol. 57, no. 4, Apr. 2024, doi: 10.1007/s10462-024-10721-6.

[25] R. Sunkara and T. Luo, “No More Strided Convolutions or Pooling: A New CNN Building Block for Low-Resolution Images and Small Objects,” Aug. 2022, [Online]. Available: http://arxiv.org/abs/2208.03641

[26] M. M. Taye, “Theoretical Understanding of Convolutional Neural Network: Concepts, Architectures, Applications, Future Directions,” Mar. 01, 2023, *MDPI*. doi: 10.3390/computation11030052.

[27] P. Wu, “Frontiers in Computing and Intelligent Systems A Survey of Few-Shot Learning Research Based on Deep Neural Network”.

[28] K. He, N. Pu, M. Lao, and M. S. Lew, “Few-shot and meta-learning methods for image understanding: a survey,” *Int J Multimed Inf Retr*, vol. 12, no. 2, Dec. 2023, doi: 10.1007/s13735-023-00279-4.

[29] Y. Song, T. Wang, S. K. Mondal, and J. P. Sahoo, “A Comprehensive Survey of Few-shot Learning: Evolution, Applications, Challenges, and Opportunities,” May 2022, [Online]. Available: http://arxiv.org/abs/2205.06743

[30] R. Wei and A. Mahmood, “Optimizing few-shot learning based on variational auto encoders,” *Entropy*, vol. 23, no. 11, Nov. 2021, doi: 10.3390/e23111390.

[31] S. Antonelli *et al.*, “Few-Shot Object Detection: A Survey,” *ACM Comput Surv*, vol. 54, no. 11 S, Jan. 2022, doi: 10.1145/3519022.

[32] Y. Li, C. L. P. Chen, and T. Zhang, “A Survey on Siamese Network: Methodologies, Applications, and Opportunities,” *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 6, pp. 994–1014, Dec. 2022, doi: 10.1109/TAI.2022.3207112.

[33] J. Porto *et al.*, “A New Siamese Network Loss for Cattle Facial Recognition in a Few-Shot Learning Scenario,” *AgriEngineering*, vol. 6, no. 3, pp. 2941–2954, Sep. 2024, doi: 10.3390/agriengineering6030169.

[34] S. Thuseethan, P. Vigneshwaran, J. Charles, and C. Wimalasooriya, “Siamese Network-Based Lightweight Framework for Tomato Leaf Disease Recognition,” *Computers*, vol. 13, no. 12, Dec. 2024, doi: 10.3390/computers13120323.

[35] G. Kertész, “Deep Metric Learning Using Negative Sampling Probability Annealing,” *Sensors*, vol. 22, no. 19, Oct. 2022, doi: 10.3390/s22197579.

[36] C. Zhu, Z. Hu, H. Dong, G. He, Z. Yu, and S. Zhang, “Construct Informative Triplet with Two-stage Hard-sample Generation,” Dec. 2021, [Online]. Available: http://arxiv.org/abs/2112.02259

[37] N. Serrano and A. Bellogín, “Siamese neural networks in recommendation,” Jul. 01, 2023, *Springer Science and Business Media Deutschland GmbH*. doi: 10.1007/s00521-023-08610-0.

[38] M. L. Ha and V. Blanz, “Deep Ranking with Adaptive Margin Triplet Loss,” Jul. 2021, [Online]. Available: http://arxiv.org/abs/2107.06187

[39] A. Panthakkan, S. M. Anzar, S. Al Mansoori, W. Mansoor, and H. Al Ahmad, “A systematic comparison of transfer learning models for COVID-19 prediction,” *Intelligent Decision Technologies*, vol. 16, no. 3, pp. 557–574, Jun. 2022, doi: 10.3233/IDT-220017.

[40] Risfendra, G. F. Ananda, and H. Setyawan, “Deep Learning-Based Waste Classification with Transfer Learning Using EfficientNet-B0 Model,” *Jurnal RESTI*, vol. 8, no. 4, pp. 535–541, Aug. 2024, doi: 10.29207/resti.v8i4.5875.

[41] S. Nigam, R. Jain, V. K. Singh, S. Marwaha, A. Arora, and S. Jain, “EfficientNet architecture and attention mechanism-based wheat disease identification model,” in *Procedia Computer Science*, Elsevier B.V., 2024, pp. 383–393. doi: 10.1016/j.procs.2024.04.038.

[42] “View of Transfer Learning for Alzheimer’s Disease Diagnosis using EfficientNet-B0 Convolutional Neural Network”.

[43] O. Risser-Maroix, A. Marzouki, H. Djeghim, C. Kurtz, and N. Lomenie, “Learning an Adaptation Function to Assess Image Visual Similarities,” Jun. 2022, [Online]. Available: http://arxiv.org/abs/2206.01417

[44] D. Durmuş, “LEARNING VISUAL SIMILARITY FOR IMAGE RETRIEVAL WITH GLOBAL DESCRIPTORS AND CAPSULE NETWORKS,” 2021.

[45] H. Steck, C. Ekanadham, and N. Kallus, “Is Cosine-Similarity of Embeddings Really About Similarity?,” Mar. 2024, doi: 10.1145/3589335.3651526.

[46] W. Chen *et al.*, “Deep Learning for Instance Retrieval: A Survey,” Oct. 2022, [Online]. Available: http://arxiv.org/abs/2101.11282

[47] M. Yang *et al.*, “DOLG: Single-Stage Image Retrieval with Deep Orthogonal Fusion of Local and Global Features.”

[48] K. Ueki, “Survey of Visual-Semantic Embedding Methods for Zero-Shot Image Retrieval,” Sep. 2021, [Online]. Available: http://arxiv.org/abs/2105.07391

[49] W. G. C. Bandara and V. M. Patel, “A Transformer-Based Siamese Network for Change Detection,” in *International Geoscience and Remote Sensing Symposium (IGARSS)*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 207–210. doi: 10.1109/IGARSS46834.2022.9883686.

[50] X. Liu, W. Gao, R. Li, Y. Xiong, X. Tang, and S. Chen, “One shot ancient character recognition with siamese similarity network,” *Sci Rep*, vol. 12, no. 1, Dec. 2022, doi: 10.1038/s41598-022-18986-z.