Pengenalan Jenis Masakan Melalui Gambar Mengunakan YOLO

Alexander William Sutjiadi, Kartika Gunadi, Leo Willyanto Santoso
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
JL. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) - 8417658
E-Mail: alex.sutjiadi@gmail.com, kgunadi@petra.ac.id, leow@petra.ac.id

ABSTRAK

Kebanyakan sistem *input* pada aplikasi informasi kalori masakan masih menggunakan *input* secara manual, Dimana *user* perlu menuliskan nama makanannya. Hal ini dapat menyebabkan kesulitan untuk *user* apabila *user* tidak mengetahui nama masakan yang ingin diketahui informasi kalorinya. Tetapi dengan berkembangnya teknologi terutama dibidang *object detection*, hal ini dapat dipermudah. Jenis masakan dapat dikenali secara otomatis melalui gambar, sehingga mempermudah *user* dan juga mempercepat proses *input*-nya.

Penelitian ini menggunakan metode *You Only Look Once* (YOLO) untuk mengenali jenis masakan dari gambar yang di-*inputkan*. Jenis model YOLO yang digunakan adalah model YOLOv3 yang telah dimodifikasi *convolution layer*-nya dengan model *Convolution Neural Network* (CNN) Xception yang memiliki akurasi dan kecepatan deteksi tinggi.

Hasil penelitian yang diperoleh adalah model YOLO yang telah dimodifikasi memiliki akurasi lebih tinggi dan kecepatan deteksi yang lebih cepat dibandingkan model YOLOv3 standar. Hasil mAP tertinggi yang dicapai adalah 85.13% dengan rata-rata waktu deteksi 0.088742 detik.

Kata Kunci: *Convolutional Neural Network, You Only Look Once, Xception, object detection*, masakan.

ABSTRACT

Most input systems in food calorie information applications still use manual input, where the user needs to write the name of the food. This can cause difficulties for the user who does not know the name of the food. But with the development of technology in object detection, this can be an easier and faster process by using images to recognize the type of food.

This research uses You Only Look Once method to recognize the type of food from the input images. The type of YOLO model used is the YOLOv3 model which has a modified convolution layer with Xception model that has high accuracy and detection speed.

The result obtained is that the modified YOLO model has higher accuracy and faster detection speed than the standard YOLOv3 model. The Highest mAP result achieved was 85.13% with 0.088742 seconds average detection time.

Keywords: *Convolutional Neural Network, You Only Look Once, Xception, object detection, food*

1. PENDAHULUAN

Pada zaman sekarang, teknologi *image classification* dan *object detection* sedang berkembang dengan pesat. Teknologi ini pun dimanfaatkan di berbagai aspek kehidupan manusia. Salah satu contoh penerapannya adalah untuk *food recognition*, yaitu untuk membantu manusia dalam mendeteksi dan mengenali jenis masakan. Teknologi ini dapat mempermudah aplikasi informasi gizi masakan dalam mendeteksi keberadaan dan jenis masakan melalui gambar sehingga tidak perlu lagi dimasukan secara manual.

Untuk mengenali jenis masakan biasanya menggunakan metode *image classification*. Metode seperti *convolutional neural network* (CNN) banyak digunakan untuk klasifikasi jenis masakan. CNN banyak digunakan untuk melakukan klasifikasi jenis masakan karena memiliki performa lebih baik dibandingkan metode konvensional yaitu yaitu dengan mengekstrak manual *image feature* seperti warna lalu dimasukan ke *classifier* seperti *support vector machines* (SVM)[9][10][16].

Tetapi, metode ini masih memiliki kekurangan yaitu hanya dapat mengklasifikasikan satu objek dalam satu gambar [14]. Oleh karena itu, mulai digunakannya pendekatan dengan metode *object detection* untuk melakukan pengenalan masakan. Pendekatan dengan *object detection* lebih unggul karena merupakan gabungan dari *localization dan classification*. Dengan pendekatan ini juga memungkinkan untuk mengenali lebih dari satu objek masakan dalam satu gambar.

Salah satu metode *object detection* yang sering digunakan adalah *You Only Look Once* (YOLO). Metode ini sering dipakai karena memiliki keunggulan kecepatan deteksi yang lebih cepat dibanding metode lain yang seperti *Faster* RCNN dan SSD [12]. Dalam pengenalan masakan, metode ini terbukti dapat mengenali beberapa masakan sekaligus dalam satu gambar, tetapi hasil akurasinya masih jauh lebih rendah dari metode yang lebih lambat seperti *sliding window* dengan *selective search* [1][11].

Untuk meningkatkan akurasi dari YOLO, dapat dilakukan modifikasi pada model CNN atau pada *Convolutional Layer* pada arsitektur YOLO. Seperti terdapat peningkatan akurasi dari versi awal ke versi berikutnya dimana pada versi pertama menggunakan sebuah custom network dengan 24 *convolutional layer* dan menggunakan Darknet-19 dengan 19 *convolutional layer* dengan 5 *maxpooling layer* pada versi kedua [13].

Tujuan dari penelitian ini adalah membuat program yang dapat mendeteksi dan mengenali jenis masakan melalui gambar menggunakan arsitektur YOLO yang dimodifikasi agar performanya lebih optimal.

2. DASAR TEORI

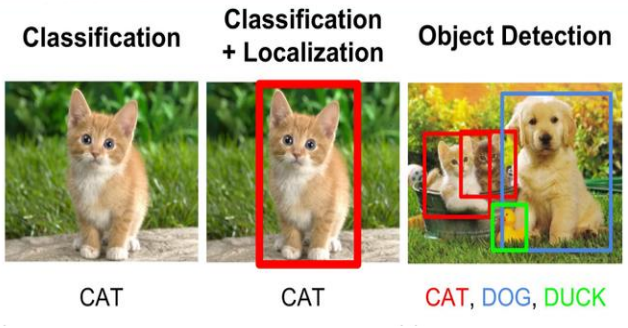
2.1 Jenis Masakan dan Kandungan Gizi

Menurut Kamus Besar Bahasa Indonesia Daring (KBBI Daring), masakan adalah hasil dari memasak atau lauk-pauk yang dimasak [2]. Dengan kata lain masakan adalah sebuah makanan atau bahan-bahan makanan yang telah diproses (dimasak). Cara memasak yang berbeda dapat mengubah kandungan gizi dari bahan makanan itu sendiri [3][6]. Oleh karena itu, tiap jenis masakan dapat memiliki kandungan gizi yang berbeda meskipun memiliki bahan makanan yang sama.

Kandungan gizi dalam tiap masakan inilah yang penting untuk diperhatikan untuk menciptakan pola makan yang sehat. Contohnya, menghindari konsumsi masakan dengan tinggi kandungan gula dan garam tinggi juga dapat membantu pola makan yang sehat [18][17] sehingga terhindar dari obesitas serta penyakit-penyakit lainnya seperti diabetes, jantung, stroke, dan kanker [19].

2.2 Object Detection

Object detection adalah sebuah teknik *computer vision* yang merupakan merupakan penyelesaian dari 2 masalah *computer vision* sekaligus yaitu *image classification* dan *localization* [15]. Untuk lebih jelasnya dapat dilihat pada Gambar 1.



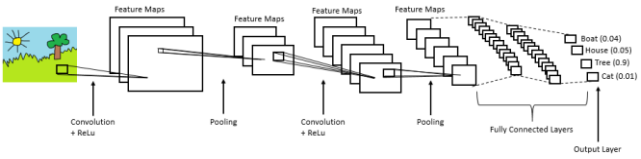
Gambar 1. *Classification, localization,* dan *Object detection*

2.3 UECFOOD-256

UECFOOD-256 adalah sebuah dataset dari University of Electro-Communications, Jepang yang berisi 256 jenis masakan. Tiap gambar memiliki informasi *bounding box* yang mengindikasikan lokasi dari masakan di gambar tersebut. Jumlah gambar tiap kategori masakan terdiri dari sekitar 100 gambar. Objek masakan pada dataset ini juga beragam, ada yang terdiri dari *single object* maupun *multiple object* dalam satu gambar masakan [8].

2.4 Convolutional Neural Network

Secara umum, setiap model CNN memiliki 3 komponen utama, yaitu *convolutional layer*, *pooling layer*, dan *fully connected layer* [5]. *Convolutional layer* adalah *layer* pertama yang berfungsi untuk mengekstrak fitur dari *input* gambar. Pada *layer* ini tiap *pixel* input akan dilakukan operasi konvolusi dengan sebuah matriks berukuran NxN yang biasanya disebut *filter*. Pada *pooling layer* akan dilakukan pengurangan dimensi *matrix* (*downsampling)* hasil dari operasi sebelumnya. *Layer* yang terakhir adalah *fully connected layer*. Pada *layer* ini *matrix* (*feature map*) hasil proses-proses sebelumnya akan diratakan (*flattened*) menjadi sebuah *vector* dan dimasukan kedalam sebuah *fully connected layer* seperti sebuah *neural network* biasa. Tujuannya untuk mengkombinasikan seluruh fitur yang telah diperoleh agar dapat menentukan *class* dari *input* (klasifikasi).

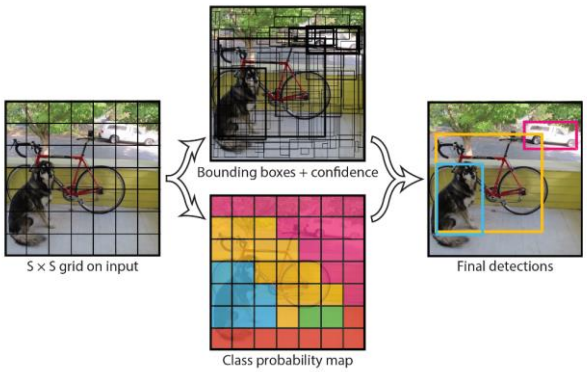


Gambar 2. Arsitektur *Convolutional Neural Network*

2.5 You Only Look Once

You Only Look Once (YOLO) adalah salah satu teknik yang digunakan untuk melakukan pendeteksian objek (*object detection*). Berbeda dengan teknik *object detection* lain seperti *sliding window* dan *region proposal-based*, YOLO menjadikan *object detection* sebagai *single regression problem* [12].

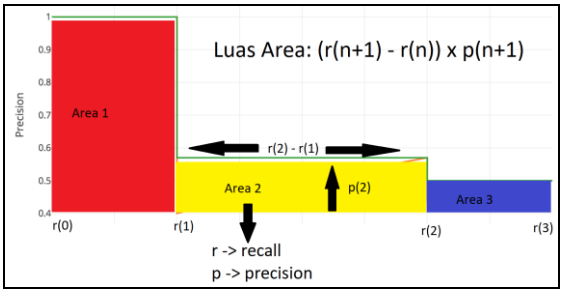
Secara umum, YOLO akan membagi input gambar menjadi sebuah *grid* berukuran SxS. Lalu dari setiap *grid cell* tersebut, akan menghasilkan 1 set prediksi dengan 2 komponen utama, yaitu koordinat *bounding box* dan *class score*. Setelah itu tiap prediksi akan dilakukan proses *non-maximum suppression* untuk mengeliminasi prediksi-prediksi yang memiliki *confident score* rendah disekitar prediksi yang memiliki *confident score* tinggi. Algoritma YOLO ditunjukan pada Gambar 3.



Gambar 3. Algoritma proses deteksi pada YOLO

2.6 *Mean Average Precission*

Mean average precision (mAP) merupakan sebuah *metrics* yang biasa digunakan untuk mengukur performa dari *object detection systems* [7]. Untuk menghitung mAP, dapat dilakukan dengan menghitung nilai rata-rata dari luas area dibawah kurva *precission-recall* tiap *class* dari dataset. Ilustrasi cara menghitung luas area dibawah kurva ditunjukan pada Gambar 4.

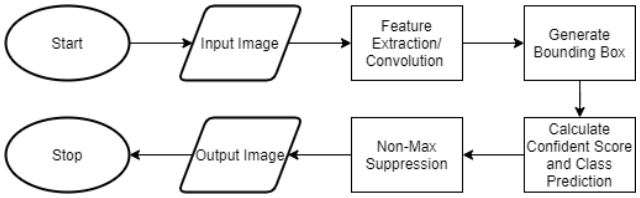


Gambar 4. Perhitungan luas area dibawah kurva

3. DESAIN SISTEM

3.1 Analisis Sistem

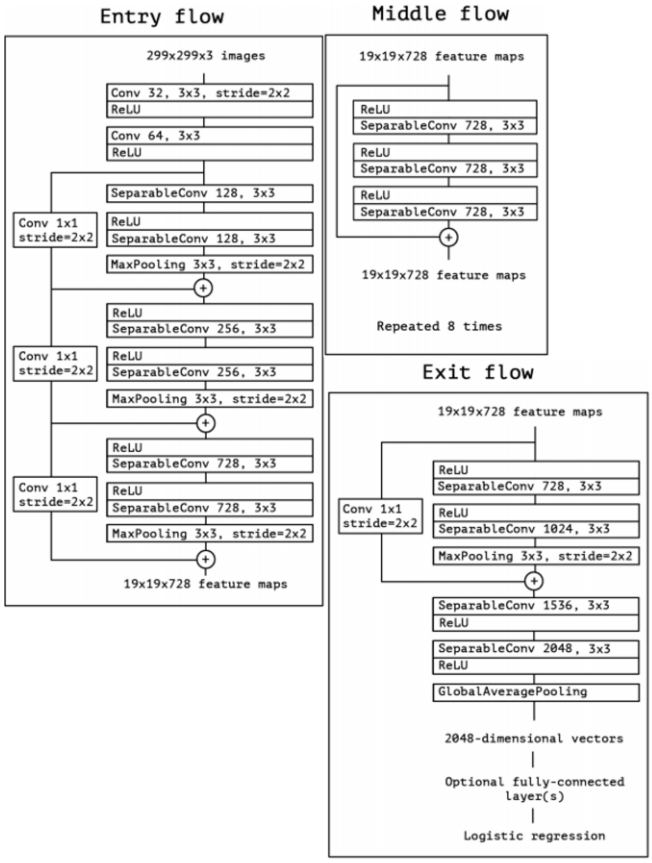
Secara garis besar, sistem pendeteksian akan dimulai dengan menerima *input* gambar. Lalu gambar tersebut akan dilakukan *feature extraction* dengan *network* yang digunakan. Setelah itu, tiap *feature* hasilnya akan digunakan untuk melakukan prediksi *bounding box* dan prediksi probabilitas *class* yang terdapat pada prediksi tersebut. *Output* yang akan dihasilkan adalah sebuah gambar dengan *bounding box* serta label *class* hasil prediksi. Alur tersebut ditunjukan pada Gambar 5.



Gambar 5. Garis besar alur pendeteksian

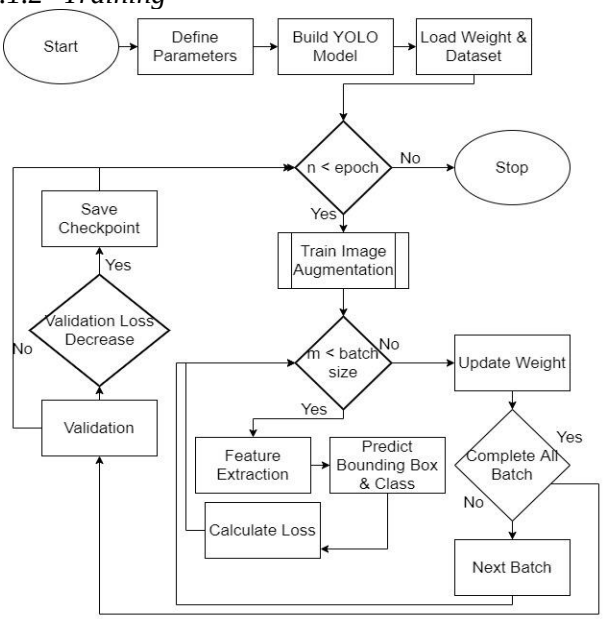
3.1.1 Modifikasi YOLO

Arsitektur YOLO akan dimodifikasi dengan menggunakan jaringan *Xception* menggantikan jaringan *Darknet*-53 sebagai *feature extractor* pada model YOLO. Jaringan ini memiliki performa yang lebih tinggi dibandingkan dengan jaringan seperti VGG-16, Resnet-152, serta Inception V3 dimana mencapai akurasi 79% untuk *top-1 accuracy* dan 94.5% untuk *top-5 accuracy* pada ImageNet dataset [4]. Detail arsitektur *Xception* ditunjukan pada Gambar 6.



Gambar 6. Arsitektur *Xception*

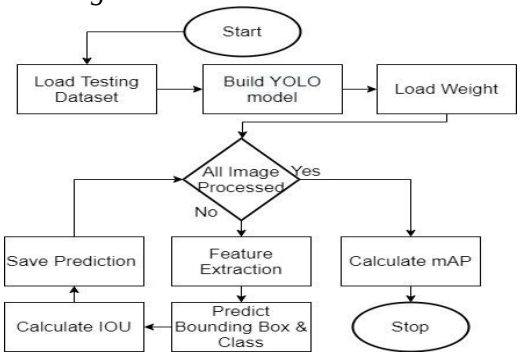
3.1.2 Training



Gambar 7. Alur *training*

Pada Gambar 7 menunjukan desain dari alur proses *training* model YOLOv3 standar dan juga YOLO-Modifikasi.

3.1.3 Testing



Gambar 8. Alur *testing*

Pada Gambar 8 menunjukan desain proses *testing* yang diawali dengan memuat *testing dataset* yang digunakan hingga menghitung mAP dari hasil prediksinya.

3.2 Desain Sistem

Pada bagian ini akan dijelaskan tentang implementasi model YOLO kedalam sebuah program yang dapat digunakan oleh *user*. Program ini akan berbentuk web dan akan menerima *input* gambar dari *user* dan menghasilkan prediksi jenis masakan beserta kandungan gizinya.

4. IMPLEMENTASI SISTEM

Sistem yang dibuat akan diimplementasikan pada komputer dengan *operating system* Windows 10 dengan spesifikasi GPU *Nvidia Geforce GTX* 960m 4gb VRAM. Bahasa pemrograman yang digunakan adalah *python* 3.6 dan menggunakan *tensorflow* sebagai *framework* untuk membantu membangun model YOLO.

5. ANALISA DAN PENGUJIAN

5.1 Pengujian Sistem

Sistem *training* dan *testing* dari model YOLOv3 standar dan YOLO modifikasi dengan Xception akan dilakukan pengujian untuk menemukan konfigurasi parameter yang terbaik. Pada tiap pengujian parameter akan terus diupdate dengan yang terbaik hasil pengujian sebelumnya. Data yang digunakan adalah 30 *class* masakan yang diambil dari dataset UECFOOD-256.

5.1.1 Pengujian Training

Model YOLOv3 Standar dan YOLO-Xception akan dilakukan *training* dengan berbagai konfigurasi parameter yang berbeda. Parameter yang digunakan yaitu *learning rate*, jumlah *epoch*, *batch size*, dan data augmentasi.

5.1.1.1 Pengujian Learning Rate

Tiap model akan diuji dengan dilakukan *training* dengan berbagai *learning rate* dan dengan ukuran *batch* 16 serta sebanyak 50 *epoch*. Hasilnya ditunjukan pada Tabel 1 dan Tabel 2.

Tabel 1. Pengujian *Learning Rate* YOLOv3

Learning rate	*Avg Train Loss*	*Avg Val Loss*	mAP
5e-2	99.91	8.87	25.14%
1e-2	3.25	4.56	0.00%
5e-3	0.50	0.62	33.78%
1e-3	0.30	0.23	29.00%
5e-4	0.19	1.20	42.80%
1e-4	**0.10**	**0.80**	**58.58%**
5e-5	0.26	0.99	45.13%
1e-5	6.05	5.71	31.93%

Tabel 2. Pengujian *Learning Rate* YOLO-Xception

Learning rate	*Avg Train Loss*	*Avg Val Loss*	mAP
5e-2	4.21	4.72	0.00%
1e-2	1.11	1.95	3.55%
5e-3	0.74	1.35	8.99%
1e-3	0.31	1.06	14.89%
5e-4	0.19	0.97	28.48%
1e-4	**0.21**	**0.90**	**31.12%**
5e-5	0.35	1.26	26.11%
1e-5	6.19	6.58	8.26%

5.1.1.2 Pengujian Jumlah Epoch

Pengujian akan dilakukan dengan *train* kedua model dengan jumlah *epoch* antara 10 hingga 50. *Learning rate* yang digunakan adalah 1e-4 dan *batch size* 16. Hasilnya ditunjukan pada Tabel 3 dan Tabel 4.

Tabel 3. Pengujian Jumlah *Epoch* YOLOv3

Epoch	*Avg Train Loss*	*Avg Val Loss*	mAP
10	12.06	10.89	21.48%
20	2.31	2.75	44.58%
30	0.68	1.35	38.15%
40	0.26	1.08	49.46%
50	0.10	0.80	58.58%
60	0.06	0.78	55.99%
70	0.03	0.79	66.55%
80	**0.02**	**0.82**	**71.31%**
90	0.04	0.91	41.30%
100	0.04	0.89	63.13%

Tabel 4. Pengujian Jumlah *Epoch* YOLO-Xception

Epoch	*Avg Train Loss*	*Avg Val Loss*	mAP
10	10.19	10.76	1.72%
20	2.11	2.52	21.03%
30	1.14	1.62	20.59%
40	0.24	0.94	35.98%
50	0.21	0.90	31.12%
60	0.11	0.93	31.73%
70	0.15	1.06	33.66%
80	0.06	0.94	43.76%
90	**0.04**	**0.90**	**48.33%**
100	0.07	0.89	33.80%

5.1.1.3 Pengujian Ukuran Batch

Pengujian akan dilakukan dengan *train* kedua model YOLO dengan *batch size* antara 2 hingga 64. *Learning rate* dan jumlah *epoch* yang digunakan adalah 1e-4 dan 80 *epoch* untuk YOLOv3 dan 90 *epoch* untuk YOLO-Xception. Hasilnya ditunjukan pada Tabel 5 dan Tabel 6.

Tabel 5. Pengujian *Batch Size* YOLOv3

Batch size	*Avg Train Loss*	*Avg Val Loss*	mAP
1	0.05	1.38	43.53%
2	0.05	1.16	65.29%
4	0.06	1.11	53.58%
8	0.02	0.98	69.67%
16	**0.02**	**0.82**	**71.31%**
32	0.19	2.13	41.70%
64	1.73	2.39	14.84%

Tabel 6. Pengujian *Batch Size* YOLO-Xception

Epoch	*Avg Train Loss*	*Avg Val Loss*	mAP
1	0.09	2.27	24.45%
2	0.08	1.13	45.56%
4	0.06	0.03	43.76%
8	0.10	0.05	44.81%

16	**0.04**	**0.90**	**48.33%**
32	0.14	0.87	36.93%
64	1.10	1.07	14.71%

5.1.1.4 Pengujian Augmentasi

Pengujian dilakukan terhadap model hasil terbaik sebelumnya, dengan melakukan *train* tambahan sebanyak 30 *epoch* menggunakan *training data* yang diaugmentasi dengan *random brightness*, *saturation*, *contrast*, *blur* serta noise. Hasil serta perbandingan dengan model tanpa augmentasi ditunjukan pada Tabel 7.

Tabel 7. Hasil dan Perbandingan Pengujian Augmentasi

Model	***Avg Train Loss***	***Avg Val Loss***	**mAP**
YOLOv3 (Tanpa Augmentasi)	0.02	0.82	71.31%
YOLOv3 (Dengan Augmentasi)	0.13	0.81	66.65%
YOLO-Xception (Tanpa Augmentasi)	0.04	0.90	48.33%
YOLO-Xception (dengan Augmentasi)	0.15	0.71	45.30%

5.1.2 Pengujian Testing

Pengujian ini untuk menemukan konfigurasi *threshold* terbaik agar dapat menghasilkan mAP maksimal. Konfigurasi *threshold* yang akan diuji adalah *confident score threshold*, *IoU threshold*, *non-maximum suppression threshold* serta pengujian terhadap berbagai tingkat *blur*. *Metric* yang akan ditunjukan adalah *true positive* (TP), *false positive* (FP), *false negative* (FN), *mean average precission* (mAP), serta *detection time*.

5.1.2.1 Pengujian Confident Score Threshold

Pengujian akan dilakukan dengan melakukan *testing* menggunakan berbagai tingkat batas *confident score* dari prediksi yang dihasilkan. Hasil pengujiannya ditunjukan pada Tabel 8 dan Tabel 9.

Tabel 8. Pengujian *Confident Score Threshold* YOLOv3

Threshold	**TP**	**FP**	**FN**	***mAP***	***Avg Detection time (detik)***
0.05	**578**	**165**	**46**	**72.17%**	**0.089428**
0.15	548	77	83	74.59%	0.087102
0.25	517	45	117	72.61%	0.092707
0.35	490	21	146	71.18%	0.091619
0.45	454	19	183	65.27%	0.088389
0.55	425	11	214	61.50%	0.089656
0.65	390	10	248	57.21%	0.097160
0.75	330	9	308	49.63%	0.091716
0.85	249	5	390	35.60%	0.090909
0.95	133	2	507	20.70%	0.088517

Tabel 9. Pengujian *Confident Score Threshold* YOLO-Xception

Threshold	**TP**	**FP**	**FN**	***mAP***	***Avg Detection time (detik)***
0.05	**592**	**390**	**36**	**48.67%**	**0.088207**
0.15	567	242	64	55.88%	0.087039
0.25	550	186	81	58.20%	0.088048
0.35	526	154	107	58.40%	0.086083
0.45	506	128	127	60.40%	0.087209
0.55	473	107	159	59.09%	0.086801
0.65	435	80	195	56.80%	0.087226
0.75	391	59	241	52.63%	0.088775
0.85	321	35	314	46.01%	0.088025
0.95	162	10	474	25.25%	0.087823

5.1.2.2 Pengujian IoU Threshold

Pengujian dilakukan dengan menggunakan berbagai tingkat *threshold* untuk tingkat kemiripan *bounding box* prediksi terhadap *ground truth bounding box*-nya. Hasil pengujiannya ditunjukan pada Tabel 10 dan Tabel 11.

Tabel 10. Pengujian *IoU Threshold* YOLOv3

Threshold	**TP**	**FP**	**FN**	***mAP***	***Avg Detection time (detik)***
0.1	**580**	**167**	**43**	**71.63%**	**0.099046**
0.2	579	166	45	71.34%	0.095189
0.3	579	165	46	71.34%	0.121977
0.4	579	165	46	71.34%	0.093104
0.5	578	164	48	71.31%	0.089420
0.6	572	163	55	70.55%	0.088513
0.7	531	161	98	64.46%	0.090967
0.8	399	158	233	41.44%	0.131308
0.9	136	153	501	6.43%	0.093068

Tabel 11. Pengujian *IoU Threshold* YOLO-Xception

Threshold	**TP**	**FP**	**FN**	***mAP***	***Avg Detection time (detik)***
0.1	**601**	**388**	**31**	**49.14%**	**0.094686**
0.2	600	386	34	49.08%	0.094603
0.3	598	386	36	48.90%	0.090079
0.4	596	385	39	48.65%	0.088052
0.5	592	383	45	48.33%	0.116828

0.6	575	383	62	46.76%	0.096059
0.7	480	382	158	38.64%	0.088483
0.8	254	380	386	17.59%	0.096190
0.9	60	380	580	2.59%	0.090558

5.1.2.3 Pengujian Non-Maximum Suppression

Pengujian akan dilakukan dengan berbagai tingkat *threshold* NMS untuk menemukan keseimbangan antara menghilangkan prediksi *duplicate* dengan yang benar *multiple object* dalam satu gambar.

Tabel 12. Pengujian NMS YOLOv3

Threshold	TP	FP	FN	*mAP*	*Avg Detection time (detik)*
0.1	**580**	**101**	**44**	**78.06%**	**0.092622**
0.2	580	103	44	77.91%	0.092266
0.3	580	105	44	77.73%	0.089627
0.4	580	116	43	76.74%	0.090664
0.5	580	167	43	71.63%	0.090009
0.6	583	332	43	55.21%	0.111178
0.7	583	784	43	30.19%	0.089798
0.8	586	1821	43	10.56%	0.089429
0.9	586	3510	43	3.50%	0.096453

Tabel 13. Pengujian NMS YOLO-Xception

Threshold	TP	FP	FN	*mAP*	*Avg Detection time (detik)*
0.1	**598**	**60**	**32**	**85.13%**	**0.088742**
0.2	598	61	32	84.99%	0.086380
0.3	599	78	31	82.08%	0.091575
0.4	600	150	31	72.31%	0.087461
0.5	601	388	31	49.14%	0.088327
0.6	600	927	31	23.69%	0.083396
0.7	601	2002	30	8.53%	0.094290
0.8	601	4352	29	1.95%	0.086616
0.9	601	8518	29	0.58%	0.116777

5.1.3 Pengujian Blur

Pengujian akan dilakukan dengan berbagai ukuran *kernel blur* dari *average blur*. Hasilnya ditunjukan pada Tabel 14 dan Tabel 15.

Tabel 14. Pengujian *Blur* pada Model Tanpa Augmentasi

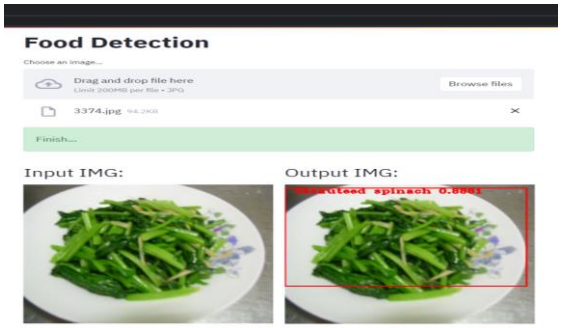
	YOLOv3		YOLO-Xception	
Kernel Size	mAP	*Detection Time* (Detik)	mAP	*Detection Time* (Detik)
3	72.32%	0.088228	79.01%	0.086059
5	62.52%	0.131128	56.89%	0.118427
7	49.21%	0.133568	44.98%	0.116540
9	38.18%	0.089719	32.08%	0.119657
11	32.66%	0.131639	24.53%	0.090610
13	25.05%	0.091152	19.50%	0.090364
15	20.79%	0.089853	15.45%	0.086845

Tabel 15. Pengujian *Blur* pada Model Dengan Augmentasi

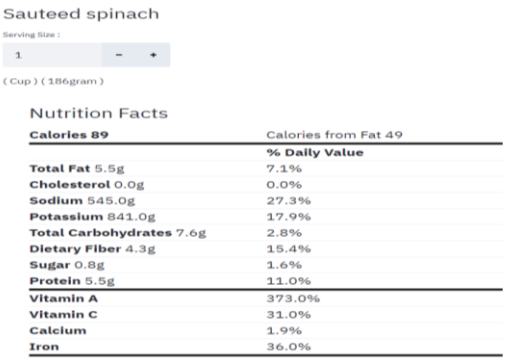
	YOLOv3		YOLO-Xception	
Kernel Size	mAP	*Detection Time* (Detik)	mAP	*Detection Time* (Detik)
3	69.99%	0.089079	77.50%	0.087087
5	68.08%	0.091293	70.14%	0.089729
7	58.48%	0.092652	60.52%	0.088952
9	51.79%	0.090681	51.34%	0.087418
11	45.63%	0.123851	38.58%	0.088001
13	38.88%	0.091752	32.82%	0.089732
15	32.04%	0.094193	26.79%	0.086153

5.2 Pengujian Aplikasi

Aplikasi ini memerlukan waktu sekitar ±2 detik untuk prediksi pertama dan sekitar 0.5 hingga 1 detik untuk prediksi berikutnya. Contoh hasil prediksi yang ditampilkan oleh program ditunjukan pada Gambar 9 dan Gambar 10.



Gambar 9. Tampilan aplikasi saat melakukan prediksi (1)



Gambar 10. Tampilan aplikasi saat melakukan prediksi (2)

6. KESIMPULAN

Berdasarkan hasil pengamatan dan pengujian terhadap model yang telah dibuat, dapat disimpulkan beberapa hal sebagai berikut:

- Pada konfigurasi parameter-parameter training dan testing terbaik, mAP model YOLO modifikasi lebih tinggi dibandingkan YOLO Standar yaitu 85.13% berbanding 78.06% seperti pada Tabel 12 dan Tabel 13
- Model YOLO yang telah dimodifikasi memiliki sensitivitas prediksi lebih tinggi sehingga banyak prediksi *false positive*. Oleh karena itu, berdasarkan hasil pengujian pada Tabel 13, akan lebih optimal jika menggunakan NMS *threshold* yang rendah untuk menyaring prediksi *false positive* tersebut.
- Berdasarkan rata-rata *detection time* pada tiap pengujian, YOLO-Xception lebih cepat 4.37% dibandingkan model YOLO standar.
- Berdasarkan hasil pengujian Tabel 14 dan Tabel 15, proses training model yang ditambahkan proses augmentasi menghasilkan mAP lebih tinggi dan kecepatan deteksi yang lebih cepat dibandingkan model YOLO tanpa proses augmentasi

Saran yang dapat dilakukan untuk mengembangkan model ini lebih lanjut yaitu:

- Menambah jumlah variasi training data tiap jenis masakan agar dapat meningkatkan keakuratan prediksi model.
- Menguji prosedur training dengan augmentasi lainnya agar dapat menghasilkan model YOLO yang memiliki performa maksimal di keadaan normal maupun blur.
- Mencoba model CNN lain agar dapat meningkatkan nilai mAP-nya

7. DAFTAR PUSTAKA

[1] Aguilar, E., Remeseiro, B., Bolaños, M., & Radeva, P. 2018. Grab, Pay, and Eat: Semantic Food Detection for Smart Restaurants. *IEEE Transactions on Multimedia*, 20(12), 3266–3275. https://doi.org/10.1109/TMM.2018.2831627

[2] Badan Pengembangan dan Pembinaan Bahasa. n.d. Hasil Pencarian - KBBI Daring. Retrieved April 5, 2021, from https://kbbi.kemdikbud.go.id/entri/masakan

[3] Brugiapaglia, A., & Destefanis, G. 2015. Effect of cooking method on the nutritional value of Piedmontese beef. August 2012, 6–10.

[4] Chollet, F. 2017. Xception: Deep learning with depthwise separable convolutions. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua, 1800–1807. https://doi.org/10.1109/CVPR.2017.195

[5] CS231n Convolutional Neural Networks for Visual Recognition. n.d.. Retrieved April 5, 2021, from https://cs231n.github.io/convolutional-networks/

[6] Domínguez, R., Borrajo, P., & Lorenzo, J. M. 2015. The effect of cooking methods on nutritional value of foal meat. *Journal of Food Composition and Analysis*, 43, 61–67. https://doi.org/10.1016/j.jfca.2015.04.007

[7] Hui, J. 2018. mAP (mean Average Precision) for Object Detection | by Jonathan Hui. *Medium*. https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173

[8] Kawano, Y., & Yanai, K. 2015. Automatic Expansion of a Food Image Dataset Leveraging Existing Categories with Domain Adaptation. In L. Agapito, M. M. Bronstein, & C. Rother (Eds.), *Computer Vision - ECCV 2014 Workshops* (pp. 3–17). Springer International Publishing. https://doi.org/10.1007/978-3-319-16199-0_1

[9] Liu, C., Cao, Y., Luo, Y., Chen, G., Vokkarane, V., & Ma, Y. (2016). Deepfood: Deep learning-based food image recognition for computer-aided dietary assessment. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9677, 37–48. https://doi.org/10.1007/978-3-319-39601-9_4

[10] Mezgec, S., & Seljak, B. K. 2017. Nutrinet: A deep learning food and drink image recognition system for dietary assessment. *Nutrients*, 9(7), 1–19. https://doi.org/10.3390/nu9070657

[11] Pouladzadeh, P., & Shirmohammadi, S. 2017. Mobile multi-food recognition using deep learning. *ACM Transactions on Multimedia Computing, Communications and Applications*, *13(3s)*, 1–21. https://doi.org/10.1145/3063592

[12] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. 2016. You only look once: Unified, real-time object detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-Decem, 779–788. https://doi.org/10.1109/CVPR.2016.91

[13] Redmon, J., & Farhadi, A. 2017. YOLO9000: Better, faster, stronger. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua, 6517–6525. https://doi.org/10.1109/CVPR.2017.690

[14] Sahoo, D., Hao, W., Ke, S., Xiongwei, W., Le, H., Achananuparp, P., Lim, E. P., & Hoi, S. C. H. 2019. Food AI: Food image recognition via deep learning for smart food logging. *ArXiv*, May.

[15] Szegedy, C., Toshev, A., & Erhan, D. 2013. Deep Neural Networks for Object Detection. *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, 2553–2561.

[16] Teng, J., Zhang, D., Lee, D. J., & Chou, Y. 2019. Recognition of Chinese food using convolutional neural network. *Multimedia Tools and Applications*, *78(9)*, 11155–11172. https://doi.org/10.1007/s11042-018-6695-9

[17] U.S. Department of Agriculture and U.S. Department of Health and Human Services. 2020. Dietary Guidelines for Americans 2020-2025. https://www.dietaryguidelines.gov/

[18] World Health Organization. 2003. Diet, Nutrition and the Prevention of Chronic Diseases.

[19] World Health Organization. 2020. Healthy diet. https://www.who.int/news-room/fact-sheets/detail/healthy-diet