BAB 2
LANDASAN TEORI

2.1 Nodul Tiroid

Nodul tiroid merupakan benjolan atau massa abnormal yang terbentuk di dalam kelenjar tiroid akibat proliferasi sel yang tidak normal [15]. Kelenjar tiroid sendiri merupakan kelenjar endokrin yang terletak di bagian anterior leher dan berfungsi mensintesis hormon triiodotironin (T3) dan tiroksin (T4) yang berperan dalam regulasi metabolisme, pertumbuhan, serta perkembangan sistem organ tubuh [16]. Nodul Tiroid dapat bersifat tunggal maupun multipel dan sering kali ditemukan secara insidental melalui pemeriksaan ultrasonografi (*USG*) [2]. Sebagian besar nodul tiroid bersifat jinak dan tidak menimbulkan gejala klinis yang signifikan, sehingga evaluasi medis umumnya difokuskan pada identifikasi kemungkinan keganasan secara akurat [17].

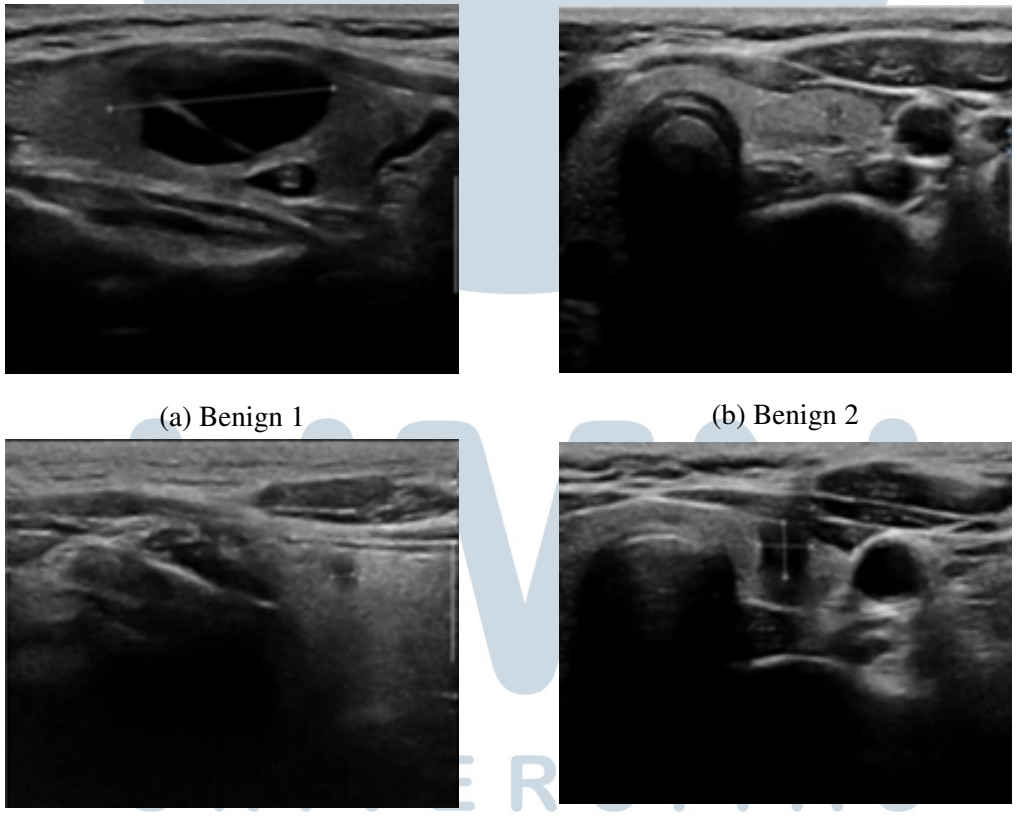
Dampak dari nodul tiroid bervariasi tergantung pada ukuran, jumlah, dan sifat biologisnya. Pada banyak kasus, nodul tidak menimbulkan gejala (*asimptomatik*), namun nodul yang berukuran besar dapat menyebabkan pembengkakan pada leher, kesulitan menelan (*disfagia*), gangguan pernapasan, atau perubahan suara akibat penekanan terhadap struktur di sekitarnya [18]. Meskipun sebagian besar bersifat jinak, sebagian kecil nodul memiliki potensi keganasan sehingga memerlukan evaluasi lebih lanjut untuk mencegah perkembangan menjadi kanker tiroid [19].

2.1.1 Jenis Nodul Tiroid

Nodul tiroid diklasifikasikan berdasarkan sifat biologisnya menjadi nodul jinak (*benign*) dan nodul ganas (*malignant*) [15]. Sebagian besar nodul bersifat jinak dan tidak memerlukan intervensi khusus, namun identifikasi yang akurat terhadap nodul ganas tetap menjadi prioritas klinis utama karena adanya peningkatan insiden global, terutama pada pasien wanita dan individu dengan usia lanjut. Nodul ganas pada kelenjar tiroid merupakan kondisi yang berpotensi berkembang menjadi kanker tiroid dan memerlukan evaluasi lebih lanjut untuk menentukan tingkat keganasan serta penanganan yang tepat [20].

Berdasarkan karakteristik ultrasonografi, nodul tiroid dibedakan menjadi nodul kistik, padat (*solid*), dan campuran. Nodul yang predominan kistik umumnya

memiliki risiko keganasan yang lebih rendah dibandingkan nodul padat, meskipun penilaian tetap harus mempertimbangkan keseluruhan fitur pencitraan [19]. Untuk meningkatkan akurasi diagnosis, digunakan sistem *Thyroid Imaging Reporting and Data System* (*TI-RADS*) yang mengklasifikasikan nodul berdasarkan fitur ultrasonografi seperti komposisi, ekogenisitas, bentuk, margin, dan kalsifikasi. Sistem ini mengelompokkan nodul ke dalam kategori TR1 hingga TR5 yang menunjukkan tingkat risiko keganasan, serta digunakan sebagai dasar dalam menentukan perlunya tindakan lanjutan seperti *Fine Needle Aspiration Biopsy* (*FNAB*), yaitu prosedur pengambilan sampel sel atau cairan dari nodul untuk diperiksa di bawah mikroskop [20]. Contoh citra ultrasonografi nodul tiroid jinak dan ganas dapat dilihat pada Gambar 2.1.



(a) Benign 1 (b) Benign 2

(c) Malignant 1 (d) Malignant 2

Gambar 2.1. Ultrasound Images of Benign and Malignant Thyroid Nodules

Sumber: [9]

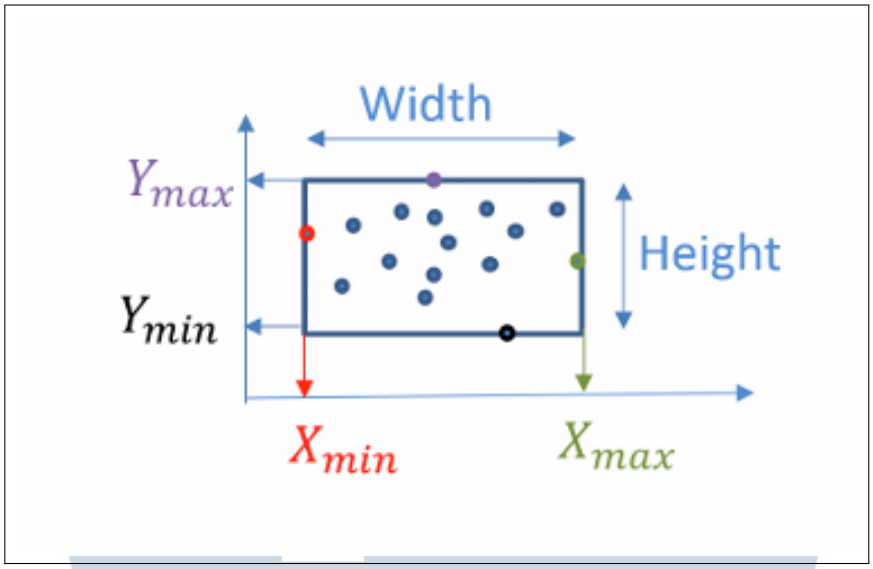
2.2 Object Detection

Object detection merupakan salah satu tugas dalam bidang *Computer Vision* yang bertujuan untuk mendeteksi keberadaan objek dalam suatu citra sekaligus menentukan lokasi objek tersebut [21]. Berbeda dengan klasifikasi citra yang hanya memberikan satu label untuk keseluruhan gambar, *object detection* harus menyelesaikan dua permasalahan secara bersamaan, yaitu pengenalan objek (*object recognition*) dan penentuan lokasi spasial (*spatial localization*). *Object detection* telah dimanfaatkan dalam berbagai aplikasi seperti pengemudian otonom, analisis citra medis, sistem pengawasan, robotika, dan *augmented reality* [22].

Pada awalnya, sistem deteksi objek mengandalkan fitur yang dirancang secara manual (*handcrafted features*) seperti *Histograms of Oriented Gradients* (*HOG*), yang mengekstraksi informasi berdasarkan distribusi arah gradien pada citra, dan *Deformable Part Models* (*DPM*), yang merepresentasikan objek sebagai kumpulan bagian-bagian yang dapat berubah posisi relatif satu sama lain. Metode-metode tersebut umumnya dikombinasikan dengan teknik *sliding window* untuk melokalisasi objek [21]. Pendekatan ini memiliki keterbatasan dalam hal representasi semantik, sensitif terhadap perubahan skala dan pencahayaan, serta kemampuan generalisasi yang rendah pada kondisi lingkungan yang bervariasi. Seiring perkembangan *deep learning*, *Convolutional Neural Network* (*CNN*) kemudian digunakan dalam *object detection* sehingga memungkinkan pembelajaran fitur secara hierarkis langsung dari data, di mana seluruh tahap mulai dari ekstraksi fitur hingga prediksi objek dilatih secara *end-to-end* dalam satu model yang terintegrasi [23].

2.2.1 Konsep Bounding Box

Bounding box merupakan representasi berbentuk persegi panjang yang digunakan untuk menunjukkan lokasi objek dalam citra. Pada tugas deteksi objek, *bounding box* berfungsi sebagai penanda area yang membatasi objek sehingga model dapat mengetahui posisi dan ukuran objek yang terdeteksi. Ilustrasi parameter bounding box dapat dilihat pada Gambar 2.2.



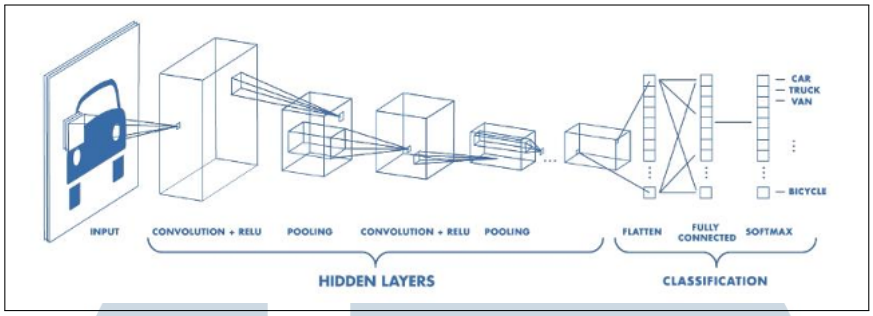
Gambar 2.2. Parameter koordinat bounding box

Sumber: [24]

Dalam metode *YOLO*, *bounding box* umumnya direpresentasikan menggunakan empat parameter utama, yaitu *x*, *y*, *w*, dan *h*. Parameter *x* menunjukkan koordinat titik tengah *bounding box* pada sumbu horizontal, sedangkan *y* menunjukkan koordinat titik tengah *bounding box* pada sumbu vertikal. Sementara itu, *w* menunjukkan lebar *bounding box* dan *h* menunjukkan tinggi *bounding box*. Keempat nilai tersebut biasanya dinormalisasi terhadap ukuran citra sehingga berada pada rentang 0 hingga 1. Selain itu, setiap *bounding box* juga dilengkapi dengan nilai kepercayaan atau *confidence score* yang menunjukkan tingkat keyakinan model terhadap keberadaan objek pada area tersebut [25].

2.3 Arsitektur Convolutional Neural Network (CNN)

Arsitektur *Convolutional Neural Network* (*CNN*) adalah salah satu pendekatan fundamental dalam *deep learning* yang secara khusus dirancang untuk memproses dan menganalisis data berbentuk citra secara efisien. Cara kerja *CNN* didasarkan pada proses ekstraksi fitur secara bertahap dan hierarkis melalui serangkaian operasi seperti konvolusi, normalisasi, serta transformasi non-linear yang saling berkaitan. Setiap lapisan dalam *CNN* memiliki peran tersendiri dalam membangun representasi fitur yang semakin abstrak dan bermakna dari lapisan awal hingga lapisan akhir [26]. Gambar 2.3 merupakan alur proses dari CNN.

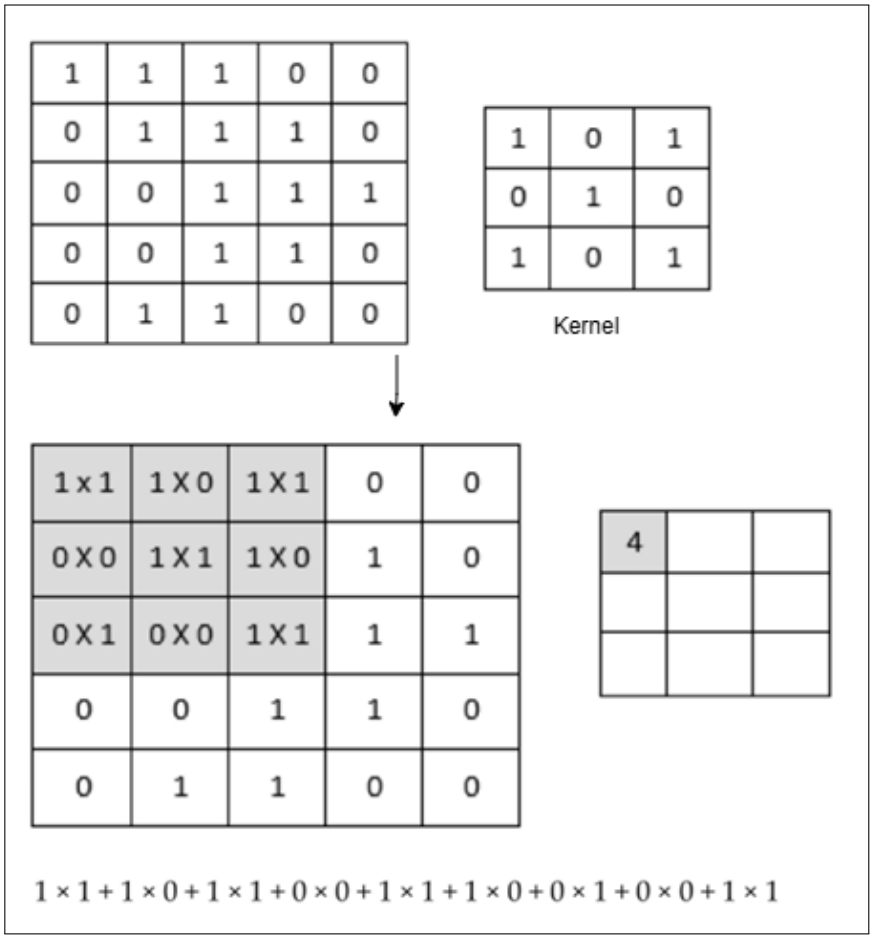


Gambar 2.3. Architecture CNN

Sumber: [27]

2.3.1 Convolutional Layer

Convolutional layer merupakan bagian utama dalam arsitektur *Convolutional Neural Network* (*CNN*) yang berfungsi untuk mengekstraksi fitur dari citra secara otomatis. Lapisan ini bekerja dengan menerapkan sejumlah filter (*kernel*) yang memiliki parameter yang dapat dipelajari (*learnable parameters*) pada citra *input* melalui operasi konvolusi [28]. Proses tersebut dilakukan dengan menggeser *kernel* pada seluruh area citra sehingga menghasilkan keluaran berupa *feature map* yang merepresentasikan pola-pola penting dari citra [27]. *Feature map* ini mampu menangkap berbagai karakteristik visual seperti tepi, tekstur, dan pola tertentu yang relevan untuk proses analisis lebih lanjut [28]. Selain itu, penggunaan beberapa lapisan konvolusi secara bertingkat memungkinkan *CNN* untuk mempelajari representasi fitur secara hierarkis, di mana lapisan awal mengekstraksi fitur sederhana, sedangkan lapisan yang lebih dalam mampu mengenali fitur yang lebih kompleks dan abstrak [29]. Ilustrasi proses pada *convolutional layer* dapat dilihat pada Gambar 2.4.



Gambar 2.4. Convolutional layer

Sumber: [26]

2.3.2 Batch Normalization

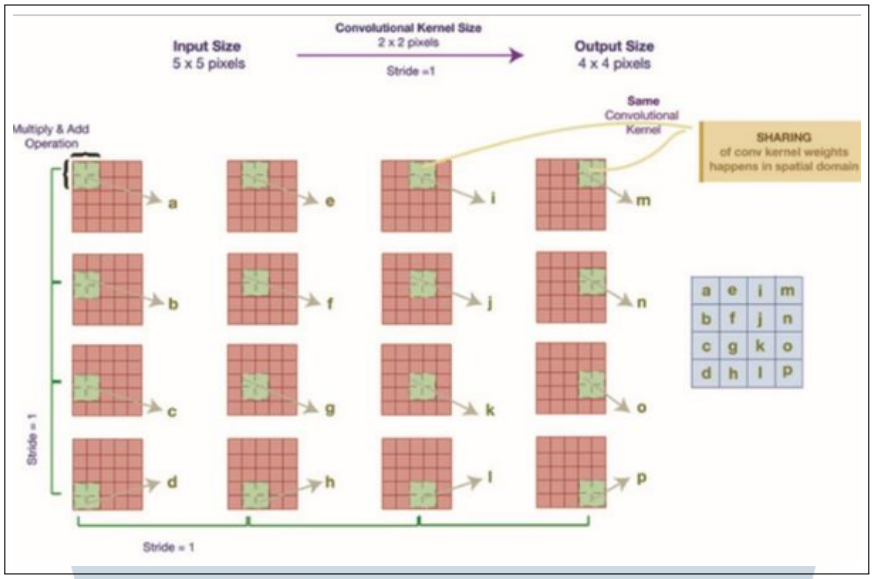
Batch Normalization adalah teknik normalisasi yang diterapkan pada keluaran *convolutional layer* dengan tujuan untuk menstabilkan distribusi data selama proses pelatihan. Teknik ini bekerja dengan menormalisasi nilai aktivasi pada setiap *batch* data menggunakan rata-rata dan variansi yang dihitung secara dinamis. Pada *YOLO26n*, *Batch Normalization* diterapkan setelah setiap operasi konvolusi, sehingga model dapat mempercepat konvergensi, mengurangi risiko *vanishing gradient*, serta memungkinkan pelatihan dengan *learning rate* yang lebih besar tanpa mengorbankan kestabilan [30].

2.3.3 Activation Function

Activation function berfungsi untuk memperkenalkan sifat non-linear ke dalam proses komputasi model, sehingga jaringan mampu mempelajari hubungan kompleks dalam data. Pada *YOLO26n*, fungsi aktivasi yang digunakan adalah *Sigmoid Linear Unit* (*SiLU*), yang memiliki karakteristik kurva lebih halus dan kontinu dibandingkan fungsi *ReLU*. Hal ini memberikan keuntungan berupa gradien yang lebih stabil selama proses pelatihan serta mampu mengurangi kemungkinan terjadinya neuron mati (*dead neuron*) yang sering menjadi masalah pada penggunaan *ReLU* konvensional [31].

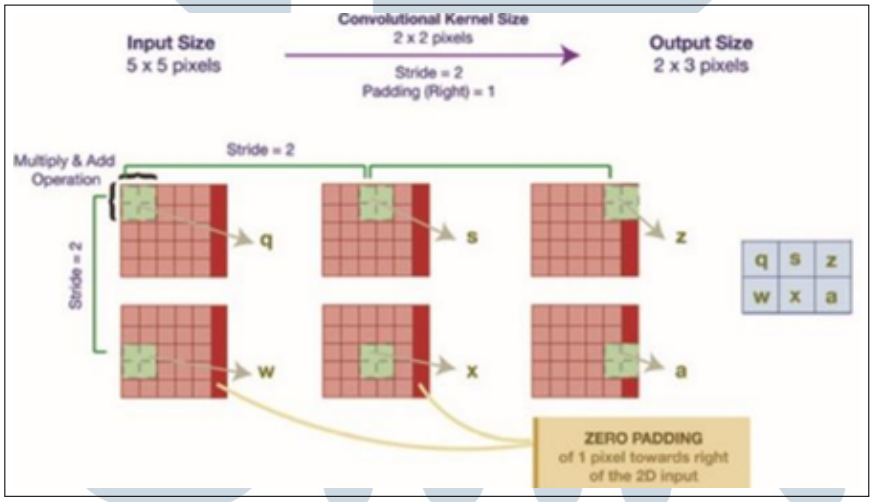
2.3.4 Strided Convolution

Stride adalah langkah pergeseran *kernel* saat melakukan operasi konvolusi pada citra, yang menentukan seberapa jauh *kernel* bergeser setiap langkahnya. *Strided convolution* adalah operasi konvolusi yang menggunakan nilai *stride* lebih dari satu, sehingga *kernel* melompati beberapa piksel setiap pergeseran. Hal ini menyebabkan ukuran *output* menjadi lebih kecil dari *input*nya, sehingga berfungsi sebagai mekanisme pengurangan dimensi spasial (*downsampling*) pada representasi fitur. Berbeda dengan *pooling layer*, *strided convolution* menggabungkan proses konvolusi dan *downsampling* dalam satu operasi sekaligus, sehingga informasi fitur dapat dipertahankan melalui bobot *kernel* yang dapat dipelajari selama pelatihan [32]. Perbedaan operasi *stride* dengan nilai *stride* 1 dan *stride* 2 dapat dilihat pada Gambar 2.5 dan Gambar 2.6.



Gambar 2.5. Stride operation when stride 1

Sumber: [30]



Gambar 2.6. Stride operation when stride 2

Sumber: [30]

2.3.5 Pooling

Pooling merupakan proses dalam CNN yang bertujuan untuk mengurangi dimensi spasial (downsampling) dari feature map dengan cara mengambil nilai representatif dari suatu area tertentu, misalnya nilai maksimum (max pooling) atau rata-rata (average pooling). Proses ini membantu mengurangi kompleksitas komputasi sekaligus mempertahankan informasi penting dari fitur [33].

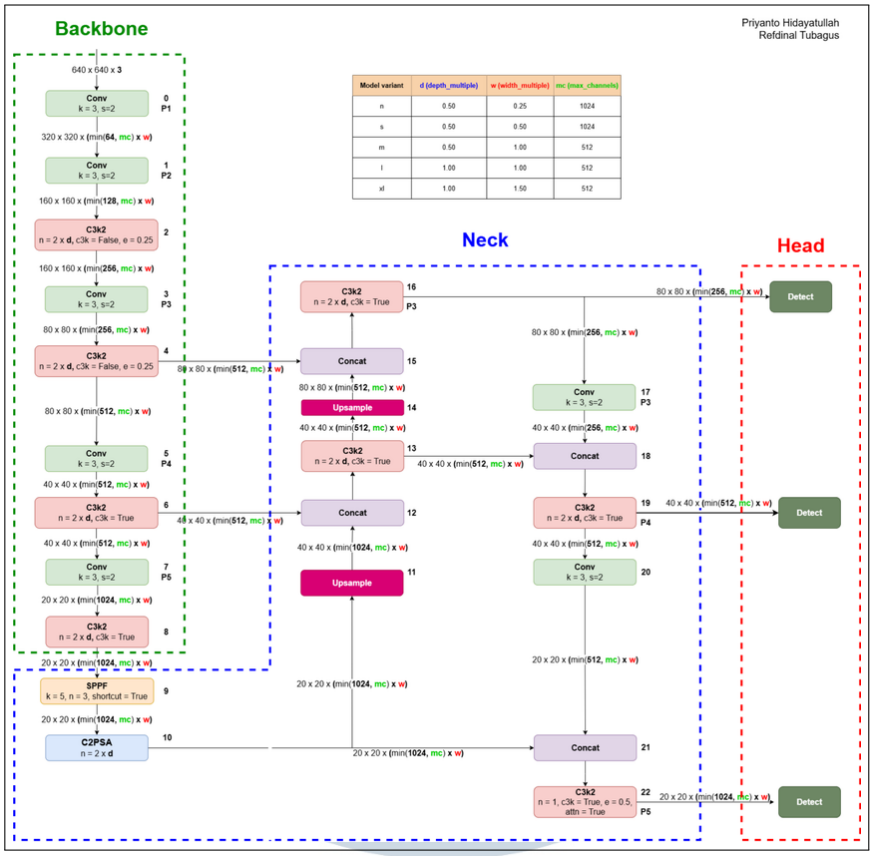
2.3.6 Augmentasi Data

Augmentasi data (*data augmentation*) merupakan teknik untuk meningkatkan keragaman data pelatihan dengan menghasilkan variasi baru dari citra yang sudah ada tanpa mengubah labelnya. Teknik ini membantu meningkatkan kemampuan generalisasi model dan mengurangi risiko *overfitting*, terutama pada dataset medis yang umumnya memiliki jumlah data terbatas [34].

Pada penelitian ini digunakan tiga teknik augmentasi data, yaitu *brightness adjustment*, *horizontal flip*, dan *noise addition*. *Brightness adjustment* digunakan untuk mengubah tingkat kecerahan citra sehingga dapat mensimulasikan variasi intensitas pada citra ultrasonografi yang disebabkan oleh perbedaan pengaturan alat atau kondisi akuisisi citra. *Horizontal flip* dilakukan dengan membalik citra secara horizontal untuk menambah variasi orientasi objek, sehingga model menjadi lebih robust terhadap perubahan posisi nodul pada citra. Sementara itu, *noise addition* dilakukan dengan menambahkan gangguan (*noise*) pada citra untuk mensimulasikan *noise* yang umum ditemukan pada citra ultrasonografi. Dengan demikian, model dapat belajar mengenali objek secara lebih baik meskipun terdapat gangguan pada citra. Kombinasi ketiga teknik tersebut dapat meningkatkan variasi data pelatihan sehingga model mampu mendeteksi nodul tiroid dengan lebih baik pada berbagai kondisi citra ultrasonografi [34].

2.4 YOLO26

YOLO26 merupakan salah satu pengembangan terbaru dalam keluarga *YOLO* (*You Only Look Once*) yang dikembangkan oleh *Ultralytics* untuk kebutuhan deteksi objek secara cepat dan efisien [35]. Model ini bekerja sebagai *single-stage detector*, yaitu proses pengenalan dan pelokasian objek dilakukan secara langsung dalam satu kali inferensi tanpa memerlukan tahap pembuatan kandidat wilayah seperti pada metode *two-stage detector*. Secara umum, arsitektur *YOLO26* terdiri dari tiga komponen utama, yaitu *backbone*, *neck*, dan *head*. Bagian *backbone* berfungsi untuk mengekstraksi fitur visual dari citra, bagian *neck* menggabungkan fitur dari berbagai skala, sedangkan bagian *head* menghasilkan prediksi berupa koordinat *bounding box*, nilai *confidence score*, dan kelas objek [35, 12].



Gambar 2.7. Arsitektur YOLO26

Sumber: [36]

Berdasarkan Gambar 2.7, arsitektur *YOLO26* tersusun atas bagian *backbone*, *neck*, dan *head*. Bagian *backbone* memproses citra masukan secara bertahap untuk menghasilkan fitur dengan ukuran resolusi yang semakin kecil, tetapi memiliki informasi yang semakin kompleks. Selanjutnya, bagian *neck* menggabungkan fitur dari beberapa skala melalui proses seperti *upsampling* dan *concatenation* agar model dapat mengenali objek dengan ukuran yang bervariasi. Bagian *head* kemudian menghasilkan prediksi deteksi pada beberapa skala fitur, sehingga model dapat mendeteksi objek kecil, sedang, maupun besar.

Dalam pengembangannya, *YOLO26* menghadirkan beberapa inovasi arsitektur yang berfokus pada efisiensi, kecepatan, dan kemudahan *deployment*. Model ini menggunakan strategi *direct regression* untuk memprediksi koordinat *bounding box* secara langsung, sehingga proses inferensi menjadi lebih sederhana dan lebih kompatibel dengan perangkat *edge*. Selain itu, *YOLO26* menerapkan inferensi *end-to-end* tanpa *Non-Maximum Suppression* (*NMS*) melalui strategi *one-to-one label assignment*, sehingga dapat mengurangi ketergantungan terhadap

tahap *post-processing* [12]. Untuk mendukung proses pelatihan, *YOLO26* juga memperkenalkan *Progressive Loss Balancing* (*ProgLoss*) dan *Small-Target-Aware Label Assignment* (*STAL*) yang dirancang untuk meningkatkan kemampuan model dalam mempelajari objek berukuran kecil [35]. Dengan karakteristik tersebut, *YOLO26* memiliki potensi untuk digunakan pada berbagai aplikasi deteksi objek secara *real-time*, termasuk pada penelitian ini untuk mendeteksi nodul tiroid pada citra ultrasonografi.

2.4.1 Varian Model YOLO26

YOLO26 tersedia dalam beberapa varian model yang dirancang untuk menyesuaikan kebutuhan antara akurasi deteksi, jumlah parameter, kebutuhan komputasi, dan kecepatan inferensi. Secara umum, varian tersebut terdiri atas *YOLO26n* (*nano*), *YOLO26s* (*small*), *YOLO26m* (*medium*), *YOLO26l* (*large*), dan *YOLO26x* (*extra large*) [36].

Perbedaan utama antar varian terletak pada ukuran jaringan, jumlah parameter, dan kompleksitas komputasi yang digunakan. Semakin besar ukuran model, semakin banyak parameter yang dimiliki sehingga kemampuan ekstraksi fitur dan akurasi deteksi cenderung meningkat. Namun, peningkatan tersebut diikuti oleh kebutuhan memori, waktu pelatihan, dan waktu inferensi yang lebih besar. Sebaliknya, model yang lebih kecil memiliki jumlah parameter yang lebih sedikit sehingga lebih ringan dan cepat dijalankan, tetapi umumnya memiliki kemampuan representasi fitur yang lebih terbatas [36].

Pada penelitian ini digunakan varian *YOLO26n* (*nano*) karena memiliki ukuran model yang paling ringan dibandingkan varian lainnya. Karakteristik tersebut memungkinkan proses pelatihan dan inferensi dilakukan dengan kebutuhan sumber daya komputasi yang lebih rendah, sekaligus tetap mempertahankan performa deteksi yang kompetitif. Oleh karena itu, *YOLO26n* dipilih sebagai model yang sesuai untuk penelitian deteksi nodul tiroid pada citra ultrasonografi [36].

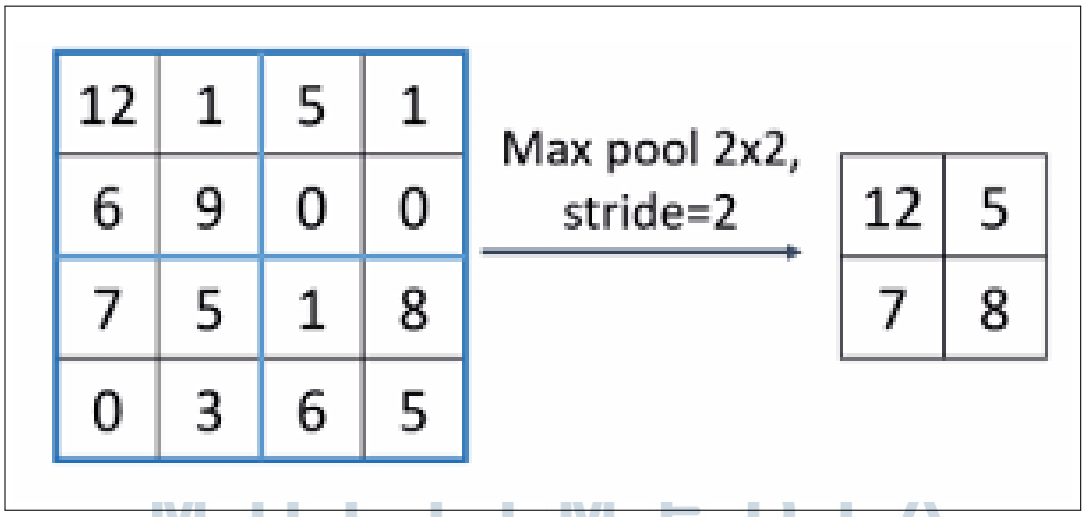
2.4.2 Cross Stage Partial with Position Sensitive Attention (C2PSA)

C2PSA (*Cross Stage Partial with Position-Sensitive Attention*) merupakan salah satu modul penting dalam arsitektur *YOLO26* yang dirancang untuk meningkatkan kualitas representasi fitur dengan menggabungkan operasi konvolusi dan mekanisme perhatian (*attention*) [36]. Modul ini bekerja dengan

membagi fitur masukan ke dalam dua jalur pemrosesan, di mana satu jalur mempertahankan informasi lokal melalui konvolusi, sementara jalur lainnya memanfaatkan mekanisme perhatian untuk menangkap hubungan jarak jauh (*long-range dependencies*) antar fitur. Hasil dari kedua jalur tersebut kemudian digabungkan untuk menghasilkan *feature map* yang lebih kaya dan informatif [37]. Pendekatan ini sejalan dengan pengembangan arsitektur *YOLO26* yang berbasis *Convolutional Neural Network* (*CNN*) dan dirancang dengan berbagai peningkatan untuk meningkatkan efisiensi serta performa deteksi objek. Dengan demikian, *C2PSA* berperan sebagai modul peningkatan (*enhancement module*) yang memungkinkan model untuk menggabungkan informasi lokal dan global secara efektif tanpa meningkatkan kompleksitas komputasi secara signifikan [36, 37].

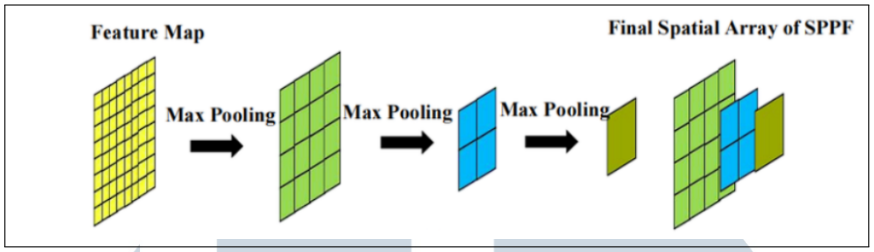
2.4.3 Spatial Pyramid Pooling Fast (SPPF)

Spatial Pyramid Pooling Fast (*SPPF*) merupakan pengembangan dari metode *Spatial Pyramid Pooling* yang digunakan untuk menangkap informasi fitur pada berbagai skala [38]. Teknik ini bekerja dengan menerapkan operasi max *pooling* secara berulang untuk menghasilkan representasi multi-skala dari *feature map* [39]. Contoh proses *max pooling* dapat dilihat pada Gambar 2.8.



Gambar 2.8. Contoh max pooling
Sumber: [33]

Ilustrasi arsitektur *SPPF* dapat dilihat pada Gambar 2.9.



Gambar 2.9. SPPF Image

Sumber: [40]

Pada *SPPF*, operasi *max pooling* dilakukan beberapa kali secara berurutan dengan ukuran *kernel* tertentu untuk memperluas *receptive field* tanpa meningkatkan jumlah parameter secara signifikan. Dengan adanya *SPPF*, model dapat memahami konteks global dan mendeteksi objek dengan ukuran yang bervariasi secara lebih efektif [39].

2.4.4 DFL-Free Regression

DFL-Free Regression merupakan pendekatan regresi *bounding box* yang digunakan pada YOLO26 untuk memprediksi lokasi objek secara langsung tanpa memanfaatkan *Distribution Focal Loss* (DFL). Pada metode berbasis DFL, koordinat *bounding box* direpresentasikan sebagai distribusi probabilitas diskret yang kemudian dikonversi menjadi nilai koordinat akhir. Meskipun pendekatan tersebut mampu meningkatkan akurasi lokalisasi objek, proses perhitungan distribusi probabilitas menambah kompleksitas komputasi dan kebutuhan memori selama pelatihan maupun inferensi [41].

Untuk mengatasi keterbatasan tersebut, YOLO26 mengadopsi pendekatan *DFL-Free Regression* yang memprediksi koordinat *bounding box* secara langsung dari *feature map* yang dihasilkan jaringan. Pendekatan ini menyederhanakan proses regresi sehingga dapat meningkatkan efisiensi komputasi dan kecepatan inferensi tanpa mengorbankan kemampuan lokalisasi objek secara signifikan. Selain menghasilkan koordinat *bounding box*, model juga memprediksi nilai *confidence score* dan kelas objek untuk menentukan keberadaan serta kategori objek yang terdeteksi[41].

2.4.5 Komponen Loss dan Optimasi

YOLO26 menggunakan komponen loss dan optimasi untuk meningkatkan efektivitas proses pelatihan model. Komponen yang digunakan meliputi *Scale-Tuned Adaptive Loss* (STAL) dan *Progressive Loss* (ProgLoss).

A Scale-Tuned Adaptive Loss (STAL)

Scale-Tuned Adaptive Loss (STAL) merupakan fungsi loss yang dirancang untuk menyesuaikan proses pembelajaran berdasarkan ukuran objek. Pendekatan ini membantu model memberikan perhatian yang lebih seimbang terhadap objek dengan skala yang berbeda sehingga dapat meningkatkan akurasi deteksi dan lokalisasi objek [12].

B Progressive Loss (ProgLoss)

Progressive Loss (ProgLoss) merupakan strategi optimasi yang mengatur kontribusi fungsi *loss* secara bertahap selama proses pelatihan. Pada tahap awal, model difokuskan untuk mempelajari pola dasar yang lebih mudah, kemudian secara bertahap meningkatkan perhatian terhadap prediksi yang lebih kompleks. Pendekatan ini membantu meningkatkan stabilitas pelatihan, mempercepat konvergensi, dan menghasilkan performa deteksi yang lebih baik [12].

2.4.6 Non-Maximum Suppression (NMS)

Non-Maximum Suppression (NMS) merupakan teknik pasca-pemrosesan yang digunakan untuk menghilangkan prediksi *bounding box* yang saling tumpang tindih pada objek yang sama. Metode ini mempertahankan *bounding box* dengan nilai *confidence score* tertinggi dan menghapus prediksi lain yang memiliki tingkat tumpang tindih tinggi, sehingga menghasilkan deteksi yang lebih akurat dan tidak redundan [12].

2.5 Evaluasi Model

Evaluasi model merupakan tahap penting dalam pengembangan sistem deteksi objek untuk mengukur sejauh mana model mampu menghasilkan prediksi yang akurat dan andal. Pada model berbasis *YOLO* (*You Only Look Once*), evaluasi

tidak hanya mempertimbangkan ketepatan klasifikasi objek, tetapi juga akurasi dalam menentukan lokasi objek melalui *bounding box*.

Metrik *accuracy* tidak digunakan sebagai indikator utama dalam penelitian ini karena kurang mampu merepresentasikan performa model deteksi objek secara menyeluruh. Hal ini disebabkan oleh ketidakmampuannya dalam mempertimbangkan kesalahan prediksi seperti *false positive* dan *false negative*, serta tidak memperhitungkan ketepatan lokasi objek yang diprediksi. Oleh karena itu, digunakan beberapa metrik evaluasi yang lebih sesuai, yaitu *Mean Average Precision* (*mAP*), *precision*, *recall*, *F1-score*, serta *confusion matrix* untuk memberikan gambaran performa model secara lebih komprehensif.

2.5.1 Mean Average Precision (mAP)

Mean Average Precision (*mAP*) merupakan metrik utama dalam evaluasi model deteksi objek yang mengukur rata-rata nilai *average precision* dari seluruh kelas objek. Nilai *average precision* diperoleh dari kurva hubungan antara *precision* dan *recall*, yang menunjukkan keseimbangan antara kemampuan model dalam mendeteksi objek yang benar dan menghindari kesalahan deteksi.

Dalam konteks deteksi objek, perhitungan *mAP* juga mempertimbangkan nilai *Intersection over Union* (*IoU*) untuk menentukan apakah prediksi *bounding box* dianggap benar atau tidak. Semakin tinggi nilai *mAP*, maka semakin baik performa model dalam mendeteksi objek secara akurat [23]. Persamaan perhitungan *mAP* ditunjukkan pada Persamaan 2.1.

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{2.1}$$

2.5.2 Precision

Precision merupakan metrik evaluasi yang digunakan untuk mengukur tingkat ketepatan prediksi positif yang dihasilkan oleh model. Pada deteksi objek, *precision* menunjukkan seberapa banyak objek yang diprediksi sebagai positif benar-benar merupakan objek yang sesuai dibandingkan dengan seluruh prediksi positif yang dihasilkan model. Nilai *precision* yang tinggi menunjukkan bahwa model memiliki sedikit kesalahan prediksi berupa *false positive*, sehingga prediksi

yang dihasilkan lebih akurat dan dapat dipercaya [40]. Persamaan perhitungan *precision* ditunjukkan pada Persamaan 2.2.

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

2.5.3 Recall

Recall merupakan metrik evaluasi yang digunakan untuk mengukur kemampuan model dalam menemukan seluruh objek yang sebenarnya ada dalam data. Pada deteksi objek, *recall* menunjukkan seberapa banyak objek yang berhasil terdeteksi dengan benar dibandingkan dengan seluruh objek positif yang sebenarnya ada. Nilai *recall* yang tinggi menunjukkan bahwa model memiliki tingkat kesalahan *false negative* yang rendah, sehingga sebagian besar objek berhasil dideteksi. Keseimbangan antara *precision* dan *recall* sangat penting untuk menghasilkan model yang optimal dalam sistem deteksi objek [40]. Persamaan perhitungan *recall* ditunjukkan pada Persamaan 2.3.

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

2.5.4 F1-Score

F1-Score merupakan metrik evaluasi yang mengukur keseimbangan antara *precision* dan *recall* melalui rata-rata harmonik kedua metrik tersebut. Nilai *F1-Score* berada pada rentang 0 hingga 1, di mana nilai yang semakin mendekati 1 menunjukkan kinerja model yang semakin baik dalam mendeteksi objek secara akurat dan lengkap [40]. *F1-Score* dihitung menggunakan Persamaan 2.4.

$$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.4}$$

2.5.5 Confusion Matrix

Confusion matrix merupakan tabel yang digunakan untuk mengevaluasi performa model klasifikasi dengan membandingkan antara nilai prediksi dan nilai sebenarnya. Tabel ini terdiri dari empat komponen utama, yaitu *true positive* (*TP*), *true negative* (*TN*), *false positive* (*FP*), dan *false negative* (*FN*). Bentuk umum *confusion matrix* ditunjukkan pada Tabel 2.1.

Tabel 2.1. Confusion Matrix

	Predicted Positive	**Predicted Negative**
Actual Positive	TP	FN
Actual Negative	FP	TN

Melalui *confusion matrix*, dapat dianalisis jenis kesalahan yang dilakukan oleh model, seperti kesalahan dalam mendeteksi objek yang tidak ada atau kegagalan dalam mendeteksi objek yang sebenarnya ada. Informasi ini sangat berguna untuk memahami kelemahan model dan melakukan perbaikan [28].