Indoor Room Recognition Menggunakan Multiple Instance Learning Convolutional Neural Networks

Yehezkiel Wuisang, Djoni H. Setiabudi, Alvin N. Tjondrowiguno
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
E-Mail: yehezkielwuisang@gmail.com, djonihs@peter.petra.ac.id, alvin.nathaniel@petra.ac.id

ABSTRAK

Pengenalan *enviroment* menjadi masalah modern yang muncul di era modern ini. Salah satunya adalah bagaimana sebuah ruangan dapat diidentifikasi jenisnya. Ruangan adalah *environment* yang sangat menantang untuk diidentifikasi karena identitas sebuah ruangan diwakili oleh berbagai macam jenis objek dalam ruangan tersebut yang ukuran dan bentuknya sendiri pun beragam. Dengan berkembangnya teknologi, khususnya *machine learning*, maka jenis ruangan dapat dikenali secara otomatis oleh sistem dengan bantuan *Image Processing* dan *Artificial Neural Network*.

Penelitian ini menggunakan algoritma Mean-Shift untuk melakukan segmentasi gambar dan metode *Convolutional Neural Network* (CNN) dibantu oleh penerapan *Multiple Instance Learning* (MIL) sehingga membentuk metode Multiple Instance Learning Convolutional Neural Network (MILCNN) untuk mengidentifikasi jenis ruangan. Selama pelatihan dan pengujian akan dilakukan penyesuaian terhadap metode agar dapat diaplikasikan dalam pengenalan jenis ruangan hanya melalui label gambar saja tanpa mencari label objek individu pada gambar.

Penelitian ini melakukan klasifikasi terhadap ruangan yang terdapat suatu gambar dengan mengenali fitur objek di dalamnya. Hasil akhir pengujian dari dataset menghasilkan persentase akurasi klasifikasi yang mencapai 43.05%.

Kata Kunci: *machine learning, artificial neural network, convolutional neural network, multiple instance learning, mean-shift, image recognition*

ABSTRACT

Environment recognition is a modern problem that appears in this modern era. One of them is how a room's type can be identified. Indoor room is a very challenging environment to identify because the identity of a room is represented by various types of objects in the room which by itself have various sizes and shapes. With the development of technology, especially machine learning, the type of room can be recognized automatically by a system with the help of Image Processing and Artificial Neural Network.

This study uses the Mean-Shift algorithm to segment images and the Convolutional Neural Network (CNN) method assisted by the application of Multiple Instance Learning (MIL) so as to form the Multiple Instance Learning Convolutional Neural Network (MILCNN) method to identify room types. During training and testing, adjustments will be made to the method so that it can be applied in recognizing room types only through image labels without looking for individual object labels on images.

This study classifies the room that contains an image by recognizing the features of the objects in it. The final result from testing the dataset produces a classification accuracy percentage that reaches 43.05%.

Keywords: *machine learning, artificial neural network, convolutional neural network, multiple instance learning, mean-shift, image recognition*

1. PENDAHULUAN

Terdapat berbagai macam tipe objek dalam sebuah ruangan di rumah-rumah pada umumnya. Objek ini dapat berupa perabotan ataupun peralatan rumah tangga yang digunakan dalam kegiatan sehari-hari seseorang. Beberapa objek yang terdapat dalam sebuah ruangan dapat mendefinisikan identitas dari ruangan tersebut. Contohnya ketika memikirkan tentang "ruang tidur", seseorang dapat menduga objek-objek yang berada atau sering ditemukan di "ruang tidur" yaitu tempat tidur, lemari pakaian, dan lain-lain. Demikian juga sebaliknya, sekumpulan objek tertentu dapat digunakan untuk mengklasifikasikan nama atau jenis dari sebuah ruangan.

Pengenalan sebuah ruangan dan objeknya juga dapat dilakukan melalui media gambar berupa foto yang diambil dari ruangan tersebut. Namun untuk mengklasifikasikan atau mengenali ruangan dari gambar dalam jumlah yang sangat besar dan beragam sangat sulit dilakukan karena keterbatasan manusia dalam melakukan pekerjaan yang berat dan banyak. Selain itu persepsi sesorang terhadap sebuah ruangan ataupun objeknya secara individual dapat berbeda-beda. Hal ini dikarenakan pengenalan ruangan oleh manusia didasarkan dari pengalaman dan pengetahuan manusia yang terbatas.

Untuk mengatasi keterbatasan manusia, teknologi *machine learning* dapat digunakan untuk melakukan tugas pengenalan ruangan berdasarkan gambar ruangan. Salah satu metode *machine learning* yang dapat digunakan yaitu Convolutional Neural Network (CNN) sebagai salah satu tipe *deep learning* yang umum diaplikasikan dalam menganalisis gambaran visual. CNN dapat mengekstrasi objek atau fitur yang terdapat dalam sebuah gambar dengan baik untuk proses training maupun penggunaan modelnya. Namun masih terdapat masalah dalam penerapan model-model CNN tradisional yang hampir seluruhnya bersifat *supervised-learning* di mana dalam aplikasinya untuk pengenalan gambar ruangan, selain dibutuhkan pemberian label jenis ruangan namun juga perlu dilakukan *semantic labelling* untuk seluruh objek dalam gambar. Hal ini menyebabkan pengembangan model memakan banyak biaya dan waktu bahkan sebelum proses learning dimulai. Selain itu, label data rentan terhadap kekeliruan karena proses penglabelan data dilakukan secara manual sehingga proses segmentasi objek dalam proses learning model CNN menjadi sulit untuk dilakukan dan menghasilkan performa yang buruk [3].

Beranjak dari masalah *supervised-learning* pada CNN tradisional maka dibutuhkan metode lain yang dapat mengurangi atau meminimalisir kebutuhan penglabelan data training. Salah satu

metode yang dapat digunakan yaitu dengan menggabungkan metode Multiple Instance Learning (MIL) dengan model CNN menjadi sebuah model bernama Multiple Instance Learning Convolutional Neural Network (MILCNN) sehingga memungkinkan terjadinya *weak supervised-learning*. Kemampuan MIL untuk memproses data dalam kelompok data yang disebut "bag" dan bekerja hanya dengan menggunakan label dari kelompok secara keseluruhan memungkinkan proses penglabelan data tidak perlu dilakukan hingga ke level instance atau objek individual [8].

Dalam pengaplikasian model-model state-of-the-art CNN untuk *indoor room recognition,* standar akurasi yang diperoleh adalah 75% namun metode tersebut masih menggunakan *supervised learning* [5]. Adapun pengaplikasian model *weak supervised learning* menghasilkan akurasi yang jauh lebih rendah dari standar akurasi tersebut, yaitu sebesar 43.08%. Hal ini disebabkan karena, walaupun model *weak supervised learning* yang digunakan sudah dapat melakukan klasifikasi hanya dengan menggunakan label ruangan saja, namun model tidak memiliki toleransi terhadap kesalahan label/*noise* pada data training [8]. Pada penelitian ini, penerapan MIL dalam proses training digunakan, selain agar proses training bersifat *weak supervised*, metode ini juga digunakan untuk mengatasi masalah mengenai toleransi kesalahan label/*noise* tersebut.

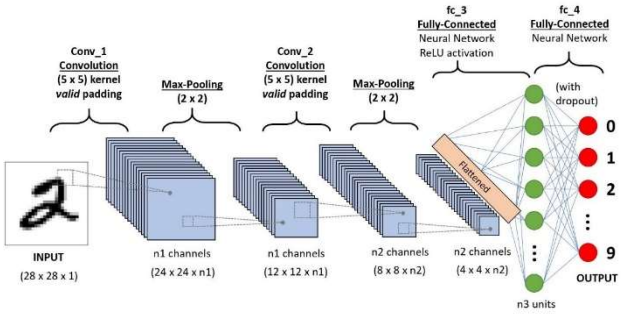
2. DASAR TEORI

2.1 *Mean-Shift*

Algoritma *mean-shift* memiliki aplikasi di bidang *image processing* dan *computer vision* yaitu dalam menentukan segmentasi atau *object tracking* dari sebuah gambar. Aplikasi *mean-shift* dalam *image processing* dilakukan dengan memperlakukan setiap *pixel* dalam gambar sebagai sebuah titik data dengan *attribute* berupa koordinat dan nilai yang merepresentasikan informasi tingkat kecerahan, warna, dan/atau fitur lainnya dari *pixel* tersebut. Kemudian akan dilakukan *clustering* terhadap *pixel-pixel* di dalam gambar yang memiliki attribut serupa. Posisi *mode* tiap *cluster* akan diperhitungkan dan jika terdapat dua *cluster* dengan jarak *mode* kurang dari *bandwidth* dan attributnya tidak jauh berbeda maka kedua *cluster* tersebut digabungkan menjadi satu *cluster.* Proses penggabungan cluster ini diulang terus menerus hingga tidak dapat terjadi penggabungan lagi, atau dengan kata lain, sudah mencapai *convergence.*

Hasil segmentasi dari *mean-shift* hanya dipengaruhi oleh ukuran kernel (*bandwidth*) yang ditentukan sehingga membutuhkan lebih sedikit intervensi manual dibandingkan algoritma segmentasi lainnya [9].

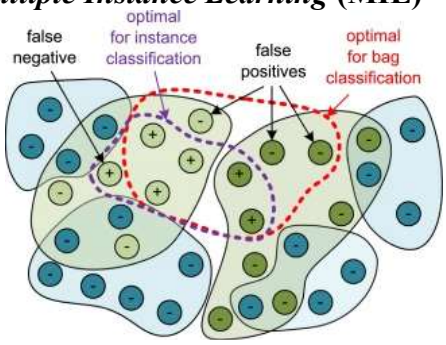
2.2 *Convolutional Neural Network* (CNN)



Gambar 1. Layer Convolutional Neural Network [6]

CNN adalah jaringan saraf yang berisi beberbagai *layer* didalamnya diantaranya adalah *convolutional layer, pooling layer, activation layer*. Contoh struktur layer dari CNN dapat dilihat pada Gambar 1. *Convolutional layer* bertugas untuk memeriksa pola dari suatu gambar dengan cara memberi filter lalu menghasilkan bobot berupa vector dan bobot tersebut akan bernilai tinggi jika dianggap melihat pola sebelumnya. Kombinasi bobot tinggi dari berbagai filter memungkinkan jaringan memprediksi konten suatu gambar. Lalu setelah memasuki tahap *convolutional layer* jaringan akan memasuki *pooling layer* yang berguna untuk mengurangi *spatial dimension* tetapi tidak untuk kedalamannya. Cara kerja *pooling layer* adalah dengan cara membagi pooling menjadi *max-pooling* dengan *output* maksimum nilai dari *sub-region* dan *average-pooling* dengan *output* rata-rata nilai dari *sub-region*. Sedangkan, *Activation Layer* berada di akhir atau diantara *convolutional layer* dan berguna untuk menentukan apakah sebuah pola yang teridentifikasi harus dieliminasi atau tidak [7]. Dapat dikatakan cara kerja dari CNN secara umum adalah dengan menganalisis *input* dan menyeleksi fitur yang dapat digunakan untuk klasifikasi dengan cara dilatih.

2.3 *Multiple Instance Learning* (MIL)



Gambar 2. Multiple Instance Learning [1]

MIL merupakan salah satu tipe *learning* dalam *Machine Learning* yang tergolong *weak supervised*. *Weak supervised learning* berarti MIL dapat berjalan dengan menggunakan data dengan label yang minimal menggunakan asumsi terstruktur berdasarkan label yang ada. Dataset dalam MIL tidak diberi label secara individual/*instance*, melainkan dikelompokan menjadi beberapa "*bag*" yang diberi label. Dari hasil *learning* MIL nantinya *bag* baru yang belum pernah ditemui dapat diklasifikasikan dan dalam beberapa kasus tiap instance juga dapat diklasifikasikan secara terpisah. MIL melakukan hal ini dengan menemukan "konsep" yang mendefinisikan sebuah *class* berdasarkan relasi antara data, baik dalam *bag* yang sama maupun berbeda.

Ilustrasi pada Gambar 2 menunjukkan bahwa MIL bekerja dengan dasar asumsi bahwa dalam setiap *bag* dengan label tertentu terdapat *instance*/fitur yang sesuai dengan label tersebut. *Bag* dinyatakan ke dalam sebuah *class* atau sebagai sebuah label selama terdapat minimal satu *instance* yang dapat mewakili fitur *class*-nya (*True Positive*) dan mengurangi pengaruh *instance* lainnya yang tidak relevan. Dalam beberapa kasus, sebuah *class*/label harus dinyatakan oleh lebih dari satu fitur. Jika terdapat lebih dari satu fitur maka pada *bag* harus terdapat minimal satu *instance* yang mewakili masing-masing fitur. Melalui proses *training* MIL, akan mencari dan mendefinisikan fitur-fitur tersebut dengan sendirinya untuk mengklasifikasikan *bag* atau *instance* baru [1].

2.4 *Multiple Learning Instance Convolutional Neural Network* (MILCNN)

MILCNN merupakan model yang didapat dari pengintegrasian konsep MIL ke dalam model dasar CNN. Berbeda dengan CNN yang menerima *input data* sebuah gambar, MILCNN menerima *input data* berupa sekumpulan gambar atau sebuah *bag* berisi

instances yang didapat dari hasil pemotongan gambar awal. Dilakukan beberapa perubahan terhadap beberapa fungsi dan komputasi yang digunakan oleh CNN pada umumnya. Beberapa komputasi yang diubah, antara lain *loss function* dan *gradient function* dimana terdapat pengaplikasian *aggregation function* didalamnya [8].

Untuk masing-masing *bag* (B) yang berisi N buah instance (x_n) dengan matriks labelnya ($y = \{0,1\}^{1xC}$) terhadap beberapa kategori klasifikasi (C), *loss function*-nya menggunakan Eq. 1.

$$f_{loss} = -\sum_{i=1}^{C} y_i \log\big(p(c_i = 1 \mid B)\big) \quad (1)$$

Di mana $p(c_i = 1 \mid B)$ merepresentasikan probabilitas *bag* tersebut untuk diklasifikasikan sebagai kategori ke-*i*. Probabilitas ini diperhitungkan menggunakan *softmax function* pada Eq. 2.

$$p(c_i = 1 \mid B) = \frac{\exp\left(\sigma(O_i)\right)}{\sum_{j=1}^{C} \exp\left(\sigma(O_j)\right)} \quad (2)$$

O_i adalah hasil *propagation* seluruh *instance* dalam *bag* untuk kategori ke-i ($O_i = \{o_{i1}, o_{i2}, \dots, o_{iN}\}$). Sedangkan σ merupakan *aggregation function* yang berfungsi untuk menyatukan hasil *propagation* seluruh *instance* menjadi satu. Salah satu penggunaan *aggregation function* dapat didefinisikan seperti pada Eq. 3.

$$\sigma(O_i) = \left(\frac{1}{N}\sum_{k=1}^{N} o_{ik}^{q}\right)^{\frac{1}{q}} \quad (3)$$

Di mana $q \in \mathbb{R}$ merupakan sebuah variabel konstan yang dapat mengatur besarnya pengaruh instance individual dalam agregasi. Untuk $q = 1$, tiap instance memiliki pengaruh yang sama rata, sehingga fungsi σ menjadi sama seperti *mean function*. Semakin besar nilai q maka pengaruh nilai terbesar pada argument akan semakin besar. Hingga untuk nilai $q \to \infty$, fungsi σ sama seperti *max function* [2].

Sedangkan fungsi gradient yang digunakan untuk *weight update* melalui *backpropagation* berhubungan dengan $\sigma(O_i)$ menggunakan diferensiasi pada Eq. 4.

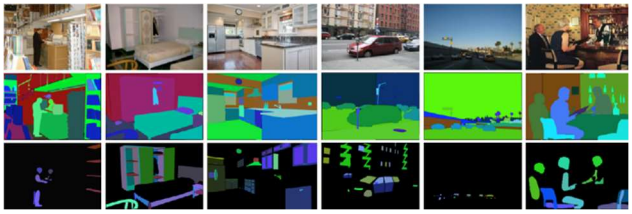
$$\frac{\partial f_{loss}}{\partial \sigma(O_i)} = -y_i + p(c_i = 1 \mid B)\sum_{j=1}^{C} y_j \quad (4)$$

3. DESAIN SISTEM

3.1 Analisis Data

Data yang digunakan adalah data yang menunjukan *indoor scene* berupa gambar berwarna (RGB) dan berformat JPG. Data dikumpulkan dari *open dataset* untuk *indoor room/scene image.* Dari dataset tersebut diambil kumpulan gambar dengan kategori-kategori jenis ruangan tertentu yang sudah ditentukan sebagai hasil klasifikasi penelitian. Dataset yang sudah diambil kemudian diseleksi dan mengeliminasi data yang tidak sesuai dengan kriteria gambar yang sudah ditentukan, yaitu terdapat minimal empat objek yang mendukung identitas ruangan; merupakan gambar keseluruhan ruangan; merupakan gambar tunggal; tidak terdapat grafik yang menutupi gambar; merupakan gambar ruangan asli; dan fokus gambar ada pada ruangan dan objeknya.

Beberapa *dataset* untuk *indoor room/scene image* yang tersedia antara lain ADE20K, ViDRILO, Places205, Places365. Dalam penelitian ini, dataset yang digunakan adalah *dataset* ADE20K. *Dataset* ADE20K merupakan sebuah *image dataset* yang disediakan oleh Massachusetts Institute of Technology (MIT) yang dapat diakses dan diunduh secara terbuka. Gambar-gambar pada dataset ini sudah dikelompokan berdasarkan labelnya dan sudah dibagi menjadi menjadi data *training* dan data *validation* untuk mendapatkan hasil yang optimal. Untuk setiap gambar disediakan juga segmentation dan hasil *masking*-nya terhadap gambar untuk mendukung *supervised learning* ataupun memvalidasi proses segmentasi yang dilakukan tersendiri seperti yang dapat pada Gambar 3.



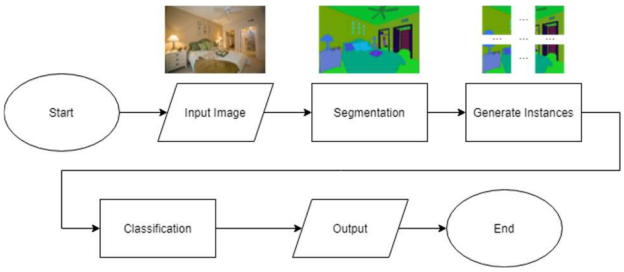
Gambar 3. Contoh data dataset ADE20K [4]

Dataset ADE20K memiliki jumlah data sebanyak 22210 file image. Setelah diseleksi berdasarkan kriteria yang telah ditentukan dalam penelitian ini, didapatkan bahwa data yang dapat digunakan dari dataset ini berjumlah 4660 file image. Dan juga terlihat bahwa tidak seluruh gambar memiliki kualitas gambar segmentasi yang sama. Beberapa gambar juga memiliki segmentasi dengan masking yang berlapis di mana gambar memiliki beberapa gambar segmentasi yang mewakili tingkat kedetailan masking yang berbeda.

Dataset yang sudah dikumpulkan dibagi menjadi dua. Yang pertama digunakan untuk proses *training*, dan yang kedua digunakan untuk proses *validation*. Proporsi pembagian data yang digunakan untuk *training* sebesar 90% dari jumlah dataset, sedangkan proporsi pembagian data yang digunakan untuk *validation* sebesar 10% dari jumlah dataset.

3.2 Desain Sistem

Analisis sistem membahas permasalahan bagaimana data yang di-*input* diproses sehingga dapat menghasilkan *output* yang relevan. Secara garis besar, sistem yang digunakan untuk mengklasifikasikan *input* gambar ruangan mencakup *preprocessing* terhadap data berupa proses segmentasi dan penyesuaian ukuran gambar, pemecahan gambar menjadi *instances* sebelum memasuki model jaringan, serta sistem *training* dan sistem *testing* dari MILCNN. Gambaran umum sistem dijelaskan dengan singkat melalui *flowchart* pada Gambar 4.



Gambar 4. Gambaran umum sistem *classification*

Sistem akan menerima *input data* berupa gambar, lalu akan dilakukan "*segmentation*" terhadap gambar agar gambar menjadi lebih sederhana dan dapat mempermudah pengekstrasian fitur dari gambar. Pada proses ini, gambar akan disegmentasi dengan menggunakan algoritma *mean-shift*. Sehingga dihasilkan gambar berupa masking dari objek-objek yang terdapat pada gambar. Hasil

segmentasi ini akan mempermudah model dalam menentukan letak, bentuk, dan orientasi dari setiap objek.

Kemudian sebelum gambar masuk ke MILCNN, perlu dilakukan perubahan terhadap bentuk data di mana satu kesatuan gambar dipecah menjadi beberapa gambar (*instance*) dan masing-masing diberi label yang sama dengan gambar awal (*bag*). Proses yang membuat beberapa gambar baru dari gambar *input* ini disebut "*generate instances*". Gambar-gambar baru didapat dari hasil pembagian gambar *input* menjadi beberapa *region* terpisah. Pada proses *training*, gambar-gambar tersebut diberi label sesuai dengan label gambar *input*.

Setelah itu akan dilakukan klasifikasi terhadap gambar menggunakan CNN (*Convolutional Neural Network*) yang sudah diintegrasikan dengan konsep MIL (*Multiple Instance Learning*) dan sudah melalui proses *training* dan *testing*. Sistem akan meng-*output*-kan teks berupa label yang didapat dari hasil klasifikasi oleh sistem.

3.2.1 Multiple Instance Learning Convolutional Neural Network (MILCNN)

Sebelum digunakan, di dalam MILCNN terdapat 3 proses mendasar, yaitu *preprocessing*, *training*, dan *validation* yang harus dilakukan terlebih dahulu agar MILCNN dapat bekerja dan melakukan tugas klasifikasi dengan baik. Ketiga proses ini dibutuhkan agar proses dapat mempelajari fitur-fitur dari masing-masing kategori ruangan dan mengaplikasikannya untuk mengenali jenis ruangan dari gambar yang sebelumnya tidak pernah dilihat oleh jaringan. Melalui proses-proses ini diharapkan akan dihasilkan sebuah model MILCNN yang *robust* dan tidak rentan terhadap perbedaan data ataupun kesalahan pada data *training*. Desain struktur model secara lengkap dapat dilihat pada Gambar 5.

Layer Type	Filter / Weight	Kernel Size	Stride Size
Input			
Convolutional + LeakyReLU**	16	3x3	1x1
Convolutional + LeakyReLU**	16	3x3	1x1
Maxpool		2x2	2x2
Convolutional + LeakyReLU**	32	3x3	1x1
Convolutional + LeakyReLU**	32	3x3	1x1
Maxpool		2x2	2x2
Convolutional + LeakyReLU**	64	3x3	1x1
Convolutional + LeakyReLU**	64	3x3	1x1
Maxpool		2x2	2x2
Convolutional + LeakyReLU**	128	3x3	1x1
Convolutional + LeakyReLU**	128	3x3	1x1
Maxpool		2x2	2x2
Convolutional + LeakyReLU**	256	3x3	1x1
Convolutional + LeakyReLU**	256	3x3	1x1
Maxpool		2x2	2x2
Convolutional + LeakyReLU**	512	3x3	1x1
Convolutional + LeakyReLU**	512	3x3	1x1
Maxpool		2x2	2x2
Flattening			
Fully Connected + LeakyReLU** + Dropout***	3600		
Fully Connected + LeakyReLU** + Dropout***	2400		
Fully Connected + LeakyReLU** + Dropout***	1600		
Fully Connected + LeakyReLU** + Dropout***	800		
Fully Connected + LeakyReLU** + Dropout***	64		
Fully Connected	10		
Aggregation			
Softmax			

* Instance is an undefined variable
** LeakyReLU alpha = 0.2
*** Dropout rate = 0.5

Gambar 5. Struktur model MILCNN

3.2.1.1 Preprocessing

Preprocessing merupakan proses yang dilakukan sebelum data memasuki proses lainnya dalam jaringan agar data yang masuk ke jaringan lebih terkontrol dan ternormalisasi. Proses *preprocessing* mencakup "*segmentation*" yang digunakan pada gambaran umum sistem *classification*.

Preprocessing untuk proses *training* dan *validation* dilakukan dengan beberapa tahap, dimulai dengan memvalidasi label yang sudah diberikan kepada data gambar dari dataset untuk menjaga relevansi data. Jika label tidak sesuai dengan daftar kategori yang ditentukan sebagai hasil klasifikasi maka data tidak digunakan dalam proses *training* dan *validation.* Selain itu, jika data tidak memiliki label yang jelas (i.e. "etc", "other") maka data dianggap sebagai "*outlier*" dan juga tidak digunakan untuk *training* dan *validation*. Selanjutnya data akan di-*resize* sehingga ukuran gambar setiap data menjadi sama. Kemudian data akan di-*shuffle* dengan mengacak urutan data. *Shuffling* data dapat membantu proses *training* secara tidak langsung.

Preprocessing juga dilakukan ketika MILCNN digunakan untuk klasifikasi setelah proses *training* dan *validation*, yaitu *testing* terhadap data *input* yang baru diterima MILCNN. Proses yang terjadi terbilang sama dengan *preprocessing* untuk proses *training* dan *validation*, namun perbedaannya adalah tidak adanya tahap validasi label dan *shuffling* karena data *input* berupa data individual, bukan dataset, yang belum diketahui dan ingin dicari klasifikasi labelnya sehingga kedua tahap tersebut tidak diperlukan.

3.2.1.2 Training

Proses *training* bertujuan agar model dapat mempelajari gambar dan menentukan fitur yang menjadi *point-of-interest* untuk mendefinisikan sebuah kategori/label. Pertama-tama, model perlu diinisialiasi dengan mendefinisikan *layer* beserta *weight*-nya dan *training method* model. Perlu didefinisikan juga nilai dari parameter *epoch*, *batch size*, dan *learning rate*. Untuk setiap *epoch*, *batch* data dengan jumlah yang sudah ditentukan masuk ke jaringan. Gambar diubah menjadi *bag of instances* dengan men-*generate instance*-nya, lalu dihitung nilai *loss*-nya menggunakan *loss function* yang kemudian digunakan untuk komputasi *gradient* oleh *gradient function*. Terakhir *weight* dalam jaringan di-*update* berdasarkan hasil komputasi *gradient* akhir dan *learning rate* yang ditentukan. *Validation* dilakukan di akhir *epoch* untuk mengetahui estimasi perkembangan pembelajaran model.

3.2.1.3 Testing

Proses *testing* dilakukan untuk mengetahui performa model setelah *training* dalam mengklasifikasikan sekelompok data yang belum pernah dilihat oleh model sebelumnya. Untuk melakukan proses *training* maka diperlukan model MILCNN dengan *weight* akhir yang didapat dari proses *training*. Model kemudian memprediksi kategori ruangan pada gambar dan mengoutputkan hasil klasifikasinya. Pada proses ini probabilitas klasifikasi dari model juga akan diperhitungkan sehingga dapat diketahui seberapa yakin model terhadap hasil klasifikasinya.

4. PENGUJIAN SISTEM

4.1 Data Pengujian

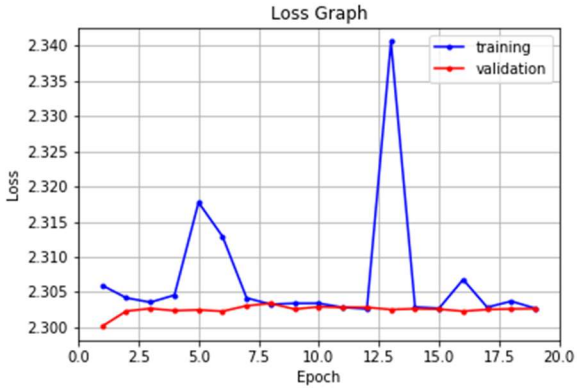
Data yang digunakan untuk pengujian terdiri dari 4660 yang terbagi menjadi 10 kategori label. Kategori-kategori tersebut beserta jumlahnya antara lain, "*bathroom*" *738* gambar, "*bedroom*" 1528 gambar, "*closet*" 79 gambar, "*corridor*" 122 gambar, "*dining_room*" 454 gambar, *garage*" 65 gambar, *kitchen*" 718 gambar, "*living_room*" 767 gambar, "*nursery*" 66 gambar, "*office*" *123* gambar. Data kemudian terbagi menjadi 4234 untuk *training* dan 426 untuk *validation*.

4.2 Pengujian Jumlah Data pada Dataset

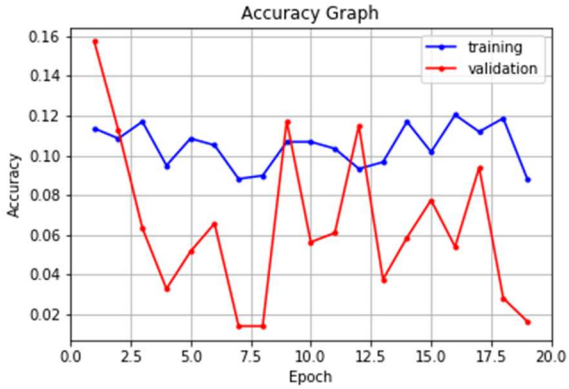
Untuk mengatasi masalah, terutama mengenai bias model, yang kemungkinan disebabkan oleh jumlah dataset yang tidak seimbang maka dilakukan pengujian dengan melakukan penyesuaian terhadap dataset *training* pengujian. Penyesuaian ini dilakukan dengan mengurangi jumlah data di dalam dataset *training* sehingga setiap kategori memiliki jumlah data yang sama dengan kategori yang memiliki data paling sedikit. Ukuran dimensi gambar pada pengujian ini adalah 1200x1200 *pixel* sehingga pemotongan terhadap gambar akan menghasilkan 16 *instances*.

Karena kategori degan jumlah data *training* paling sedikit adalah "*garage*" dengan jumlah 59 data *training*, maka data dalam tiap label diseleksi secara acak sejumlah 59 data sehingga jumlah data dalam dataset *training* berubah menjadi 590 data. Dataset *training* baru ini dinamakan "*balanced dataset*". *Balanced dataset* kemudian diujikan kembali melalui proses *training* tanpa ada perubahan lain.

Perkembangan proses *training* pengujian ini dapat dilihat pada Gambar 6 dan Gambar 7. Dari pengujian ini didapati bahwa rata-rata waktu yang dibutuhkan untuk menjalankan satu *epoch training* terhadap dataset *training* baru ini adalah 40 menit. Sedangkan, *loss* dan akurasi sepanjang proses *training* dan *validation* masih tidak berjalan dengan baik. Nilai *loss* yang dihasilkan lebih kecil dari sebelumnya karena perbedaan variasi data, namun tetap tergolong stagnant dan hanya mengalami perubahan yang relatif signifikan pada *epoch* tertentu saja. Akurasi juga tidak tidak terdapat kenaikan baik untuk data *training* maupun data *validation*. Sehingga untuk sementara disimpulkan bahwa keseluruhan masalah tidak disebabkan oleh perbedaan jumlah dataset.



Gambar 6. Grafik *loss* pengujian *balanced dataset*

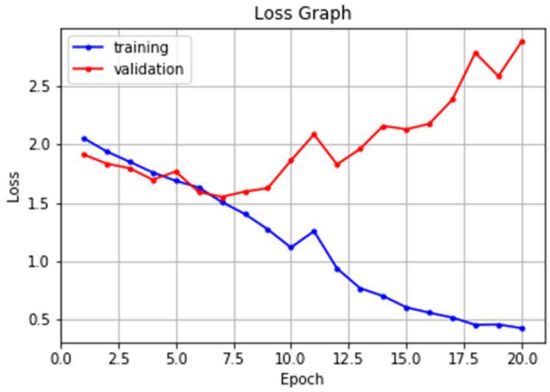


Gambar 7. Grafik akurasi pengujian *balanced dataset*

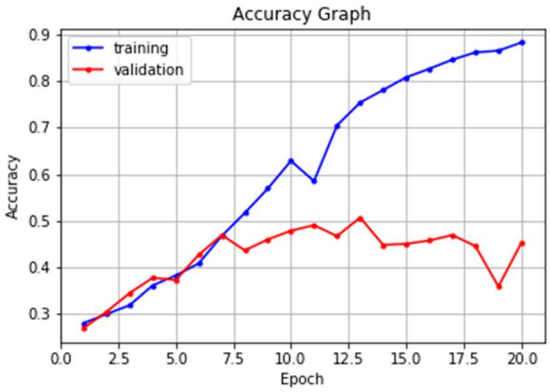
4.3 Pengujian Jumlah Region/Instance Gambar

Untuk memastikan bahwa model dapat melakukan proses *training* dengan pembelajaran fitur yang baik maka dilakukan pengujian dengan mengubah ukuran dimensi gambar menjadi 300x300 *pixel* sehingga tidak ada pemotongan terhadap gambar. Data yang masuk ke dalam model hanya terdiri dari satu instance yaitu data itu sendiri secara utuh. Model kemudian diujikan kembali menggunakan dataset *training* tanpa ada perubahan lain.

Perkembangan proses *training* pengujian ini dapat dilihat pada Gambar 8 dan Gambar 9. Dari pengujian didapati bahwa rata-rata waktu yang dibutuhkan untuk menjalankan satu *epoch training* terhadap dataset *training* adalah 41 menit. Terhadap data *training*, *loss* yang dihasilkan terus menurun dan akurasi terus meningkat, mengindikasikan bahwa model dapat mempelajari fitur dengan baik. Namun terhadap data *validation*, *loss* yang dihasilkan menurun dan mulai naik kembali pada *epoch* tertentu, begitu juga akurasi yang dihasilkan naik dan mulai menurun kembali pada *epoch* yang sama. Hal ini disebabkan karena terjadinya *overfitting* di mana model mulai menghafalkan data *training* dan masih adanya bias yang cukup signifikan untuk mempengaruhi klasifikasi.



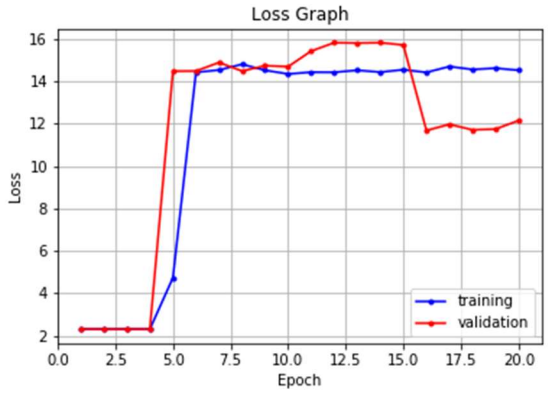
Gambar 8. Grafik *loss* pengujian satu *instance*



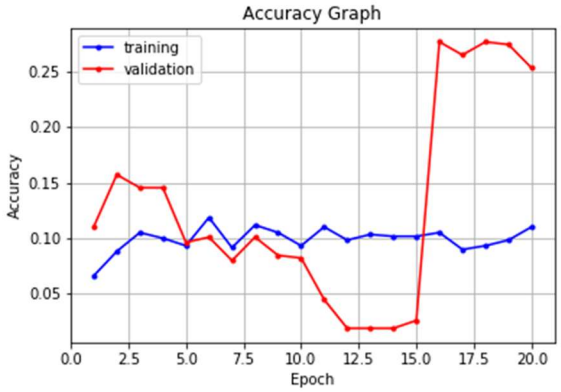
Gambar 9. Grafik akurasi pengujian satu *instance*

Pengujian model tanpa pemotongan gambar ini juga dilakukan terhadap dataset *training* pada pengujian sebelumnya, yaitu *balanced dataset* (dataset *training* berjumlah 590 data dengan jumlah yang sama rata tiap labelnya). Perkembangan proses *training* pada pengujian ini dapat dilihat pada Gambar 10 dan Gambar 11. Dari pengujian ini didapati bahwa rata-rata waktu yang dibutuhkan untuk menjalankan satu *epoch training* adalah 6 menit. Terhadap data *training*, *loss* yang dihasilkan mengalami kenaikan

yang sangat signifikan secara tiba-tiba pada *epoch* tertentu, dan akurasinya tidak mengalami kenaikan. Hal ini dikarenakan pengurangan dataset *training* membuat dataset tidak baik untuk digunakan karena hilangnya banyak informasi fitur yang harus dipelajari oleh model untuk dapat membedakan masing-masing label. Sehingga dari kedua pengujian ini, dapat disimpulkan bahwa dataset awal sudah merupakan dataset yang baik untuk digunakan dalam *training* dan perubahan terhadap jumlah dataset *training* hanya akan mengurangi kualitas *training*.



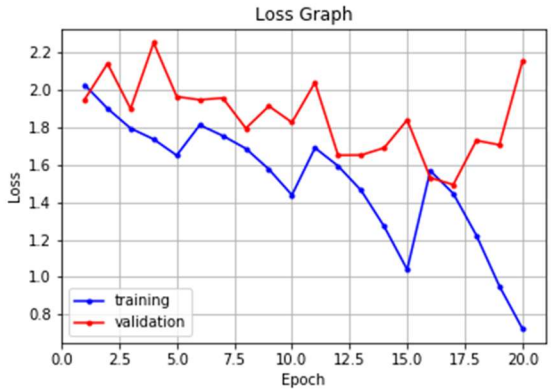
Gambar 10. Grafik *loss* pengujian satu *instance* dengan *balanced dataset*



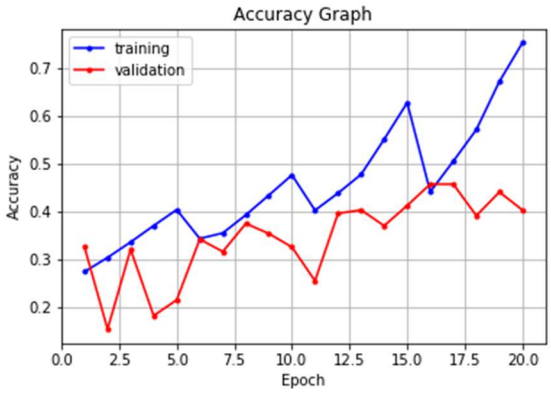
Gambar 11. Grafik akurasi pengujian satu *instance* dengan *balanced dataset*

Dari informasi-informasi yang didapatkan melalui pengujian-pengujian sebelumnya maka diusulkan penyesuaian jumlah instance secara iterative terhadap *epoch training*. Di mana jumlah instance akan dimulai dari satu instance kemudian bertambah setiap interval *epoch* tertentu menjadi empat instance, sembilan instance, dan seterusnya hingga mencapai target jumlah region yang ditentukan. Dengan cara ini model dapat mempelajari terlebih dahulu fitur label secara keseluruhan dan menggunakan informasi fitur tersebut untuk mempelajari fitur yang lebih spesifik ketika gambar dipotong menjadi bagian gambar yang lebih kecil secara bertahap. Untuk selanjutnya usulan ini akan disebut sebagi metode "*iteratively increasing instance*".

Dilakukan pengujian berdasarkan usulan demikian dengan interval *epoch* setiap 5 *epoch* selama 20 *epoch training*, dan target jumlah instance sejumlah 16 region 300x300 *pixel* (atau sama dengan target ukuran dimensi data 1200x1200 *pixel*) terhadap dataset awal. Perkembangan proses *training* pada pengujian ini dapat dilihat pada Gambar 12 dan Gambar 13. Dari pengujian ini didapatkan bahwa waktu yang dibutuhkan untuk satu *epoch training* berbeda-beda tiap *epoch* bergantung pada jumlah instance di *epoch* itu dengan rata-rata keseluruhan waktu *training* selama 20 *epoch* adalah 2 jam 3 menit. Terhadap data *training*, *loss* yang dihasilkan selalu menurun kecuali pada setiap interval *epoch* yang ditentukan di mana *loss* selalu sedikit naik terlebih dahulu. Begitu juga terhadap data validasi, dihasilkan nilai *loss* yang terkontrol dan akurasi yang cenderung meningkat. Masih ada kecenderungan untuk bias yang terlihat dari probabilitas masing-masing label saat klasifikasi.



Gambar 12. Grafik *loss* pengujian *iteratively increasing instance*



Gambar 13. Grafik akurasi pengujian *iteratively increasing instance*

5. KESIMPULAN DAN SARAN

Dari hasil analisis perancangan, implementasi, dan pengujian dapat disimpulkan bahwa:

- Metode *Multiple Instance Learning Convolutional Neural Network* adalah metode yang rentan mengalami bias dalam mengklasifikasikan *indoor room scene*;
- Dibutuhkan penyesuaian jumlah region instance yang dihasilkan dari pemotongan gambar untuk mempelajari fitur objek dalam *indoor room scene* yang bentuk dan ukurannya bervariasi. Penyesuaian jumlah region dilakukan secara bertahap mulai dari satu region instance;
- Metode *Multiple Instance Learning Convolutional Neural Network* kesulitan mencapai akurasi melebihi 75% dalam melakukan tugas *Indoor Room Recognition.*

Saran yang diberikan untuk mengembangkan sistem lebih lanjut lagi, antara lain:

- Penambahan *dataset* lain yang relevan untuk digunakan sebagai pembanding hasil *training;*
- Penambahan *layer* pada model CNN agar dapat mempelajari fitur dengan lebih baik;
- Penggantian fungsi agregasi pada model agar dapat mengontrol pengaruh *instance* terhadap *output* model terutama membatasi dan menyeleksi *noise instance* yang tidak baik digunakan untuk *training*.

6. DAFTAR PUSTAKA

[1] Carbonneau, M., Cheplygina, V., Granger, E. & Gagnon, G. 2018. Multiple Instance Learning: A Survey of Problem Characteristics and Applications. *Pattern Recognition*, 77, 329-353. DOI= https://doi.org/10.1016/j.patcog.2017.10.009

[2] Duffner, S. & Garcia, C. 2020. Multiple Instance Learning for Training Neural Networks under Label Noise. *In 2020 International Joint Conference on Neural Networks (IJCNN). IEEE, 1-7.* DOI= https://doi.org/10.1109/IJCNN48605.2020.9206669

[3] Espinace, P., Kollar, T., Soto, A., & Roy, N. 2010. Indoor scene recognition through object detection. In *2010 IEEE International Conference on Robotics and Automation*. IEEE, 1406-1413. DOI= https://doi.org/10.1109/ROBOT.2010.5509682

[4] MIT CSAIL. n.d. *ADE20K*. Retrieved October 10, 2020 from MIT CSAIL Vision Datasets. URI= https://groups.csail.mit.edu/vision/datasets/ADE20K/

[5] Othman, K.M. & Rad, A.B. 2019. An Indoor Room Classification System for Social Robots via Integration of CNN and ECOC. *Applied Sciences*, 9, 470. DOI= https://doi.org/10.3390/app9030470

[6] Saha, S. 2018. *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way*. Retrieved October 16, 2020 from Medium. URI= https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[7] Shubham, J. 2018. *What exactly does CNN see*? Retrieved September 13, 2020 from Medium. URI= https://becominghuman.ai/what-exactly-does-cnn-see-4d436d8e6e52

[8] Sun, M., Han, T.X., Liu, Ming-Chang, & Khodayari-Rostamabad, A. 2016. In *2016 23rd International Conference on Pattern Recognition (ICPR)*. IEEE, 3270-3275. DOI= https://doi.org/10.1109/ICPR.2016.7900139

[9] Zhou, H., Wang, X., & Schaefer, G. 2011. Mean Shift and Its Application in Image Segmentation. *Studies in Computational Intelligence*, 339, 291-312. DOI= https://doi.org/10.1007/978-3-642-17934-1_13