BAB II

TINJAUAN PUSTAKA

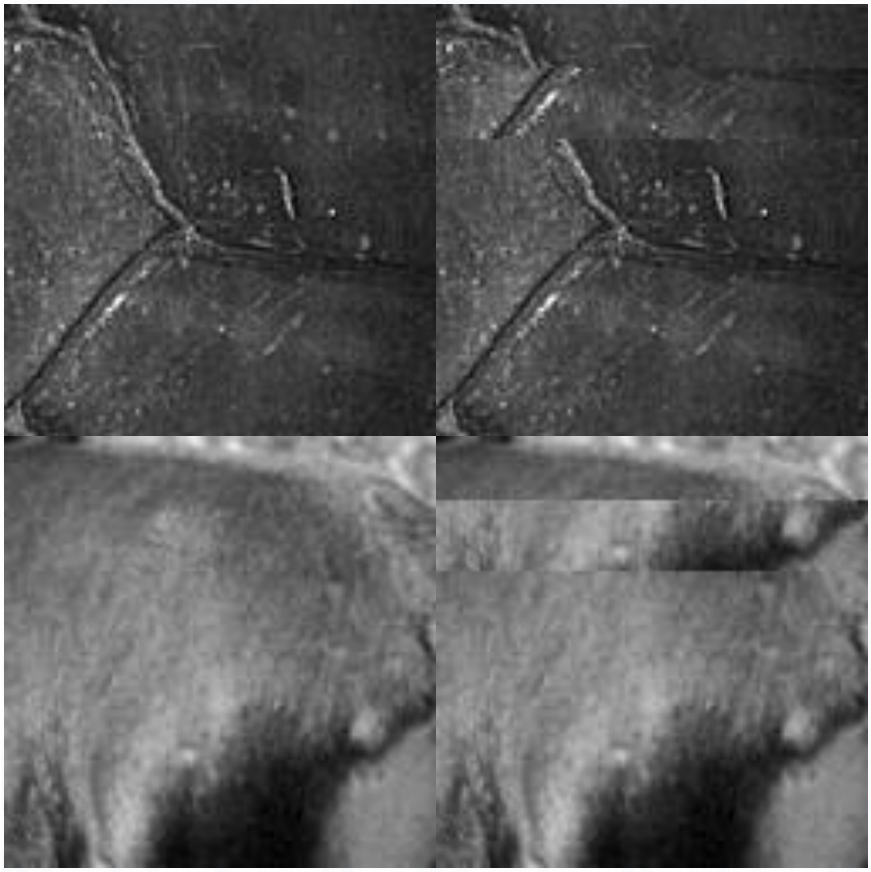
2.1 *Digital watermarking*

Digital watermarking adalah sebuah proses untuk menanamkan tanda atau pola unik pada sebuah objek atau informasi dalam bentuk digital. Awalnya, *watermark* dibuat oleh organisasi pemerintahan untuk mencegah peniruan dokumen. Pada perkembangannya, bentuk *watermarking* menjadi lebih beragam, seperti *logo* dan *emblem* pada sebuah *brand*. Hingga saat ini, *digital watermarking* banyak ditanamkan pada dokumen, gambar, suara, dan video sebagai bentuk perlindungan terhadap informasi digital. Penggunaan *digital watermarking* tergolong efisien karena *watermark* mampu mempertahankan keaslian konten dan data meskipun terdapat upaya manipulasi. Dengan demikian, *digital watermarking* digunakan sebagai identitas untuk mengklaim *copyright* dan mencegah duplikasi yang tidak sah [5]. Contoh gambar yang sudah ditanamkan *watermark* terdapat pada Gambar 2.1.

Gambar 2.1 Contoh Digital *Watermark*ing pada Gambar [12]

2.2 *Splicing*

Tindakan pemalsuan gambar semakin umum terjadi pada era digital saat ini. Salah satu teknik yang paling sering digunakan adalah *image splicing*. *Splicing* merupakan salah satu teknik pemalsuan gambar dengan menambahkan satu atau lebih area pada gambar ke area gambar lainnya [13]. Untuk mengidentifikasi pemalsuan gambar yang dilakukan dengan teknik *splicing*, deteksi *image splicing* menjadi penting untuk dilakukan. Tujuan utama dari deteksi *splicing* adalah mendeteksi apakah sebuah gambar yang diberikan adalah gambar gabungan yang dihasilkan dengan memotong dan menggabungkan dua atau lebih gambar yang berbeda dengan tujuan untuk menyesatkan informasi [14]. Melalui deteksi *splicing*, sebuah gambar bisa dipastikan keasliannya. Untuk lebih jelasnya, Gambar 2.2 akan menunjukkan perbedaan gambar asli dengan gambar yang sudah dirusak menggunakan teknik *splicing*.



Gambar 2.2 Contoh Perbedaan *Authentic* (Kiri) dan *Splicing* (Kanan) *Dataset* DVMM [15]

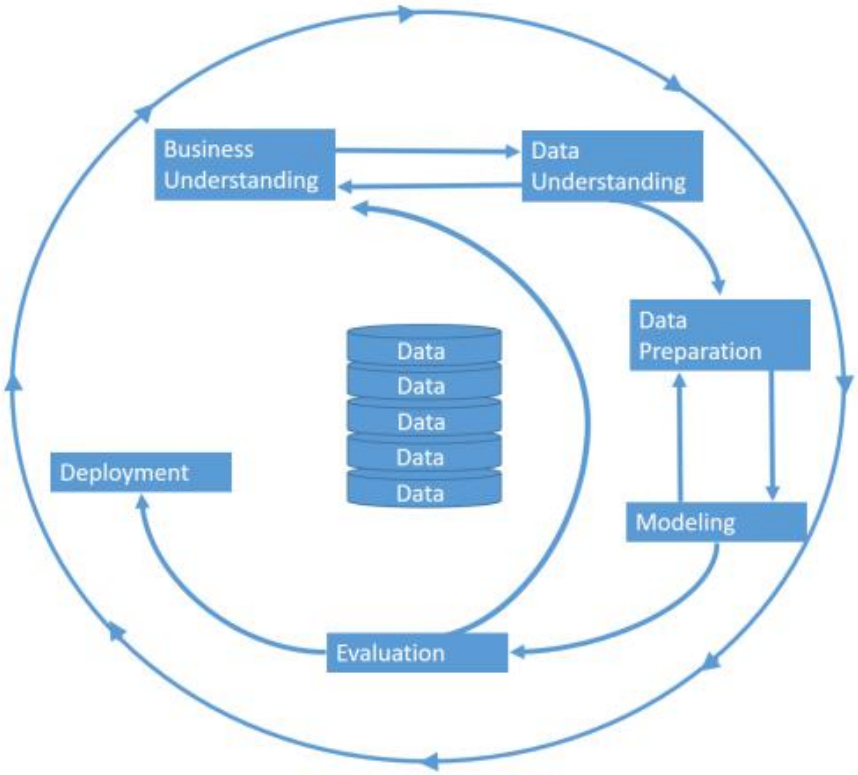
2.3 *Deep learning*

Deep learning merupakan turunan dari *machine learning* yang berbasis jaringan saraf konvensional dengan keunggulan yang jauh lebih tinggi dibandingkan pendahulunya. Hal ini disebabkan oleh kemampuan *deep learning* yang mampu membangun model pembelajaran *multi-layer* menggunakan transformasi dan teknologi grafik secara bersamaan. Teknologi *deep learning* baru-baru ini memiliki kinerja yang luar biasa dalam berbagai bidang, seperti pemrosesan audio, data visual, *natural language*, dan lain-lain dibandingkan model lainnya. *Feature extraction* dilakukan secara otomatis menggunakan algoritma *deep learning* sehingga para peneliti bisa mengekstrak *features* meskipun sumber daya dan pengetahuannya terbatas. Algoritma *deep learning* mempunyai aksitektur representasi data *multi-layer* dimana *layer* pertama akan mengekstrak *low-level features* hingga *layer* terakhir akan mengekstrak *high-level features* [10].

2.3 Metodologi *deep learning*

Dalam proses pembuatan model *deep learning*, ada beberapa metodologi yang bisa digunakan, yaitu CRISP-DM, KDD, dan SEMMA. Berikut ini adalah penjelasan masing-masing metodologi mengenai tahapan apa saja yang harus dilalui pada penggunaan metodologi tersebut.

Cross-Industry Standard Process for Data Mining (CRISP-DM) memiliki 6 tahapan yang harus dijalankan saat membangun model *deep learning*. Setiap hasil tahapan akan menentukan apakah kita harus melanjutkan tahapan berikutnya atau mengulangi tahapan tersebut. Oleh karena itu, tahapan CRISP-DM sifatnya tidak kaku karena kita bebas mengulangi tahapan tertentu untuk mendapatkan hasil yang paling optimal. Gambar 2.3 menujukkan kerangka berpikir metode CRISP-DM.



Gambar 2.3 Tahapan CRISP-DM [16]

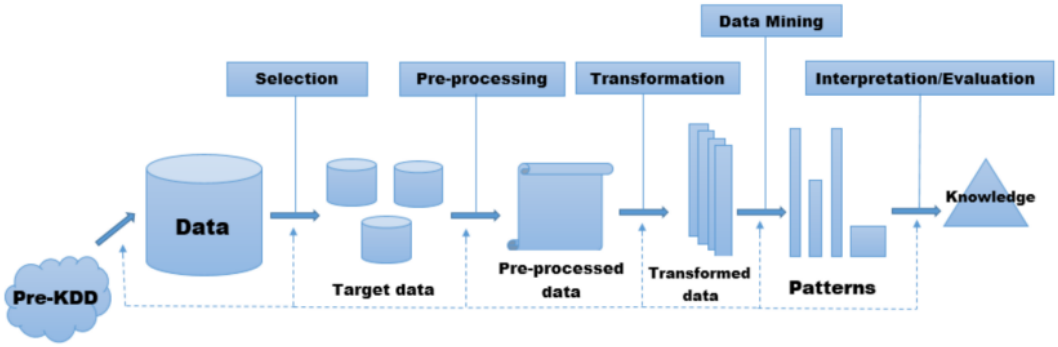
Tahap-tahap CRISP-DM terdiri atas:

1. *Business understanding*: memahami tujuan dan kebutuhan *project* dari perspektif bisnis kemudian ditransformasi menjadi sebuah masalah sebagai landasan awal pembuatan rencana project untuk mencapai tujuan bisnis.
2. *Data understanding*: mengumpulkan data awal untuk mengetahui tujuan memahami data. Mengidentifikasi kulitas data, mencari wawasan pada data awal, dan mendeteksi *subset* yang menarik menjadi bagian dari *data understanding* ini.
3. *Data preparation*: setelah data dikumpulkan, data harus disiapkan supaya dapat membuat *dataset* final. Beberapa aktivitas *data preparation*, seperti pemilihan tabel, *record*, atribut, transformasi, dan pembersihan data.
4. *Modelling*: pemilihan dan penerapan berbagai teknik pemodelan pada *project*. Pengukuran parameter dilakukan pada model-model tersebut untuk memperoleh hasil yang optimal. Tahap ini seringkali mengembalikan kita ke

tahap *data preparation* karena setiap model memiliki kebutuhan *dataset* yang berbeda-beda.

5. *Evaluation*: evaluasi model diperlukan untuk memastikan hasil akhirnya memenuhi tujuan bisnis yang diinginkan sejak awal. Evaluasi dilakukan secara menyeluruh untuk memutuskan hasil model yang paling optimal.
6. *Deployment*: model akhri akan diimplementasikan pada tahap ini. Bergantung pada kebutuhan bisnis, *deployment* bisa dilakukan dengan sederhana, seperti mengintegrasikan model pada sistem operasi atau lebih kompleks.

Knowledge Discovery in Databases (KDD) adalah sebuah proses untuk mendapatkan pengetahuan yang bermanfaat dari basis data. Fokus KDD adalah menemukan pengetahuan dari data, mulai dari cara data disimpan dan diakses, bagaimana menjalankan algoritma pada *dataset* yang besar tetapi tetap efisien, dan bagaimana hasilnya dapat diinterpretasikan dan divisualisasikan dengan baik. Tahapan KDD merupakan proses berulang yang dipaparkan pada Gambar 2.4 di bawah ini:

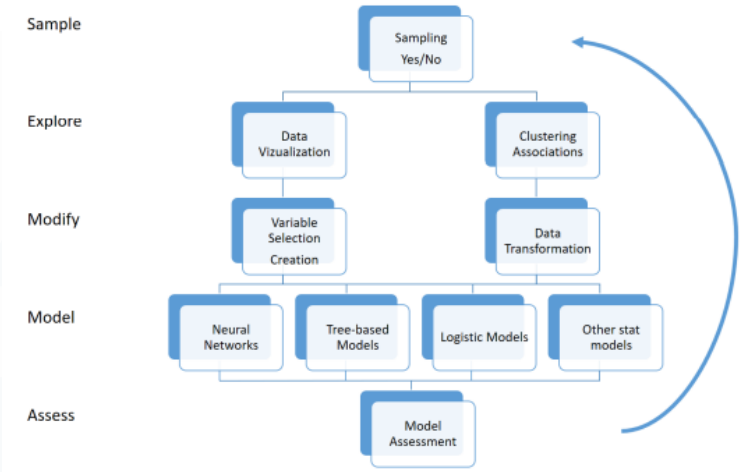


Gambar 2.4 Tahapan KDD [16]

1. *Pre-KDD*: penyelidikan untuk memahami domain *project* yang akan dikembangkan dan apa yang perlu dilakukan untuk memperoleh pengetahuan yang relevan. Hasil akhirnya adalah sebuah tujuan *project* dari sudut pandang *end-user*.

2. *Selection*: pembuatan *dataset* target berdasarkan data-data yang diperoleh pada tahap sebelumnya. Data-data yang didapatkan akan diintegrasikan menjadi satu *dataset* yang berfokus pada sebagian variabel atau sampel data.
3. *Pre-processing*: tahap pembersihan data dengan cara menghapus *noise*, mengumpulkan informasi yang diperlukan untuk pemodelan, mengatasi bagian data yang *missing*, dan memperhitungkan informasi urutan waktu dan perubahan yang diketahui.
4. *Transformation*: tahap menyiapkan data untuk proses *data mining*. *Feature* yang merepresentasikan data akan ditemukan pada tahapan ini. Metode yang umum digunakan pada tahap ini adalah pengurangan dimensi, seperti *feature selection and extraction*.
5. *Data mining*: memiliki beberapa langkah untuk menjalankannya, yaitu memilih metode *data mining* berdasarkan tujuan KDD pada langkah awal, memilih algoritma dan metode untuk menemukan pola-pola penting dalam data, dan pengulangan implementasi algoritma *data mining* untuk mencari pola-pola yang menarik pada *dataset* demi memperoleh hasil yang optimal.
6. *Interpretation/Evaluation*: interpretasi atau evaluasi pola-pola yang ditemukan pada langkah sebelumnya untuk memastikan hasilnya sesuai dengan tujuan KDD yang ditetapkan pada langkah awal. Kembali ke tahapan sebelumnya juga memungkinkan untuk menghasilkan beberapa perubahan.
7. *Post-KDD*: mengambil tindakan berdasarkan hasil akhir yang sudah diperoleh. Pengetahuan yang didapatkan bisa digunakan secara langsung ataupun diimplementasikan ke dalam sistem serta dibuat dokumentasi atau laporannya.

Sample, Explore, Modify, Model, Asses (SEMMA) adalah sebuah *toolset* untuk mengeksekusi tugas-tugas inti dalam pengembangan model. SEMMA berfokus pada pengembangan model dan banyak digunakan pada SAS Enterprise Mine *software*. Sifat tahapan SEMMA tidak kaku, sehingga perpindahan langkah baik maju atau mundur dapat dilakukan sesuai kebutuhan. SEMMA memiliki 5 tahapan seperti yang tertera pada Gambar 2.5:



Gambar 2.5 Tahapan SEMMA [16]

1. *Sample*: pengambilan sampel data untuk membangun model. Tahap ini juga membagi data-data yang dikumpulkan menjadi sampel *training*, *validation*, dan *testing*.
2. *Explore*: tahap eksplorasi pola-pola dan hubungan yang menarik untuk memahami data. Pemahaman data berguna untuk mendapatkan ide dan membantu mengambil kesimpulan. Tahap ini bisa dilakukan dengan visualisai ataupun analisis statistik.
3. *Modify*: modifikasi data hasil eksplorasi agar sesuai dengan model yang ingin dibangun. Tahap ini juga mencakup segmentasi tambahan dan pembuatan variabel baru.
4. *Model*: pembuatan model dengan menerapkan teknik pemodelan pada data dan variabel untuk mendapatkan model yang dapat diandalkan untuk memprediksi hasil atau mengklasifikasikan data.
5. *Assess*: evaluasi hasil dan kinerja model terhadap sampel *validation* dan *testing*. Melalui evaluasi ini, akan ditemukan kesimpulan apakah model bermanfaat dan dapat diandalkan [16].

2.4 Algoritma yang digunakan

Convolutional Neural Network (CNN) adalah metode *deep learning* yang mengambil *input* gambar dan menetapkan bias dan bobot yang dapat dipelajari pada

berbagai objek dalam gambar untuk membedakannya. Dibandingkan dengan metode *deep learning* lainnya, CNN membutuhkan *pre-processing* yang lebih sedikit. CNN bekerja dengan merakit kembali jaringan saraf kovensional tetapi dengan karakteristik khusus yang terdiri atas bobot dan bias yang bisa dipelajari oleh CNN. Setiap jaringan CNN menerima banyak *input* kemudian melakukan perkalian titik yang secara opsional diikuti oleh nonlinier. Tujuannya untuk mengekstrak *feature* penting dari sumber data lokal terkait [11].

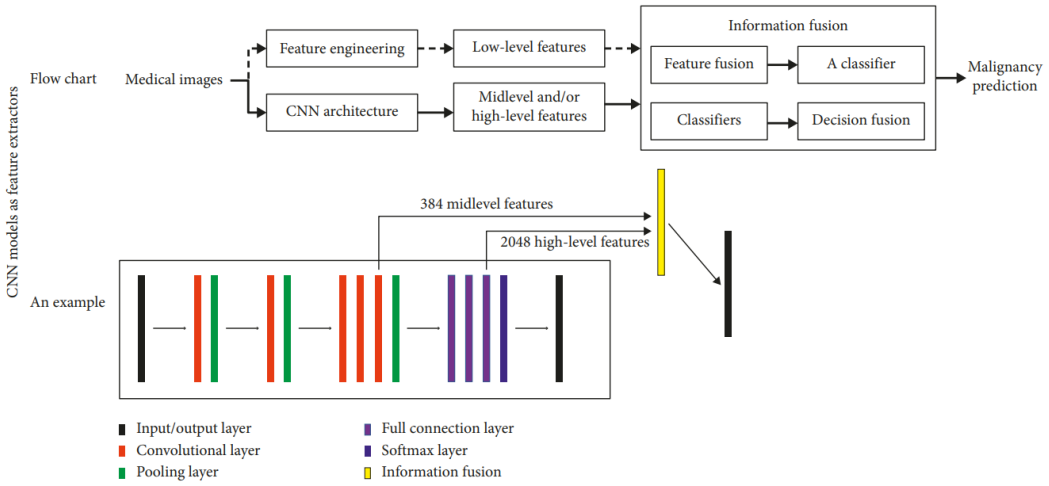
CNN memiliki beberapa lapisan (*layer*) yang bekerja untuk memproses gambar, yaitu [17]:

1. *Convolutional layer*

   Lapisan pertama bertanggung jawab untuk mengekstrasi *features* yang ada dalam *input* gambar digital berdasarkan filter yang sudah ditetapkan pada operasi konvolusi. Pada lapisan ini juga terdapat sebuah kernel inti berukuran kecil yang bergerak pada seluruh wilayah *input* gambar untuk menghasilkan *feature map*. *Feature map* akan menampung informasi lokasi pola atau *feature input data*.
2. *Pooling layer*

   Lapisan kedua bertanggung jawab untuk mengurangi ukuran data pada *feature map* yang dihasilkan oleh *convolutional layer* sebelumnya. Metode yang sering digunakan untuk merealisasikannya adalah *max pooling*. *Max pooling* bekerja dengan menetapkan nilai maksimum dari setiap wilayah *pooling*. Metode *max pooling* dilakukan dengan tujuan meningkatkan invarian pada pergeseran dan mengurangi *overfitting* pada model nantinya.
3. *Fully connected layer*

   Lapisan terakhir bertanggung jawab untuk menghubungkan *node* lapisan sebelumnya dengan *node* lapisan *output* dengan tujuan klasifikasi data. Pada lapisan ini juga dilakukan perhitungan *loss* (*error*) pada model untuk menentukan apakah optimasi diperlukan pada model akhir atau tidak.



Gambar 2.6 *Feature extraction* pada CNN [18]

Gambar 2.6 merupakan alur kerja CNN ketika melakukan *feature extraction* menggunakan lapisan-lapisannya. *Feature extraction* yang diperoleh memiliki dua pendekatan yang berbeda, yaitu *feature* yang diikuti oleh pengklasifikasi dan *feature* yang diprediksi oleh satu atau lebih pengklasifikasi [18]. *Feature extraction* pada CNN dilakukan secara otomatis saat proses *training data*. Oleh karena itu, *layer-layer* yang ada pada CNN akan menyesuaikan bobot filter selama proses *training data* untuk mengekstrak fitur-fitur penting pada *dataset*.

Konsep dasar CNN memanfaatkan seluruh struktur lapisan untuk mengambil *feature* penting dari *input* guna memprediksi atau melakukan segmentasi berdasarkan *feature* penting tersebut. Selain dapat dijabarkan dalam bentuk teori, cara kerja CNN dengan lapisan-lapisannya juga memiliki rumus matematika yang mencakup operasi konvolusi, aktivasi, dan *pooling*. Rumus matematika ini juga dapat menjelaskan bagaimana lapisan-lapisan CNN berperan dalam pengenalan objek, klasifikasi, dan segmentasi *dataset* [19]. CNN bekerja dengan konvolusi gambar atau memecah *input* gambar menjadi lebih kecil kemudian diproses dalam jaringan neural untuk menghasilkan representasi *feature* yang dibutuhkan. Rumus untuk menghasilkan *layer* konvolusi adalah [20]:

$$W_{out} = \frac{W-F+2P}{S} + 1 \quad (1)$$

Keterangan rumus:

W: Ukuran input (lebar atau tinggi, tergantung pada sumbu mana filter diterapkan).

F: Ukuran filter (lebar atau tinggi filter).

P: *Zero-padding* - jumlah padding nol yang ditambahkan ke sekitar input.

S: *Stride* (langkah) - seberapa jauh filter bergerak setiap kali filter bergerak melintasi input.

Setelah dibuat *layer* konvolusi, representasi *feature* yang dihasilkan akan diproses oleh *max-pooling layer* untuk menyimpan informasi penting. Informasi penting akan diteruskan ke *fully-connected layer* untuk diprediksi. Output dari *max-pooling layer* bisa dirumuskan sebagai berikut [20]:

$$W_{out} = \frac{W-F}{S} + 1 \quad (2)$$

Keterangan rumus:

W: Ukuran input (lebar atau tinggi, tergantung pada sumbu mana filter diterapkan).

F: Ukuran filter (lebar atau tinggi filter).

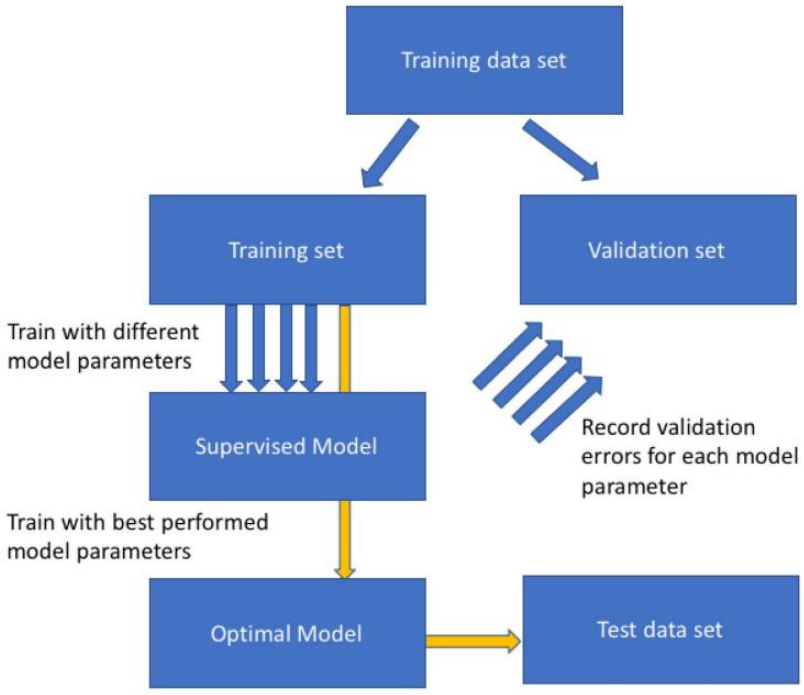
S: *Stride* (langkah) - seberapa jauh filter bergerak setiap kali filter bergerak melintasi input.

2.5 Data *Training*, Data Validasi, dan Data *Testing*

Dalam sebuah proses penelitian deteksi pemalsuan gambar, *dataset* yang digunakan dibagi menjadi 3 kelompok, yaitu data *training*, data validasi, dan data *testing*. Data *training* digunakan untuk melatih model agar mampu mendeteksi objek tertentu pada *dataset*. Selama proses pelatihan, model dilatih untuk memahami pola atau *feature* tertentu yang terkandung dalam *dataset*. Setelah model selesai dilatih, model akan diuji kembali menggunakan dua kelompok data yang berbeda, yaitu data validasi dan data *testing* [21].

Data validasi digunakan untuk mengevaluasi model prediksi setelah dilatih. Melalui evaluasi menggunakan data validasi, model prediksi dibandingkan dengan label sebenarnya yang ada pada data validasi. Tujuan evaluasi menggunakan data validasi adalah mengukur sejauh apa model dapat mendeteksi objek tertentu yang belum diketahui saat proses pelatihan. Identifikasi model apakah *overfitting* atau *underfitting* akan terlihat pada proses ini [22].

Evaluasi menggunakan data *testing* dilakukan untuk pengujian keseluruhan akurasi dan kinerja model prediksi. Data yang digunakan pada proses ini harus berbeda dengan data *training* dan data validasi. Hasil proses ini memberi gambaran akurat mengenai kemampuan model mendeteksi objek pada *dataset* yang digunakan [21]. Alur kerja proses pembuatan dan pengujian model prediksi menggunakan data *training*, data validasi, dan data *testing* dapat dilihat pada Gambar 2.7.

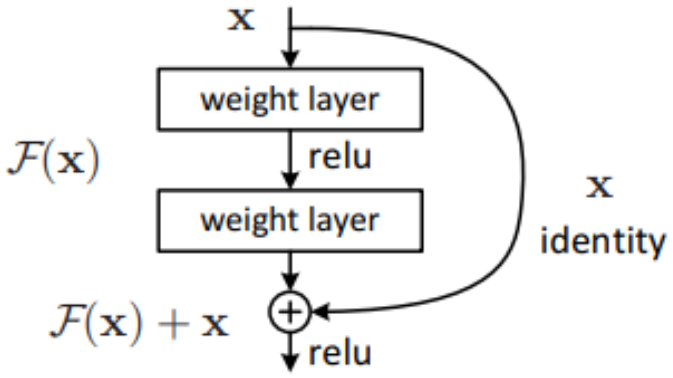


Gambar 2.7 Alur Kerja Pemilihan Model Deteksi [21]

2.6 Algoritma Pembanding

2.6.1. ResNet-50

Residual Neural Network (ResNet) merupakan model yang sudah dilatih sehingga tidak memerlukan konfigurasi yang kompleks untuk mengatur *layer* di dalamnya. Oleh karena itu, ResNet bisa langsung diimplementasikan untuk membuat klasifikasi gambar [23]. Namun, untuk beberapa kasus khusus perlu dilakukan beberapa penyesuaian pada *layer* untuk memenuhi syarat kebutuhan kasus yang bersangkutan. ResNet bekerja dengan konsep *residual block* yang mengirim informasi dari lapisan *input* ke lapisan *output* melalui *shortcut connections*. Melalui *shortcut connections*, informasi yang dikirimkan dapat sampai tanpa masalah yang berarti. Tujuannya untuk mencegah penurunan performa. Biasanya, setelah ResNet terdapat sebuah angka, dimana pada penelitian ini adalah 50. ResNet-50 berarti struktur ResNet yang digunakan memiliki 50 lapisan *residual network*. Struktur *layer* Res-Net terlihat pada Gambar 2.8.

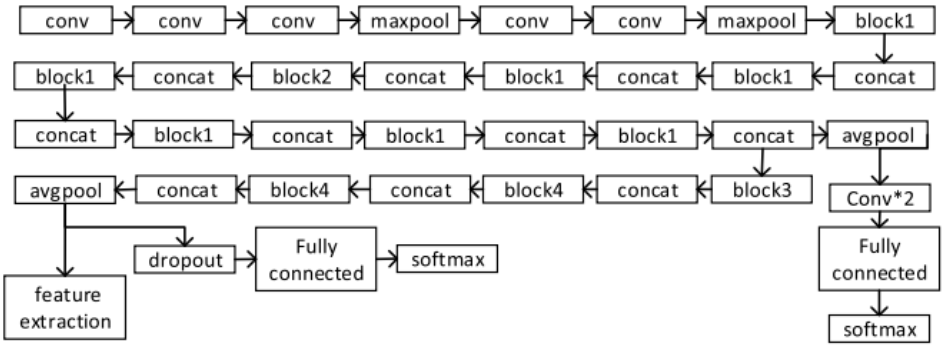


Gambar 2.8 Struktur *Layer* ResNet [24]

2.6.2. Inception-V3

Inception-V3 merupakan bagian dari arsitektur Inception yang dikenal karena efisiensinya dalam mengenali objek dalam citra atau gambar [25]. Inception-V3 menggunakan modul *inception* untuk mengekstrak fitur dan memahami konteks pada gambar. Modul ini terbentuk dari beberapa operasi konvolusi sehingga lapisan yang digunakan identik dengan CNN. Untuk

mengatur jumlah parameter, Inception-V3 menggunakan metode faktorisasi. Pada umumnya, Inception-V3 telah dilatih menggunakan *dataset* ImageNet yang besar sehingga bisa digunakan untuk klasifikasi gambar pada *dataset* khusus. Oleh karena itu, Inception-V3 termasuk dalam metode *transfer learning*, terutama untuk klasifikasi gambar. Struktur Inception-V3 terdapat pada Gambar 2.9.



Gambar 2.9 Struktur Inception-V3 [25]

2.7 Evaluasi Model

2.7.1. *Confusion matrix*

Confusion matrix digunakan untuk mengklasifikasikan jumlah data *train* yang benar dan salah [26]. Pada umumnya, *confusion matrix* berbentuk tabel berukuran 2x2 dengan 2 kelas, yaitu positif dan negatif. Dari *confusion matrix*, bisa diperoleh prediksi yang benar dari model yang dihasilkan dari proses *training*. Struktur *confusion matrix* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Struktur *Confusion matrix* [27]

Class	***Classified as Positive***	***Classified as Negative***
Positive	*True Positive (TP)*	*False Negative (FN)*
Negative	*False Positive (FP)*	*True Negative (TN)*

True positive merepresentasikan jumlah prediksi benar yang bernilai positif, *true negative* merepresentasikan jumlah prediksi benar yang bernilai negatif, *false positive* merepresentasikan jumlah prediksi salah yang bernilai positif, dan *false negative* merepresentasikan jumlah prediksi salah yang

bernilai negatif. Asumsi positif dan negatif ditentukan oleh konteks masalah klasifikasi tertentu pada *dataset*.

2.7.2. *ROC curve*

Receiver Operating Characteristic (ROC) merupakan kurva yang digunakan untuk melihat nilai prediksi suatu model pada suatu penelitian. Visualisasi, pengaturan, dan performa algoritma dapat dilihat menggunakan metode ROC [28]. Hubungan antara *false positive rate* dan *true positive rate* akan ditonjolkan oleh kurva ROC ini. Tujuan akhir dari *ROC curve* adalah mengevaluasi seberapa baik sebuah model dalam membedakan kelas positif dan kelas negatif. *ROC curve* akan memberikan informasi rinci mengenai kemampuan model mengenali *true positive* dari total prediksi positif (*precision*), mengenali *true positive* dari total kasus positif sebenarnya (*recall*), menampilkan rata-rata *precision* dan *recall* (*F1-Score*), dan jumlah kasus (*support*). Seluruh informasi tersebut, kecuali *support*, bisa direpresentasikan menggunakan rumus matriks evaluasi pada Tabel 2.2.

Tabel 2.2 Rumus Matriks Evaluasi *ROC curve* [26]

Accuracy	=	$\frac{TP + TN}{(TP + TN + FP + FN)}$
Precision	=	$\frac{TP}{(TP + FP)}$
Recall	=	$\frac{TP}{(TP + FN)}$
F1-Score	=	$2 * \frac{(Precision * Recall)}{(Precision + Recall)}$

2.8 *Tools* yang digunakan

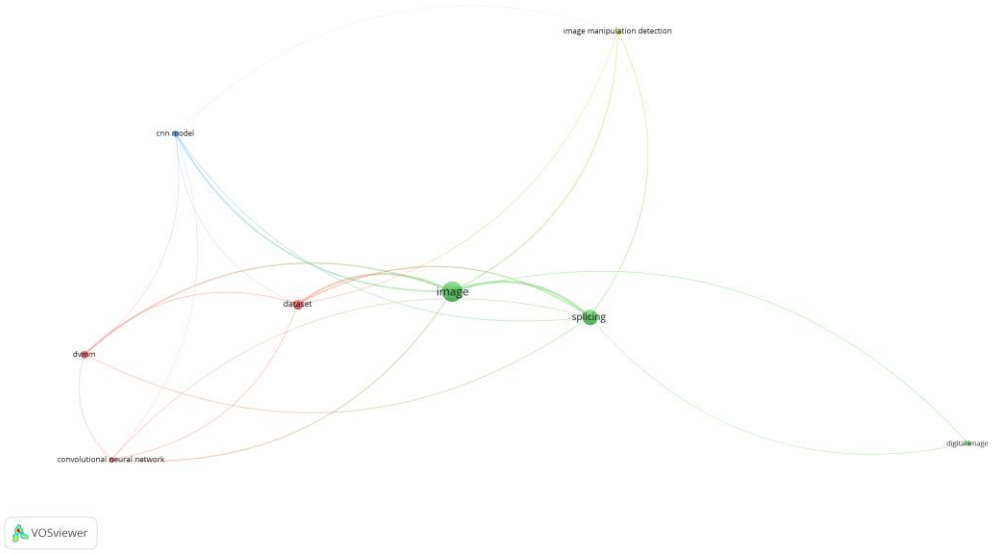
Penelitian ini menggunakan python sebagai bahasa pemrograman untuk membuat model, mulai dari *pre-processing* sampai evaluasi model. Python merupakan bahasa pemrograman yang berfokus pada prinsip perancangan untuk membaca kode [29]. Sifat python yang *open source* membuatnya populer digunakan oleh para *programmer* dalam pengembangan model dan *software*.

Beberapa *library* yang terkenal dan mudah untuk di-*install* adalah NumPy, Pandas, Matplotlib, dan TensorFlow. Pada penelitian ini, Python dijalankan pada Jupyter Notebook yang difasilitasi oleh miniconda.

Jupyter notebook memungkinkan penggunanya untuk membuat dokumen berbasis web yang berisi kode, teks naratif, hasil visualisasi, dan gambar. Struktur cell yang digunakan pada jupyter notebook memungkinkan kode yang dibuat dieksekusi satu per satu. Terdapat juga fitur markdown untuk membuat teks naratif yang biasanya digunakan untuk menjelaskan kode. Selain python, jupyter notebook juga mendukung R, Julia, dan banyak bahasa pemrograman lainnya.

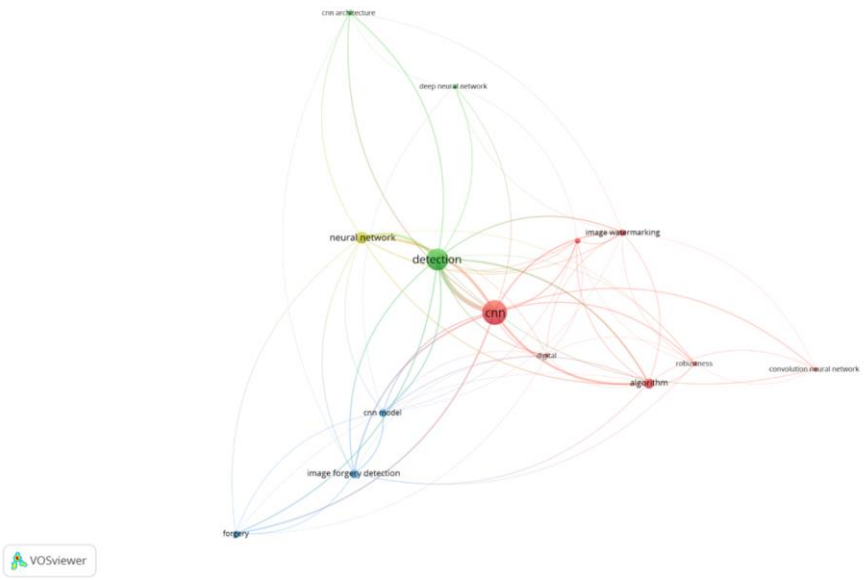
2.9 Pemetaan Penelitian

Pemetaan penelitian adalah serangkaian proses untuk mendapatkan rincian, tahapan penelitian, dan evaluasi kerangka pengetahuan pada sebuah bidang penelitian tertentu. Tujuan dilakukan pemetaan penelitian adalah memperoleh berbagai pandangan secara menyeluruh terkait penelitian terdahulu, identifikasi tren, ataupun mendapatkan celah yang mengarahkan ke penelitian di masa depan. Pada penelitian ini, pemetaaan akan dilakukan pada dua bidang, yaitu deteksi *image forgery splicing* pada *dataset* DVMM menggunakan CNN dan deteksi *watermark* sebagai bentuk deteksi pemalsuan gambar secara aktif. Untuk mempermudah proses pemetaan penelitian, digunakan Publish or Perish untuk memperoleh studi terdahulu dan bantuan VosViewer untuk membentuk visualisasi pemetaan penelitian. Hasil pemetaan penelitian terdapat pada Gambar 2.10 dan Gambar 2.11.



Gambar 2.10 Hasil Pemetaan Penelitian Deteksi *Splicing* pada Dataset DVMM

Pada Gambar 2.10 dapat dilihat bahwa penelitian terdahulu banyak yang berfokus pada *image* dan *splicing*. Namun, yang menggunakan model CNN belum terlalu banyak. Oleh karena itu, penelitian ini akan menguji deteksi *splicing* pada dataset DVMM menggunakan CNN.



Gambar 2.11 Hasil Pemetaan Penelitian Deteksi *Watermark* Menggunakan CNN

Pada Gambar 2.11 dapat dilihat bahwa penelitian terdahulu banyak yang berfokus pada *detection* dan CNN. Namun, yang mendeteksi *watermark* untuk *active image forgery detection* masih sangat sedikit. Oleh karena itu, penelitian ini akan menggunakan model CNN untuk mendeteksi *watermark*.

2.10 Penelitian Terdahulu

Berdasarkan pemetaan penelitian menggunakan Publish or Perish dan VosViewer, ditemukan beberapa penelitian terdahulu yang dijadikan acuan pada penelitian ini. Penelitian-penelitian tersebut adalah sebagai berikut:

Tabel 2.3 Penelitian Terdahulu

No	Penulis	Judul	Jurnal	Algoritma	*Dataset*	Akurasi	Kelebihan	*Future Scope*
Splicing Detection* DVMM *Dataset								
1.	Zhang, Q., Lu, W., Wang, R., & Li, G. (2020) [14].	*Digital image splicing detection based on Markov features in QDCT and QWT domain*	*Multimedia Tools and Applications*	*Support* Vector Machine (SVM)	DVMM	89,88% (Ditambah DWT)	Perbandingan penelitian dengan yang terdahulu dijelaskan secara rinci dan lengkap.	Jumlah *dataset* sedikit.
2.	Moghaddasi, Z., Jalab, H. A., & Noor, R. M. (2019) [13].	*Image splicing forgery detection based on low-dimensional singular value decomposition of discrete cosine transform coefficients*	*Neural Computing and Applications*	Singular Value Decomposition (SVD)	DVMM	64,13% (*Base*) 97,15% (Kernel PCA)	Banyak *dataset* yang diuji menggunakan algoritma dan optimasi yang sama.	Model belum mampu mendeteksi letak *splicing* dan belum mampu mendeteksi *copy-move* dan *retouching*.

3.	Kaur, N., Jindal, N., & Singh, K. (2020) [30].	*A passive approach for the detection of splicing forgery in digital images.*	*Multimedia Tools and Applications*	SVM	DVMM	97,8% (Ditambah DWT)	Banyak *dataset* yang diuji dengan algoritma yang sama.	*Dataset* bisa diganti menjadi gambar medis untuk kebutuhan masyarakat luas.
Digital *Watermark*ing *Detection*								
4.	Ma, N., Zhao, X., & Bolin, M. (2019) [31].	*An End-to-End Solution for Effectively Demoting Watermarked Images in Image Search*	*arXiv preprint arXiv:1901.09473.*	Resnet-50	*Web search*	84,85%	Model *transfer learning* cukup banyak, akurasi cukup baik.	Sumber *dataset* yang digunakan bisa dijelaskan secara spesifik.
5.	Sushma, S., Sindhu, C. S., Sowmya, K. L., Kumar, N. N. R., & Surya, T. *Watermark Detection* Using Deep Learning. (2022) [6].	*Watermark Detection Using Deep Learning*	*International Journal for Advanced Research in Science & Technology*	*You Only Look Once* (YOLO)	*Personal dataset*.	93%	Hasil akurasi sangat baik.	Menambahkan *confusion matrix* untuk validasi dan *testing*.
6.	Tirumala, S. S., Jamil, N., &	*A Deep Neural Network Approach for Classification of*	*Intelligent Technologies and*	*Deep Neural*	*No watermark no*	77,9% (*Original IW*)	Jenis *watermark* yang diteliti	Akurasi masih bisa

	Malik, M. A. (2019) [5].	*Watermarked and Non-watermarked Images.*	*Applications: First International Conference*	*Network* (DNN)	*degraded* (NWND)		tidak hanya *image*.	ditingkatkan lagi.
7.	Tayyab, M., Marjani, M., Jhanjhi, N. Z., & Hashem, I. A. T. (2021) [4].	*A Light-weight Watermarking-Based Framework on Dataset Using Deep Learning Algorithms.*	*2021 National Computing Colleges Conference (NCCC)*	*Convolutional Neural Network (CNN)* dan *Artificial Neural Network (ANN)*	CIFAR-10 dan CIFAR-100	Kurang lebih 97-98%	Akurasi *deep learning* sangat bagus dan cukup ringan untuk diimplementasikan pada *dataset* besar.	Memerlukan algoritma penanaman *watermark* pada *dataset* sebelum bisa digunakan.
8.	Alzahrani, A. (2022) [1].	*Detecting Digital Watermarking Image Attacks Using a Convolution Neural Network Approach.*	*Security and Communication Networks 2022*	*Convolutional Neural Network* (CNN)	*Standard Repository from Kaggle*	98%	Akurasi sangat tinggi dan *dataset open source*.	Menambahkan deteksi *watermark* pada *dataset*, tidak hanya penyerangan.
9.	Deeba, F., Kun, S., Dharejo, F. A., Langah, H., & Memon, H. (2020) [32].	*Digital watermarking using deep neural network.*	*International Journal of Machine Learning and Computing*	*Deep Neural Network* (DNN)	*Six well known grey images Lena-image*	54.7% (*training*) dan 51.89% (*testing*)	Model mampu menanam *watermark* pada *dataset*.	Hasil akurasi masih bisa ditingkatkan.

					dan *standard image-set5*			
10.	Zhang, J., Gu, Z., Jang, J., Wu, H., Stoecklin, M. P., Huang, H., & Molloy, I. (2018) [33].	*Protecting intellectual property of deep neural networks with watermarking*	*Proceedings of the 2018 on Asia conference on computer and communications security*	*Deep Neural Network* (DNN)	EMNIST *dataset*	100% dalam verifikasi kepemilikkan hak kekayaan intelektual.	Akurasi sangat tinggi dan teruji mampu mendeteksi kepemilikkan model DNN.	Menambahkan deteksi *watermark*, tidak hanya implementasi *watermark* untuk melindungi model DNN.

Penelitian mengenai deteksi *image forgery* teknik *splicing* pada *dataset* DVMM bertujuan untuk melihat akurasi yang dihasilkan jika deteksi dilakukan dengan algoritma *deep learning* CNN. Ketiga penelitian sebelumnya menggunakan algoritma *machine learning*, seperti *suppport vector machine* (SVM) [14] [30] dan *singular value decomposition* (SVD) [13]. Selain itu, untuk mencapai akurasi yang lebih baik, beberapa penelitian terdahulu menggunakan teknik *image processing discrete wavelet transform* (DWT) [14] [30] dan menggunakan kernel PCA untuk mengurangi dimensi *dataset* DVMM [13]. Sedangkan penelitian ini hanya ingin menguji akurasi CNN tanpa optimasi untuk mendeteksi *splicing* pada *dataset* DVMM.

Penelitian untuk membuat model deteksi *digital watermark* pada gambar memiliki beberapa kebaruan dibandingkan dengan penelitian terdahulu dimana penelitian ini menggunakan CNN untuk mendeteksi *digital watermark* pada *dataset*. Penelitian terdahulu yang mendeteksi *digital watermark* hanya menggunakan aksitektur CNN pada berbagai metode *transfer learning* [31] dan *You Only Look Once* (YOLO) [6]. Selain itu, penelitian ini berfokus pada deteksi *watermark* pada gambar dimana penelitian sebelumnya memanfaatkan algoritma CNN atau *deep learning* lainnya untuk mendeteksi penyerangan terhadap *watermark* [4] [1] [32]. Penelitian terdahulu juga ada yang fokus pada penggunaan *watermark* untuk melindungi model DNN [33]. Sedangkan penelitian dilakukan menguji apakah algoritma CNN yang sudah dioptimasi dapat mendeteksi *digital watermark* lebih baik dibandingkan dengan *deep* CNN *transfer learning* ataupun algoritma pendeteksi lainnya, seperti ResNet-50 dan Inception-V3 [31].